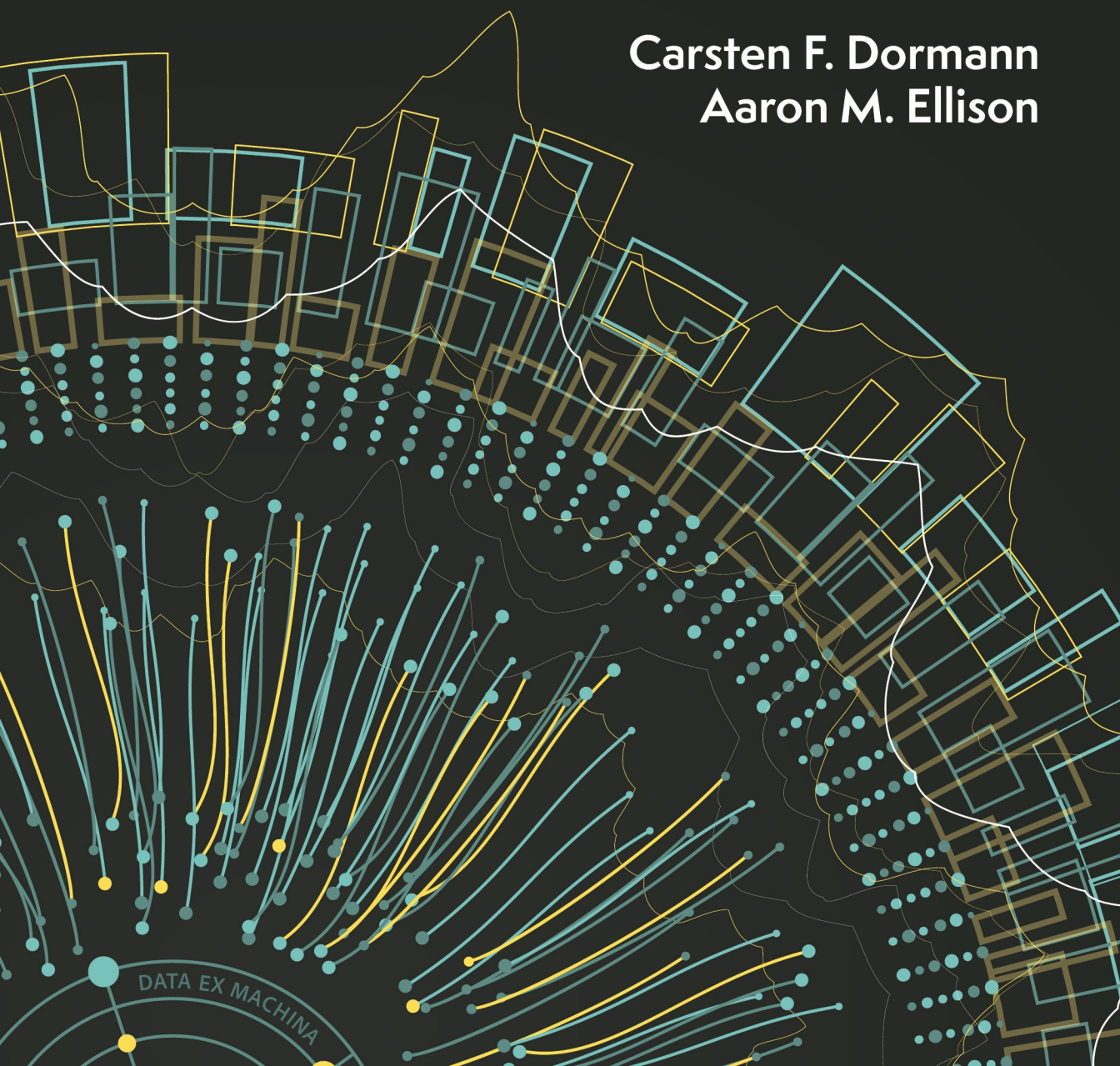
STATISTICS BY SIMULATION

A Synthetic Data Approach

Carsten F. Dormann
Aaron M. Ellison



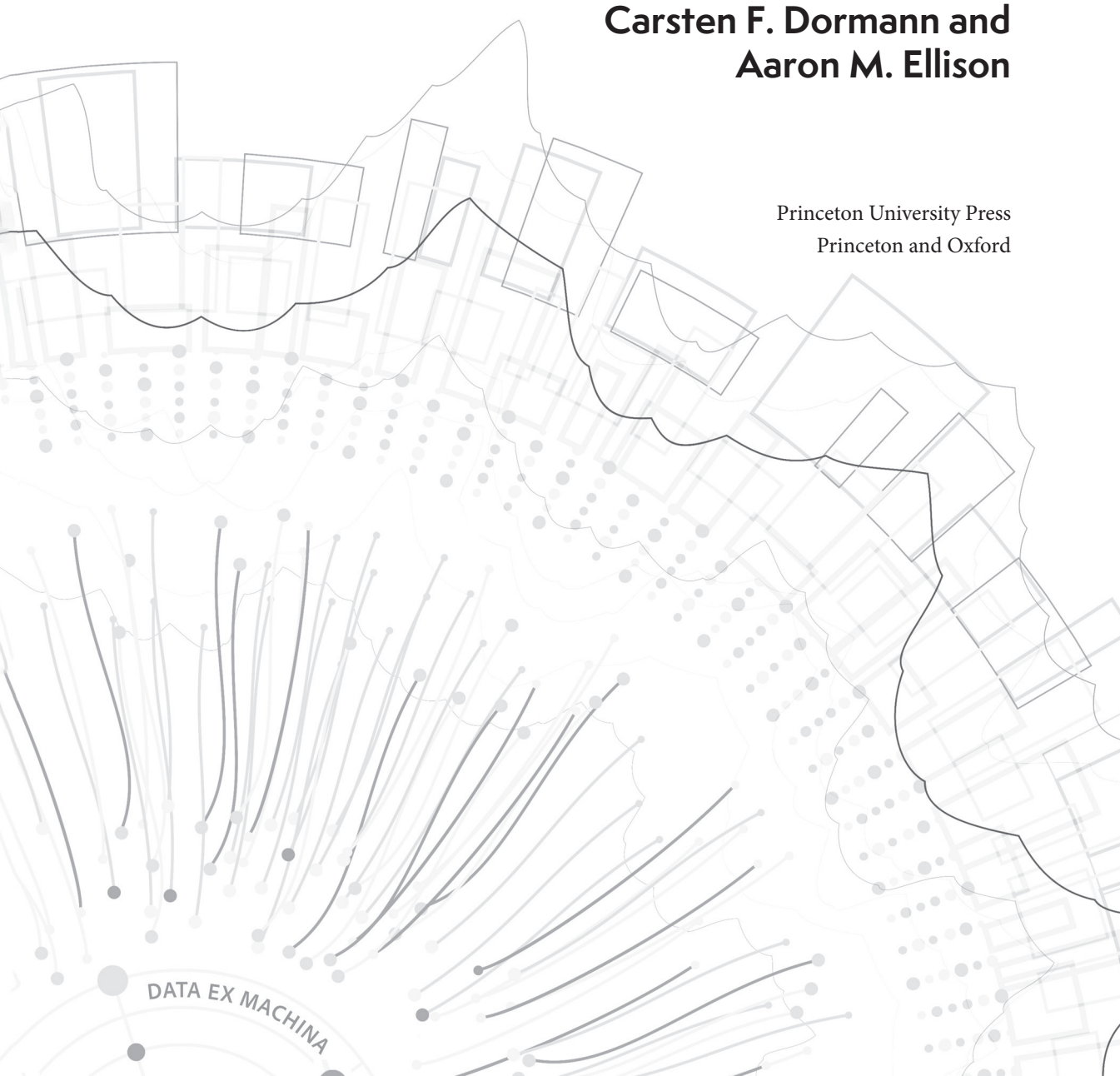
STATISTICS BY SIMULATION

STATISTICS BY SIMULATION

A Synthetic Data Approach

Carsten F. Dormann and
Aaron M. Ellison

Princeton University Press
Princeton and Oxford



Copyright © 2025 by Princeton University Press

Princeton University Press is committed to the protection of copyright and the intellectual property our authors entrust to us. Copyright promotes the progress and integrity of knowledge created by humans. Thank you for supporting free speech and the global exchange of ideas by purchasing an authorized edition of this book. If you wish to reproduce or distribute any part of it in any form, please obtain permission.

Requests for permission to reproduce material from this work
should be sent to permissions@press.princeton.edu

Published by Princeton University Press
41 William Street, Princeton, New Jersey 08540
99 Banbury Road, Oxford OX2 6JX

press.princeton.edu

All Rights Reserved

GPSR Authorized Representative: Easy Access
System Europe - Mustamäe tee 50, 10621
Tallinn, Estonia, gpsr.requests@easproject.com

ISBN 9780691273891
ISBN (pbk.) 9780691258775
ISBN (e-book) 9780691258768
LCCN: 2024054271

British Library Cataloging-in-Publication Data is available

Editorial: Sydney Carroll and Johannah Walkowicz
Production Editorial: Jaden Young
Text and Jacket/Cover Design: Wanda España
Production: Jacqueline Poirer
Publicity: William Pagdatoon

Jacket/Cover Credit: Cover illustration: Smeagorl / Alamy Stock Vector

This book has been composed in Semplicita Pro and Minion Pro

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

[*A*]*deoque obnoxii sumus sorti, ut sors ipsa pro deo sit, qua deus probatur incertus.*[1]

—Pliny, the Elder, *Naturalis historia*, II, VII (ca. 77 CE)

[1] [*A*]*nd we are so much at the mercy of chance that Chance herself, by whom God is proved uncertain, takes the place of God.* In Pliny, the Elder, with an English translation by H. Rackham (1938), *Naturalis historia*, English and Latin, volume 1. Harvard University Press, Cambridge, MA, USA, pp. 184–185.

Table of contents

Preface xi
Acknowledgments xv

Part I
Propositi: Why and how to simulate 1

1 General Introduction 3
1.1 What are simulated data? 4
1.2 Simulated data are specific 5
1.3 Yes, scientists really simulate data 6
1.4 There are many good reasons to simulate data 9
1.5 Useful background knowledge to use this book most effectively 10
1.6 Notational conventions 11
1.7 Structure, organisation, and flow 12
1.8 Summary 13

2 The basics of simulating data and the need for computational competence 17
2.1 A road map for simulation in statistics 17
2.2 Two simple examples 18
2.3 More complex examples 21
2.4 Simulating autocorrelated data 33
2.5 Simulation versus randomisation techniques 35
2.6 Summary 43

Part II
Ante mensuram: Prospective simulations of study designs and their power 45

3 Think before you act 47
3.1 The illusion of truth: A case study 47

3.2 The question comes first 48
3.3 Setting expectations, defining hypotheses 49
3.4 Testing hypotheses and assessing their support 55
3.5 Pre-registration 67
3.6 Summary 69

4 Prospective simulation of statistical power 73

4.1 Simple group comparisons 74
4.2 How many data points do we need for a simple correlation? 76
4.3 Is "recruit until significant" problematic? 80
4.4 How long does a time series have to be? 85
4.5 Improving estimates: Is the experiment powerful enough? 91
4.6 Summary 101

Part III
Post mensuram: Simulations in statistical analysis 103

5 Assumptions: Is that one important? 105

5.1 Linear regression requires the data to be normally distributed 106
5.2 Regression models also assume that errors in predictor variables are negligible or unimportant 110
5.3 The intended, rather than the realised, manipulation is an admissible predictor variable 112
5.4 ANOVA requires homoscedasticity 116
5.5 Multiple testing and the inflation of false positives 123
5.6 Hyper-distributions in mixed-effect models are normal 131
5.7 Correlations among predictors are the same outside the range of the observed data 137
5.8 Summary 145

6 Folklore: Is that rule-of-thumb true or useful? 153

6.1 Model selection does not always improve interpretation 154
6.2 Selecting one of two correlated predictors does not mitigate collinearity in regression and machine learning 165
6.3 It is not OK to categorise continuous predictor variables 172
6.4 Use Monte Carlo simulation when data are heteroscedastic 180
6.5 Time series should not be detrended by default 190
6.6 Machine learning and Big Data do not obviate rules-of-thumb 200
6.7 Summary 204

7 Workflows and pipelines can introduce and propagate artefacts 211

7.1 What can we do about missing data? 212
7.2 Types of missing data 212

7.3 Imputation of missing predictors 213
7.4 Estimating values for censored observations 222
7.5 Pre-selecting predictors 232
7.6 Regression on residuals 242
7.7 Error propagation 245
7.8 Workflow: Stringing multiple statistical steps into an analytical pipeline 251
7.9 Summary 259

Part IV
Post exemplum: Diagnostic simulations 267

8 Evaluating models: How well do they really fit? 269
8.1 Learning from the prior 270
8.2 What does a model tell us, and what does it not tell us? 273
8.3 Visualising more complex effects: conditional, marginal, and partial plots 276
8.4 Model diagnostics 281
8.5 Predicting with confidence is not the same as confidence in prediction 288
8.6 Iterative learning: New priors from old posteriors 305
8.7 Outlook 306
8.8 Summary 307

9 Post hoc alternatives to retrospective power analysis 311
9.1 Reprise: Prospective power analysis 312
9.2 What is retrospective power analysis? 313
9.3 Post hoc alternatives to retrospective power analysis 318
9.4 Summary: Most retrospective analyses should be avoided 332
9.5 Coda: What would a Bayesian do instead? 334

Part V
In posterum: Simulations for new methods 339

10 Combining studies: Meta-analysis and federated analysis 341
10.1 Whence the data? 341
10.2 From meta-analysis through federated analysis to complete analysis 342
10.3 Meta-analysis 345
10.4 Individual participant-level meta-analysis 352
10.5 One-step federated analysis 354
10.6 Multi-step federated analysis 358
10.7 Complete data analysis 360

10.8 Conclusions and outlook 367
10.9 Summary 370

11 Putting it through its paces: Does this new method work? 375

11.1 Unit testing 376
11.2 Dimensional analysis 379
11.3 Comparisons 380
11.4 Intellectual advancement 382
11.5 Intuitive understanding 382
11.6 Model-agnostic number of parameters: Generalised degrees of freedom 389
11.7 Know your limits 397
11.8 Summary 398

12 Outroduction: How far should we push simulations? 403

12.1 Stochastic weather forecasting 403
12.2 Infusing fake signals to test the workflow at LIGO 404
12.3 Virtual LIDAR scanning 406
12.4 Advanced simulation may be neither possible nor desirable 406

Appendix A
Useful R functions for data simulations 409

A.1 Drawing random values from a distribution 409
A.2 Doing things repeatedly: `for`-loops and `replicate` 410
A.3 Shuffling, resampling, and bootstrapping: `sample()` 417
A.4 Little helpers 418
A.5 Dedicated simulation packages 421

Index 423

Preface

Teaching statistics necessitates finding that delicate balance between fostering deeper understanding and providing recipes. On the one hand, deeper understanding often requires abstract detachment from specific application, yet not everyone is born an abstract thinker. On the other hand, recipes are bound to fail as soon as the statistical analysis becomes a bit more complicated, the data a bit more structured, the method a bit more sophisticated. Indeed, as LaDeau (2010) noted in the context of hierarchical Bayesian models, we may be doing analysis whose properties are not even known or proven; how can we possibly offer recipes?

Guiding ideas, general principles, and rules-of-thumb may be more suitable for avoiding common mistakes (Good & Hardin 2012). However, we often have encountered such folklore that seemed to have gained the status of general principles and rules without justification (see Chapter 6). But how can we differentiate between myth and reality unless someone else has explicitly addressed it in the context of the problem (or a similar one) that we are facing in our own analysis?

Time and again we have found ourselves doing, or recommending that our students and colleagues similarly do, small simulations to see whether our intuition is correct, a rule-of-thumb makes sense, or violating an assumption really presents a problem, etc. Such simulations can be useful in any scientific discipline and at every stage of scientific inquiry—from designing and preregistering a study, through organising an analytical workflow, modelling the data, and evaluating model fit, to meta-analysis and the design of new indices and methods. Instead of relying on rules-of-thumb for designing and analysing our studies, simulations can be used to plan ahead, pilot observational and experimental designs, check the effects of violating analytical assumptions, and inspect and validate fitted models. Data simulations can help anticipate uncertainty, visualise model fits, and foster a deeper understanding of the approaches used throughout a research project. We recognise that using simulations in these ways will not be new or surprising to professional statisticians. However, we are unaware of another text that allows (post)graduate students[1] or "domain" scientists and practitioners who use statistics on a day-to-day basis to learn how to do simulations and use them effectively to gain a better understanding of statistical analysis.

We emphasize that this is not a book from which to learn statistics. Rather, we expect that readers of this book will already have a fundamental understanding of basic statistical concepts on the level of what we have presented in our previous books (Gotelli &

[1] Both "graduate" and "postgraduate" students refer to students who have completed an undergraduate degree.

Ellison 2012; Dormann 2020), especially distributions, multiple regression and GLMs. We encourage readers who are interested in using data simulations to learn statistics to consult Vasishth & Broe (2010). However, with this book, we make doing and using simulations accessible to scientists in fields ranging from sociology and psychology, through neurobiology and ecology, and on to chemistry and physics; our examples and case studies are drawn from the wide range of disciplines where simulations are routinely used. We acknowledge that some of these examples and case studies take a relatively deep dive into the details that may matter only to the cognoscenti of particular fields, but the details really do matter in practice when using statistics and data simulations in any field.

Simulations are useful for frequentist and Bayesian methods (and statisticians) alike, and we do not take sides herein in the longstanding philosophical debate on their relative merits or consistency (e.g., Ellison 1996, 2004; Ellison & Dennis 2010). We are both statistical pragmatists; in our own research we use analytical and inferential methods appropriate to the question at hand. There is no need for a complex hierarchical model when the results of a simple *t*-test will be just as meaningful and clearer to interpret (e.g., Murtaugh 2007), but other problems are simply intractable without resorting to a full-scale Bayesian assault. Many of our examples include the reporting of *p*-values; simulation is a remarkably valuable tool for understanding what *p*-values are and what they are not (Schervish 1996). In our own work, and in this book, we follow the guidance of the American Statistical Association on the appropriate use of *p*-values and eschew their use in asserting the evidence for or against null hypotheses (Wasserstein & Lazar 2016).

The book is organised in five parts. First, we introduce the reasoning behind simulations using familiar and practical examples (Part I). We then proceed to show how to use simulations to plan experiments or observational studies and preregister them (Part II); how to use simulations to explore the consequences of violating the assumptions of standard statistical models, differentiate myths from reality in common statistical guidelines and rules-of-thumb, and develop analytical workflows (Part III); and how to take advantage of simulations to explore model fits and assess them with model diagnostics while not drowning in the swamp of retrospective power analysis and post hoc justifications for "non-signficant" results (Part IV). Finally, in Part V, we illustrate the importance of using simulations for synthesising and modelling data across many studies using classical meta-analysis and the emerging and powerful federated networks of data analysis for integrating, analysing, and synthesising sensitive, protected, or otherwise siloed data. We conclude Part V with a discussion of the crucial role of simulation in the development of new statistical indices, metrics, and methods, and the associated emerging standards that require simulation as a condition for publishing such methods in ecology and medicine.

We hope that any scientist who wants to learn how to better understand the statistical methods they already use, analyse data using methods beyond those illustrated in standard introductory textbooks, or develop new methods and indices for themselves and others will find something of value in this book. All of our examples are accompanied by reproducible code (in the `R` programming language: R Core Team 2023) that can be modified easily for related problems or translated into comparable software or programming languages such as `Python` or `Julia` using large language models such as Llama, StarCoder, or BLOOM.[2] Detailed footnotes and extensive bibliographies in each chapter provide additional pointers to similar problems and comparable simulations in a range of fields.[3] The 12 chapters of the

[2] Although ChatGPT is the most familiar large language model, it is not the best choice for coding and other programming applications (e.g., Ozkaya 2023; Cooper et al. 2024).

[3] Our use of footnotes is inspired by two sources. First, G. Evelyn Hutchinson's classic 1978 textbook, *An Introduction to Population Biology*, brought new meaning and value to footnotes in science textbooks. In many ways,

book could be comfortably covered in a semester-long, discipline-specific course in applied statistics for (post)graduate students or advanced undergraduates.

References

Cooper, N., Clark, A. T., Lecompte, N., et al. (2024). Harnessing large language models for coding, teaching and inclusion to empower research in ecology and evolution. *Methods in Ecology and Evolution*, 15(10), 1757–1763. doi: 10.1111/2041-210X.14325.

Dormann, C. F. (2020). *Environmental Data Analysis: An Introduction with Examples in R*. Heidelberg, Germany: Springer.

Ellison, A. M. (1996). An introduction to Bayesian inference for ecological research and environmental decision-making. *Ecological Applications*, 6(4), 1036–1046. doi: 10.2307/2269588.

Ellison, A. M. (2004). Commentary on statistics and the scientific method in ecology by Brian Dennis. In M. L. Taper & S. R. Lele (Eds.), *The Nature of Scientific Evidence* (pp. 362–367). Chicago, IL, USA: University of Chicago Press.

Ellison, A. M. & Dennis, B. (2010). Paths to statistical fluency for ecologists. *Frontiers in Ecology and the Environment*, 8(7), 362–370. doi: 10.1890/080209.

Good, P. I. & Hardin, J. W. (2012). *Common Errors in Statistics (And How to Avoid Them)*, 4th edition. Hoboken, NJ, USA: Wiley.

Gotelli, N. J. & Ellison, A. M. (2012). *A Primer of Ecological Statistics*, 2nd edition. New York, NY, USA: Oxford University Press.

Hutchinson, G. E. (1978). *An Introduction to Population Biology*. New Haven, CT, USA: Yale University Press.

LaDeau, S. L. (2010). Advances in modeling highlight a tension between analytical accuracy and accessibility. *Ecology*, 91(12), 3488–3492. doi: 10.1890/09-2249.1.

Murtaugh, P. A. (2007). Simplicity and complexity in ecological data analysis. *Ecology*, 88(1), 56–62. doi: 10.1890/0012-9658(2007)88[56:SACIED]2.0.CO;2.

Ozkaya, I. (2023). Application of large language models to software engineering tasks: Opportunities, risks, and implications. *IEEE Software*, 40(3), 4–8. doi: 10.1109/MS.2023.3248401.

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Schervish, M. J. (1996). *P* values: what they are and what they are not. *The American Statistician*, 50(3), 203–206. doi: 10.1080/00031305.1996.10474380.

Vasishth, S. & Broe, M. (2010). *The Foundations of Statistics: A Simulation-Based Approach*. Heidelberg, Germany: Springer.

Vodolazkin, E. (2018). *Solovyov and Larionov*. Translated from the Russian by Lisa C. Hayden. London, England, UK: Oneworld Publications.

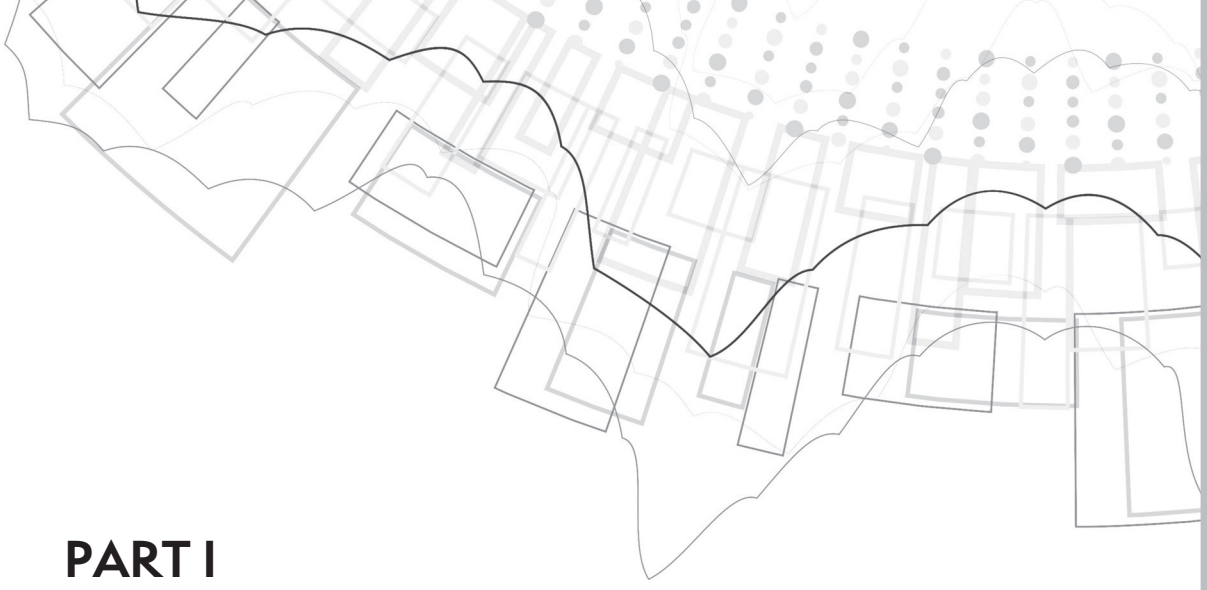
Wasserstein, R. L. & Lazar, N. A. (2016). The ASA statement on *p*-values: Context, process, and purpose. *The American Statistician*, 70(2), 129–133. doi: 10.1080/00031305.2016.1154108.

we learned more from the footnotes while using this text than we did from the main text. His dedication to Latin was also inspiring. Second and paraphrasing the historian Solovyov, the main character in Eugene Vodolazkin's 2018 novel *Solovyov and Larionov*, footnotes not only express respect for one's predecessors but also reveal that there is no longer any realm of knowledge where we are first in line.

Acknowledgments

Both authors have learned a lot about effectively using simulations in statistics through their many years of teaching biometry and biostatistics to undergraduates and (post)graduate students in Germany and the United States of America. Carsten tested some of the examples on BSc and MSc students in Environmental Science at the University of Freiburg. Such trials clarified some didactic issues for us while causing some frustration and confusion for them; we are thankful for their endurance and resilience! We also are grateful to the anonymous reviewers and our colleagues who provided useful critiques and feedback on early versions of various chapters or the entirety of this book. We would especially like to express our gratitude to Dr. Daniela Zöller, at the Institute of Medical Biometry of the University Clinic Freiburg, for talking us through federated analysis (Chapter 10) and providing detailed feedback on that chapter; and our friends and colleagues Nick Gotelli (University of Vermont) and Florian Hartig (Universität Regensburg) for carefully reading and critiquing the entire manuscript. Ikedichi Azuh (University of Freiburg) kindly provided the reworked Fig. 6.6. Finally, we thank Sydney Carroll, our editor at Princeton University Press, for believing in and taking on this project, and her team at the Press for its high-quality production.

We began working on this book while Aaron was a Natural Sciences Fellow at the Swedish Collegium for Advanced Study in 2022. His continued work on it was supported by a Humboldt-Forschungspreis from Germany's Alexander von Humboldt-Stiftung, which enabled him to work closely with Carsten in Freiburg during two extended research stays in 2023.



PART I

Propositi: Why and how to simulate

Our shelves are replete with statistics textbooks that range from the introductory texts we used as undergraduate students to advanced and highly technical monographs on narrow, abstruse topics. Why is there need for another one, and why this one in particular?

Teaching and learning statistics requires finding that delicate balance between encouraging the exploration of abstract statistical concepts and providing specific recipes for solving particular problems. As students and teachers, we regularly use simulations to help maintain our balance between diving into abstraction or relying on intuition, and resorting to tried-and-true recipes or well-known but unsubstantiated rules-of-thumb. Such simulations can be useful in any scientific discipline and at every stage of scientific inquiry—from planning and study design, through data collection, analysis, and modelling, and on to developing new methods. The value of simulations will not be news to professional statisticians. However, we are unaware of another text aimed at (post)graduate and advanced undergraduate students or "domain" scientists and practitioners who use statistics on a day-to-day basis from which they can learn how to do simulations and use them effectively to gain a better understanding of statistical analysis. With this book, we make doing and using simulations accessible to scientists in fields ranging from sociology and psychology, through neurobiology and ecology, and on to chemistry and physics; our examples and case studies are drawn from the wide range of disciplines where simulations are routinely used.

The two chapters in this first part of the book are a gentle introduction to using simulation in our day-to-day statistical work. Chapter 1 sets the overall tone of the book, provides general guidance for its effective use, and supplies an outline for finding key topics of interest in subsequent chapters. Chapter 2 uses a range of simple examples to illustrate the nine steps for developing meaningful simulations; highlights the key values of simulation and how much can be achieved with it; and serves as an introduction to moving from algorithms and pseudocode to reading and writing formal R code. The most important simulation-specific functions in R are reviewed in the Appendix.

1

General introduction

In 2009, a poster at the Annual Meeting of the Human Brain Mapping conference caused a stir. It showed significant brain activity in response to the subject being shown photographs of humans in emotional valence. The subject was an Arctic salmon—and it was dead (Bennett et al. 2009). In the same year, Kriegeskorte et al. (2009) highlighted a disturbing amount of circular reasoning and double-dipping in neuroscience, particularly in fMRI analyses, where the test statistics were not independent of the selection criteria and common analyses produced spurious results. Later research comparing how statistical software designed and used by different manufacturers of fMRI machines handled spatial autocorrelation in the voxels revealed an error rate of up to 70% (Eklund et al. 2016). Yet even when done correctly, effects can be tiny and to reliably detect them requires sample sizes much larger than usually published (Marek et al. 2022). Clearly, fMRI-based science needed improvement. But are the statistical approaches and analyses used in fMRI any different from those used in other scientific fields every day?

All statistical methods make some assumptions about the independence, distributions, representativeness, etc. of the samples and the data collected from them. Most classical statistical methods have been mathematically (analytically) proven to yield "asymptotically unbiased estimates." This phrase means that a statistical estimate will converge to its "true" value (that's the "unbiased" bit) given an infinite number of observations (that's the "asymptotic" bit) of randomly-collected samples (observations or data points) that conform to all the assumptions made in the statistical model. However, no one collects an infinite amount of data and residual assumptions are never 100% met, so it is important to ask how reliable our estimates are for finite or even rather small datasets. As a corollary, we also want to know how small is too small (i.e., how many observations are too few for us to compute unbiased, or at least reliably informative, estimates), how large is large enough, and whether our conclusions are robust to minor violations of model assumptions.

Furthermore, statistical methods generally have been developed as stand-alone tools. For example, general linear models (GLMs, of which the familiar analysis of variance [ANOVA] is a special case) are used to identify the expected (or average) response of one variable as a function of one or more predictor variables. Principal component analysis (PCA) is used to reduce the dimensionality of a large and often unwieldy set of variables (either predictors or responses) to a more manageable, smaller number (usually two or three) of composite variables that are linear combinations of the original data. Used by itself, PCA works well as a descriptive, hypothesis-generating tool, whereas a GLM works well as a predictive, hypothesis-testing tool. But what happens if we combine a PCA and a GLM in workflow? Will the PCA-GLM approach still "work?" And what about more complicated methods and

workflows using machine learning or neural networks? Will the resulting estimates still be unbiased or will uncertainty and errors propagate throughout the workflow and create statistical artefacts?

Questions such as these rarely can be answered for even a small number of statistical methods, hardly ever by non-mathematicians, and may even defy analytical solutions. Instead, mathematicians and statisticians rely on simulations to answer them.

This book aims at providing an entry to understanding statistics using simulations. By going through many examples, both conceptually and with code, we hope to lower the bar so that scientists who use statistics can simulate data and associated analyses themselves. Although simulations are a standard tool in statistical research and the development of new statistical methods, we do not present or test general methods. Rather, we illustrate the value of simulations by working through key issues and exemplar datasets in specific contexts representing a broad range of research areas—from sociology and archaeology to ecology and physics, and many in between.

What you will learn from this chapter

- What simulated data are, why they are important, and important ways in which they are used.
- How the book is organized and what you need to know to use this book effectively.
- How to read, interpret, and use the Code Boxes.
- What notational conventions are used for mathematical expressions throughout the book.

1.1 What are simulated data?

Simulated data are data similar to what we plan to collect or have already collected, but are created using computational algorithms.[1] We think it is helpful to think of such data as emerging from a "data-generating model" (DGM). The central challenge addressed in this book is to define such a data-generating model and then to use it effectively. When simulating data, we usually take advantage of random-number generators to include random variation in the simulated dataset that is similar to the random variation that occurs in real data. Examples of such random variation include differences among sampled individuals (i.e., between-subject variance), random positions, and residual errors, among many others.

Computers can simulate data efficiently and rapidly. When we run many simulations (e.g., thousands of them, each starting with a different random "seed"), we end up with replicate datasets for which we know the true underlying parameter values, the sampling design, underlying correlations, etc. When we analyse these simulated data, we can quickly check for and identify biases in our estimates and determine whether our methods are reliable and robust. We also can use simulations to test model fits and generate *p*-values. Pedagogically, it can be argued that if we cannot simulate our data, we probably do not know what we are doing when we are analysing them. Even if we think we know what we are doing, simulating data opens our eyes to what is simple and what is difficult in statistical analysis.

[1] Other terms you may encounter for simulated data include "synthetic data," "fake data," or "invented data." Fake data and invented data have negative connotations and suggest that we are "cooking the books" by doing something wrong or illegitimate. "Synthetic data" is fine, but in the statistical sciences, "simulated" is used more frequently than "synthetic."

Simulated data can take many forms, from a small number of observations of a single variable to a very large number of observations generated by a sophisticated model of a complex system driven by different underlying stochastic processes and sampled with a different stochastic sampling process (e.g., Zurell et al. 2010). Think, for example, about modelling traffic jams or representing the associated emergency calls as a way of collecting data. Both processes can be simulated at the level of the data (which is the focus of this book) or at the deeper level of the process(es) that could plausibly generate the data. The latter approach is touched upon in Chapter 12.

One more thing before we start. Simulations and reality have a lot in common.[2] In both, strong effects will shine through but weak effects will be hard to discern through the fog of high variability (Silver 2012).

1.2 Simulated data are specific

Simulation sacrifices generality for ease of use. That is, we typically use simulations to ask questions about a specific dataset or a very specific type of statistical analysis. Unlike an analytical proof, the results of a simulation study rarely are applicable to a wide range of statistical methods, even those that share underlying mathematical structures. Despite its lack of generality, however, simulation is a great tool to use to explore how a particular kind of statistical method works and what it can actually tell us about the kinds of data we can collect or already have on hand.

Any real dataset is but a single instance (and, we hope, a random one) of all the possible datasets that could have been collected from the population of interest. Although we might have sampled more or fewer subjects or sites, used a different method, or collected the data at a different time, we did not. We are usually interested in analysing such specific datasets and using the analyses to make more general inferences about all the datasets we did not collect, i.e., the population.

When we simulate data, we have such a specific dataset in mind, but we do not restrict ourselves to the exact values in our real dataset. Rather, we are asking how much quantitative information can be gleaned from this specific instance of a more general kind or type of dataset.[3] In other words, is the information in the data sufficient for us to extract a signal from the noise? To answer this question using data simulation, we repeatedly generate data with a similar underlying structure and apply our intended analysis to them. Since we defined and set the parameters in our simulations, we can now check: How often do we find the effect we programmed into the simulation? Are our parameter estimates biased? If we are interested in testing our model, we can also ask whether the *p*-value of the output correctly reflects the number of significant effects across the simulations.

If a statistical method applied to our simulated data yields unbiased estimates or the expected distribution of *p*-values, then we can be more confident that the application of the method to the analysis of our single and unique real dataset will be similarly unbiased and the *p*-value will accurately reflect the statistical "significance" of our results. But remember the dead salmon. If the application of our method to our simulated data yields, for example,

[2] In fact, many science-fiction writers and even some dyed-in-the-wool philosophers have asserted that reality may be nothing more than a simulation (e.g., Adams 1979; Bostrom 2003).

[3] We distinguish here between types and instances of a variable or a dataset. A *type* is a more general category, whereas an *instance* is a specific example of the type. For example, the value 10 is an instance of a variable whose type is "integer." An interesting feature of types is that they are hierarchical, so that objects that are usually thought of as types also can be used as instances of other objects.

biased estimates, numerical anomalies, or incorrect standard errors, then we should not trust a similar analysis of the real dataset.

1.3 Yes, scientists really simulate data

Although some mathematicians still frown upon simulations, there are many interesting problems for which closed-form analytical solutions have not yet been identified. Using computers to simulate data and explore a range of possible solutions of analytically insoluble problems now is recognised as a legitimate approach to solving them; one of the first and perhaps most familiar examples is the solution to the four-colour map problem (Appel & Haken 1989; see Meyer & Schmidt 2012 for others). For many statistical methods, simulations are the only way available to validate established and proposed methods.

Not only do many scientists simulate data and analyse them,[4] statisticians and many others teaching both theoretical and applied statistics actively encourage this practice (e.g., Crawley 2007; Jones et al. 2014; Morris et al. 2019; McElreath 2020; Crump et al. 2021; Donovan et al. 2021). Really, there is no reason not to, and we illustrate the power of simulation with a few straightforward examples in the following subsections and in the next chapter. These motivate the more complex problems described in the rest of the book and should whet your appetite for doing simulations yourself.

1.3.1 Geographic biases in reported crime rates

Governments collect and analyse data on crimes to identify where they are committed and the kinds of individuals most likely to commit them. Systematic reviews of these data have suggested that most crimes reported to police occur in a small number of neighbourhoods (Lee et al. 2017). This result has been used to develop strategies that, their proponents assert, will most efficiently deploy people and resources to those areas where crimes are most likely to occur. Critics point to the racial, socioeconomic, and other biases on which these spatially-targeted strategies are based and that they perpetuate. Simulations have been used extensively to support both sides of the debate (Liu & Eck 2008). An important, but rarely asked question, however, is whether the data used to build the crime simulations are themselves biased.

Buil-Gil et al. (2022) used simulation to address this question. They examined potential sources of bias in reported crime statistics in the UK's Crime Survey for England and Wales (CSEW). These researchers simulated three things for the city of Manchester, England: a population of people, the number of crimes experienced by each individual in the population, and the number of those crimes that were actually reported to the police (Figure 1.1). Buil-Gil et al. (2022) then asked whether the bias in the number of reported crimes changed from the smallest spatial scale (a neighbourhood of $\approx$125 households) to larger ones (communities [collections of neighbourhoods], precincts [collections of communities], and wards [collections of precincts]).

The results were striking. First, the simulations accurately reflected the CSEW data that nearly two-thirds (62%) of all committed crimes are not reported to the police. But the variation in non-reporting was largest among neighbourhoods (inset plot in Figure 1.1). That is, different neighbourhoods, each of which tends to be demographically more homogeneous than the larger communities, precincts, or wards, report crimes at different rates. These

[4] Any list of fewer than thousands of references is unfair to those doing simulations, but here are a few more recent examples: Royston & Sauerbrei (2014); Boulesteix et al. (2018); Jayasekera et al. (2018); Boulesteix et al. (2020); Adams & Collyer (2022); Martínez-Santalla et al. (2022); DiRenzo et al. (2023).

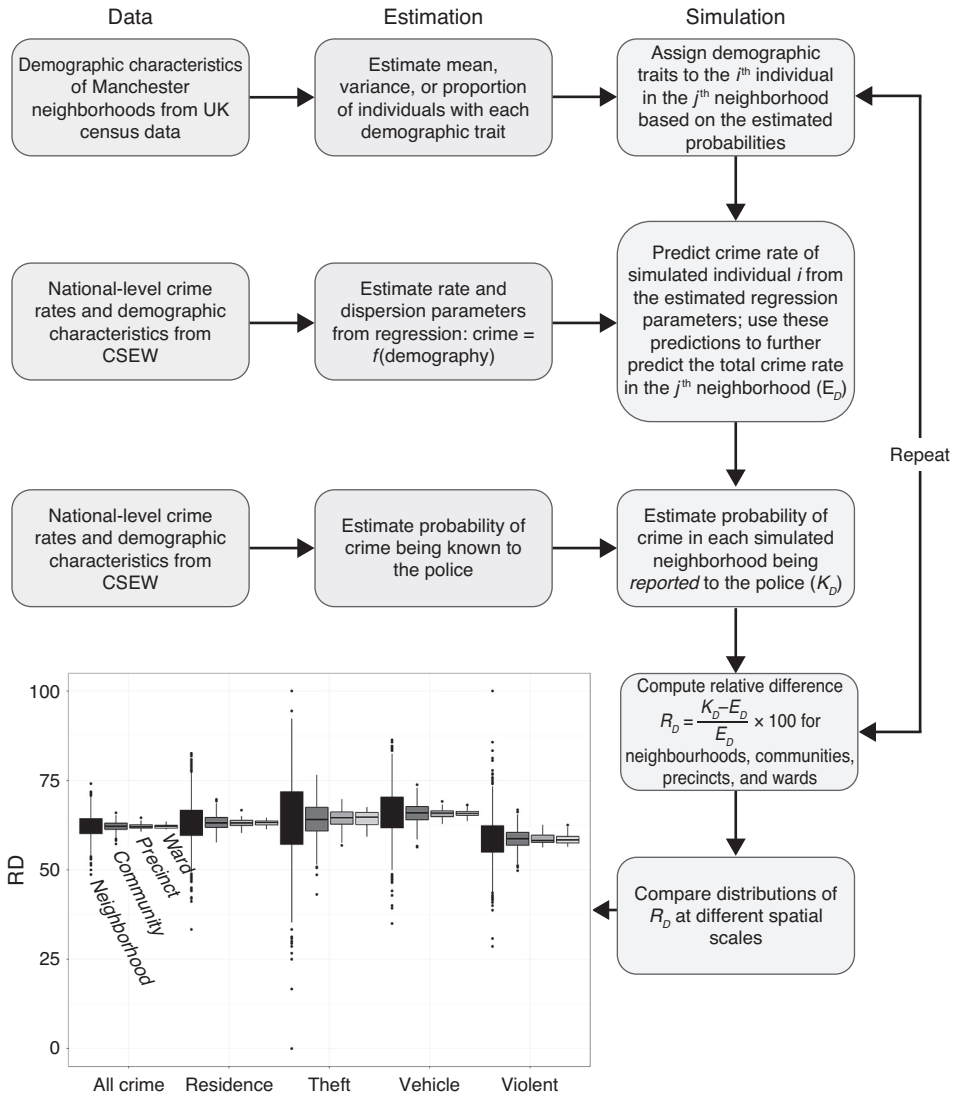


Figure 1.1 Workflow and results of the simulation exploring geographic bias in reported crime rates. The workflow, derived from the description of the simulation in Buil-Gil et al. (2022), includes obtaining data from government sources, estimation using negative binomial and logistic regressions, simulations, and visualisations. The inset figure is reproduced from Figure 1 of Buil-Gil et al. (2022) (CC-BY-4.0).

differences in reporting rates can lead to biases in the data that are used in criminological research and caution against blind acceptance of crime reduction strategies targeted at individual communities ("micro-geographies").

1.3.2 Handling data gaps

FLUXNET[5] is an international network of hundreds of stations that use the eddy-covariance technique to measure the flux of CO_2 and water above vegetation in a standardised way (Baldocchi 2003). The apparatus is very sensitive but requires air movement for measurements;

[5] https://fluxnet.org.

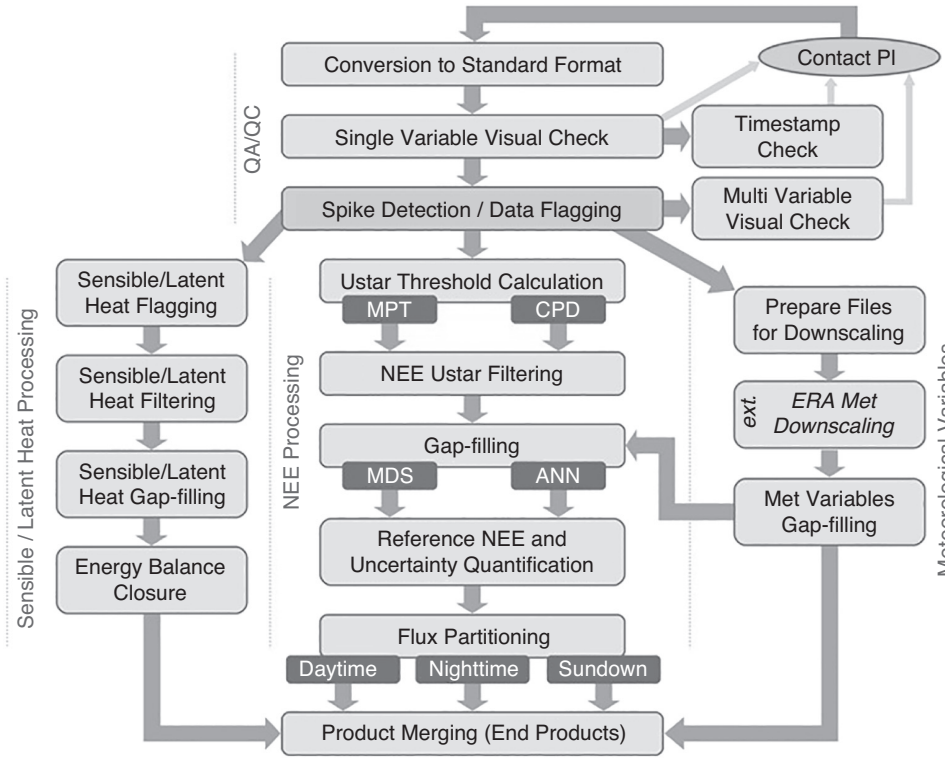


Figure 1.2 Workflow for gap filling of eddy covariance data in the FLUXNET network. Image in the public domain from FLUXNET (nd).

CO_2 flux cannot be measured at very low wind speeds, which happen regularly and especially at night and result in gaps in the data streams. Because these data gaps occur non-randomly, they need to be "filled" before the dataset can be analysed to avoid biased results.

Over the years, and after many, many simulations and comparisons of gap-filling methods, a sophisticated data pre-processing pipeline has been established (Figure 1.2). This is now the default workflow applied to all raw data entering the FLUXNET repository and data-processing chain. However, this workflow and the resulting estimates of the flux of carbon between forests and the atmosphere depends strongly on how the "U-star" ("Ustar" in Figure 1.2) threshold is determined. Simulations have shown that small changes in U-star could change the conclusion that a forest takes up more carbon from the atmosphere than it releases to its opposite—that the forest actually contributes planet-warming CO_2 into the atmosphere (Ellison et al. 2006).

1.3.3 Data augmentation in convolutional neural networks

Automatic classification of images is used widely—and often inappropriately or to ill effect—by many people, organisations, and government agencies. Classification, also called the "tyranny of the discontinuous mind" by Richard Dawkins (2011), can be improved by providing more and more data to the classifying algorithm. In lieu of genuine new data, computer scientists training convolutional neural networks (CNNs) on images use data augmentation: simulation of new data points based on old ones. Typical methods of data augmentation are flipping or rotating images, re-sizing or re-colouring them, adding noise to them, or increasing their grain. More recently, generative adversarial networks (GANs)

have introduced more sophisticated image mimicry but rely similarly on the variability in the training data. If each training image is altered in all combinations of such manipulations, sample size can be increased ten-fold with only marginal computational cost and at no cost to data collectors (e.g., Gopal et al. 2020; Nandhini Abirami et al. 2021; Li et al. 2023).

The second step of these simulations is to see whether data augmentation using GANs actually improves the classifier. Classifying the same images is uninformative; only classification of new, previously analysed, and validated images can tell us whether the method of data augmentation actually works. Indeed, it does improve classification accuracy for new images (e.g., Perez & Wang 2017; Mikołajczyk & Grochowski 2018), but not for the reasons initially suggested. Rather, the main reason seems to be that data augmentation using GANs adds noise, making the classifier avoid focusing on the specifics of an image, such as the stark contrast of the light reflection on the wet nose of a dog. As a result, data augmentation is now seen as a regularising or shrinkage approach rather than as a way to simulate new data (Steiner et al. 2022).

1.3.4 Null models in network analysis

When the value of an index depends in a complicated and non-linear way on its inputs, its distribution can be difficult to derive mathematically. One relevant case is the structure of bipartite (two-part) networks, which show the relationships between, for example, actors and the movies in which they appear, directors and the boards on which they serve, or pollinators and the flowers they visit. A description of what network structure we could expect based on how many movies and popular actors there are (i.e., in how many movies an actor has appeared) requires simulating the networks.

For example, when trying to identify modules ("cliques") of actors (or pollinators), we use a "null model": an algorithm to simulate a reference baseline. For either Boolean networks (e.g., an actor is in a movie or not) or networks with variable (but integer) number of interactions (e.g., a pollinator visits a flower N times), a modularity algorithm uses the sum of observations for each combination of actors and movies or pollinators and flowers, respectively, to simulate thousands of random networks against which the observed network is compared to determine if it is "structured" or random (e.g., Newman & Girvan 2004; Newman 2006; Barber 2007).

1.4 There are many good reasons to simulate data

Data simulation often is motivated by a specific problem with real data, such as violation of assumptions (e.g., normality, homoscedasticity), missing data, or unbalanced designs. However, there are many other benefits to simulating data besides demonstrating the validity or soundness of a particular method applied to a specific dataset. Perhaps most importantly, data simulation helps us to work through new problems that turn up while we are planning our research or analysing our data, and assess the validity of our proposed solutions when we are unsure of them.

While we were writing this book, ChatGPT, CoPilot, Llama, and many other large language models (LLMs) masquerading as artificial intelligence (AI) came online (cf. Searle 1982). Although many pundits (and not a few of our students and colleagues) have asserted that Big Data spelled the end of data analysis (e.g., Anderson 2008) and that AI and LLMs will make it unnecessary to learn how to write code (Hutson 2022), statisticians and programmers are still in high demand. Why? Even the best LLMs "hallucinate": although they may confidently provide an answer to a question (a "prompt"), the answer may be inaccurate or completely wrong (e.g., Lehr et al. 2024). Hallucinations are most common when the

data used to train the LLM had errors or did not include enough information to address a particular prompt, when the prompt does not have sufficient context, or if the prompt concerns methods, packages, or programming languages that are not commonly used. Thus, it is incumbent on us to be sceptical of responses given by any AI or LLM. Answers should always be checked for accuracy and code should be tested carefully to determine whether it works the way we expect (e.g., Cooper et al., 2024). Independent data simulations are one of many useful tools to check the responses of an LLM and to use it effectively (see also Lubiana et al. 2023; van Dis et al. 2023).

In this book, our main focus is on using simulations to better understand what statistical models and analyses do, and in doing so, to improve our statistical competence. However, simulations increasingly are being used as part of scientific publications. In recent years, scientific societies and journal publishers have begun to develop guidelines for reporting the results of simulations so that readers can trust and maximally benefit from them (Morris et al. 2019; DiRenzo et al. 2023).

1.5 Useful background knowledge to use this book most effectively

This book is for anybody doing statistical analysis who already is familiar with basic concepts such as probability and likelihood and has skills with standard hypothesis-testing and model-fitting methods such as t-tests, χ^2-statistics, and GLMs. Our previous books (Gotelli & Ellison 2012; Dormann 2020) can be used as refreshers. If you are starting from scratch, the `OpenIntro` project[6] has excellent open-source resources for teaching and learning basic statistics. Its latest book, the second edition of *Introduction to Modern Statistics* (Çetinkaya-Rundel & Hardin 2024), covers basic concepts in modelling, hypothesis testing, and statistical inference; includes detailed exercises and scripts in the `R` programming language, and introduces simulation and randomisation tests with its discussion of the foundations of statistical inference. Fieberg (2024) focuses more tightly on alternatives to ordinary linear regression for analysing observational data collected by ecologists. His text includes many good examples of using data simulation and, like us, takes a pragmatic approach to the utility of both frequentist and Bayesian approaches.

We also assume familiarity with at least one statistical programming language, such as `R`, `Python`, `Julia`, `Matlab`, or `Mathematica`, and the ability to construct scripts in it from descriptive algorithms and pseudocode. We provide code for all our examples written in the `R` programming language (R Core Team 2023) and presented in a standard form for all the algorithms and examples in the text (Code Box 1.1).[7]

All the code may be downloaded from the online code repository associated with this book (Dormann & Ellison 2024). With only a few exceptions, we eschew the `tidyverse` and pipes (Wickham et al. 2019); all data wrangling and functions are written in "standard" `R` and will run in either the classic `R` GUI (Venables et al. 2023) or `RStudio` (RStudio Team 2020). Similarly, with the exception of the graphics in the Appendix, all plots are generated using `ggplot` (Wickham 2016). Unless the primary goal of a particular code snippet is generating a specific plot, we do not include the plotting commands in the Code Boxes. However, the code to generate all the plots is included in the online code repository. Finally, the online repository also includes additional examples and associated code that we have developed since writing the book, and we encourage our readers and others interested in data simulation to contribute their own examples and code to the repository.

[6] https://openintro.org

[7] There are packages in `R` or `Python` that simulate data for you (e.g., `conjurer` and `simstudy` in `R`; see Appendix A for others), but which hide what actually is going on. So, we use them sparingly.

Code Box 1.1: How to read the code in the Code Boxes

```
# Code is presented in the same way throughout the text

# Lines or phrases beginning with a "hash tag" or "pound sign" (#) are comments
    that describe the subsequent or associated executable commands. If these
    lines are pasted into an R script, they will not run. For example, the line

# Load required libraries

# tells you that the subsequent lines will include the contributed packages
    (R "libraries") necessary for the executable code to run successfully.

# Lines indented with one or more spaces and lacking a leading character (such
    as a # or a |) are executable code. If these are pasted into an R script,
    they will run. For example, the line

 mean(x)

# will return the average of the variable "x".

# Lines beginning with a vertical bar (|) include the output (returned to the
    console) if executable code is executed. These lines should not be pasted
    into an R script, as they are neither comments nor executable code. For
    example,

| [1] 13.75

# would be the output from running the previous command (mean(x)).
```

1.6 Notational conventions

We have endeavoured to use consistent notation for mathematical expressions throughout the book, following Casella & Berger (2002). An upper-case letter (e.g., X) indicates a random variable, whereas its corresponding lower-case letter (e.g., x,) is a specific value from it. The set of all N values of $\{X_1, X_2, \ldots, X_N\}$ is denoted as X or $\{X\}$. We capitalise the sample size, N, throughout the main text, but in Code Boxes, it may appear in either lower case or upper case, depending on how it is defined in existing R functions.[8] We do not use vector notation (e.g., $\mathbf{X}$) for a matrix X; the difference between scalars and vectors should be clear from context.

Population-level ("true" and usually unknown) parameters are written with Greek letters (e.g., the population mean μ and its variance σ^2), whereas sample parameters normally are written with Roman letters (e.g., the sample mean, $\bar{x}$, and its variance, s^2). Modelled estimates are indicated with "hats" (e.g., the estimated population variance is written as $\hat{\sigma}^2$ and the estimated sample variance is written as $\hat{s}^2$).

[8] Note that commands, object names, etc. in R are case-sensitive: "Bird" and "bird" would be two different kinds of flying animals.

The names of libraries, functions, and variables used in R (or other programming languages) are set in `monospaced font`; function names always include parentheses, as in `t.test()`, to serve as a reminder that they need to have objects, parameters, or options provided to them (e.g., `t.test(x, y, type="paired")`).

1.7 Structure, organisation, and flow

The book is constructed in such a way that the two chapters in this Part I set the tone. After that, we proceed through topics following the same path that we would normally use to develop a research project: ask a question and develop hypotheses; come up with a sampling protocol or experimental design; collect, visualise, summarise, and analyse the data; fit and test models and (re)check their assumptions; and state our conclusions together with our degree of certainty about them. We show that using data simulation helps us do all these steps better, improves our science, and give us more confidence in the conclusions we draw from our research.

In the two chapters in Part II of this book, we show how to use simulations to plan surveys, observations and experiments from the time we come up with an idea until we actually start the study. For example, we can investigate how a proposed sample size will affect the probability of detecting weak effects. Simulation might also suggest that we abandon a proposed study if we cannot muster the necessary resources and logistics to reliably test our hypotheses. Thinking longer about a study *before* we do it is always beneficial. Data simulation encourages us to think first and act later, and forces us to be explicit about our assumptions and expectations.

The three chapters in Part III get us ready to analyse our hard-earned data. Before we analyse the data we have collected, we should check to see if our data satisfy key distributional assumptions (Chapter 5). We also can use simulations to check whether the method or model we plan to use to analyse our data makes sense or is it simply folklore: a rule-of-thumb that "everybody knows" or has "always been done" (Chapter 6). In a similar vein, individual models may work fine by themselves but exhibit strange behaviour when linked together in a workflow or analytical pipeline. Data simulations can help us discover that before we get so far into a tunnel that we cannot turn around and extricate ourselves (Chapter 7).

The two chapters in Part IV showcase ways to use data simulations to evaluate the fit of our models and what to do if they exhibit strange behaviour or yield results that are not statistically significant. If our models work as expected, all is well; simulations let us evaluate the fit of models we know to be reliable (Chapter 8). If they do not and our model yields really strange or unexpected results when we analyse our real data, we also can use simulations to probe the method or model more deeply, identify the origin of the weird results, and have firmer grounds from which to judge whether the results are strange but true or only an odd artefact of the specific sample or method (Chapter 8). Finally, if—despite all our careful planning, scrupulous data collection, and robust analysis—our results are not statistically significant, simulation shows us that we should not seek absolution in retrospective power analysis (Chapter 9). At the same time, we can use retrospective design analysis (Gelman & Carlin 2014) to test whether our statistically significant results are as meaningful as we think they are (Chapter 9).

The three chapters in Part V close the circle. We can use all the approaches presented in Parts II–IV to synthesise existing work using meta-analysis or federated analyses (Chapter 10), or to design and test new statistical methods or create new indices (Chapter 11). Using simulations offers an efficient and effective way to demonstrate the appropriateness of a new procedure or detect its flaws. Finally, Chapter 12 returns to where we started this

Table 1.1: Where to find detailed discussion of key statistical topics that recur throughout this book.

Topic	Chapter.Section
Assumptions:	
– Normally-distributed data	2.3; 5.1; 5.6; 8.4
– Homoscedasticity	2.2; 5.4; 8.4
– Independence	2.4; 4.2; 5.5–5.6; 6.5–6.6
– Stationarity	5.7; 12.2
Bayesian inference:	
– Bayes' Theorem	3.3; 6.4
– Prior vs. posterior distributions	8.1; 8.6
– MCMC and its relatives	7.7; 8.1; 11.6
Categorical vs. continuous predictors	6.3; 8.5
Collinearity	6.2; 7.5; 7.8
Confidence intervals	2.3; 3.4; 8.5; 9.3
Effect size	3.4; 4.1; 7.6; 9.1–9.3; 10.3–10.7
Imputation of missing data	6.1; 7.3–7.4; 7.8
Machine learning vs. classical statistics	6.2; 6.6; 7.5; 8.4–8.5; 11.4–11.5
Meta-analysis	3.4; 10.3–10.4
Model selection	6.1; 7.8; 10.2; 11.3
Power analysis	
– a priori (prospective)	3.4; 4 [entire chapter]
– post hoc (retrospective)	9 [entire chapter]; 10.3
Time-series analysis	2.4; 4.4; 6.4–6.5; 11.5

book: using simulations for planning large and very complex research programs that are addressing some of the most difficult questions scientists have posed to date.

We encourage you to read and work through the chapters in order. However, each chapter is mostly self-contained and you should be able to jump to, and work with, any part or chapter of the book in which you are most interested. Some key statistical topics and concepts pertain to more than one method or model, or recur in different contexts. Table 1.1 provides an overview of the most important of these topics and points to where we present them in detail. For more fine-grained direction, please consult the table of contents and the index.

1.8 Summary

- Simulated data are created using computational algorithms.
- Simulated data are similar, but not identical, to data we have already collected or that we plan to collect.
- Simulating data requires a "data-generating model" that normally implies a specific question, dataset, or type of statistical analysis.
- One of the most important uses of data simulation is to determine how much information about a large population can be gleaned from the smaller sample of it that has been collected.

- Other common uses of data simulation include testing assumptions, working through unanticipated or unexpected problems that came up while planning a research project or analysing the data, assessing the validity of proposed solutions, and determining whether the results of using large language models are real or hallucinatory.
- To use this book most effectively, you should already be familiar with basic statistical concepts such as probability and likelihood, and have skills with standard hypothesis-testing and model-fitting methods such as t-tests, χ^2-statistics, and GLMs.

References

Adams, D. (1979). *The Hitchhiker's Guide to the Galaxy*. London, UK: Pan/Macmillan.

Adams, D. C. & Collyer, M. L. (2022). Consilience of methods for phylogenetic analysis of variance. *Evolution*, 76(7), 1406–1419. doi: 10.1111/evo.14512.

Anderson, C. (2008). The end of theory: The data deluge makes the scientific method obsolete. *WIRED*, 16(7). https://www.wired.com/2008/06/pb-theory/.

Appel, K. & Haken, W. (1989). *Every Planar Map Is Four Colorable*. Providence, RI, USA: American Mathematical Society.

Baldocchi, D. D. (2003). Assessing the eddy covariance technique for evaluating carbon dioxide exchange rates of ecosystems: Past, present and future. *Global Change Biology*, 9(4), 479–492. doi: 10.1046/j.1365-2486.2003.00629.x.

Barber, M. (2007). Modularity and community detection in bipartite networks. *Physical Review E*, 76(6), 1–9. doi: 10.1103/PhysRevE.76.066102.

Bennett, C. M., Miller, M. B., & Wolford, G. L. (2009). Neural correlates of interspecies perspective taking in the post-mortem Atlantic Salmon: An argument for multiple comparisons correction. *NeuroImage*, 47, S125. doi: 10.1016/S1053-8119(09)71202-9.

Bostrom, N. (2003). Are you living in a computer simulation? *Philosophical Quarterly*, 53(211), 243–255.

Boulesteix, A.-L., Binder, H., Abrahamowicz, M., et al. (2018). On the necessity and design of studies comparing statistical methods. *Biometrical Journal*, 60(1), 216–218. doi: 10.1002/bimj.201700129.

Boulesteix, A.-L., Groenwold, R. H. H., Abrahamowicz, M., et al. (2020). Introduction to statistical simulations in health research. *British Medical Journal Open*, 10(12), e039921. doi: 10.1136/bmjopen-2020-039921.

Buil-Gil, D., Moretti, A., & Langton, S. H. (2022). The accuracy of crime statistics: Assessing the impact of police data bias on geographic crime analysis. *Journal of Experimental Criminology*, 18, 515–541. doi: 10.1007/s11292-021-09457-y.

Casella, G. & Berger, R. L. (2002). *Statistical Inference*. 2nd edition. Duxbury Press/Thomson Learning.

Çetinkaya-Rundel, M. & Hardin, J. (2024). *Introduction to Modern Statistics*. https://openintro.org: OpenIntro, 2nd edition.

Cooper, N., Clark, A. T., Lecompte, N., et al. (2024). Harnessing large language models for coding, teaching and inclusion to empower research in ecology and evolution. *Methods in Ecology and Evolution*, 15(10), 1757–1763. doi: 10.1111/2041-210X.14325.

Crawley, M. J. (2007). *The R Book*. Chichester, West Sussex, UK: John Wiley & Sons.

Crump, M. J. C., Navarro, D., & Suzuki, J. (2021). Answering questions with data: Introductory statistics for psychology students. DOI: 10.17605/OSF.IO/JZE52. Accessed 1 February 2024.

Dawkins, R. (2011). The tyranny of the discontinuous mind. *New Statesman*, 100(5084/5085), 54–57.

DiRenzo, G. V., Hanks, E., & Miller, D. A. W. (2023). A practical guide to understanding and validating complex models using data simulations. *Methods in Ecology and Evolution*, 14(1), 203–217. doi: 10.1111/2041-210X.14030.

Donovan, T. M., Brown, M., & Katz, J. E. (2021). R for fledglings. https://www.uvm.edu/~tdonovan/RforFledglings. Accessed 1 February 2024.

Dormann, C. & Ellison, A. M. (2024). Data ex machina. https://osf.io/3skv4/.

Dormann, C. F. (2020). *Environmental Data Analysis: An Introduction with Examples in R*. Heidelberg, Germany: Springer.

Eklund, A., Nichols, T. E., & Knutsson, H. (2016). Cluster failure: Why fMRI inferences for spatial extent have inflated false-positive rates. *Proceedings of the National Academy of Sciences, USA*, 113, 7900–7905. doi: 10.1073/pnas.1612033113.

Ellison, A. M., Osterweil, L. J., Hadley, J. L., et al. (2006). Analytic webs support the synthesis of ecological datasets. *Ecology*, 87(6), 1345–1358. doi: 10.1890/0012-9658(2006)87[1345:AWSTSO]2.0.CO;2.

Fieberg, J. R. (2024). *Statistics for Ecologists: A Frequentist and Bayesian Treatment of Modern Regression Models*. Minneapolis, MN, USA: University of Minnesota Libraries Publishing.

FLUXNET (n.d.). Data Processing 101: Pipeline and Procedures. https://fluxnet.org/data/aboutdata/data-processing-101-pipeline-and-procedures/. Accessed 16 January 2024.

Gelman, A. & Carlin, J. (2014). Beyond power calculations: Assessing type S (sign) and type M (magnitude) errors. *Perspectives on Psychological Science*, 9(6), 641–651. doi: 10.1177/1745691614551642.

Gopal, A., Gandhimaruthian, L., & Ali, J. (2020). Role of general adversarial networks in mammogram analysis: A review. *Current Medical Imaging*, 16(7), 863–877. doi: 10.2174/1573405614666191115102318.

Gotelli, N. J. & Ellison, A. M. (2012). *A Primer of Ecological Statistics*, 2nd edition. New York, NY, USA: Oxford University Press.

Hutson, M. (2022). AI learns to write computer code in 'stunning' advance. DOI: 10.1126/science.adg2088. Accessed 26 May 2024.

Jayasekera, J., Li, Y., Schechter, C. B., et al. (2018). Simulation modeling of cancer clinical trials: Application to omitting radiotherapy in low-risk breast cancer. *JNCI: Journal of the National Cancer Institute*, 110(12), 1360–1369. doi: 10.1093/jnci/djy059.

Jones, O., Maillardet, R., & Robinson, A. (2014). *Introduction to Scientific Programming and Simulation Using R*, 2nd edition. Boca Raton, FL, USA: Chapman & Hall/CRC.

Kriegeskorte, N., Simmons, W. K., Bellgowan, P. S. F., & Baker, C. I. (2009). Circular analysis in systems neuroscience: The dangers of double dipping. *Nature Neuroscience*, 12(55), 535–540. doi: 10.1038/nn.2303.

Lee, Y., Eck, J. E., O, S., & Martinez, N. N. (2017). How concentrated is crime at places? A systematic review from 1970 to 2015. *Crime Science*, 6, 6. doi: 10.1186/s40163-017-0069-x.

Lehr, S. A., Caliskan, A., Liyanage, S., & Banaji, M. R. (2024). ChatGPT as research scientist: probing GPT's capabilities as a research librarian, research ethicist, data generator, and data predictor. *Proceedings of the National Academy of Sciences USA*, 121(35), e2404328121.

Li, J., Yu, Z., Yu, L., et al. (2023). A comprehensive survey on SAR ATR in deep-learning era. *Remote Sensing*, 15(5), 1454. doi: 10.3390/rs15051454.

Liu, L. & Eck, J., Eds. (2008). *Artificial Crime Analysis Systems: Using Computer Simulations and Geographic Information Systems*. Hershey, PA, USA: Information Science Reference.

Lubiana, T., Lopes, R., Medeiros, P., et al. (2023). Ten quick tips for harnessing the power of ChatGPT in computational biology. *PLoS Computational Biology*, 19(8), 1–9. doi: 10.1371/journal.pcbi.1011319.

Marek, S., Tervo-Clemmens, B., Calabro, F. J., et al. (2022). Reproducible brain-wide association studies require thousands of individuals. *Nature*, 603(7902), 654–660. doi: 10.1038/s41586-022-04492-9.

Martínez-Santalla, S., Martín-Devasa, R., Gómez-Rodríguez, C., et al. (2022). Assessing the nonlinear decay of community similarity: Permutation and site-block resampling significance tests. *Journal of Biogeography*, 49(5), 968–978. doi: 10.1111/jbi.14351.

McElreath, R. (2020). *Statistical Rethinking: A Bayesian Course with Examples in R and STAN*, 2nd edition. Boca Raton, FL, USA: Chapman & Hall/CRC.

Meyer, K. R. & Schmidt, D. S., Eds. (2012). *Computer Aided Proofs in Analysis*. New York, NY, USA: Springer.

Mikołajczyk, A. & Grochowski, M. (2018). Data augmentation for improving deep learning in image classification problem. In *2018 International Interdisciplinary PhD Workshop (IIPhDW)* (pp. 117–122). Świnoujście, Poland.

Morris, T. P., White, I. R., & Crowther, M. J. (2019). Using simulation studies to evaluate statistical methods. *Statistics in Medicine*, 38(11), 2074–2102. doi: 10.1002/sim.8086.

Nandhini Abirami, R., Durai Raj Vincent, P. M., Srinivasan, K., et al. (2021). Deep CNN and deep GAN in computational visual perception-driven image analysis. *Complexity*, 2021, 5541134. doi: 10.1155/2021/5541134.

Newman, M. E. J. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences, USA*, 103(23), 8577–8582. doi: 10.1073/pnas.0601602103.

Newman, M. E. J. & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, 69(2), 1–15. doi: 10.1103/PhysRevE.69.026113.

Perez, L. & Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. *arXiv*, 1712.04621. doi: 10.48550/arXiv.1712.04621.

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Royston, P. & Sauerbrei, W. (2014). Interaction of treatment with a continuous variable: Simulation study of power for several methods of analysis. *Statistics in Medicine*, 33(27), 4695–4708. doi: 10.1002/sim.6308.

RStudio Team (2020). *RStudio: Integrated Development Environment for R*. RStudio, PBC., Boston, MA.

Searle, J. (1982). The myth of the computer. *The New York Review of Books*, 29 (April 29), 3–7. https://www.nybooks.com/articles/1982/04/29/the-myth-of-the-computer/.

Silver, N. (2012). *The Signal and the Noise: Why So Many Predictions Fail—But Some Don't*. New York, NY, USA: Penguin Press.

Steiner, A., Kolesnikov, A., Zhai, X., et al. (2022). How to train your ViT? Data, augmentation, and regularization in vision transformers. *arXiv*, 2106.10270. doi: 10.48550/arXiv.2106.10270.

van Dis, E. A. M., Bollen, J., van Rooij, R., et al. (2023). ChatGPT: Five priorities for research. *Nature*, 614(7947), 224–226. doi: 10.1038/d41586-023-00288-7.

Venables, W. N., Smith, D. M., & The R Core Team (2023). An Introduction to R. https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf. Accessed 11 February 2024.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. New York, NY, USA: Springer-Verlag, 2nd edition.

Wickham, H., Averick, M., Bryan, J., et al. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43), 1686. doi: 10.21105/joss.01686.

Zurell, D., Berger, U., Cabral, J. S., et al. (2010). The virtual ecologist approach: Simulating data and observers. *Oikos*, 119(4), 622–635. doi: 10.1111/j.1600-0706.2009.18284.x.

2

The basics of simulating data and the need for computational competence

The three examples in Chapter 1 illustrated the use of simulation in sociology, ecosystem science, and image processing using machine learning. These examples and all of the others we present in this book follow a general nine-step approach, but there are devilish details in any specific case (see also Morris et al. 2019; DiRenzo et al. 2023). We start this chapter with an outline of this approach. Then, to put us in the right frame of mind for the later chapters and because simulations are best learned by doing, we apply this approach to some well-known problems of increasing complexity. These examples highlight the key values of simulation and how much can be achieved with it.

This chapter also serves as in introduction to reading and writing R code (see Code Box 1.1 in Chapter 1 for the stylistic conventions we use in the Code Boxes) and will help you determine how comfortable you are with reading and writing it yourself. We summarise and describe how to use the most useful R commands for simulations in Appendix A. Venables et al. (2023) is the standard introductory reference for the R language; Donovan et al. (2021) provides a more user-friendly entrée to R and there are innumerable other online books, tutorials, videos, etc. that will be revealed with even a cursory search on the Internet.

What you will learn from this chapter

- The nine steps for developing a meaningful data simulation and a data generating model.
- The two most important ingredients for a successful data simulation.
- How to turn a description of a model into working code using examples from basic probability, ANOVA, confidence intervals, and time-series data.
- The differences between data simulations, randomisation methods used in non-parametric statistics, and resampling methods such as the bootstrap and the jackknife.

2.1 A road map for simulation in statistics

The nine steps to follow for developing a meaningful simulation are:

1. Formulate the specific question or model behind the simulation.
2. Specify exactly what the data that you want to simulate should look like (i.e., their distribution).
3. Identify the equations that you need to simulate the data (i.e., the process or process model).

4. Formulate the steps of the analysis and the final statistic(s) or parameter(s) of interest.
5. Identify the functions in your software that allow you to simulate the data.
6. Set a random number seed for reproducibility.
7. Implement all the above steps on a single dataset.
8. Repeat the analysis N times, storing the statistics or parameters of interest for each repetition.
9. Visualise and evaluate the statistics or parameters from the simulations to answer your question.

Steps 1 and 9 are the most important. First and foremost, it is crucial to have a clear question in mind and a model that identifies exactly what we want to find out about it. This is because simulations are specific and we typically cannot explore ill-defined questions, a large range of possible questions, poorly-specified models, or extensive parameter spaces with a single set of simulations. It is all too easy to start off with a rough idea, a vague question, or an unquantifiable "conceptual model" and then get bogged down in muddled thoughts and sprawling code.

For example, the question "Is an ANOVA robust to non-normality?" sounds specific enough, but it is not. How do we quantify robustness? Which statistic do we need to compute to determine whether an ANOVA is robust or not? Does it matter which non-normal distribution best describes the data or residuals? Before we even start thinking about how to simulate, we not only need to be clear about what we are simulating, but we also need to know what we will do with the results of the simulation. This is where Step 9 comes in. Once we've gone through the mechanics of simulating data and models (Steps 2–8), we actually count, visualise, compare, and evaluate the parameter values or test statistics that will allow us to answer the specific question we posed in Step 1. Step 9 must follow logically from Step 1.

2.2 Two simple examples

Our first two examples focus on Steps 1 and 9—framing and answering the question—and make apparent the power and value of even the simplest computational simulations.

2.2.1 Pólya's urn

In an urn there two balls: a red one and a blue one.[1] One ball is picked at random, its colour inspected, and then it is put back into the urn. Another ball of the same colour is then added to the urn; after N such random draws and additional balls have been added to the urn after each draw, the urn contains $2 + N$ balls. The Hungarian mathematician George (György) Pólya asked a simple and clear question: after (infinitely) many such draws and additions (i.e., as $N \to \infty$), what is the proportion of blue balls in the urn (Mahmoud 2008)?

The R code for this simulation is straightforward (Code Box 2.1).

In this first simulation, only 20% of the balls in the urn are blue after 100 moves. However, if we run the simulation again with a different seed, the proportion of blue balls is 87% (Code Box 2.2)!

We plot the two trajectories from Code Boxes 2.1 and 2.2 to visualise the results of our two simulations (Figure 2.1, left panel). We also could increase the number of moves to 1,000

[1] An urn is a vase with a rounded body on a pedestal, a narrow neck above the body, and a cover atop the neck. In probability and statistics, an urn is an imaginary container that can hold an infinitely large population from which a sample of N objects can be drawn using a defined probability distribution.

Code Box 2.1: Simulating a single Pólya urn

```
set.seed(123)
urn <- c("r", "b") # Start with one red, one blue
N <- 100 # Number of moves (random draws + additions)
for (i in 1:N) { urn <- c(urn, sample(urn, size=1)) }
table(urn[-c(1,2)])
```

```
| urn
|  b  r
| 20 80
```

Code Box 2.2: Simulating a single Pólya urn, *continued*

```
set.seed(321) # Re-run the simulation with a different seed
urn2 <- c("r", "b") # Start with one red, one blue
N <- 100 # Number of moves (random draws + additions)
for (i in 1:N) { urn2 <- c(urn2, sample(urn2, size=1)) }
table(urn2[-c(1,2)])
```

```
| urn2
|  b  r
| 87 13
```

```
# Compute proportions (and plot the results over time)
propBlue <- cumsum(ifelse(urn[-c(1:2)]=="b", 1, 0))/c(1:100) #urn from Code Box
    2.1
propBlue2 <- cumsum(ifelse(urn2[-c(1:2)]=="b", 1, 0))/c(1:100) #urn2 from this
    Code Box
runs <- data.frame(blue=c(propBlue, propBlue2), trials=rep(1:100, 2), run=rep
    (1:2, each=100))
```

or more,[2] but the results remain very similar. As it turns out, running many such simulations (Code Box 2.3) yields any proportion of blues between 0 and 1.[3]

The distribution of the proportion of blue balls across all the simulations is nearly uniform (Figure 2.1, right panel), but the values of 0 and 1 become increasingly improbable as $N \to \infty$. More specifically, the distribution of the proportion of blue balls in an infinite number of Pólya's urns is a beta-binomial distribution with parameters N, α, and β, where N is the number of draws, and α and β are the initial number of balls of each colour in the urn.

The Pólya urn model is interesting in that the trajectories converge to a constant value, but one that is purely random and dependent on the initial 10–20 steps of the random

[2] Try this yourself by replacing `N <- 100` with `N <- 1000` (or with whatever number of replicates you'd like) in Code Box 2.2.

[3] In some instances of running Code Box 2.3, a warning message will be returned that there were some runs (e.g., 1%) without *any* blue (or red) ball being added to the urn.

Code Box 2.3: Simulating many Pólya urns

```
blueProportion <- 1
for (j in 1:5000){ # Simulate 5000 urns
  set.seed(2+j)
  urn <- c("r", "b") # Start each run with one red, one blue
  for (i in 1:N) { urn <- c(urn, sample(urn, size=1)) }
  blueProportion[j] <- table(urn[-c(1:2)])["b"]/ N
}

sum(is.na(blueProportion)) / 5000 # 1% without blue!

| [1] 0.01

sum(blueProportion == 1, na.rm=T) / 5000 # 1% only blue!

| [1] 0.0106
```

sequence of draws. The model has been referred to as a "rich get richer" sequence, as the colour initially gaining an upper hand will stabilise at a higher value than the other. Several variations of Pólya's urn model exist, including one that leads to the "Chinese restaurant" distribution and another that simulates genetic mutations (Hoppe 1984).[4]

This kind of process is not intuitive and this simple example illustrates that simulations can be very useful for exploring situations of which we have little understanding or for which we have no general analytical solutions.

2.2.2 Heterogeneity of variances in ANOVA

We return here to our initially ill-framed question about the robustness of ANOVA. We home in on a common—and more specific—question about ANOVA: does heterogeneity of variances among the different groups being compared lead to the spurious result of detecting differences between groups that actually do not exist? To answer this question, we could plot a frequency distribution of the *F*-values computed by the ANOVA and see if heterogeneity of variances changes our conclusions.[5]

The *F*-statistic computed in an ANOVA represents the ratio of explained to unexplained variance. The *F*-value is evaluated against its expected *F*-distribution to compute a *p*-value for the probability of observing the data, given the computed and more extreme values of *F* (Johnson et al. 1995). Variances that are unequal among the different groups (i.e., heteroscedastic variances) affect the computation from the data of the *F*-value, but not the reference (expected) *F*-distribution. So, if heteroscedasticity were to lead to larger *F*-values, and hence lower *p*-values, we would run the risk of detecting differences that do not exist (i.e.,

[4] The Chinese restaurant distribution is the probability distribution on the number of tables in the Chinese restaurant process (Aldous 1985). This process is the result of seating n customers at an infinite number of circular tables. As in Pólya's urn, after the first customer sits at the first table, the next one sits at either the first table or another one. Subsequent customers choose to sit either at an occupied table with a probability proportional to the number of customers already at the mth table, or at an unoccupied table. After n customers have been seated, they occupy $m \leq n$ tables.

[5] There are many other possible questions about robustness of ANOVA. We address some of these in more detail in Chapter 5.

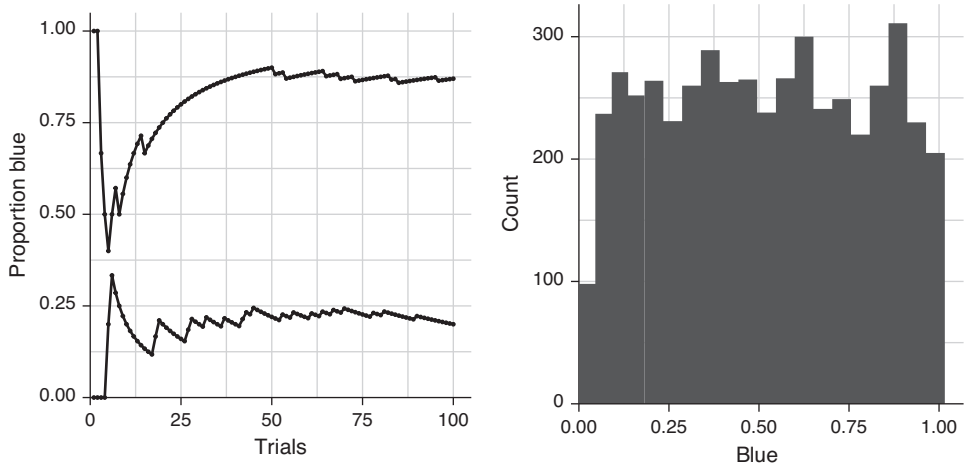


Figure 2.1 *Left*: Two Pólya urn sample runs, both converging, but to different values. *Right*: An histogram of the final proportions of blue balls from 5,000 such runs.

false negatives) and committing a type-II statistical error (for a more detailed description of type-I and type-II statistical errors, see Table 3.1 in Chapter 3).

To illustrate this problem, we compared 1,000 simulated datasets, each with three groups with equal variances ("homoscedastic" data), with 1,000 datasets with higher variances in one of the three groups ("heteroscedastic" data). For each dataset, we computed the resulting *F*-statistic (Code Box 2.4).

The results of our simulations are shown in Figure 2.2. The distribution of the *F*-statistics from homoscedastic (dark grey distribution) and heteroscedastic (light grey distribution) data differed substantially. The take-home message is that when assumptions of the model or test are violated, the resulting distribution(s) of our parameter(s) or test statistic(s) of interest will be shifted, squashed, or trimmed relative to the distribution expected when our data meet the required assumptions of the model or the test. The difference from expectation will depend on the specific question, dataset, and violations of assumptions. In this case, *F*-values were more conservative for the heteroscedastic simulations, meaning that a truly existing difference would be more likely to be overlooked (light-grey distribution in Figure 2.2). In short: heteroscedasticity reduces statistical power. The actual shape and location of the statistic for violated assumptions has to be found anew using different simulations for each question.

2.3 More complex examples

The previous two examples focused only on Steps 1 and 9. To really understand data simulation, though, we need to dive into the finer details. We go through all nine steps in our next two examples. In the remaining two examples in this section, and in the rest of the book, we will skip over the steps that do not require detailed documentation.

2.3.1 What is a 95% confidence interval?

A classic example of the value of simulation is to illustrate the true meaning of a 95% confidence interval (CI). What a 95% CI describes is annoyingly difficult to phrase accurately, and it is frequently misinterpreted (Ellison & Dennis 2010). However, a simulation clearly conveys the actual meaning of a 95% CI (Gelman & Nolan 2002).

Code Box 2.4: Simulation of homoscedastic and heteroscedastic data for ANOVA

```
set.seed(123)
# Homoscedastic case (all variances equal)
simu1 <- replicate(1000, { # 1000 simulations
 group1 <- rnorm(74, mean=0.2, sd=0.1)
 group2 <- rnorm(34, mean=0.2, sd=0.1)
 group3 <- rnorm(62, mean=0.3, sd=0.1)
 dataset <- data.frame("response"=c(group1, group2, group3), "groups"=c(rep
     ("group1", 74), rep("group2", 34), rep("group3", 62)))
 summary(aov(response ~ groups, data=dataset))[[1]]$"F value"[1] #Stores the
     F-values
})

# Heteroscedastic case (variance of group 3 greater than the variances of the
    other two groups)
simu2 <- replicate(1000, { # 1000 simulations
 group1 <- rnorm(74, mean=0.2, sd=0.1)
 group2 <- rnorm(34, mean=0.2, sd=0.1)
 group3 <- rnorm(62, mean=0.5, sd=0.4)
 dataset <- data.frame("response"=c(group1, group2, group3), "category"=c(rep
     ("group1", 74), rep("group2", 34), rep("group3", 62)))
 summary(aov(response ~ category, data=dataset))[[1]]$"F value"[1] #Stores the
     F-values
 })
```

A 95% CI gives the range of values in which the true estimate (usually the mean, but any statistical estimator has a confidence interval) lies in 95% of the cases.[6] For any given sample, we do not know whether the true mean is in the interval or not, but across many samples it should be in the interval 95% of the time.[7] In other words, when we compute 95% CIs for each of 100 samples with identical means and variances, in 5 of cases the 95% CIs will not overlap with the true mean. Is that the right interpretation? We answer this question using simulation, and detail all nine steps of the process.

1. **Question**: Will 5 samples out of 100 repetitions have CIs that do not include the true mean μ?
2. **Data to simulate**: This is a single-sample simulation, so we simulate a vector of random values drawn from a Gaussian (normal) distribution.

[6] The confidence interval was introduced by Jerzy Neyman in 1935 (although most refer to his 1937 article, as the earlier one was only a reply to Fisher's comment on a slightly different topic in 1934). It "resolved" an issue that had bothered non-Bayesian statisticians (although at the time, that term did not yet exist: all statisticians were de facto Bayesians until around 1918): "The calculation . . . is not convenient, as both multipliers . . . depend on the unknown probability function *a priori* of θ" (Neyman 1935, p. 113). Neyman and like-minded statisticians were uncomfortable with what we now call the prior (the Bayesian prior probability distribution); he introduced his computation of confidence intervals to get around priors—at great interpretational expense for scientists and statisticians ever since.

[7] The situation is in fact slightly more complex, as the true mean is typically not known and hence the 95% CI only includes an unbiased estimate of the true mean of all the replicates in 95% of an infinite number of trials or for a large number of trials of asymptotically unbiased models.

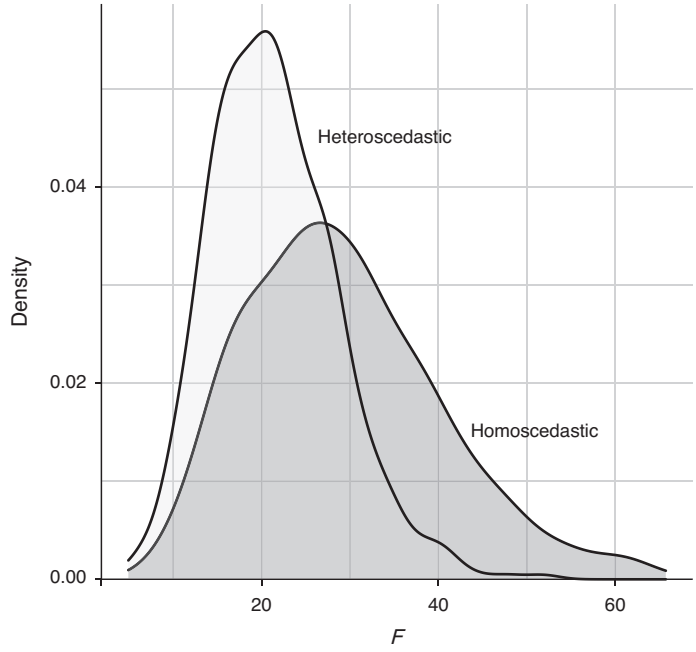


Figure 2.2 The distribution of 1,000 *F*-values from each of two different ANOVAs of simulated data (1,000 replicates each) with three categorical levels. The simulation contrasted a situation in which the three different groups had unequal variances (heteroscedastic; light grey)—in this case, one level had higher variance than the other two—with the situation in which all three groups had the same variance (homoscedastic; dark grey).

3. **Simulation equation**: Because our random values $Y = \{Y_1, Y_2, \ldots, Y_N\}$ are drawn from a Gaussian distribution with mean $= \mu$ and standard deviation $= \sigma$, we must first define μ and σ. For this example, we set the mean = 10 and the standard deviation = 2. Our simulation equation is then $y \sim \mathcal{N}(\mu = 10,\ \sigma = 2)$.
4. **Analytical steps**:
   (a) The 95% CI $= [\overline{Y} - c \cdot$ standard error of the mean, $\overline{Y} + c \cdot$ standard error of the mean$]$. Thus, we need to compute the 0.025 (or 0.975) quantile of the *t*-distribution with $N - 1 = 9$ degrees of freedom.[8]
   (b) Simulate a sample with a random draw of 10 observations from the normal distribution specified in step 3.
   (c) Compute the sample mean and the sample standard deviation of that sample, and from those, the standard error of the mean $= \frac{s}{\sqrt{N}}$.
   (d) Store estimates of lower and upper bounds of each 95% CI.[9]

[8] Estimates of both μ and σ (i.e., $\overline{Y}$ and s) have to be estimated from the data. For a sample drawn from a normal distribution, the 95% CI is computed based on the *t*-distribution. For a given sample size N, we can compute the value c at which $\Pr(X \leq c) = 0.025$ (or $\Pr(X \geq c) = 0.975$). The *t*-distribution is symmetric, so we can look at either the lower (0.025%) or the upper (97.5%) quantile).

Then, the 95% CI around the population mean μ is $\Pr\left(\overline{Y} - \frac{cs}{\sqrt{N}} \leq \mu \leq \overline{Y} + \frac{cs}{\sqrt{N}}\right) = 0.95$, with $\overline{Y}$ and s being the sample mean and the standard deviation, respectively. For different desired CIs, use different quantiles.

Finally, for quick computation of 95% CIs, scientists often use $c = 1.96$ in the previous equation. But this value is correct only for "large" sample sizes ($N > 100$), not for smaller ones.

[9] At this point we also may want to compare our computation with a standard function that computes the CI, just to verify our coding.

Code Box 2.5: Simulation of a single 95% confidence interval

```
# Set the "true" parameter values
 mean.true <- 10
 sd <- 2

# Generate a single set of observations
 obs <- rnorm(10, mean=mean.true, sd=sd)

# Find the appropriate quantile of the t-distribution for computing the 95% CI
 fact <- qt(0.975, df=9)
 fact

| [1] 2.262157

# Compute the estimated mean and the lower and upper bounds of its 95%
    confidence interval
 c(mean(obs) - fact*sd(obs)/sqrt(10), mean(obs), mean(obs) + fact*sd(obs)/sqrt
     (10))

| [1]  8.004720  9.410052 10.815384

# Compare our simulated results with those computed by R's built-in t-test
    function
 t.test(obs)$conf.int # Confirms the previous computation

| [1]  8.00472 10.81538
| attr(,"conf.level")
| [1] 0.95
```

5. **Functions**: In R code, we use the function `rnorm()` to draw random numbers, and functions `mean()` and `sd()` to compute the sample mean and its standard error. Function `qt()` gives us the value for c.
6. **Write the code** (see Code Box 2.5).
7. **Set the random number seed** (in the example below, we use '12345' but any integer will do).
8. **Repeat**: 100 repetitions suffice for this illustration, but normally we would use 1,000 or more (see Code Box 2.6).
9. **Compute the parameters or summary statistics and visualise the results**: Our target statistic is a Boolean (true/false) indicator of whether the 95% CI of an individual simulation includes the true mean. Count how often the lower bound > 10 and how often the upper bound < 10. Sum both counts. This is our final value of interest. For this example, a good visualisation would be a plot of the mean and the CI of each sample, along with a reference line indicating the true mean, $\mu = 10$ (Figure 2.3).

Note that there are many things we did not simulate: the effect of sample size, changing the true standard deviation, or using the rule-of-thumb value for c (1.96 or any other value "close to" 1.96; see footnote 8) instead of the exact value of the 0.975 quantile of the

Code Box 2.6: Repeated simulation of 95% confidence intervals

```
set.seed(12345)
simu <- replicate(100, { # 100 Simulations
  obs <- rnorm(10, mean=mean.true, sd=sd)
  c("lower"=mean(obs) - fact*sd(obs)/sqrt(10), "mean"=mean(obs), "upper"=mean(
      obs) + fact*sd(obs)/sqrt(10))
})
simu <- cbind.data.frame("N"=1:100, t(simu))
length(which(simu$lower > 10 | simu$upper < 10))
```

```
[1]  4
```

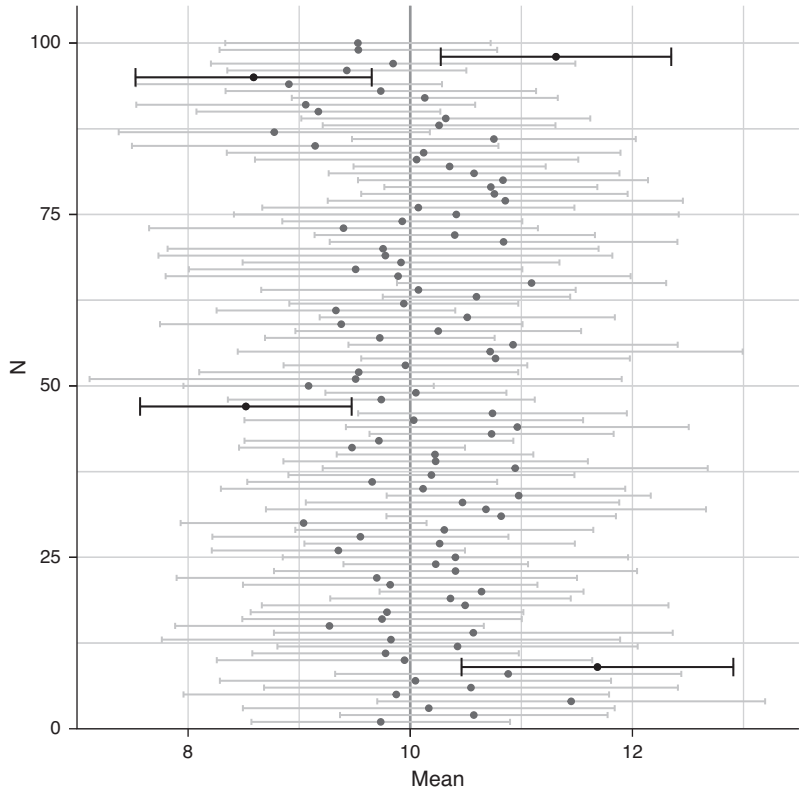


Figure 2.3 Sample confidence intervals of 100 samples of size 10 from a normal distribution with $\mu = 10$ and $\sigma = 2$. The four points and lines in black are the four samples generated by the code in Code Boxes 2.5 and 2.6 for which the 95% confidence interval does not overlap with the true mean (vertical line).

t-distribution, or …. Finally, a more robust simulation would have many more than 100 replicates (at least 1,000 or 10,000). But the idea should be clear nonetheless.

Four of the 100 samples have confidence intervals (black points and confidence intervals in Figure 2.3) that *do not* include the true mean, only one fewer than the expected five samples. If we had run more replicate simulations, we would expect that the number of

non-overlapping confidence intervals would more closely approach 5% of the simulations. Use the code in Code Boxes 2.5 and 2.6 to try it yourself!

This simple example shows two of the most important ingredients to a simulation: a clear objective and a high level of replication.

2.3.2 Robustness of the Kolmorogov-Smirnov test to small sample sizes

The traditional Kolmorogov-Smirnov (K-S) test compares the distribution of data we have observed to data that would have been collected given an assumed continuous distribution. Formally, for a set of N observations $X = \{X_1, X_2, \ldots, X_N\}$, the K-S test tests whether X was drawn from a specific probability distribution $G(x)$.

How does the K-S test work? For each observed datum X_i, with X ordered from smallest to largest, we first compute the proportion of data less than or equal to x: $F_N(x) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{I}(X_i \leq x)$, where $\mathrm{I}(X_i \leq x)$ is the indicator function, which is equal to 1 if it is true $(X_i \leq x)$ and 0 if it is false $(X_i > x)$. Second, we compare the empirical (observed) distribution $F_N(x)$ with a reference cumulative distribution $G(x)$. Finally, the largest absolute difference between $F_N(x)$ and $G(x)$ is the K-S statistic: $D_N = \sup_x |F_N(x) - G(x)|$. The hypothesis that the observed data were drawn from the assumed continuous distribution is rejected at critical level α if $\sqrt{N}D_N > K_\alpha$, where K is Kolmogorov-distributed and $\Pr(K \leq K_\alpha) = 1 - \alpha$.[10]

A key assumption of the K-S test is that the parameters of the reference distribution are known, but this is rarely the case. In practice, as in our simulation (Code Box 2.7), the parameters of the reference distribution $G(x)$ must be estimated from the data. For example, if the reference distribution is the normal distribution, then $G(x) = \mathcal{N}(x \mid \mu = \bar{x}, \sigma = s)$, where $\bar{x}$ and s are, respectively, the sample mean and standard deviation of our sample x.

However, when the parameters of the reference distribution are unknown and must be estimated from the data, the critical values of the Kolmogorov distribution generally are incorrect. It is possible to adjust the computations to yield the correct statistic, but for a non-statistician, implementing these corrections requires swimming in deeper mathematical waters (which are referred to as "empirical process" *in litt.*, e.g., van der Vaart & Wellner 1996; Shorack & Wellner 2009). Luckily, for the special but common case of the normal distribution, an analytical approximation exists: the Lilliefors test (Dallal & Wilkinson 1986). However, when we have only few data points from a normal distribution, they often look anything but normal. We ask: Do both the K-S test and the Lilliefors test work correctly for small datasets?

To address this question, we simulated a small dataset from a normal distribution and ran a K-S test on it. We repeated this procedure 1,000 times and recorded how often the K-S test identified the data as non-normal for $P(\alpha \leq 0.05)$. An acceptable type-I error rate would be $\approx 5\%$ of the cases. We then did the same for the Lilliefors test. For clarity, here are the nine steps:

1. **Question**: Do the K-S and Lilliefors tests have the correct error rate (5%) when the parameters of the reference distribution are estimated from a small sample?
2. **Data to simulate**: 10 draws from a standard normal distribution (which is sometimes abbreviated as φ; its cumulative distribution function is then abbreviated as Φ).
3. **Simulation equation**: $X \sim \phi(x) = \mathcal{N}(\mu = 0, \sigma = 1)$.

[10] The Kolmogorov distribution is a parameter-free distribution, i.e., its value is identical for all sample sizes and the expression $\sqrt{N}D_N$ accounts for the sample size.

Code Box 2.7: Simulating and comparing the K-S and Lilliefors tests

```
#Load required libraries
 library(nortest) # For the lillie.test function

# Run each test once to ensure that the code works. Note that if we do not
    specify the mean and sd, the defaults are mean = 0 and sd = 1. Of course we
     could use any other value for the mean and sd.
 A <- rnorm(10)
 ks.test(A, "pnorm", mean=mean(A), sd=sd(A))

|   One-sample Kolmogorov-Smirnov test
|
| data:  A
| D = 0.20825, p-value = 0.7056
| alternative hypothesis: two-sided

 lillie.test(A)

|   Lilliefors (Kolmogorov-Smirnov) normality test
|
| data:  A
| D = 0.20825, p-value = 0.2521

# Simulate 1000 times:
 set.seed(234)
 res <- replicate(1000, {
     A <- rnorm(10)
     c(ks.test(A, "pnorm", mean=mean(A), sd=sd(A))$p.value, lillie.test(A)$p.
         value)
 })
 rowSums(res < 0.05)

| [1]  0 46
```

4. **Steps**:
   (a) Draw x as a specific realisation of $X = \{X_1, \ldots, X_{10}\}$.
   (b) Compute the p-value from the K-S statistic D_N relative to that of a normal distribution with parameters $\mu = \bar{x}$ and $\sigma = s_x$.
   (c) Using the same parameter estimates, compute the p-value using the Lilliefors test.
   (d) Store both p-values.
5. **Functions**: In R, we need the functions `rnorm()` to draw random numbers; `mean()` and `sd()` to compute means and standard deviations; and `ks.test()` and `nortest::lillie.test()` to compute the two types of test statistics.[11]

[11] The double colon preceding the function `lilli.test()` means that the function is included in the library preceding the double colon (in this case, the `nortest` library). Hence, the library needs to be loaded at the beginning of the code using the `library()` command as in Code Box 2.7.

6. **Set random number seed** (in this example, we use '234', but any integer will do).
7. **Repeat** 1,000 times.
8. **Process the statistics**: Count the number of p-values < 0.05 for each test. For 1,000 replicates we would expect that $5\% = 0.05 \cdot 1000 = 50$ of the p-values would be < 0.05.

Note that there are many things we did not investigate with this simulation, including whether the K-S test is reliable for distributions other than the normal; whether non-normal data are wrongly identified as normal; or what is the minimum sample size at which the K-S test becomes unreliable.

The result is that *none* of the 1,000 simulations are classified as being significantly different from normality according to the K-S test. In contrast, 46 of the 1,000 Lilliefors simulations—just shy of the expected 5%—were classified as being significantly different from normality. Our simulations confirmed that the K-S test has an unacceptable error rate, but even for a very small sample size, the Lilliefors has an acceptable error rate.[12]

You can show that the K-S test will continue to fail as the sample size gets larger by changing the sample size.[13] This continued failure is a consequence of using the data to estimate the mean and standard deviation for the reference distribution, which the K-S test assumes are known independently. In contrast, if you set these values to their true values of 0 and 1, respectively, the K-S test will have the correct, nominal 5% error rate.[14]

2.3.3 Visualising the variability of fitted model

When we fit a model, we often report only the parameter estimates and some measure of their variability, or we simply plot the fitted line or curve and its confidence interval. Such tables or plots do identify the uncertainty in the parameter estimates but do not convey as well the variability implicit in the uncertainty itself. Simulation can help us better visualise this latter variability. For example, rather than plotting only the fitted curve and its shaded confidence interval, we can plot some or all of the curves consistent with our model to more clearly and intuitively illustrate how different they might be (for a useful example, see Aguirre-Gutiérrez et al. 2022). The better the model, the more similar the curves.

There are two common ways to compute parameter uncertainty in a regression context. The first is analytical, and is derived from the curvature of the likelihood at its maximum (estimated using the so-called Hessian matrix, the matrix of second derivatives at the maximum-likelihood estimate). It is returned by all statistical software as the "standard error."[15]

[12] This may appear a bit strange: Isn't it great that the K-S test correctly identified all simulated data as indistinguishable from a normal? Sure; but if the K-S test states an error rate of 5% then it should have an error rate of 5%, not 0%. Also, since type-I and type-II errors are inversely related (see Chapter 4), over-estimated type-II error rates typically result in under-estimated type-I error rates. In other words: it is great that the K-S test gets all the true normal variates right but it is likely that it will, therefore, also get a few non-normal variates wrong. Of course, this is also something that could be simulated.

[13] You can do this by increasing the value passed to `rnorm()` and re-running the simulation in Code Box 2.7.

[14] You can do this by changing the `ks.test()` command in Code Box 2.7 to `ks.test(A, "pnorm", mean=0, sd=1)`. Note that this change also assumes that the reference distribution you used in the `rnorm` command has `mean=0` and `sd=1`. If you pass different values to `rnorm()`, you must pass those same values to `ks.test()`.

[15] The standard error of a parameter is an estimate of its true standard deviation. Although we usually refer to the standard error of the mean ($= \frac{\sigma}{\sqrt{N}}$) as simply the "standard error," it is better to explicitly name the parameter

The second way to compute parameter uncertainty uses bootstrapping (Efron 1982). Bootstrapping begins with a set of samples, each repeatedly drawn at random, with replacement, from the observed data. The model is re-fit to each random sample, yielding a new set of estimated parameters of the model. The process is repeated many times, and the standard deviations of the distributions of the models' parameters provide estimates of their standard errors. For large datasets, the analytical and bootstrap estimates of the standard errors converge, but for small datasets, the analytical approach may underestimate the standard errors.

Code Box 2.8 shows how we simulated a two-variable dataset, fit a logistic regression model to the data, stored the parameters computed analytically and with a bootstrap, and plotted the results.[16]

When we used the analytical method of computing the standard errors of the parameters, we simulated random draws from the fitted parameter distribution (i.e., Monte Carlo simulation).[17] Because the parameters may be correlated, we drew our samples from the multivariate normal distribution of the parameter estimates. For each parameter combination, we then plotted the resulting curve (Figure 2.4, left).

When we used bootstrapping to compute the standard errors of the parameters, we could directly use the bootstrap estimates for plotting the lines (Figure 2.4, right). In this case, the Monte Carlo draws from the fitted GLM yielded very similar results as the bootstrapping, indicating that our 123 points were enough for an asymptotic approximation. However, any one of the curves in either of the two plots could be the true underlying data-generating function. How certain are you now of the accuracy of the fitted model itself?

2.3.4 Does it matter if the *t*-test is significant but the residuals are skewed?

The Student's *t*-test (Gosset, a.k.a. "Student," 1908) may be one of the most commonly used statistical tests. Even though the *t*-test assumes (conditional) normality, it is commonly asserted that using a *t*-test to test for differences between two groups is robust to deviations from the assumption of normality. However, statistically significant differences revealed by a *t*-test may be a result of non-normal data as opposed to true differences between the groups.[18] We can use simulation to investigate the effect of skewness on falsely detecting differences between two groups (i.e., a type-I statistical error) by evaluating the nominal coverage of the *t*-test when the data are drawn from a skewed distribution.

First, we confirmed for ourselves that the *t*-test works fine for Gaussian data (Code Box 2.9). When we simulated 1,000 datasets with the same mean and standard deviation

for which the standard error is estimated. In a simple linear regression model, for example, we can estimate the standard error for the estimated value of the intercept and the standard error for the estimated value of the slope.

[16] In this example, we explicitly show the code used to plot the results because the point of these simulations is the visualisation itself.

[17] Why is it called Monte Carlo simulation? The physicist Nicholas Metropolis wrote that he came up with the term in 1947 as a name for a computationally intensive statistical method suggested by the mathematicians and physicists John von Neumann and Stanislaw Ulam to solve the problem of how neutrons would diffuse in fissionable material used in nuclear weapons. The method had become possible after ENIAC—the first electronic computer—had been successfully built. Because Ulam had often spoken about his uncle Michał who would borrow money from relatives because he just had to go and gamble in Monte Carlo (thus directly confronting the laws of probability), Metropolis used "Monte Carlo" to refer to the then-classified method (Metropolis 1987). The first unclassified publication about the Monte Carlo method appeared two years after the name was coined (Metropolis & Ulam 1949).

[18] One could always use the Wilcoxon signed-rank test for non-normal data, but that is not the point here.

Code Box 2.8: Using analytical and bootstrap methods to visualise uncertainty in a GLM

```
# Load required libraries
 library(mvtnorm) # For multivariate random normal variables
 library(purrr)   # For the pmap and pipe functions
 library(dplyr)   # For tibbles
 library(ggplot2) # For plotting

# Set seeds, constants, and other variables
 set.seed(345)
 X <- runif(123, 0, 1)
 Y <- rbinom(123, size=1, prob=plogis(1 - 2*X))

# Analytical method:
 fm <- glm(Y ~ X, family=binomial)
 vcov(fm)

|             (Intercept)          X
| (Intercept)   0.1372207 -0.2075565
| X            -0.2075565  0.4232422

# Simulations
 sims <- rmvnorm(n=100, mean=coef(fm), sigma=vcov(fm))
 fun <- function(x, a, b) plogis(a + b*x)
 coefs <- as.data.frame(sims)
 colnames(coefs) <- c("intercept", "slope")
 curves <- coefs %>%
   pmap(function(...){
     dots <- tibble(...)
     stat_function(data = dots, aes(0),
           fun = function(x) fun(x, dots$intercept, dots$slope),
                   xlim = c(0, 1),
                   color=rgb(.2, .2, .2, .3))
   })

# Bootstrap method:
# Single bootstrap run:
 coef(glm(Y ~ X, family=binomial, data=dats[sample(nrow(dats), replace=T),]))

| (Intercept)           X
|   0.4505363  -1.7076292

# 100 replicates:
 boots <- replicate(100, coef(glm(Y ~ X, family=binomial, data=dats[sample(nrow(
     dats), replace=T),])))
 coefs.boot <- as.data.frame(t(boots))
 colnames(coefs.boot) <- c("intercept", "slope")
 curves <- coefs.boot %>%
```

```
  pmap(function(...){
    dots <- tibble(...)
    stat_function(data = dots, aes(0),
                  fun = function(x) fun(x, dots$intercept, dots$slope),
                  xlim = c(0, 1),
                  color=rgb(.2, .2, .2, .3))
  })
```

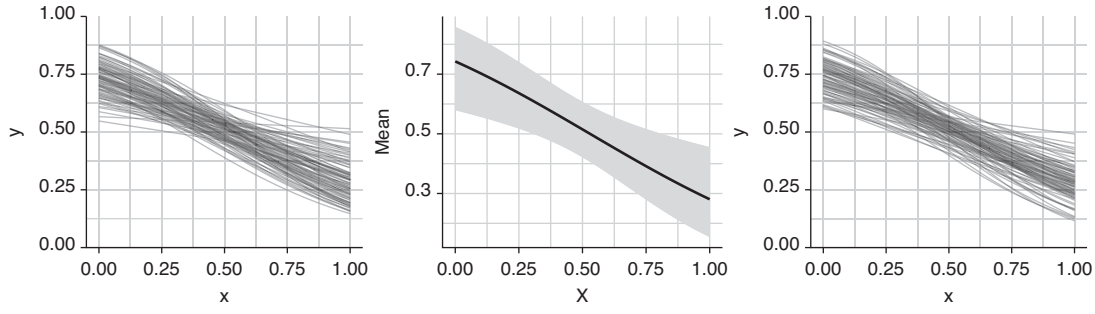


Figure 2.4 One hundred simulations of curves with randomly drawn parameter combinations for the same fitted logistic regression (*left*) and 100 bootstrapped regressions (*right*). Compare these two plots with the more traditional visualisation of the best-fit regression and its 95% confidence envelope (*centre*). Which one best conveys information about the uncertainty in the model?

Code Box 2.9: *t*-test on simulated Gaussian data

```
set.seed(1234)
simu <- replicate(1000, {
  sim <- data.frame("group"=rep(c("A", "B"), each=10), "surface"=rnorm(20,
       mean(obs$surface), sd(obs$surface)))
  with(sim, t.test(surface ~ group))$p.value
})
sum(simu < 0.05)/1000
```

```
[1] 0.043
```

as the pooled observations, a *t*-test found a significant difference in 4.3% of the cases, very close to the expected error rate of 5%.[19]

In contrast, when we drew our samples from a population with the same mean and standard deviation but from a log-normal distribution (Code Box 2.10), the *t*-test detected a difference in only 3% of the cases, even though it should detect it in 5%. In other words, for these data and parameter estimates, the log-normal *lowered* the type-I error rate (*p*-value) from 0.05 to around 0.03, making the *t*-test a bit more conservative.

In answer to our question ("What is the effect of skewness on type-I error rates of a *t*-test?"), we conclude that the *t*-test is *even more likely* to be correct when the data are

[19] Because we sampled from all the data pooled, we would expect the *t*-test to return "no significant difference" for ≈ 95% of the simulations.

Code Box 2.10: *t*-test on simulated log-normal data

```
library(MASS)
 fit.lognormal <- fitdistr(obs$surface, "log-normal")
 simu.lognormal <- replicate(1000, {
   sim <- data.frame("group"=rep(c("A", "B"), each=10), "surface"=rlnorm(20, fit.
        lognormal$estimate[1], fit.lognormal$estimate[2]))
   with(sim, t.test(surface ~ group))$p.value
 })
 sum(simu.lognormal < 0.05)/1000

[1] 0.031
```

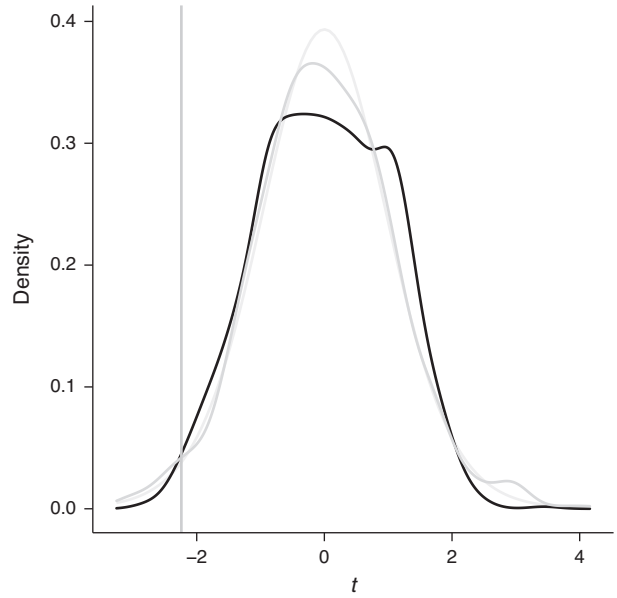


Figure 2.5 Simulating the t distribution for normally-distributed data (darker grey) gives a near-perfect match to the analytical equation for a t distribution with $2N - 2 = 18$ degrees of freedom (light grey). Using log-normal data, however, leads to some distortion in the peak (black) and the distribution is platykurtic (it has relatively "thin" tails); very small differences between the randomly assigned values were less common. However, this had little effect on the significance of the observed t-value for our data (vertical line) because the left tails of all the simulated distributions coincidentally were more similar than their right tails.

log-normally distributed than if they are Gaussian, as the type-I error rate is lower than expected (Figure 2.5). In contrast, if we had found a type-I error rate substantially higher than 5%, we would have concluded that the log-normal makes the detection of a spurious effect more likely.

The reason that the t-test on log-normal data is more conservative than a t-test applied to normally-distributed data is that the sample standard deviation s of the log-normal distribution is much larger. Hence, the t-statistic becomes smaller, moving towards non-significance: $t = \frac{\Delta Y}{\sigma/\sqrt{N}}$. We note that if the t-test is not significant, it may be tempting to calculate its power retrospectively. This is not a good idea (see §9.2 in Chapter 9).

2.4 Simulating autocorrelated data

The previous examples in this chapter have shown that a simulation can be very straightforward to implement once we have decided on the exact aim of the simulation. However, one common topic that poses additional challenges is the simulation of autocorrelated data. Autocorrelation comes in many forms, including temporal, spatial, and phylogenetic (Ives & Zhu 2006; Cressie & Wikle 2011). All three can be simulated in similar ways, so we illustrate the basic concepts of autocorrelated data with a simulation of temporal data. Other examples of working with autocorrelated data and simulating them are in §4.4 in Chapter 4, and §6.4 and §6.5 in Chapter 6.

Before simulating autocorrelated data, we first must identify the proximate cause of the observed autocorrelation. For example, if we observe the temperature at midnight over the course of a year, the observations will be highly autocorrelated. This temporal autocorrelation is caused by small daily differences and clear seasonal patterns in the amount of solar radiation reaching the Earth.

We need to distinguish such temporal dependence of Y (here, midnight temperature) on (i.e., autocorrelated with) X (here, solar radiation) from residual temporal autocorrelation. The latter case means that the residuals of a regression of Y on X (i.e., $resid_i = Y_i - \beta X_i$) are autocorrelated. We should be more concerned with residual autocorrelation than simple temporal dependence because residual autocorrelation violates the i.i.d. assumption of standard regression and machine-learning models.[20] For most simulations, this means that we have to simulate autocorrelation in the error term, *not* the observations (e.g., the dependent variable(s) in a model).[21] How do we do that?

We start by noting that temporal autocorrelation means that the observations Y_i made at time t, which we denote as $Y_{i(t)}$ (or more concisely, Y_t) are correlated with those at Y_{t-k}, where k is the temporal lag between the observations. An intercept-only autocorrelation model with lag (k) = 1 would be:

$$Y_t = \beta_0 + \beta_1 Y_{t-1} + \varepsilon_t. \tag{2.1}$$

In this case, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. That is, it is i.i.d. Gaussian with mean $\mu = 0$.[22] Algebraically, we can reformulate equation 2.1, pulling the autocorrelation term from the predictor term into the error term:

$$Y = \beta_0 + \varepsilon\,, \tag{2.2}$$

where $\varepsilon \sim \mathcal{N}(0, \Sigma)$, $\Sigma = \sigma^2 R$, and $R = \begin{pmatrix} 1 & \rho & \cdots & 0 & 0 \\ \rho & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & \rho \\ 0 & 0 & \cdots & \rho & 1 \end{pmatrix}$.

The square matrix R has values = 1 on the diagonal, so all the corresponding values on the diagonal of $\Sigma = \sigma^2$. The additional entries on the first off-diagonals are correlation

[20] i.i.d means that the observations are independent and identically distributed. That is, the underlying random variables y each have the same probability distribution as all the others and are mutually independent of all the others.

[21] That is, $Y_i = \beta X_i + \varepsilon_i$, with $\varepsilon \sim \mathcal{N}(0, \Sigma)$ and $\Sigma \neq I$. See Equations 2.2 and 2.3, and footnote 23, below.

[22] The normal distribution is defined by two parameters, the mean, μ, and its variance, σ^2: $\mathcal{N}(0, \sigma^2)$. Sometimes, as in our previous examples, we write this as $\mathcal{N}(\mu, \sigma)$, in which we have substituted the standard deviation σ for the variance σ^2. When we look at autocorrelated errors, however, we always use the variance parameterisation, as Σ is known as the "variance-covariance matrix" of the error term.

Code Box 2.11: Simulation of temporally autocorrelated data

```
set.seed(123)
rho = 0.7; sigma = 2; ts.mean.target <- 0
a <- ts.mean.target * (1-rho)
y <- rnorm(1, ts.mean.target, sigma)
for (i in 2:100) y[i] = a + rho* y[i-1]  + rnorm(1, 0, sigma)
```

coefficients for lag $k = 1$. If there are higher-order lags, additional off-diagonals also would have non-zero entries.[23]

We continue with the simplest case of lag $k = 1$. We can simulate autocorrelated error terms by first simulating uncorrelated noise and then adding in autocorrelation:

$$\varepsilon_t = a + \rho\varepsilon_{t-1} + \xi_t\,, \tag{2.3}$$

where a is the intercept not including the bias due to the autocorrelation, $\xi_t \sim \mathcal{N}(0, \sigma)$ is uncorrelated ("white") noise, and ρ describes the strength of temporal autocorrelation in terms of Pearson's correlation coefficient (these are the same values of ρ as in the off-diagonals of the variance-covariance matrix Σ). Note that the time series $\{\varepsilon\}$ has mean = $a/(1-\rho)$, not, as one might assume, a itself.[24]

To simulate autocorrelated data, we start with some value for $Y_{t=0}$ and then iteratively compute the values for Y_{t+1} based on the desired autocorrelation coefficient ρ, overall noise σ, and target mean value $a/(1-\rho)$, which typically would be set to 0 because we assume that ε has mean 0 (Code Box 2.11).

The result of this simulation is the time series shown in Figure 2.6, where we also illustrate diagnostic plots for autocorrelation and partial autocorrelation.[25] This figure shows clearly that the time series is autocorrelated at the desired level (first bar of the partial autocorrelation plot is at 0.7) and no other lag makes a substantial contribution to the autocorrelation (none of the values of the partial autocorrelation coefficient other than that at lag $k = 1$ exceed the significance bounds indicated by the dashed grey lines).

We conclude this simulation by showing that temporal autocorrelation affects inference. Knowing from the simulation that the expectation of Y, $E(Y)$, equals zero, we could (incorrectly) use an ordinary least squares (OLS) model to estimate $\bar{Y}$ and test its difference from 0 (Code Box 2.12).

The result reveals that the OLS model incorrectly detects a significant difference between the simulated Y and its expected value of 0 (a type-I error, or false positive). This is because the standard error of the estimated mean is very small and the model assumes the data are i.i.d.

The correct analysis would use a generalised least squares (GLS) model with an appropriately specified term for the temporal autocorrelation. Since we simulated data with a

[23] Generally speaking, converting between the variance-covariance matrix Σ and the correlation matrix R is a bit more involved. Let "diag" represent the diagonal of a matrix. Then $R = \text{diag}(\Sigma)^{-1/2}\,\Sigma\text{diag}(\Sigma)^{-1/2}$. In general, Σ needs to be standardised by the variances of ε, and R must be positive definite (i.e., $\rho \in (-a, a)$ for some a in Eq. 2.3). In this simple example, we assume that the variances (the values on the diagonal) are all equal (homoscedastic): $\text{diag}(\Sigma)^{-1/2} = 1/\sigma$ and therefore $R = \frac{1}{\sigma^2}\Sigma$ and $\Sigma = \sigma^2 R$.

[24] Higher lags can be simulated analogously: $\varepsilon_t = a + \rho^{(1)}\varepsilon_{t-1} + \ldots + \rho^{(k)}\varepsilon_{t-k} + \xi_t$. Note that we need $N + k$ data points for a time series of length N.

[25] These can be obtained quickly using the R commands `acf(y)` and `acf(y, type="partial")`.

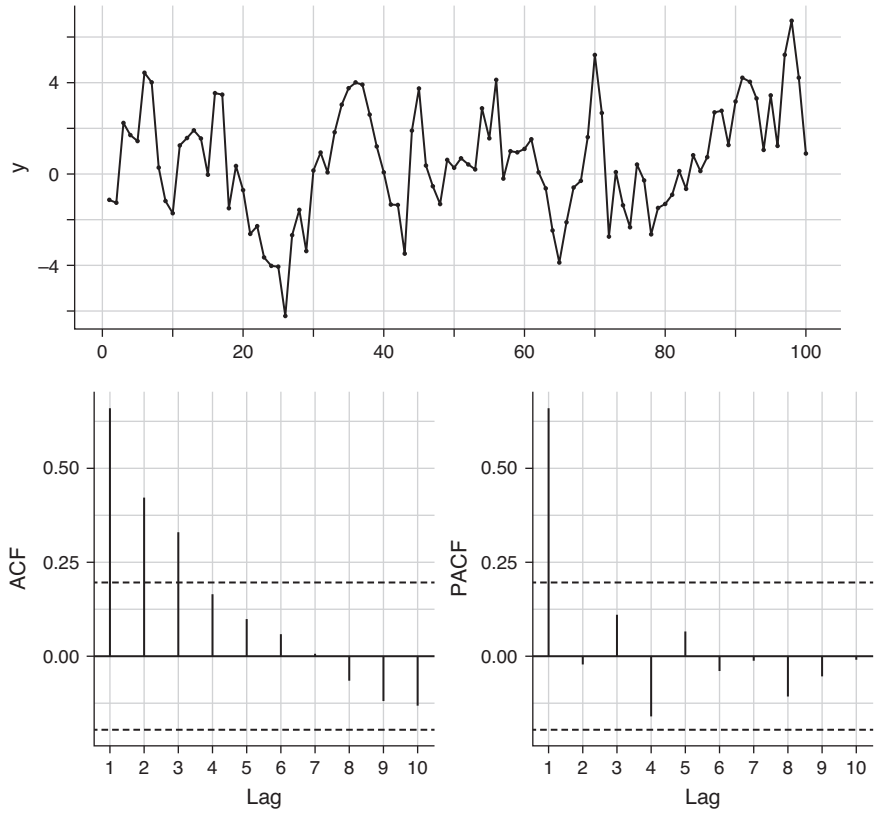


Figure 2.6 Simulated time series of 100 steps with a lag-1 autocorrelation of $\rho = 0.7$ (Code Box 2.11). The initial value in the simulation was $Y_1 = -1.57$ (drawn randomly), and $\sigma = 2$. Lower panels show the lagged autocorrelation and lagged partial autocorrelation, respectively; dashed lines are 95% confidence intervals, outside of which the (partial) autocorrelation is significantly different from zero.

lag-effect of one time step, the model would have the aptly-named "AR-1" structure (for autoregressive model with lag 1; Code Box 2.13).

Although the GLS estimate for $E(Y)$ is very similar to that obtained using an OLS model (compare the values for the `Intercept` in Code Boxes 2.12 and 2.13), the standard error (`Std.Error`) of the GLS model is more than twice as large. As a result, the estimate is no longer significantly different from 0. We avoided the false positive and got a true negative instead. Additionally, the GLS also gives us an estimate for the strength of temporal autocorrelation: the parameter `Phi`, which equals 0.63. This compares well with the actual value used in the simulation (`rho` = 0.7 in Code Box 2.11).

Finally, we note that this is but a small example demonstrating the advantages of data simulation. We can go much further with even this example and check whether we understood the statistics behind what we are doing (e.g., §11.1.2 in Chapter 11).

2.5 Simulation versus randomisation techniques

To conclude this chapter, we illustrate key differences between data simulations and the perhaps more familiar randomisation and resampling approaches used with non-parametric statistics. The literature on randomisation- and resampling-based approaches is large and rich (e.g., Efron 1982; Siegel & Castellan 1988; Härdtle et al. 2004; Wasserman 2005; Carsey

Code Box 2.12: An OLS model analysis of temporally autocorrelated data

```
# This Code Box uses the object "y" from Code Box 2.11

 summary(lm(y ~ 1)) # Fit y to a constant

| Residuals:
|    Min      1Q  Median      3Q     Max
| -6.7932 -1.7675 -0.1503  1.4194  6.1474
|
| Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept)   0.5814     0.2428   2.394 0.0185 *
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| Residual standard error: 2.428 on 99 degrees of freedom
```

Code Box 2.13: GLS analysis of temporally autocorrelated data

```
# This code continues to use the object "y" from Code Box 2.11

# Load required libraries
 library(nlme) # For the gls() function

 summary(gls(y ~ 1, correlation=corAR1(form = ~1)))

| Generalized least squares fit by REML
|  Model: y ~ 1
|       AIC      BIC    logLik
|  408.8552 416.6405 -201.4276
|
| Correlation Structure: AR(1)
| Formula: ~1
| Parameter estimate(s):
|       Phi
| 0.6722837
|
| Coefficients:
|                Value Std.Error  t-value p-value
| (Intercept)0.5543876 0.5451855 1.016879 0.3117
|
| Standardized residuals:
|          Min           Q1          Med           Q3          Max
|    -2.74775941    -0.70683413  -0.05006202     0.58737581  2.50738737
|
| Residual standard error: 2.462462
| Degrees of freedom: 100 total; 99 residual
```

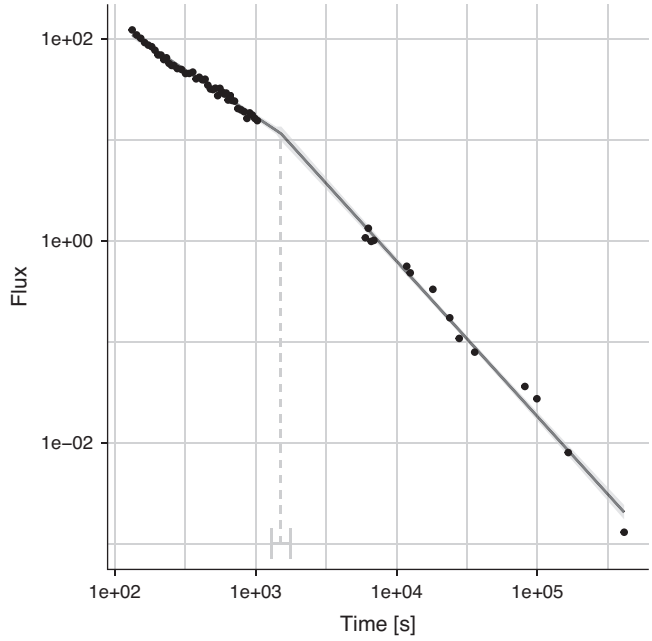


Figure 2.7 Decay of the light curve of gamma-ray burst GRB050525a (X-ray telescope data from Blustin et al. 2006 and see NASA 2009 for additional details). Flux is measured in 10^{-11}erg cm^{-2}s^{-1} 2–10 keV, where an erg is 10^{-7} Joule and "2–10 keV" describes the energy of the photons measured (in this case, the range of the X-rays). Both axes are log-scaled. The fitted line (with 95% CI) is from a piece-wise regression (see Code Box 2.14); the estimated breakpoint is indicated by the vertical dashed line (with 95% CI).

& Harden 2014; Manly 2017; Gibbons & Chakraborti 2020; Kolassa 2020; Barr 2021). These same books barely touch on data simulations like the ones we present in this book, even though it is only a small conceptual step from resampling and randomisztion to data simulation. In brief, randomisations use only the available (i.e., observed) data, whereas in data simulations, additional (i.e., simulated) data are used. As an exemplar, we use data describing the time-course of the light intensity of a gamma-ray burst from a far-away galaxy that was observed on Earth in 2005 (Blustin et al. 2006).

Gamma-ray bursts are the most energetic explosions known in our universe. Astrophysicists hypothesise that when a supernova collapses into a neutron star or a black hole, it releases a focused beam of gamma radiation of enormous intensity along its rotational axis. In 2005, a specific gamma-ray burst was observed for almost a week, during which the energy constantly decreased and, accordingly, the wavelength increased. Our concern here is the observation that the slope of energy decay in the first hundreds of seconds appears to differ afterwards (indicated by the breakpoint in Figure 2.7).

Breakpoint analysis using segmented regression gives estimates of the breakpoint and the slopes before and after it, but no *p*-value for the probability that the two slopes are different (Code Box 2.14).

The breakpoint is estimated to lie at $10^{3.179\pm0.067} = 1{,}510$ (95% $CI = [1{,}294, 1{,}762]$) seconds (or ≈1.5e+03 s in Figure 2.7), and corresponds with the gap at the end of the intense initial measurement period. Across the breakpoint, the slope changes by −0.61 (the `Estimate` for `U1.log10(time)` in Code Box 2.14), from −0.93 to −1.54. However, the column for the *p*-value of this slope change is empty. The reason is that the slope change estimate is conditional on a breakpoint being detected in the first place. Since that probability is

Code Box 2.14: Fitting a segmented regression model to the gamma-ray burst data

```
# Load required libraries
 library(faraway)   # Includes the gamma-ray data
 library(segmented) # For segmented regression functions

# Fit a breakpoint model
 fm <- segreg(log10(flux) ~ seg(log10(time), npsi=1), data=gammaray)
 summary(fm) # Note that there is no p value for the 2nd slope estimate

|   ***Regression Model with Segmented Relationship(s)***
|
| Call:
| segreg(formula = log10(flux) ~ seg(log10(time), npsi = 1), data = gammaray)
|
| Estimated Break-Point(s):
|                    Est. St.Err
| psi1.log10(time) 3.179  0.067
|
| Coefficients of the linear terms:
|                Estimate Std. Error t value Pr(>|t|)
| (Intercept)     4.00584    0.08411   47.62   <2e-16 ***
| log10(time)    -0.92548    0.03193  -28.98   <2e-16 ***
| U1.log10(time) -0.61464    0.04169  -14.74       NA
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.05625 on 59 degrees of freedom
| Multiple R-Squared: 0.9976,  Adjusted R-squared: 0.9974
|
| Boot restarting based on 6 samples. Last fit:
| Convergence attained in 2 iterations (rel. change 1.032e-13)
```

unknown, or at least is not obvious, the statistical significance of the change in slope cannot be tested meaningfully.[26]

So how can we test whether there is a breakpoint or not? The simplest way is to fit two models, one with a breakpoint and one without a breakpoint, and compare them with a likelihood-ratio test (LRT) evaluated against a χ^2-distribution with (in this case) two degrees of freedom—one for the changed slope, and one for the breakpoint (Code Box 2.15).

The χ^2-test statistic, which equals $2\times$ the difference between the log-likelihoods of the two models, is highly significant. We conclude from this that the model with a breakpoint fits the data better than the model without one. Can we trust this result?

[26] The Davies test can be employed in this situation. It puts a breakpoint at different positions of x and then evaluates the slope change. This serves as an expectation for the change in slope, *if* a breakpoint is imposed; see the documentation accompanying the R function `segmented::davies.test()`.

Code Box 2.15: Testing the fit of the segmented regression of gamma-ray burst with a likelihood-ratio test

```
# This code uses the "gammaray" data object and the breakpoint model "fm" fitted
      in Code Box 2.14.

# Load required libraries
 library(lmtest)    # includes the lrtest() function

# Fit a model without a breakpoint
 flmNo <- lm(log10(flux) ~ log10(time), data=gammaray)

# Compare the model with a breakpoint (fm) to the model without a breakpoint
 (flmNo) lrtest(flmNo, fm)

| Likelihood ratio test
|
| Model 1: log10(flux) ~ log10(time)
| Model 2: log10(flux) ~ seg(log10(time), npsi = 1)
|   #Df LogLik Df Chisq Pr(>Chisq)
| 1   3 44.472
| 2   5 93.982  2 99.02  < 2.2e-16 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# save the observed X2-value of LRT:
 obsLRT <- lrtest(flmNo, fm)$Chisq[2]
```

The χ^2-test is asymptotically correct; but with only 63 observations, we do not know how close we are to this asymptote. We can use one of two strategies to make the analysis more robust. First, we can rely on the LRT but use bootstrapping or data perturbation to allow for more uncertainty in the observed likelihood ratio. Alternatively, we can use randomisation or simulation to generate an approximation to the χ^2-distribution based on what the data for the null hypothesis (no breakpoint) would look like.

BOOTSTRAPPING THE LRT

We introduced bootstrapping above in §2.3.2. Here, we are interested in the distribution of bootstrapped log-likelihood ratios and whether they are larger than the critical value for the quantile function of the χ^2-distribution with two degrees of freedom.[27] The code for bootstrapping the breakpoint model is in Code Box 2.16.

The bootstrap yields a confidence interval on the observed LRT. Because none of the values is less than the critical value, the result of the original LRT is confirmed: the breakpoint model fits significantly better than the no-breakpoint model (with $p < 1/1{,}000 = 0.001$).

[27] The quantile function is typically written as the inverse of the cumulative density function, $Q(p) = F_x^{-1}(p)$. In R, these functions work in the same way as the random number generating functions (§A.1 in Appendix A), but they are prefixed with `q` instead of `r`. Here, we would use `qchisq(0.05, df=2, lower.tail=F)`, which returns the value of 5.99 for χ^2_2 and $\alpha = 0.05$.

Code Box 2.16: Bootstrapping the segmented regression model of the gamma-ray burst data

```
# This code uses the "gammaray" data object created in Code Box 2.14

# Load required libraries
 library(lmtest)    # includes the lrtest() function

 set.seed(123)
 bs <- replicate(1000, {
  dats <- gammaray[sample(nrow(gammaray), replace=T), ] # Bootstrapping
  onefm <- segreg(log10(flux) ~ seg(log10(time), npsi=1), data=dats) # Fit a
      breakpoint model
  oneflm <- lm(log10(flux)  ~ log10(time), data=dats) # Fit a no-breakpoint model
  lrtest(oneflm, onefm)$Chisq[2] # LRT
 })
 summary(bs)    # From which we can derive confidence intervals

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
|   62.17   97.18  105.17  107.21  116.40  168.48

 sum(bs < 5.99) / length(bs)

| [1] 0
```

PERTURBING THE DATA

Data perturbation is a statistical approach that adds noise to the observed data. Data perturbation is applied commonly to training deep learning models, in which perturbations are used to augment training data to boost sample size (Hazan et al. 2016, and see §1.3.3 in Chapter 1). Continuing with our gamma-ray example, we simply add random noise to the observations and then recompute the log-likelihood. We use the standard deviation of the residuals from the no-breakpoint model to parameterise the distribution of the noise that we add to the observations. We repeat this process many times, which, as for bootstrapping, results in a distribution of log-likelihood ratios (Code Box 2.17.)

The results are similar to those of the bootstrapping, although altogether closer to 0. Still, all 1,000 perturbations are far away from the critical value of 5.99, thus confirming that the breakpoint model fits the data better than the no-breakpoint model ($p < 0.001$).

RESAMPLING

A typical approach in non-parametric statistics is the use of resampling, or data shuffling.[28] When we resample the sequence of Y data in a regression, any signal of X is destroyed. Applying this idea to the breakpoint regression at hand means shuffling `flux` but maintaining the values of `time`. The reshuffled data are then used to fit the two models and compute their

[28] Both bootstrapping and resampling sample from the data. The difference between them is that bootstrapping samples from the data *with replacement*, whereas resampling samples from the data without replacement. Resampling an entire dataset recovers all the data, but in a different order, which is pointless. Instead, we restrict resampling to the response variable, when it becomes equivalent to randomising or "shuffling" the data (Efron 1982).

Code Box 2.17: Applying data perturbation to the segmented regression of the gamma-ray burst data

```
# This code uses the "gammaray" data object created in Code Box 2.14

# Load required libraries
 library(lmtest)    # includes the lrtest() function

# Add random noise to the data.
 pert <- log10(gammaray$flux) + rnorm(nrow(gammaray), 0, sd(residuals(flmNo)))

 set.seed(123)
 perts <- replicate(1000, {
  pert <- log10(gammaray$flux) + rnorm(nrow(gammaray), 0, sd(resi))
  onefm <- segreg(pert ~ seg(log10(gammaray$time), npsi=1)) # Fit a breakpoint
       model
  oneflm <- lm(pert  ~ log10(gammaray$time)) # Fit a no-breakpoint model
  lrtest(oneflm, onefm)$Chisq[2] # LRT
 })
 summary(perts)

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
|  39.41   62.67   66.45   66.94   70.34   89.36

sum(perts < 5.99) / length(perts)

| [1] 0
```

log-likelihood ratios (Code Box 2.18). There should be no breakpoint with the randomised data, so the resulting log-likelihood ratios should approximate a χ^2-distribution for the null hypothesis, i.e., of no effect (and no breakpoint). If the observed log-likelihood ratio is very different from those computed for many randomisations, we can assume the breakpoint model to be significantly different from the no-breakpoint model.

Resampling shows that occasionally a χ^2-value much higher than 5.99 can be expected from this kind of data: the maximal observed, in 1,000 runs, is just under 15. Still, all the permutations of the values remain substantially below the observed critical value for the LRT and hence support the evidence for a breakpoint ($p < 0.001$).

DATA SIMULATION

In contrast to resampling, data simulation generates many new datasets drawn from a distribution with parameters similar to those observed. In this case, we can create no-breakpoint data and run both models to get a distribution of LRT values for this specific setting (Code Box 2.19). As with the resampling methods, we can then fit breakpoint and no-breakpoint models to the simulated dataset and use LRTs to compare them. It is thus similar to the previous randomisation (Code Box 2.18), but each time we use new data instead.

The data simulation is much more constrained than the permutation and yields values closer to the actual χ^2-distribution. Here the maximal LR-value is 6.8, and hence far below the observed value of 99. Again, the breakpoint model is supported by a wide margin ($p < 0.001$).

Code Box 2.18: Resampling of the gamma-ray burst data

```
# This code uses the "gammaray" data object created in Code Box 2.14

# Load required libraries
 library(lmtest)    # includes the lrtest() function

 resam <- replicate(1000, {
  set.seed(Sys.time()) # Use a different seed for each perturbation
  x <- log10(gammaray$flux[sample(nrow(gammaray), replace=F)])
  onefm <- segreg(x ~ seg(log10(gammaray$time), npsi=1)) # Fit a breakpoint model
  oneflm <- lm(x ~ log10(gammaray$time) ) # Fit a no-breakpoint model
  rm(x)
  lrtest(oneflm, onefm)$Chisq[2] # LRT
})

 summary(resam)

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
|  0.1798  1.5938  2.6382  3.0712  3.4122 14.5281

 sum(resam > obsLRT) / length(perts)

| [1] 0
```

Code Box 2.19: Full simulation of the gamma-ray burst data

```
# This code uses the "gammaray" data object created in Code Box 2.14

# Load required libraries
 library(lmtest)    # includes the lrtest() function

# First, run and evaluate a single simulation

 # Fit a no-breakpoint model to the original data.
  summary(flmNo <- lm(log10(flux) ~ log10(time), data=gammaray)) #
 # Use the parameter estimates from the fitted model to define the simulation
  set.seed(123)
  simu <- coef(flmNo)[1] + coef(flmNo)[2] * log10(gammaray$time) + rnorm(
       nrow(gammaray), 0, sd(residuals(flmNo)))

# Run many simulations
 set.seed(123)
 sims <- replicate(1000, {
  simu <- coef(flmNo)[1] + coef(flmNo)[2] * log10(gammaray$time) + rnorm(
       nrow(gammaray), 0, sd(residuals(flmNo)))
  onefm <- segreg(simu ~ seg(log10(gammaray$time), npsi=1))
  oneflm <- lm(simu ~ log10(gammaray$time))
```

```
 lrtest(oneflm, onefm)$Chisq[2]
})
summary(sims)

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
| 0.06946 1.00988 1.72102 2.17949 3.28649 6.76714

sum(sims > obsLRT) / length(sims)

| [1] 0
```

IS THERE A "RIGHT" APPROACH?

The four approaches all address the same question, but the two strategies differ in whether they compute an interval for the single observed likelihood ratio (bootstrapping and perturbation) or recompute the reference distribution (resampling and simulation). There is no "wrong" approach to take, and statisticians use all of them in different contexts. However, the comparison between the randomisation approaches (bootstrapping, resampling) and the simulation-based ones (perturbation is a simple version of a full data simulation) clarifies what we will do in the rest of the book: use new datasets created with simulations rather than working solely with the data provided.

2.6 Summary

- The nine steps for developing a meaningful data simulation and a data generating model include formulating a question and a model; specifying the data distribution; identifying the process model (i.e., the equations needed to simulate the data); formulating each step for the analysis; determining which functions are needed for the simulation; setting a random "seed" to ensure that the simulation is reproducible; running the code once to make sure it works correctly; repeating the runs a large number of times; and visualising and evaluating the output to answer the question.
- The two most important ingredients for a successful data simulation are having a clear objective and running a sufficiently large number of replicate simulations.
- Bootstrapping, data perturbation, resampling, and data simulation address the same questions but differ in whether they use only the observed data (bootstrapping, perturbation, resampling) or additional data derived from a data-generating model (data simulation). Bootstrapping and data perturbation estimate a confidence interval for a single statistic whereas resampling and data simulation recompute the entire reference distribution. None of these approaches are "wrong," but which one is the "right" one to use depends on the question, the model, and the context.

References

Aguirre-Gutiérrez, J., Berenguer, E., Oliveras Menor, I., et al. (2022). Functional susceptibility of tropical forests to climate change. *Nature Ecology & Evolution*, 6(7), 878–889. doi: 10.1038/s41559-022-01747-6.

Aldous, D. J. (1985). *Exchangeability and Related Topics*. Berlin, Germany: Springer.

Barr, D. J. (2021). *Learning Statistical Models through Simulation in R: An Interactive Textbook*, volume 1.0.0. https://psyteachr.github.io/stat-models-v1. Accessed 10 December 2024.

Blustin, A. J., Band, D., Barthelmy, S., et al. (2006). Swift panchromatic observations of the bright gamma-ray burst GRB050525a. *The Astrophysical Journal*, 637(2), 901–913. doi: 10.1086/498425.

Carsey, T. M. & Harden, J. J. (2014). *Monte Carlo Simulation and Resampling Methods for Social Science*. Los Angeles, CA, USA: Sage.

Cressie, N. & Wikle, C. K. (2011). *Statistics for Spatio-Temporal Data*. Hoboken, NJ, USA: John Wiley & Sons.

Dallal, G. E. & Wilkinson, L. (1986). An analytic approximation to the distribution of Lilliefors's test statistic for normality. *The American Statistician*, 40(4), 294–296. doi: 10.1080/00031305.1986.10475419.

DiRenzo, G. V., Hanks, E., & Miller, D. A. W. (2023). A practical guide to understanding and validating complex models using data simulations. *Methods in Ecology and Evolution*, 14(1), 203–217. doi: 10.1111/2041-210X.14030.

Donovan, T. M., Brown, M., & Katz, J. E. (2021). R for Fledglings. https://www.uvm.edu/~tdonovan/RforFledglings. Accessed 1 February 2024.

Efron, B. (1982). *The Jackknife, the Bootstrap, and Other Resampling Plans*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Ellison, A. M. & Dennis, B. (2010). Paths to statistical fluency for ecologists. *Frontiers in Ecology and the Environment*, 8(7), 362–370. doi: 10.1890/080209.

Gelman, A. & Nolan, D. (2002). *Teaching Statistics: A Bag of Tricks*. New York, NY, USA: Oxford University Press.

Gibbons, J. D. & Chakraborti, S. (2020). *Nonparametric Statistical Inference*, 6th edition. Boca Raton, FL, USA: CRC Press.

Gosset, W. S. (1908). The probable error of a mean. *Biometrika*, 6(1), 1–25. doi: 10.1093/biomet/6.1.1.

Härdtle, W. K., Müller, M., Sperlich, S., & Werwatz, A. (2004). *Nonparametric and Semiparametric Models*. Berlin: Springer.

Hazan, T., Papandreou, G., & Tarlow, D., Eds. (2016). *Perturbations, Optimization, and Statistics*. Cambridge, MA, USA: MIT Press.

Hoppe, F. M. (1984). Pólya-like urns and the Ewens' sampling formula. *Journal of Mathematical Biology*, 20(1), 91–94. doi: 10.1007/BF00275863.

Ives, A. R. & Zhu, J. (2006). Statistics for correlated data: phylogeny, space, and time. *Ecological Applications*, 16(1), 22–32. doi: 10.1890/04-0702.

Johnson, N. L., Kotz, S., & Balakrishnan, N. (1995). *Continuous Univariate Distributions*, volume 2, 2nd edition. New York, NY, USA: John Wiley & Sons.

Kolassa, J. E. (2020). *An Introduction to Nonparametric Statistics*. Boca Raton, FL, USA: Chapman & Hall.

Mahmoud, H. (2008). *Pólya Urn Models*. Boca Raton, FL, USA: Chapman & Hall/CRC.

Manly, B. F. J. (2017). *Randomization, Bootstrap and Monte Carlo Methods in Biology*. New York, NY, USA: Chapman and Hall/CRC, 3rd edition.

Metropolis, N. (1987). The beginning of the Monte Carlo method. *Los Alamos Science*, Special Issue 1987, 125–130.

Metropolis, N. & Ulam, S. (1949). The Monte Carlo method. *Journal of the American Statistical Association*, 44(247), 335–341. doi: 10.1080/01621459.1949.10483310.

Morris, T. P., White, I. R., & Crowther, M. J. (2019). Using simulation studies to evaluate statistical methods. *Statistics in Medicine*, 38(11), 2074–2102. doi: 10.1002/sim.8086.

NASA (2009). SWIFTGRB: The Swift Gamma-Ray Burst Catalog; GRB 050525A. https://heasarc.gsfc.nasa.gov/docs/swift/archive/grbsummary/grb_pages/2005/grb050525a/grb050525a.html. Accessed 1 May 2024.

Neyman, J. (1935). On the problem of confidence intervals. *The Annals of Mathematical Statistics*, 6(3), 111–116. doi: 10.1214/aoms/1177732585.

Neyman, J. (1937). Outline of a theory of statistical estimation based on the classical theory of probability. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 236(767), 333–380. doi: 10.1098/rsta.1937.0005.

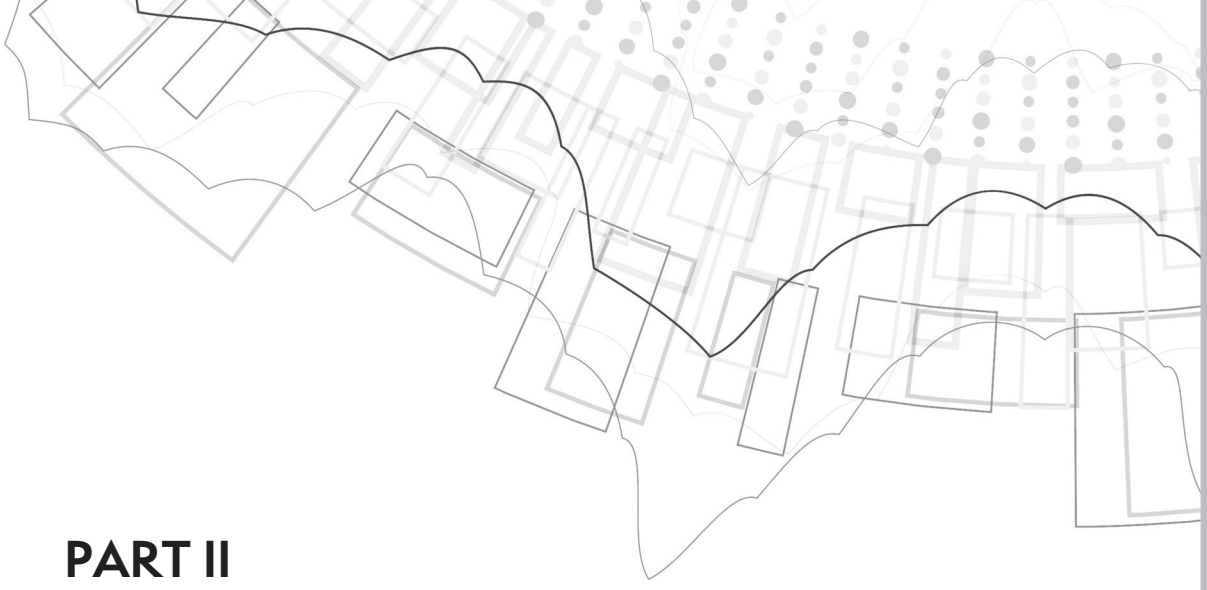
Shorack, G. R. & Wellner, J. A. (2009). *Empirical Processes with Applications to Statistics*, revised edition. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Siegel, S. & Castellan, N. J. (1988). *Nonparametric Statistics for the Behavioral Sciences*. New York: McGraw-Hill.

van der Vaart, A. W. & Wellner, J. A. (1996). *Weak Convergence and Empirical Processes*. New York, NY, USA: Springer.

Venables, W. N., Smith, D. M., & The R Core Team (2023). An Introduction to R. https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf. Accessed 11 February 2024.

Wasserman, L. (2005). *All of Nonparametric Statistics*. New York, NY, USA: Springer.



PART II

Ante mensuram: Prospective simulations of study designs and their power

The process of doing a scientific research project includes identifying a hypothesis of interest, planning and designing the project to test the hypothesis, executing the project and collecting the data, analysing and interpreting the data, and finally, publishing the results and their interpretation. This section of the book focuses on the planning and design phase of a research project, and on how simulations can help make explicit its assumptions, challenges, and benefits.

Why is planning and design so important? Scientific research projects usually are very costly in terms of time, labour, or money, and these resources can be hard to come by. On the one hand, scarce resources should be marshalled and used effectively to ensure that the project can be done successfully and that the key hypotheses can be tested credibly. On the other hand, if other researchers have already collected and published enough data to support or reject the hypotheses of interest, it would be a waste of limited resources to test them again. In Bayesian statistical language, we would say that the project is justifiable and the design is appropriate if the expected likelihood distribution of the data to be collected (the "likelihood") is expected to be substantially more informative (that is, the likelihood distribution will be sharper) than the prior probability distribution (the "prior"). In contrast, the study is likely to be a waste of time if the prior is expected to be much sharper than the expected likelihood.

In the next two chapters, we show how we can use simulations to address these issues by considering four topics in detail: project planning through synthesis of existing data; using such syntheses to design experiments (whether manipulative, observational, or "natural"); preregistering them to avoid experimenter bias (Chapter 3); and determining the power of different experimental designs to detect effects (Chapter 4).

3

Think before you act

All research begins with an idea, a question, or an hypothesis. These come from observations—we see a cat catch a bird and wonder if this was a common or unusual event, how the cat caught the bird, or whether we can predict how other cats will catch other birds. We then use additional observations along with experiments, statistics, and models to answer our questions, support or reject our hypotheses, and draw inferences, and use predictor variables outside of the range we observed or measured to forecast events that could occur in the future. It can be tempting to start making more observations right away, immediately set up an experiment and collect as much data as possible, and use as many statistics as it takes to find significant support for the hypothesis of interest. We should resist these temptations. Studies that are thought out carefully and designed well will yield clear answers, robust hypothesis tests, and reliable inferences and forecasts. Ill-conceived and poorly designed studies will not.

In this chapter, we lay out a series of steps that use simulations to help us carefully think through our questions, design clear studies to answer them, and plan in advance the appropriate analyses needed to test our hypotheses. We also advocate for going through the process of formally pre-registering a study to increase its robustness, reliability, and reproducibility, enhance the credibility of its results and conclusions, and minimise the unconscious biases that all of us bring to our work (e.g., Nosek et al. 2018, 2022). We illustrate these ideas by working through a single case study that should motivate you to try these ideas on your own nascent research projects.

What you will learn from this chapter

- How data simulation can be used to communicate uncertainty and minimize biases of researchers.
- The importance of setting prior expectations and stating clear and testable hypotheses.
- What is statistical power and how it is related to effect size, sample size, and statistical significance.
- What are questionable research practices and how to avoid them using preregistration and simulation.

3.1 The illusion of truth: A case study

It is "common knowledge" that we are awash in "fake news." "Facts" proliferate faster than data are collected, "experts" can be found to support the facts, and the social-media echo

chambers reinforce the "truth" du jour. Why all the scare quotes? The use of terms such as "common knowledge," "fake news," "facts," "experts," and "truth" often communicate certainty while hiding unconscious biases (Fallis 2015). Three important reasons to use simulations are to refine our ideas while exposing our biases, test our assumptions, and more clearly communicate the uncertainty in our results.

While philosophers have pondered the idea and reality of truth (e.g., Kirkham 1992), psychologists have asked a more specific question (Bastick 2021): How do people decide whether a statement is true or false? Decades of research have uncovered the "illusory truth effect": if a person repeatedly hears a statement then they will be more likely to consider that statement to be true than if they hear it only once or after they have heard it for the first time (Dechêne et al. 2010). Fazio et al. (2019) set out to examine a specific and previously untested instance of the illusory truth effect: the hypothesis that when individuals repeatedly hear an assertion of an obviously true or false statement that their belief in the truth or falsity of the statement should not change. To test this hypothesis, they used a combination of data simulation, a controlled experiment, and pre-planned (and pre-registered) statistical analyses that included bootstrapping to determine statistical significance. The results were unsettling. Fazio et al. (2019) found that the illusory truth effect is independent of the actual truth or falsity of a statement. That is, even obviously false statements, such as the Earth is a perfect square or an elephant weighs less than an ant, become more believable after they are repeated.

By pre-registering their study, Fazio et al. (2019) completely documented their proposed and actual methods and analysis (i.e., their complete workflow; see Chapter 7 for further discussion and additional examples of workflows) and provided the code they used for prospective simulations (Pennycook 2019). Thus, their study provides a compelling and complete example of how to use simulations effectively throughout the process of hypothesis generation, planning, and designing an appropriate experiment to test the hypothesis.

3.2 The question comes first

The illusory truth effect has been studied by psychologists and cognitive scientists at least since the 1970s (Dechêne et al. 2010). There is broad consensus based on the results of individual studies and of meta-analyses (see Chapter 10 for an in-depth discussion of meta-analysis) of the truth of the illusory truth effect, but Dechêne et al. (2010) also distinguished and identified many factors that could affect its size and strength.

All researchers and authors of the papers reviewed by Dechêne et al. (2010) did agree, however, that repeated assertion of an obviously true or false statement should not lead to a change in an individual's belief in its truth or falsehood. In other words, the illusory truth effect should only apply to ambiguous statements. Fazio et al. (2019) set out to test this hypothesis, a test which has taken on broader importance as the number of unabashedly lying politicians and deep-fake videos has increased dramatically.[1] More generally, Fazio et al. (2019) asked: does the illusory truth effect apply to all statements or only to some of them?

[1] Fazio et al. (2019) restricted the motivation for their study to a test of the hypothesis that the illusory truth effect applies only to statements that are demonstrably neither true nor false. Nonetheless, their concluding sentence (Fazio et al. 2019, p. 1710) illustrated their awareness of the broader context for their study and its results: "These findings have large implications for daily life where people are often repeatedly exposed to both plausible and implausible falsehoods. Even implausible falsehoods may slowly become more plausible with repetition."

3.3 Setting expectations, defining hypotheses

Let's take a step back from the specifics of the illusory truth effect and consider the general process that scientists use to answer questions and test hypotheses.

People have been observing and manipulating the world around us for millennia, and following the "scientific method" in making observations and doing experiments for centuries.[2] Thus, observations and experiments rarely lack precedents; all of our work is built on the work of others.[3] That, in turn, means we ought to have some expectation of what we will find, and we implicitly or explicitly develop and modify our research designs accordingly. This part of the scientific method is part of the deductive step we take from the broad framework of an underlying theory to a specific hypothesis to be tested (Hepburn & Andersen 2021). Our observational or experimental study is then designed to test this specific hypothesis. Other researchers are more likely than not to have worked on a similar question before. We can use the results of such studies not only to inform our research designs but also to suggest a range of plausible outcomes and parameterise our prospective simulations.

3.3.1 Formulating prior expectations

Before we start observing a population, sampling it, or designing an experiment, we should articulate what we expect to learn from it. We begin our research by combing the literature to identify previous research on the same topic done by others in the same field or related ones. These literature reviews may be haphazard and informed by one's experience or research colleagues and mentors, or they may be structured and systematic (e.g., Page et al. 2021). Either way, the results of these studies can be combined using meta-analysis to estimate strengths of observed effects or experimentally-induced treatments, gauge the accuracy and precision of existing measurement devices, or identify potential sources of error and unknown background noise (e.g., Koricheva et al. 2013; Page et al. 2021; Schmid et al. 2021; and see Chapter 10 for further discussion of meta-analysis).

The results from literature reviews, syntheses, and associated meta-analyses then can be used in many different and effective ways. In designing an experiment, the most important use of these results is to estimate effect sizes and their variances that might be observed, or to use them in initial simulations of statistical power and detectable effect sizes, given a sampling design. The results of these simulations can be used to suggest more appropriate (larger or smaller) sample sizes and allocation of individuals to treatment groups; identify key variables to measure; test scripts for different statistical analyses that could be applied to the expected data; and estimate the statistical power of planned frequentist analyses.

The use of prior information is especially important in Bayesian analyses. Using Bayes' theorem (Bayes 1763), a new study updates what is known before the study was done—the aptly-named prior, $P(\theta)$—with the new data (x) that have been collected after the study is complete. The latter—the posterior $P(\theta|x)$—is defined as:

$$P(\theta|x) = \frac{P(x|\theta)p(\theta)}{P(x)}. \tag{3.1}$$

[2] Although most "Western" scientists would probably date the origin of the scientific method to Roger Bacon (Clegg 2013) and René Decartes (Ladyman 2001), its roots are much deeper and not restricted to the "West" or the "Global North." In fact, the origins of what we now consider to be "Western science" can be found in Mesopotamia and areas of the world that lie within those countries currently identified as China, India, and Egypt. See, for example, Needham (1981) and the ever-evolving timeline on Wikipedia (Wikipedia Contributors 2022).

[3] Writing to Robert Hooke in 1675, Isaac Newton wrote of his scientific discoveries that "if I have seen further [than others], it is by standing on the shoulders of giants." The origin of this sentiment pre-dates Newton's assertion of it, likely by many centuries.

The other term in the numerator of Bayes' theorem is the likelihood of the data, $P(x|\theta)$ (often written as $\mathcal{L}(\theta|x)$); the term in the denominator is a normalising constant, $P(x) = \int_\theta f(x_1, \ldots, x - N, \theta) d\theta$.[4]

There are three interpretations of the prior, $P(\theta)$: (1) a frequency distribution whose parameters reflect analysis and synthesis of existing data; (2) an "objective" or rational statement of belief about the initial parameters or their distributions, given initial ignorance of them; and (3) a subjective measure of what the investigator actually believes about the parameters of interest and their distributions before they do their experiment (Cox & Hinkley 1974). Whether we use a prior based on quantitative (meta-)analysis and synthesis or a more qualitative one derived from formal or informal elicitation methods (e.g., Azzolina et al. 2021; Falconer et al. 2022), a well-designed study whose likelihood $\mathcal{L}(\theta|x)$ is more informative (or "sharper") than the prior probability distribution will increase our knowledge about the phenomenon or process of interest (see §4.5 in Chapter 4 and, e.g., Stefan et al. 2022).

3.3.2 What will the most important figure be?

Learning what others have found out about your question of interest and formulating a prior probability distribution are important first steps in any research project. They also suggest an entrée to using data simulations. We should flash forwards to what we expect to be the final product of our research: a presentation we will deliver about the work (as an oral or poster presentation at a scientific meeting) or the paper that we will write about it. What will the central figure (or table) of the talk or paper look like? We may not know where the data will fall, but based on our literature review, meta-analysis, or $P(\theta)$, we should have a good idea about what the axes, scales, and labels of the graph ought to look like; how these will translate to the row and column titles of the associated dataset or data table; and the units, accuracy, and precision of the key variables.

To illustrate this idea, consider our case study of the illusory truth effect. Figure 3.1 is a mock-up of the key plot that would illustrate how the illusory truth effect studied and tested by Fazio et al. (2019) might vary with the degree of truth of a given statement. Creating a blank plot like that shown in Figure 3.1 focuses our attention on three essential questions that we should answer to guide the design and analysis of this or any other research study, for defining a data-generating model, and for running data simulations:

1. How will the variables be quantified and measured?
2. What is the expected (hypothesised) relationship between the variables?
3. What test is appropriate to assess the likelihood of (support for) the hypothesis?

3.3.3 Quantifying and measuring the variables of interest

Translating our ideas into an hypothesis usually starts with a conceptual model. A blank figure such as Figure 3.1 can help us turn this conceptual model into a more quantitative one. In our case study of the illusory truth effect, we need a way to quantify and measure the initial plausibility of a statement and model how this plausibility might be expected to change after the statement has been repeated one or more times. Fazio et al. (2019) wrote this model as:

$$a_{ij} \sim p_i + f_i, \tag{3.2}$$

[4] The analytical intractability in most circumstances of computing the denominator in Bayes' theorem led to the development of Markov chain Monte Carlo ("MCMC") and other computationally-intensive methods that have much in common with data simulations (Robert & Casella 2011).

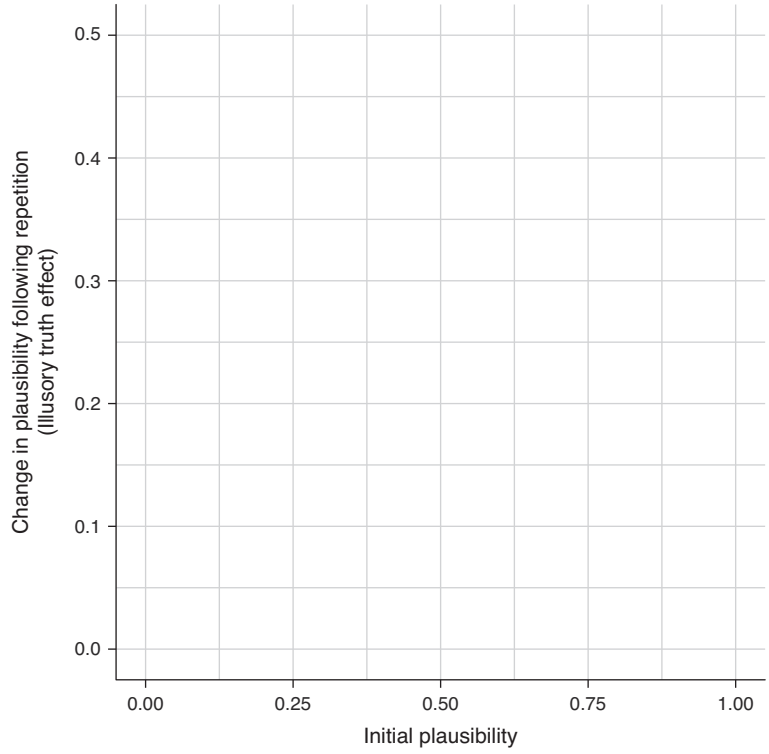


Figure 3.1 A template graph for illustrating the results of an experiment testing the effect of the perceived truth of a statement on the illusory truth effect. The values on the *x*-axis range from 0 for a demonstrably false statement to 1 for a demonstrably true statement. The values on the *y*-axis range from 0 (no effect of repetition) to some positive value for the illusory truth effect. The maximum value shown here for the *y*-axis is somewhat larger than the maximum estimated from meta-analyses (≤ 0.4) for the illusory truth effect (Dechêne et al. 2010).

where a_{ij} is the perceived accuracy of statement i by individual j, p_i is the plausibility of statement i, and f_i is the increase in the mean perceived accuracy of statement i attributable to its repetition. Thus, for a "novel" statement (i.e., one that an individual hears for the first time), $f_i = 0$ and $a_{ij} \sim p_i$.

Because any given individual j is likely to have a different perception of the accuracy of statement i, we can treat a_{ij} as a Gaussian random variable. Thus, for a novel statement N, the expected value of a_{ij} is p_i, its distribution is

$$a_{ij} \sim \mathcal{N}(p_i, \sigma_N^2), \tag{3.3}$$

and σ_N^2 is its expected variance. Following repetition R of the same statement, the expected value of a_{ij} increases by f_i:

$$a_{ij} \sim \mathcal{N}([p_i + f_i], \sigma_R^2). \tag{3.4}$$

Note that the variance σ_R^2 of a_{ij} also may change after the statement has been repeated (that is, σ_N^2 does not have to equal σ_R^2).

How do we model f_i? Fazio et al. (2019) wanted to test the hypothesis that the illusory truth effect only applies to ambiguous statements, not demonstrably false or true ones. We first consider three alternatives to this hypothesis:

1. The effect of repetition on perceived accuracy should increase with plausibility;
2. The effect of repetition on perceived accuracy should decrease with plausibility;
3. The effect of repetition on perceived accuracy should be greatest for the most ambiguous statements (plausibility = 0.5).

For each of these three alternatives, f_i is expected to vary with p_i in some way. Fazio et al. (2019) modelled the first two as a linear increase or decrease with plausibility, where m (such that $m > 0$) is the strength of the effect of hearing a statement for the second and subsequent times:

1. $f_i = mp_i$;
2. $f_i = m(1 - p_i)$;

and the third as a symmetrical function peaking at $p_i = 0.5$:

3. $f_i = 2m\left(\frac{1}{2} - |\frac{1}{2} - p_i|\right)$.

Once we have articulated these three alternatives (and of course, there are many other ways that f_i could vary with p_i), the null expectation becomes clear:

0. $f_i = m$.

That is, if the repetition effect is independent of the plausibility of a statement, then repetition should have the same effect on the perceived accuracy of a statement no matter how implausible it is.

There is one more step to take. The illusory truth effect measures the change in an individual's assessment of the truth or falsity of a statement, not a change in its perceived accuracy. The change in assessment from true to false is like an on-off switch: the assessment is that a statement is either true or false. However, the perception of accuracy in our model is the continuous variable a_{ij} (Equations 3.2 and 3.4). Thus, for any individual, we need to convert their perception of accuracy (the continuous random variable a_{ij}) into a decision about truth or falsity (a binary random variable that can take on values of True $\equiv 1$ or False $\equiv 0$). We use a reasonable decision rule: if $a_{ij} > 0.5$ then the statement would be considered to be true. Otherwise, if $a_{ij} \leq 0.5$, then the statement would be considered to be false.

Now, we have all the pieces we need to simulate our four alternative hypotheses. We use Equations 3.2 and 3.4 and the decision rule "if $a_{ij} > 0.5$ then the statement is considered to be true" to simulate a population of individuals who determine the accuracy of statements with an initial range of plausibilities.

First, we define constants and variables that will be used in the simulation (Code Box 3.1).[5]

Our simulation (Code Box 3.2) uses the constants and objects defined in Code Box 3.1—the size N of the population to simulate (`N` = 100,000 in this example); the range of initial plausibilities (from obviously false [`plausibility` $= -1$] to clearly true [`plausibility` $= 2$]); the standard deviations ($\sqrt{\sigma_N^2}$ and $\sqrt{\sigma_R^2}$) of the normal distributions in equations 3.3 and 3.4 (which Fazio et al. 2019 set to the same value, $\sqrt{\sigma^2}$, and which we set as `sdev` = 0.5); and the repetition effect (the vector of `fluencies`)—to compute the

[5] Fazio et al.'s (2019) choices for a_i, p_i, and the decision rule differ from how we might have defined them. Specifically, it is a bit strange to use a plausibility score $p_i \in [-1, 2]$ rather than $p_i \in [0, 1]$ or to treat a_{ij} as a Gaussian random variable that is then converted into a True or False statement using a cutoff at 0.5, rather than a Bernoulli draw. These differences show that there is not always a single right way to define the truth in a data simulation.

Code Box 3.1: Simulate a population evaluating the accuracy of statements with different plausibilities (*part 1*)

```
# Following Fazio et al. (2019), we simulate a population of N/2 individuals
# assessing the plausibility of a novel statement and N/2 individuals
# assessing the plausibility of a repeated statement.
# This R code was converted from the Matlab code preregistered by
# Pennycook et al. (2019) at https://osf.io/w4k2c/

# Specify constants
 N <- 100000 # number of subjects
 sdev <- 0.5 # standard deviation of normal distribution from which accuracy
      scores will be sampled (Note that Fazio et al. [2019] call this "mu")

# Use a vector to specify three different levels of the effect of repetition on
     the plausibility of a statement.
# Fazio et al. call this variable "fluencies"
 fluencies <- c(.1, .2, .4)  #vector of fluencies

# Set up a dummy variable (a vector with length equal to our number of models
     for the effect of repetition on perceived accuracy. Fazio et al. (2019)
     call this dummy variable "skewes", whose values can be:
#      0: Model 0: constant repetition effect size
#     -1: Model 2: repetition effect size decreases with plausibility
#     +1: Model 1: repetition effect size increases with plausibility
#     +2: Model 3: repetition effect size is maximal at the midpoint of
     plausibility

 skewes <- c(0, -1, 1, 2)

# Define a finely divided range of initial plausibility ratings

 plausibility <- seq(-1,2,.01) #vector of plausibilities

# Define blank variables (arrays) for storing output
scoresR <- array(NA, c(length(plausibility), length(fluencies)+1, length(
    skewes)))
  dimnames(scoresR)[[3]] <- paste("RepEffect.",skewes, sep="")
  dimnames(scoresR)[[2]] <- c("Plausibility", paste("Fluencies.",fluencies,
      sep=""))
  scoresR[,1,] <- plausibility #first column is plausibility vector
scoresN <- array(NA, c(length(plausibility), length(fluencies)+1, length(
    skewes)))
  dimnames(scoresN)[[3]] <- paste("RepEffect.",skewes, sep="")
  dimnames(scoresN)[[2]] <- c("Plausibility", paste("Fluencies.",fluencies,
      sep=""))
  scoresN[,1,] <- plausibility #first column is plausibility vector
```

Code Box 3.2: Simulate a population evaluating the accuracy of statements with different plausibilities (*part 2*)

```
# Simulate the effect of repetition on perceived accuracty for N participants
    for the different models of effects of repetition as a function of
    plausibility and fluencies.

# This code uses the objects defined or created in Code Box 3.1:
# --The constants N and sdev
# --The vectors skewes, fluencies, and plausibility
# --The empty arrays scoresN and scoresR

set.seed(12345) # For reproducibility

for (i in 1:length(skewes)) {
  for (j in 1:length(fluencies)) {
    for (k in 1:length(plausibility)) {
           # Each of these if statements correspondes to a model of processing
               fluency as a function of plausbility.
      if (skewes[i]==0){  # Repetition effect is independent of plausibility
        fluencyEffect <- fluencies[j]
      } else if(skewes[i]==-1){ # Repetition effect decreases linearly with
          plausibility
        fluencyEffect <- fluencies[j] * (1 - plausibility[k])
      } else if(skewes[i]==1){ # Repetition effect increases linearly with
          plausibility
        fluencyEffect <- fluencies[j] * plausibility[k]
      } else if(skewes[i]==2){ # Repetition effect is a symmetrical function of
          plausiblity
        fluencyEffect <- fluencies[j] * 2 * (.5 - abs(.5 - plausibility[k]))
      }
      # Simulate N/2 (here, 50,000) novel subjects (hear a statement once):
        NFsample <- rnorm(N/2, plausibility[k], sdev) # If the statement is heard
             only once, its plausibility is set to the initial plausibility
        NFsample <- NFsample > 0.5  # The decision rule for TRUE
        scoresN[k,j+1,i] <- mean(NFsample) # mean value for each combination of
            skewes, fluencies, and plausibilities
      # Simulate N/2 (here, 50,000) repeated subjects (hear a statement twice):
        Fsample <- rnorm(N/2, (plausibility[k] + fluencyEffect), sdev) # If a
            statement is heard twice, its plausibility is modified by each of
            our four hypothetical relationships
        Fsample <- Fsample > 0.5 # The decision rule for TRUE
        scoresR[k,j+1,i] <- mean(Fsample)
    }
  }
}
```

effect of repetition of a statement on its perceived accuracy. This effect (equivalent to the variable f_i in equations 3.2 and 3.4) is computed in the variable `fluencyEffect` in Code Box 3.2 and refers to what psychologists call "processing fluency."[6]

3.3.4 What is the expected (hypothesised) relationship between our variables of interest?

Our first step was to draw a blank graph (Figure 3.1). The axes of that graph focused our attention on the key variables, how to quantify them, and how they might vary with our experiment (Equations 3.2–3.4). Now, we can use the results of our simulation (Code Box 3.2) to let us think about what the actual version(s) of our plot might look like.

Following Fazio et al. (2019), we considered three different values for the strength of the repetition effect m (the variable `fluencies` in Code Box 3.1) that relates f_i in Equations 3.2–3.4 (`fluencyEffect` in Code Box 3.2) to p_i (`plausibility` in Code Box 3.1) in each of our four hypotheses (coded with the dummy variable `skewes` in Code Box 3.1). These values span the range of fluency effects estimated from the meta-analysis published by Dechêne et al. (2010).[7]

The results of running these simulations for our four alternative hypotheses and three different values for m (the strength of the repetition effect) within each of the four hypotheses are shown in Figure 3.2. The top left panel of Figure 3.2 illustrates the unsettling results of the null expectation, $f_i = m$: even when the effect of repetition is independent of plausibility, there is an illusory truth effect—it is "hump-shaped" and symmetrical, with a maximum when the perceived truth is most ambiguous. This is because only around $0.5 \pm m$ will the threshold cause a change in expressed opinion between True and False.[8] Not surprisingly, repetition has less of an effect on more extreme statements (perceived truth equals 0 [absolutely false] or 1 [absolutely true]), but repetition still has a positive, non-zero effect.

3.4 Testing hypotheses and assessing their support

A testable hypothesis is a quantitative one, but quantitative doesn't mean complicated. For example, a simple, testable hypothesis of the illusory truth effect would be that there is no effect of repetition on an individual's decision as to whether a statement is true or false (H_0: $m = 0$) versus the alternative that there is some effect of repetition (H_A: $m \neq 0$). However, literature reviews, meta-analyses and syntheses, and especially simulations can help us make our hypotheses more precise.

Indeed, the results of the simulations shown in Figure 3.2 suggest two things that we should look for and test in a subsequent empirical study. First, the illusory truth effect should have a hump shape regardless of whether or not repetition equally increases the perceived truth of a statement. Second, the value of the perceived truth of a statement that gives the strongest illusory truth effect (i.e., the "location" along the x-axis of the hump) indicates

[6] Dechêne et al. (2010, p. 238) noted that researchers who study the illusory truth effect generally agree that it is mediated by *processing fluency* (italics in the original): "processing a repeated statement is experienced as unexpectedly fluent—in other words, the processing fluency is experienced as discrepant from a comparison standard—and this discrepancy subsequently affects judgments of truth." Fazio et al. (2019, p. 1705) restated this idea more succinctly: "repeated statements are more easily processed and understood, and this processing fluency is used as a signal for truth." See also Unkelbach (2007); Unkelbach & Greifeneder (2018).

[7] You can change the values of `fluencies` and `plausibilities` in Code Box 3.1 to explore other areas of the parameter space with your own simulations. You could also consider and simulate hypotheses that have different and even nonlinear relationships between f_i and p_i.

[8] Note that this would be less problematic with a Bernoulli random draw, but the results would be qualitatively similar.

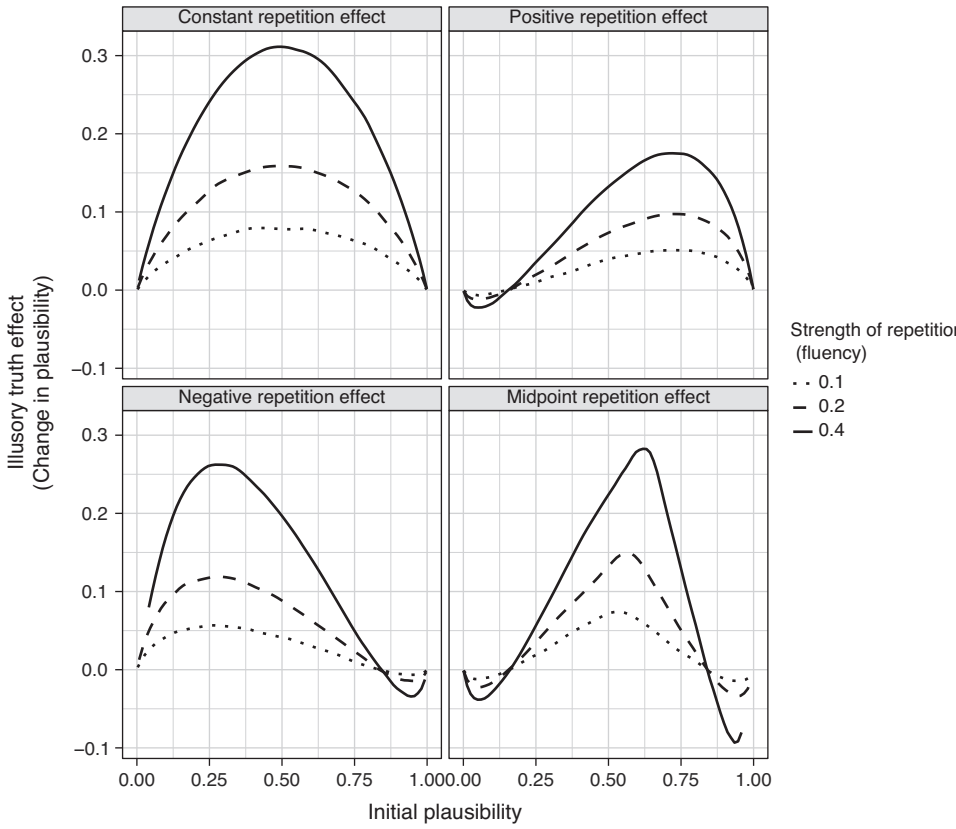


Figure 3.2 Results of simulating the illusory truth effect as a function of plausibility (p in Equations 3.2–3.4) for three different strengths of repetition (m) and four different hypothesised ways (f_i) that repetition could change the perception that a statement is true. The top left panel is the simulated null model, in which f_i is independent of p_i (model 0: $f_i = m$). The other three panels illustrate the three different alternatives. When the effect of repetition on perceived accuracy increases linearly with plausibility (model 1: $f_i = mp_i$; top right panel); decreases linearly with plausibility (model 2: $f_i = m(1 - p_i)$; lower left panel); or is largest for the most ambiguous statements (model 3: $f_i = 2m\left(\frac{1}{2} - |\frac{1}{2} - p_i|\right)$; lower right panel).

the type of relationship between repetition and plausibility (i.e., which of the four models is likely to be true).

Now that we have a question (§3.2), a model (Equations 3.2–3.4 in §3.3.3), and alternative testable hypotheses (§3.3.4), we are ready to design the study to test them. Identifying the key parameters—those that define the shape and strength of the illusory truth effect—and testing the hypothesis that those parameters are significantly different from zero (i.e., no effect) requires a replicated study design that minimises to the extent possible the probability (actually, two probabilities) that we will reach an incorrect conclusion.

3.4.1 A four-fold path to adequate replication

Most experiments and observational studies have a control group corresponding to the null hypothesis (in which no treatment is applied or the effect of interest is not present) and one or more treatment groups corresponding to the alternative hypotheses (to each of which a different treatment is applied or the effects of interest are present). There are literally hundreds of different kinds of sampling and experimental designs (e.g., Thompson 2012; Montgomery

2020), but all of them require more than one observation (or replicate) in each treatment group. The question is, how many replicates do we need, and why do we need so many?

More specifically, the actual question we need to answer in designing our study is how much replication is needed for us to be able to detect a statistically significant effect, *if it exists*. If the sampling effort is too limited, the number of samples too small, or the accuracy and precision of the measurements too low, the study may have inadequate statistical power to detect an effect. So, to answer our question, we first need three other pieces of information: how large an effect of our treatments do we expect to see; and given that we see an effect, how much confidence do we want to have that our results are neither false positives (and statistically significant) nor false negatives (and not statistically significant)?

ESTIMATING AND ANTICIPATING THE EFFECT SIZE

We expect that our (experimental) treatment will have an effect on a variable of interest and that the effect will differ between the treatment and the control (or among the different treatments). These effect sizes can either be raw (i.e., the actual, observed values), or standardised (Borenstein 2009). The former can be compared only if the exact same response variable was measured (e.g., systolic blood pressure or carbon-dated age of an artefact). In all other cases, and most notably in meta-analysis (Chapter 10), only standardised effect sizes can be compared or aggregated across studies.

Although the list of possible (un)standardised effect sizes is very long (Wikipedia Contributors 2023), all of them fundamentally quantify the relationship between two variables of interest. Such relationships can be expressed by a difference, a distance, an odds ratio, or a correlation coefficient, and they can be computed for continuous and discrete variables (Borenstein 2009; Fleiss & Berlin 2009).

The most basic effect size is the difference Δ between the means of two variables that measure the same response on the same scale and with the same instrument:

$$\Delta = \mu_1 - \mu_2, \tag{3.5}$$

where μ_1 and μ_2 are the true means of random variables y_1 and y_2. Of course, we rarely know the true values of these random variables, so we estimate Δ from the sample means:

$$D = \overline{Y}_1 - \overline{Y}_2. \tag{3.6}$$

Because Δ and D can, in theory, take on any value in $(-\infty, \infty)$ and because we are often interested in comparing effect sizes across studies, we generally report the standardised effect size (which also makes meta-analysis—Chapter 10—considerably simpler). For a comparison between groups, the most common standardised effect size measures the difference between groups in multiples of the within-group standard deviations.[9] The true standardised effect size δ is

$$\delta = \frac{\mu_1 - \mu_2}{\sigma}, \tag{3.7}$$

and its estimate based on the sampled data is

$$d = \frac{\overline{Y}_1 - \overline{Y}_2}{s_{\text{within}}}, \tag{3.8}$$

[9] Note the similarity here with the difference computed for a *t*-test. However, the *t*-test quantifies the difference in multiples of the standard error of the estimated difference, and hence depends on the sample size. In contrast, the (standardised) effect size is independent of the sample size.

Table 3.1: Correct inference and possibilities of type-I and type-II errors. A well-designed experiment will simultaneously minimise the probabilities of the two types of errors.

	Study result	
"Reality"	*No significant effect observed*	*Significant effect observed*
H_0 *is true*	Correct inference	False positive (type-I error), α
H_0 *is false*	False negative (type-II error), β	Correct inference

where

$$s_{\text{within}} = \sqrt{\frac{(N_1 - 1)s_1^2 + (N_2 - 1)s_2^2}{N_1 + N_2 - 2}}, \tag{3.9}$$

s_1 and s_2 are the sample standard deviations, and N_1 and N_2 are the sample sizes.[10]

Standardised effect sizes can be computed for many different experimental designs, not only for mean effects within an experiment but also (when comparing across studies in a meta-analysis) for summary statistics, such as the familiar t, F, and p (Borenstein 2009). In addition, when comparing two continuous variables using correlation or regression, the sample estimate r of the correlation coefficient ρ can be used as an estimate of the effect size.[11]

All of the preceding discussion presupposes that the experiment has already been done, so that you have the samples and results from which to compute mean responses, their variances, and relevant test statistics. But we have not yet done the experiment. Rather, we are still determining sample sizes and need an estimate of the effect size that we expect see and, to compute the standardised effect size that we will need to determine our sample size, the expected variances. Where might those numbers come from? Literature reviews, pilot studies, and simulations!

POTENTIAL ERRORS IN HYPOTHESIS TESTING

The next two steps on our path to adequate replication are to determine critical values for false positives and false negatives. We would like to take both of these steps at once and have our design simultaneously satisfy two criteria first identified by Neyman & Pearson (1933). First, it should keep the probability of getting a false positive result (what we call a type-I error) within acceptable limits. Second, our statistical power should be large enough so that we have a high probability of avoiding a false negative (what we call a type-II error) (Table 3.1).

Researchers normally fix the acceptable or "critical" level of a false positive (denoted by α) to be equal to 0.05. The familiar p-value that is calculated for a statistical test and reported in many scientific papers is the probability of committing a type-I error. If $p \leq \alpha$, we reject our null hypothesis of no effect and assert that our results are statistically significant.[12]

[10] The standardised effect size d (Equation 3.8) is often referred to as Cohen's d, and it is slightly biased, especially for small sample sizes. The bias-corrected estimate, called Hedges' g, is equal to $d \cdot J_{\text{df}}$, where $J_{\text{df}} = 1 - \frac{3}{4\text{df}-1}$ and df equals the degrees of freedom used to estimate s_{within} in Equation 3.9 (Borenstein 2009). Because $J_{\text{df}} < 1$, $|g| < |d|$. But in practice, J_{df} will be nearly equal to 1.0 as long as df is even moderately large (i.e., ≥ 10), so in practice, there will be little difference in results of using d or g as an estimate of the standardised effect size, especially in meta-analyses of many studies (see McGrath & Meyer 2006, for a detailed comparison).

[11] In meta-analysis, Fisher's z-score is used in place of the sample correlation coefficient r as a measure of the standardised effect size: $z = 0.5 \cdot \ln\left(\frac{1+r}{1-r}\right) = \text{atanh}(r)$ (Borenstein 2009). See Chapter 10 for additional discussion.

[12] Note that this assertion implicitly assumes that if p is low, then the probability of observing an effect of our experimental treatment, given that it actually exists (the top left cell in Table 3.1), is high and equal to at least $1 - \alpha$.

Although researchers commonly use the $p \leq \alpha = 0.05$ rule for statistical significance, there is no similarly agreed-upon critical level for β, the acceptable level of obtaining a false negative (the upper right cell in Table 3.1). Many researchers follow Cohen (1965) and set $\beta = 0.2$ (a 20% false negative rate) but rather than reporting β directly, they report the "power" of the study, where power = $1 - \beta$.[13]

RELIABLE HYPOTHESIS TESTING WITH ACCEPTABLE STATISTICAL POWER REQUIRES MORE REPLICATES THAN YOU MIGHT EXPECT

Given an estimated effect size, a pre-defined critical level α, and a pre-defined statistical power $1 - \beta$, we can compute the sample size required for our experiment that will result in a true positive or negative result for an expected effect size (and its variance). In fact, if we know any three of these (e.g., sample size, effect size, α), we can determine analytically the fourth (in this example, power, $1 - \beta$) for most relatively common study designs and their associated test statistics (Cohen 1988). But there is no general equation that will work for all designs and all test statistics.

By way of an example, consider what is perhaps the simplest statistical test: the *t*-test. It may come as a surprise that we need a sample size close to 400 *in each group* to attain

---

Many scientists and virtually all non-scientists would then estimate the probability that the actual effect exists to be $1 - p$ (or at least $1 - \alpha$). In fact, as has been pointed out repeatedly in the literature, this is rarely, if ever, the case (e.g., Berger & Selke 1987; Howson & Urbach 1991). Rather, the probability that an actual effect exists may be as much as an order of magnitude smaller. The statistical reason for this is that the *p*-value is computed from a cumulative probability function of the test statistic, e.g., $p(t \geq t_\alpha)$, and the subsequent assertion that, for example, $p < \alpha = 0.05$ is mute about whether $t \approx t_\alpha$ (and hence $p = 0.05$) and $t \gg t_\alpha$ (in which case $p \ll 0.05$). Specifying an exact *p*-value does not ameliorate this problem because doing so misinterprets the relationship between p and α (the a priori decision rule for committing a type-I error). For further discussion, see Greenland et al. (2016) and Wasserstein & Lazar (2016). Finding a new study that does not detect the pattern can be expected even for small *p*-values (Edwards et al. 1963; Berger & Selke 1987). This fact may contribute to the perception that fields such as medicine and psychology are enmeshed in a reproducibility crisis (e.g., Ioannidis 2012; Gelman & Carlin 2014; Goodman et al. 2016; Szucs & Ioannidis 2017; Nosek et al. 2022).

[13] You might wonder why the normally acceptable critical value for α, 0.05, is so much lower than the normally acceptable critical value for β, 0.20. Neyman & Pearson (1933) discussed this point extensively and their reasoning has been followed implicitly ever since. They started with the premise that there is one null hypothesis (H_0) but many alternatives (H_i). If H_0 is falsely rejected (type-I error), the researcher has made a mistake that will be the same regardless of which H_i is true. In contrast, if one of the many possible H_is are rejected, the mistake will not be the same as if another H_i is rejected. Hence, the critical value for a type-I error should be more stringent than the critical vale for committing a type-II error.

Without citing Neyman & Pearson (1933), Cohen (1965, p. 98) followed their reasoning in suggesting a "conventional" value of $\beta = 0.20$ "*when no other basis is available* [and that] like all conventions, this value is arbitrary, but *it is, I believe, reasonable*" [emphasis added]. Cohen (1965) accepted the conventional $\alpha = 0.05$ critical level for a type-I error and then went on to suggest that it was about four times worse than committing a type-II error. Although Cohen (1965, p. 99) added that he "would have no serious quarrel with anyone who claimed that the factor should be three or five (or even two or six)," no one seems to have taken up that cudgel.

The rationale for down-weighting the value of type-II errors becomes more apparent when discussing these errors in the capitalist marketplace. There, type-I errors are called "producer" errors and type-II errors are called "consumer" errors. For example, a company producing a new drug and bringing it to market does not want to assert an effect of the drug when it does not actually exist (a false positive in which a true H_0 is asserted to be false). This error would affect every consumer, so the critical value to the drug producer for falsely rejecting H_0 should be stringent. The consumer, on the other hand, doesn't want to be taking a drug for which the company asserted there was no negative side-effect, but in fact it had such side-effect—on that particular consumer. Because potential effects among different consumers are likely to vary greatly, the financial impact on the producer of making a type-II error is much less than the impact of making a type-I error. That is, the producer can afford a larger critical value for making a type-II error. On the other hand, the precautionary principle advocates minimising type-II errors while caring less about type-I errors. The ethics of the relative weighting of type-I and type-II errors have been discussed extensively by philosophers (e.g., Shrader-Frechette 1988).

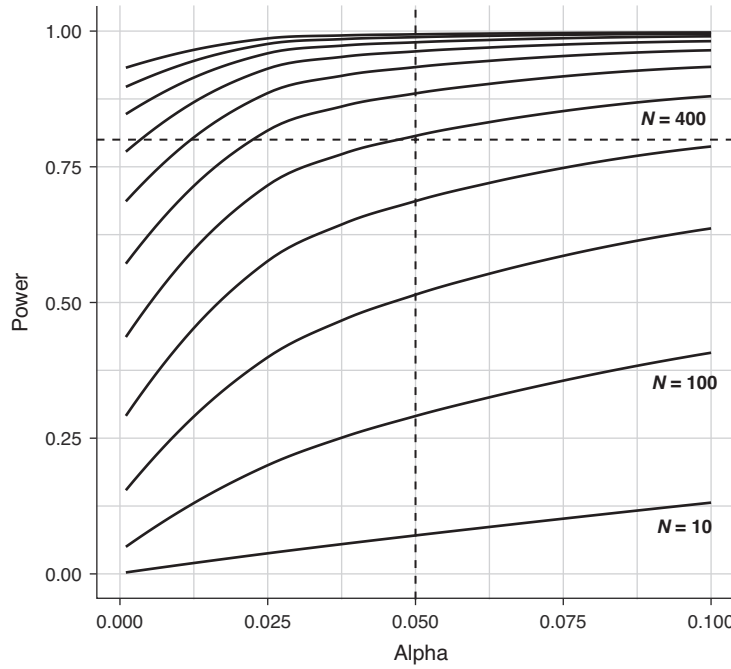


Figure 3.3 The relationship between the acceptable level of a type-I error (α; vertical dashed line) and statistical power ($1 - \beta$; horizontal dashed line) of a t-test for sample sizes ranging from 10 to 1,000 in intervals of 100 and a standardised effect size of 0.2 (i.e., $\frac{\overline{Y}_1 - \overline{Y}_2}{s} = 0.2$, where $\overline{Y}_i$ is the sample mean of the ith group and s is the pooled sample standard deviation).

both a low (5%) false-positive rate and relatively high (80%) power for the two-sample t-test when testing the hypothesis that two groups differ by 20% in their response to a given experimental treatment (Figure 3.3; Code Box 3.3). Note that the results shown in Figure 3.3 are not from a simulation. That is because in this simple example, the prospective or a priori power of an unpaired, two-sample t-test, like that of other common statistical tests (e.g., one-sample and paired t-tests, the chi-square and other tests for frequency tables, and one- and two-way ANOVAs), can be computed exactly, given the sample size, effect size, and the fixed acceptable level of a type-I error, α (Cohen 1988). However, simulations are needed to estimate the prospective power of most other statistical tests (Chapter 4).

These kinds of calculations highlight that if there are not enough resources for an adequate sampling effort, to make accurate and precise measurements, or to run the experiment for long enough, the study will be underpowered and doomed from the start. But expensive need not mean only the billions of dollars spent on space exploration programmes, multinational particle accelerators, decades-long longitudinal studies, or "warp-speed" vaccine development. It can also mean that years of laborious post-graduate research efforts were done for no good reason—a loss keenly felt by the researchers themselves.

3.4.2 Sample size and the power to detect and measure the illusory truth effect

REPRISE: WHAT WAS OUR QUESTION?

We now return to our case study. Following Fazio et al. (2019), we are interested in predicting the direction of the illusory truth effect as a function of the perceived truth of a statement (Figures 3.1 and 3.2). We ask, which of the hypotheses (H_0 to H_4) illustrated

Code Box 3.3: Compute the power for a two-sample *t*-test for a range of sample sizes

```
# Compute the power of a t-test for sample sizes 10, 100, 200, ..., 1000 and
    critical levels alpha from 0.001 to 0.1
 library(pwr)  # Load required library

 df <- data.frame(matrix(ncol = 4, nrow = 0))
 for (n in c(10, seq(100,1000,100))){
  a <- pwr.t.test(d=.2, n=n, sig.level=seq(.001,.1,.001), type="two.sample",
       alternative="two.sided")
  b <- cbind(a$d, a$n, a$sig.level, a$power)
  df <- rbind(df,b)
}
```

in Figure 3.2 is best supported by the data? The key parameter of interest is the value along the *x*-axis where the peak of the relationship between the illusory truth effect and the statement's perceived truth occurs. Whether we use a linear regression model with a quadratic term or a local regression model (e.g., loess, splines), we would use the location (*x*-value) of the local maximum (*y*-value) of the fitted model as our estimate of this key parameter. If the null hypothesis—that the illusory truth effect is independent of the perceived truth of a statement—is true, we expect the location of the peak of the relationship to be ≈ 0.5.

Secondarily, Fazio et al. (2019) also tested for what they called an "overall illusory truth effect"—that, regardless of their plausibility, repeated statements were more likely to be considered to be true than statements heard only once. This question looks at the magnitude of the illusory truth effect. To test it, they used a paired *t*-test for which each observation was a single participant in the study and the two variables were the proportion of statements heard only once ("novel" statements) that each participant rated as true and the proportion of statements heard twice ("repeated" statements) that the same participants rated as true. The standardised difference of the means of these two variables is the overall effect size of the illusory truth effect.

Sample sizes

Fazio et al. (2019) presented a set of 80 true/false questions with different degrees of plausibility[14] to each of 546 participants (of which 503 completed the study). The 80 questions were divided into two sets ("A" and "B"); half of the participants (randomly assigned) saw Set A as the novel set of questions and Set B as the repeated set of questions, and the other half of the participants saw Set B as novel and Set A as repeated. Were these sample sizes appropriate? Given an estimated effect size and conventional values of $\alpha = 0.05$ and $\beta = 0.2$ (i.e., power $= 0.8$), we can easily determine the required sample size for the overall illusory truth effect, but not so easily, if at all, for the location of the plausibility of a statement at which the illusory truth effect is maximised.

For the overall illusory truth effect, the prior estimate of the standardised effect size was 0.49 (95% CI = [0.45, 0.55]; Dechêne et al. 2010).[15] This is a "medium" effect size for a

[14] The plausibility of each statement was rated by individuals in a previous, unpublished study; the complete set of questions is provided in Pennycook (2019), available at https://osf.io/8qcy3.

[15] These values, from Table 1 of Dechêne et al. (2010), are for between-item assessments in which judgements of the truth of a set of repeated statements are compared to judgements of a different set of novel statements.

t-test.[16] The corresponding estimated sample size needed to detect a standardised effect size of 0.49 would be 35 pairs.[17] So, the ≈ 500 participants used by Fazio et al. (2019) should have been more than adequate to detect the expected effect size; in fact, their sample size should have been able to detect an effect size of 0.12.[18] Indeed, Fazio et al. (2019) reported that 52% of repeated statements and 48% of the novel statements were judged to be true by the 503 participants. These values are significantly different from one another ($t_{502} = 9.19$, $p < 0.001$) and, given the sample standard deviations of 0.161 and 0.164 for the two variables[19] and Equations 3.8 and 3.9, we estimated $d = 0.24$ for the overall effect size of the illusory truth effect. This is half the standardised effect size for the illusory truth effect reported by Dechêne et al. (2010), but still easily detectable give the sample size.

However, we do not have enough information to determine the sample size we need to detect a specific plausibility. Plausibility is the predictor variable from which the illusory truth effect is estimated. As is normally the case in regression models, the predictors are assumed to be measured (or defined) without error (but see §5.2 in Chapter 5) and in fact there is no information on the variability of the plausibility of each of the 80 questions.[20] You might think that if we could compute the effect size then we would be able to identify the location (x-value) where plausibility = 0.5, given $\alpha = 0.05$, power = 0.8, and a sample size of 80 questions. Indeed we can (it is $d = 0.44$ for a two-tailed t-test), but that result assumes that we will be comparing samples of plausibilities from two normally distributed populations. Is that a reasonable assumption? We do not know.

AFTER THE STUDY—FOR THE SAKE OF COMPLETENESS

Finally, we return to the key question of interest: the directional relationship between the illusory truth effect and the perceived truth of a statement. The points and solid black curve in Figure 3.4 reproduce the results reported by Fazio et al. (2019); the points in Figure 3.4 represent the illusory truth effect for each of the 80 questions averaged over all 503 participants and the solid black curve is the best-fit quadratic regression to these data. The dashed curve and the shaded confidence regions are our additions to their figure (Code Box 3.4).

A visual comparison of Figures 3.2 and 3.4 suggests that the scaling constant m relating the effect of plausibility p on processing fluency f is small, on the order of 0.1, and that the magnitude of the illusory truth effect for each question (≈ 0.05 at the peak of the curve) is similar in size to the overall illusory truth effect observed across all the participants (0.04). This is comforting, but we are really interested in *where* along the x-axis that peak occurs,

---

The standardised effect size for within-item assessments—the effect of repetition on the judgement of truth of a novel statement compared to the judgement of truth of the same statement after it has been repeated—is somewhat smaller (0.39; 95% CI = [0.32, 0.47]).

[16] Cohen (1988) distinguished "small," "medium," and "large" effect sizes detectable using a t-test, corresponding to standardised effect sizes of 0.2, 0.5, and 0.8, respectively.

[17] Using the `R` function `pwr.t.test(n = NULL, d = .49, sig.level = 0.05, power = .8, type = "paired")` from the `pwr` library. We note that most statisticians assert that it is better to use the actual effect size Δ (Equation 3.5) or D (Equation 3.6) rather than Cohen's d (Equation 3.8) when computing prospective power (e.g., Lenth 2007; Gelman & Carlin 2014). Correll et al. (2020) argued that "T-shirt sizing" of effect sizes should be abandoned as arbitrary, inconsistent, and illogical. Cohen (1988) himself said that such categorisation of effect sizes should be done only as a last resort and only if no other option (e.g., prior studies, meta-analysis, or systematic reviews) were available.

[18] Use the same `pwr.t.test()` function as in footnote 17, but with `n = 500 and d = NULL`.

[19] Fazio et al. (2019) did not report these sample standard deviations, so we calculated them from the `full range.csv` dataset provided by Pennycook (2019).

[20] As noted above in footnote 14, the study from which the questions were drawn is unpublished. Further, the plausibilities were grouped into deciles and questions selected from within the deciles.

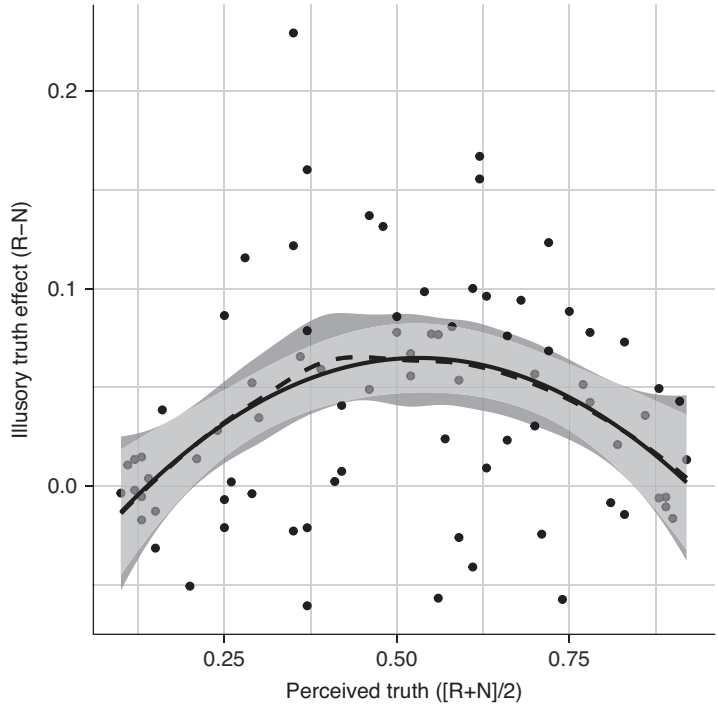


Figure 3.4 Relationship between the perceived truth (average proportion of statements rated "true" by $N = 503$ individuals) and the strength of the illusory truth effect. Data from Pennycook (2019). The solid black line (with 95% confidence interval in light grey) is the quadratic regression as fit by Fazio et al. (2019), and the dashed black line (with 95% confidence interval in dark grey) is a loess curve with span = 0.75 of the data.

Code Box 3.4: Fitting and plotting curves and their confidence intervals to the illusory truth dataset

```
# Load required libraries
 library(ggplot2)

# Read in the data (this is a local copy of the data file downloaded from https
    ://osf.io/9zjre)
 statement.effects <- read.csv("../Data/FazioExperiment/StatementEffects.csv")

# Plotting curves and associated confidence intervals is done quickly and easily
     with ggplot

 ggplot(statement.effects, aes(x=perceived_truth, y=illusory_truth_effect)) +
    geom_point() +
    geom_smooth(color="black", lty=2, fill="black") + # adds a loess curve with
        confidence interval
    geom_smooth(color="black", method=lm, formula=y ~poly(x,2), fill="white") +
        # Adds a quadratic regression with confidence interval
    geom_smooth(color="black", lty=2, se=FALSE) + # Re-plots the loess curve to
        make it visible on top
    labs(x="Perceived truth ([R+N]/2)", y="Illusory Truth Effect (R-N)") + #
        Labels the axes
    theme_minimal()
```

Code Box 3.5: Identifying locations of peaks of curves

```
# We use the same dataset -- statement.effects -- as in Code Box 3.4

# First, find the location of the peak of the quadratic regression model
 lm.out <- lm(illusory_truth_effect~poly(perceived_truth, 2, raw=T), data=
      statement.effects)
# Get the predicted values using the predict() function.
 perceived_truth <- seq(min(statement.effects$perceived_truth), max(statement.
      effects$perceived_truth), length=1000)
 pred <- data.frame(perceived_truth, illusory_truth_effect=predict(lm.out,
      newdata=data.frame(perceived_truth)))

# Since this is a parabola, all we need is the x value associated with the
     maximum y-value
 max_y <- max(pred[2])
    max_x <- pred[pred[2]==max_y,1]
    max_x

| [1] 0.53

# Second, find the location of the peak of the loess model
 loess.out <- loess(illusory_truth_effect~perceived_truth, data=statement.effects
      , degree=2, span=.75)
 first_diff <- diff(loess.out$fitted)     # diff() calculates differences between
       consecutive elements in a vector
# Extract the corresponding x and y values for the minima (and maxima)
 res <- cbind(loess.out$x[c(which.min(first_diff), which.max(first_diff))], loess
      .out$fitted[c(which.min(first_diff), which.max(first_diff))])
 colnames(res) <- c("x", "y")
 rownames(res) <- c("min", "max")
 max_x <- res[1,1] # We only need the x value associated with the minimum y value
       because that's where the difference = 0
 max_x

| [1] 0.37
```

not how high it is. We can easily identify the location of the peak by finding where along these polynomial curves the slope equals zero (Code Box 3.5).

The locations of the peaks differ between the quadratic regression (0.53) and the local regression (0.37). The former is close to that expected by the null model (H_0), whereas the latter would suggest H_2, a negative effect of repetition. We next ask, which model(s) do these estimates actually support and do they differ from one another?

Fazio et al. (2019) used bootstrapping to determine the 95% CI on the perceived truth value for which the illusory truth effect was maximised.[21] We did the same, for both the quadratic regression and the loess smooth (Code Box 3.6).

[21] See §2.3.3 in Chapter 2 for a detailed discussion of bootstrapping.

Code Box 3.6: Bootstrapped confidence intervals of locations of peaks of curves

```
# Load required libraries
 library(boot) # We use R's boot library for fast bootstrapping

# First, we write a function for fitting the quadratic regression model and
    extracting the location of its peak
 peak_function.lm <- function(formula, data, indices) {
    d <- data[indices,] #indices is the number of bootstrap replicates
    lm.out <- lm(formula, data=d)
    perceived_truth <- seq(min(d$perceived_truth), max(d$perceived_truth), length
        =1000)
    pred <- data.frame(perceived_truth, illusory_truth_effect=predict(lm.out,
        newdata=data.frame(perceived_truth)))
    max_y <- max(pred[2])
    max_x <- pred[pred[2]==max_y,1]
    return(max_x)
}

# We also write a function for fitting the loess model and extracting the
    location of its peak
 peak_function.loess <- function(formula, degree, span, data,indices) {
    d <- data[indices,]
    loess.out <- loess(formula, data=d, degree=degree, span=span) #includes
        options to vary degree and span of the loess model
    first_diff <- diff(loess.out$fitted)
    max_x <- loess.out$x[which.min(first_diff)]
    return(max_x)
}

# Bootstrap the models using the same dataset as in Code Box 3.4
 statement.effects <- read.csv("../Data/FazioExperiment/StatementEffects.csv")

# Bootstrap (5000 replicates) the quadratic regression model with the
    peak_function.lm() function
 set.seed(1234)
 reps.lm <- boot(data=statement.effects, statistic=peak_function.lm, R=5000,
     formula=illusory_truth_effect~poly(perceived_truth, 2, raw=T))

#Bootstrap (5000 replicates) the loess model with the peak_function.loess()
    function
 set.seed(1234)
 reps.loess <- boot(data=statement.effects, statistic=peak_function.loess, R
     =5000, formula=illusory_truth_effect~perceived_truth, degree=2, span=.75)

#Print 95% bias-corrected confidence intervals (we use bias correction because
    of the small number of observations (80))
 lm.ci <- boot.ci(reps.lm, type="bca")
 loess.ci <- boot.ci(reps.loess, type="bca")
```

```
 matrix(rbind(unlist(lm.ci)[c(1,2,7,8)], unlist(loess.ci)[c(1,2,7,8)]),
     nrow=2, dimnames = list(c("lm", "loess"), c("Reps", "Estimate", "Lower",
40   "Upper")))

#Extract the relevant values from boot.ci output

|       Reps Estimate  Lower  Upper
45 | lm    5000 0.53      0.49   0.59
| loess 5000 0.37      0.25   0.41
```

The bootstrapped 95% confidence intervals suggest that there is a difference in the location of the peak illusory truth effects predicted by the quadratic regression (which supports H_0: the illusory truth effect is independent of the plausibility of the statement) and the loess model (which supports H_2: the illusory truth effect is higher for implausible statements). Although the sample size of participants (503) was more than adequate to detect even a "small" *overall* illusory truth effect, 80 questions likely were insufficient to determine the shape, peak, and location of the peak of the curvilinear relationship between plausibility and the illusory truth effect (Figure 3.4). Indeed, to get the smooth curves shown in Figure 3.2, the initial simulations used 300 statements with differences in plausibility between the questions = 0.01 (Pennycook 2019). We leave it as an exercise for the reader to modify the code in Code Boxes 3.4–3.6 to develop a set of simulations to determine how many questions would be needed to have confidence in choosing among the alternative hypotheses.[22]

3.4.3 Significance and power are not the only error rates of interest

In many applications, including medical diagnoses and industrial quality-control, practitioners often are more interested in the sensitivity and specificity of a test than in the probability of type-I or type-II statistical errors. The sensitivity of a diagnostic or quality-control test is equal to the number of true positives divided by the sum of the number of true positives plus the number of false negatives. That is, a test is considered to have high sensitivity if, for example, it correctly detects that a patient has a disease *when in fact they*

[22] Fazio et al. (2019) incorrectly suggested that their conclusions would would have been stronger if they had analysed responses for each participant rather than by averaging responses to each of the 80 statements across all the participants. Pennycook (2019) in an addendum to their pre-registration document suggested that the variance was "large" among the $N = 503$ individuals, so they planned to repeat the experiment with > 1,000 individuals and use a sequential analysis. They wrote (at https://osf.io/vc9k2):

> After analyzing the results from our original planned sample (500 in the pretest, 364 in the exp), we realized that the data was far noisier than we expected. Thus, we need a substantially larger N to detect any meaningful patterns. Rather than discard the data we have already collected, we decided to use a sequential analysis approach to increase our N while adjusting our alpha for the final analysis to avoid an inflated error rate. Our new recruitment goal for the initial pretest is 1,500 participants. We expect a similar return rate as the first round of recruitment which will leave ≈1,090 participants in the final sample. Thus, we have completed our initial analysis with 33% of the final sample and will repeat the analysis with 100% of the final sample. Using a linear spending function (power family function) for a two-sided analysis controlling for alpha of .05 (estimated using the `alpha*t^phi` function with `phi` = 1 found in the `GroupSec` package for `R`; see Lakens 2014); the alpha boundary for the final analysis is 0.038.

The repeated analysis has not yet been published, but there is no reason to think that the results would be an improvement over a study using more questions. There also are additional perils to using sequential analyses and associated stopping rules, which we discuss in detail in §4.3 of Chapter 4.

have it. In contrast, the specificity of a test is equal to the number of true negatives divided by the sum of the number of true negatives and the number of false positives. That is, a test is considered to have high specificity if the test does not detect the disease when in fact the patient is well. Although there is an apparent similarity between sensitivity and power, and between specificity and a *p*-value (= the probability of committing a type-I error), there are important differences. Most importantly, sensitivity and specificity are post hoc calculations derived from empirical data, whereas prospective power is derived analytically (or estimated through simulation) before any data are collected. See Chapter 9 for additional discussion of this topic. Conflating specificity and *p*-values, or sensitivity and power, is also at the root of the "prosecutor's fallacy" (see Chapter 5, §5.5 and footnote 18 in that chapter).

3.5 Pre-registration

If we have gone through all the steps outlined in the previous sections—posing a question and a testable hypothesis, creating blank figures and tables to identify the key variables and their relationships, systematically reviewing and synthesising existing work to set (prior) expectations or probabilities and plausible effect sizes, using simulations to winnow possible sampling or experimental designs, and running a prospective power analysis—the next step should be to do the project, collect the data, analyse them, test the hypothesis, and publish the work, right? Not so fast. The last thing to do before turning the key is to pre-register the study design and the plan for analysing the data.

3.5.1 Questionable research practices and other potholes on the road to publication

A major risk to drawing credible conclusions from inferential statistics is HARKing—Hypothesizing After the Results are Known (Kerr 1988)—which results from the so-called "researcher degrees of freedom" (Simmons et al. 2011, p. 1359):

> In the course of collecting and analyzing data, researchers have many decisions to make: Should more data be collected? Should some observations be excluded? Which conditions should be combined and which ones compared? Which control variables should be considered? Should specific measures be combined or transformed or both?
>
> It is rare, and sometimes impractical, for researchers to make all these decisions beforehand. Rather, it is common (and accepted practice) for researchers to explore various analytic alternatives, to search for a combination that yields "statistical significance," and to then report only what "worked."

If we do not articulate our hypothesis before doing the study and analyse it in a way that would preclude changing the analysis afterwards, we might think we are free to explore many different hypotheses and analyses with our data. Such practices, although not necessarily ill-intentioned or malicious, greatly inflate the probability of a type-I error: erroneously finding a significant effect (Table 3.1). Why is that? Because the probability that more than one analysis done *on the same dataset* will produce a false-positive finding at a given critical level α is necessarily greater than α. Indeed, using simulations, Ioannidis (2005) argued that the results of more than half of all scientific publications are "false" and that the combination of HARKing and multiple testing is the primary cause of these falsehoods.

HARKing is but one contributor to the reproducibility crisis (see footnote 12, above, and Ioannidis 2012; Nosek et al. 2022). Similar questionable research practices ("QRPs") include: presenting only the evidence that supports one's favoured hypothesis while leaving

the rest unreported ("cherry-picking"); the deliberate and time-consuming repeated analysis of a dataset with many different statistical methods with the goal of obtaining a statistically significant result, which is then the only one presented in the paper ("*p* hacking"); testing every conceivable association between as many different combinations of observed and measured variables as possible in the hopes that something "significant" will emerge ("fishing expeditions"); and extensively testing and non-parametrically modelling as may of these relationships as possible among the dozens to thousands or more variables found in enormous datasets that have been scraped from databases, especially when done without regard to their quality or commensurability ("data dredging" or "data mining"). Rubin (2020) and Andrade (2021) provide thorough reviews of QRPs.

3.5.2 Pre-registration helps avoid QRPs

A common and effective prescriptive remedy for QRPs is the pre-registration of the design and expected analysis of a research project (Nosek et al. 2018). Pre-registration refers to the formal documentation and posting in an approved registry of the specifications of the hypothesis of interest and the precise methods of how it will be tested.[23] Typical details of a pre-registration document include how the data will be collected, checked for quality ("cleaned"), imputed (to deal with missing values or incomplete observations), and mathematically transformed (only if required by the specified analysis), and the analytical method to be used to test the hypothesis. The last item should include a detailed specification of the statistical model, its assumptions, prospective power, and the software (and its version) that will be used on a particular computing platform and operating system.

Pre-registration has been a feature of clinical trials in medicine for many years,[24] and increasingly is being advocated in other fields (Purgar et al. 2024). Although there are salient critiques of the value of pre-registration on both technical (Lakens 2019) and philosophical (Rubin 2020) grounds, the tide of public opinion is strongly in favour of it. Many journals accept pre-registered methods and protocols for peer review before the study is actually done, and if the peer review is successful, commit (at least provisionally) to accepting the paper describing the results of the "Registered Report."[25]

Although Fazio et al. (2019) did include simulation code in their pre-registration document (Pennycook 2019), this is not common practice. There are two challenges to the inclusion of simulations in a pre-registration document. First, how do we decide which previous studies are similar enough to provide insight into our simulations? Second, how do we to turn information in these studies into priors for our simulations? Both questions are foci of meta-analyses and require formulating inclusion criteria (to address the first question) and the statistical formalities of how to use such data (to address the second question; see also §10.3 in Chapter 10). Such a simulation, together with the systematic review and meta-analysis needed to parameterise it, could even be worth publishing on its own ahead of doing the complete research project.

[23] For example, the US National Institutes of Health supports an online registry, clinicaltrials.gov, which as of late 2022 held the methodological details of nearly 500,000 publicly and privately funded studies from the USA and > 200 other countries and sovereign states [Accessed 20 November 2022]. Close to 100,000 more studies in other fields are available through the Open Science Framework's aggregation of registries (https://osf.io/registries/discover [Accessed 20 November 2022]).

[24] For example, in the United States, pre-registration of clinical trials has been required by law since 2007 (42 CFR Part 11, as amended 17 November 2022).

[25] Registered Reports are now reviewed by more than 300 well-known, international peer-reviewed journals. See Center for Open Science (2022).

3.6 Summary

- Data simulation requires a priori specification of hypotheses, prior expectations, and models. Thus, data simulations can transparently illuminate researcher biases and communicate the uncertainty (variance) in prior estimates of effect sizes. Data simulations and bootstrapping also can more clearly illustrate post hoc (posterior) uncertainty in parameter estimates than simple assertions of variance estimates or confidence intervals.
- Statistical significance—the p-value, which is equal to the probability of falsely rejecting a true null hypothesis—and β, which is the probability of falsely accepting a false null hypothesis, are related to one another through their joint dependence on sample size and effect size. It is conventional to assert statistical significance if $p \leq 0.05$ and to strive for statistical power $1 - \beta = 0.8$. For even the simplest statistical tests, the sample size needed to detect a relatively small effect size can be surprisingly large.
- Questionable research practices (QRPs, including HARKing) arise when hypotheses are poorly articulated and analyses are not (or only weakly) defined prior to undertaking a study. Defining a data-generating model from prior information and using it for data simulations can help to clarify hypotheses and determine appropriate analyses once the data have been collected. Using these simulations as part of a study's pre-registration can further minimise researcher biases and QRPs.

References

Andrade, C. (2021). HARKing, cherry-picking, P-hacking, fishing expeditions, and data dredging and mining as questionable research practices. *Journal of Clinical Psychiatry*, 82(1), 20f13804. doi: 10.4088/JCP.20f13804.

Azzolina, D., Berchialla, P., Gregori, D., & Baldi, I. (2021). Prior elicitation for use in clinical trial design and analysis: A literature review. *International Journal of Environmental Research and Public Health*, 18(4), 1833. doi: 10.3390/ijerph18041833.

Bastick, Z. (2021). Would you notice if fake news changed your behavior? An experiment on the unconscious effects of disinformation. *Computers in Human Behavior*, 116, 106633. doi: 10.1016/j.chb.2020.106633.

Bayes, T. (1763). An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, 53, 370–418. doi: 10.1098/rstl.1763.0053.

Berger, J. O. & Selke, T. (1987). Testing a point null hypothesis: The irreconcilability of *P* values and evidence (with comments and rejoinder). *Journal of the American Statistical Association*, 82(397), 112–139. doi: 10.2307/2289131.

Borenstein, M. (2009). Effect sizes for continuous data. In H. Cooper, L. V. Hedges, & J. C. Valentine (Eds.), *Handbook of Research Synthesis and Meta-analysis* (pp. 221–235). New York, NY, USA: Russell Sage Foundation, 2nd edition.

Center for Open Science (2022). Registered Reports: Peer review before results are known to align scientific values and practices. https://www.cos.io/initiatives/registered-reports. Accessed 20 November 2022.

Clegg, B. (2013). *Roger Bacon: The First Scientist*. New York, NY, USA: Carroll & Graf Publishers.

Cohen, J. (1965). Some statistical issues in psychological research. In B. B. Wolman (Ed.), *Handbook of Clinical Psychology* (pp. 95–121). New York, NY, USA: McGraw-Hill.

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences*, 2nd edition. New York, NY, USA: Academic Press.

Correll, J., Mellinger, C., McClelland, G. H., & Judd, C. M. (2020). Avoid Cohen's 'small,' 'medium' and 'large' for power analysis. *Trends in Cognitive Sciences*, 24(3), 200–207. doi: 10.1016/j.tics.2019.12.009.

Cox, D. R. & Hinkley, D. V. (1974). *Theoretical Statistics*. London, UK.: Chapman & Hall.

Dechêne, A., Stahl, C., Hansen, J., & Wänke, M. (2010). The truth about the truth: A meta-analytic review of the truth effect. *Personality and Social Psychology Review*, 14(2), 238–257. doi: 10.1177/1088868309352251.

Edwards, W., Lindman, H., & Savage, L. J. (1963). Bayesian statistical inference for psychological research. *Psychological Review*, 70(3), 193–242. doi: doi.org/10.1037/h0044139.

Falconer, J. R., Frank, E., Polaschek, D. L. L., & Joshi, C. (2022). Methods for eliciting informative prior distributions: A critical review. *Decision Analysis*, 19(3), 189–204. doi: 10.1287/deca.2022.0451.

Fallis, D. (2015). What is disinformation? *Library Trends*, 63(3), 401–426. doi: 10.1353/LIB.2015.0014.

Fazio, L. K., Rand, D. G., & Pennycook, G. (2019). Repetition increases perceived truth equally for plausible and implausible statements. *Psyconomic Bulletin & Review*, 26, 1705–1710. doi: 1.3758/s13423-019-01651-4.

Fleiss, J. L. & Berlin, J. A. (2009). Effect sizes for dichotomous data. In H. Cooper, L. V. Hedges, & J. C. Valentine (Eds.), *Handbook of Research Synthesis and Meta-analysis*, 2nd edition (pp. 238–253). New York, NY, USA: Russell Sage Foundation.

Gelman, A. & Carlin, J. (2014). Beyond power calculations: Assessing type S (sign) and type M (magnitude) errors. *Perspectives on Psychological Science*, 9(6), 641–651. doi: 10.1177/1745691614551642.

Goodman, S. N., Fanelli, D., & Ioannidis, J. P. A. (2016). What does research reproducibility mean? *Science Translational Medicine*, 8(341), 341ps12. doi: 10.1126/scitranslmed.aaf5027.

Greenland, S., Senn, S. J., Rothman, K. J., et al. (2016). Statistical tests, *P* values, confidence intervals, and power: A guide to misinterpretations. *European Journal of Epidemiology*, 31(4), 337–350. doi: 10.1007/s10654-016-0149-3.

Hepburn, B. & Andersen, H. (2021). Scientific Method. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2021 edition.

Howson, C. & Urbach, P. (1991). Bayesian reasoning in science. *Nature*, 350(6317), 371–374. doi: 10.1038/350371a0.

Ioannidis, J. P. A. (2005). Why most published research findings are false. *PLoS Medicine*, 2(8), e124. doi: 10.1371/journal.pmed.0020124.

Ioannidis, J. P. A. (2012). Why science is not necessarily self-correcting. *Perspectives on Psychological Science*, 7(6), 645–654. doi: 10.1177/1745691612464056.

Kerr, N. L. (1988). HARKing: Hypothesizing after the results are known. *Personality and Social Psychology Review*, 2(3), 196–217. doi: 10.1207/s15327957pspr0203_4.

Kirkham, R. L. (1992). *Theories of Truth: A Critical Introduction*. Cambridge, MA, USA: MIT Press.

Koricheva, J., Gurevitch, J., & Mengersen, K. L., Eds. (2013). *Handbook of Meta-analysis in Ecology and Evolution*. Princeton, NJ, USA: Princeton University Press.

Ladyman, J. (2001). *Understanding Philosophy of Science*. London: Routledge.

Lakens, D. (2014). Performing high-powered studies efficiently with sequential analyses. *European Journal of Social Psychology*, 44(7), 701–710. doi: 10.1002/ejsp.2023.

Lakens, D. (2019). The value of preregistration for psychological science: A conceptual analysis. *Japanese Psychological Review*, 63(3), 221–230. doi: 10.24602/sjpr.62.3_221.

Lenth, R. V. (2007). Statistical power calculations. *Journal of Animal Science*, 85(13 Suppl), E24–E29. doi: 10.2527/jas.2006-449.

McGrath, R. E. & Meyer, G. J. (2006). When effect sizes disagree: The case of *r* and *d*. *Psychological Methods*, 11(4), 386–401. doi: 10.1037/1082-989X.11.4.386.

Montgomery, D. C. (2020). *Design of Experiments*, 10th edition. New York, NY, USA: Wiley.

Needham, J. (1981). *Science in Traditional China: A Comparative Perspective*. Cambridge, MA, USA: Harvard University Press.

Neyman, J. & Pearson, E. S. (1933). The testing of statistical hypotheses in relation to probabilities *a priori*. *Mathematical Proceedings of the Cambridge Philosophical Society*, 33(4), 492–510. doi: 10.1017/S030500410001152X.

Nosek, B. A., Ebersole, C. R., DeHaven, A. C., & Mellor, D. T. (2018). The preregistration revolution. *Proceedings of the National Academy of Sciences, USA*, 115(11), 2600–2606. doi: 10.1073/pnas.1708274114.

Nosek, B. A., Hardwicke, T. E., Moshontz, H., et al. (2022). Replicability, robustness, and reproducibility in psychological science. *Annual Review of Psychology*, 73, 719–748. doi: 10.1146/annurev-psych-020821-114157.

Page, M. J., McKenzie, J. E., Bossuyt, P. M., et al. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *Journal of Clinical Epidemiology*, 134(1), 178–189. doi: 10.1016/j.jclinepi.2021.03.001.

Pennycook, G. (2019). Supplemental materials for "Repetition increases perceived truth equally for plausible and implausible statements." https://osf.io/w4k2c. Accessed 23 May 2023.

Purgar, M., Glaszio, P., Klanjscek, T., Nakagawa, S., & Culina, A. (2024). Supporting study registration to reduce research waste. *Nature Ecology & Evolution* 8(8) 1391–1399. 10.1038/s41559-024-02433-5.

Robert, C. & Casella, G. (2011). A short history of Markov chain Monte Carlo: Subjective recollections from incomplete data. *Statistical Science*, 26(1), 102–115. doi: 10.1214/10-STS351.

Rubin, M. (2020). Does preregistration improve the credibility of research findings? *The Quantitative Methods for Psychology*, 16(4), 366–390. doi: 10.20982/tqmp.16.4.p376.

Schmid, C. H., Stijnen, T., & White, I., Eds. (2021). *Handbook of Meta-Analysis*. Boca Raton, FL, USA: Chapman & Hall/CRC.

Shrader-Frechette, K. (1988). Producer risk, consumer risk, and assessing technological impacts. *Impact Assessment*, 6(3–4), 155–164. doi: 10.1080/07349165.1988.9725656.

Simmons, J. P., Nelson, L. D., & Simonsohn, U. (2011). False-positive psychology: Undisclosed flexibility in data collection and analysis allows presenting anything as significant. *Psychological Science*, 22(11), 1359–1366. doi: 10.1177/0956797611417632.

Stefan, A. M., Katsimpokis, D., Gronau, Q. F., & Wagenmakers, E.-J. (2022). Expert agreement in prior elicitation and its effects on Bayesian inference. *Psychonomic Bulletin & Review*, 29(5), 1776–1794. doi: 10.3758/s13423-022-02074-4.

Szucs, D. & Ioannidis, J. P. A. (2017). When null hypothesis significance testing is unsuitable for research: A reassessment. *Frontiers in Human Neuroscience*, 11, 390. doi: 10.3389/fnhum.2017.00390.

Thompson, S. K. (2012). *Sampling*, 3rd edition. New York, NY, USA: Wiley.

Unkelbach, C. (2007). Reversing the truth effect: Learning the interpretation of processing fluency in judgments of truth. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, 33(1), 219–230. doi: 10.1037/0278-7393.33.1.219.

Unkelbach, C. & Greifeneder, R. (2018). Experiential fluency and declarative advice jointly inform judgments of truth. *Journal of Experimental Social Psychology*, 79, 78–86. doi: 10.1016/j.jesp.2018.06.010.

Wasserstein, R. L. & Lazar, N. A. (2016). The ASA statement on *p*-values: Context, process, and purpose. *The American Statistician*, 70(2), 129–133. doi: 10.1080/00031305.2016.1154108.

Wikipedia Contributors (2022). Timeline of the history of the scientific method—Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Timeline_of_the_history_of_the_scientific_method&oldid=1119623237. Accessed 16 January 2024.

Wikipedia Contributors (2023). Effect size—Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Effect_size&oldid=1159322505. Accessed 15 July 2023.

4

Prospective simulation of statistical power

If a project is worth doing, it should be done well. Most scientific studies take months or years of advance planning.[1] A crucial step in this planning is the determination of sample size—how many replicates should the experiment have for each treatment group or stratum—and, for longitudinal studies, how long should they run? For example, we might hypothesise that honey bees are more sensitive to wireless electromagnetic radiation than humans, as the former are smaller and seemingly more fragile than the latter (e.g., Treder et al. 2023). An experiment placing wireless routers next to bee hives (and controls with routers where the wireless is switched off) may seem to us to be an obvious experiment with which to test this hypothesis. But what can it yield? How strong would we expect the effect to be? If the effect is small, how many hive-control combinations do we need to detect it? Either way, for how long should the experiment run? Finally, if we do not detect an effect, can we then say there really is none, or have we committed a type-II statistical error (Table 3.1) and reported a false negative (i.e., what is the power of our experimental design)?

In most cases, the more resources we invest, the larger the probability of finding an effect, if it exists (see §4.1), but the return on our investment will diminish in parallel. Similarly, the accuracy of an estimate of an effect size increases with the square root of the sample size. Although more is always better, we are more interested in identifying an acceptable probability of detecting an effect for the lowest reasonable cost. Working hard to collect data only to find out after the fact that the sampling or experimental design never had a chance to detect a weak effect is very frustrating. Such frustration can and should be reduced by thinking ahead—from study design through the end of the data analysis (Chapter 3).

Using simulations, we can do a dry-run of the entire experiment and see if it is worth doing. Using simulations also forces us to be specific about how much noise we expect to find, what kind of measurements we will take, how strong we expect the effects to be, and how we plan to analyse the data. Being forced to formulate our expectations is a good thing because by doing so we reduce the risk of a fruitless endeavour.[2]

In the following sections, we first explore sample-size calculations for simple two-sample comparisons analogous to the WiFi-bee question above and for which analytical solutions

[1] The James Webb Space Telescope is a contemporary extreme example. It was planned, designed, and tested for more than 30 years before it was finally launched at the end of 2021. The pre-launch costs exceeded US $10 billion, and it is expected to be in service for at least a decade.

[2] We acknowledge that the exact predictive power cannot be quantified. Every small error in effect size or minor violation of distributional assumptions will change the estimated power. That said, it is much better to try and estimate power as best we can and plan ahead than to shrug and run blindly after undetectable effects.

also are available. We also illustrate how many samples we need before an estimator of the correlation between two variables becomes stable; the answer is surprising. We then move on to somewhat more complicated designs, including longitudinal studies with and without stopping rules and for detecting trends in monitoring of ecological phenomena through time. We close this chapter with pointers to even more difficult problems in determining sample sizes and computing statistical power that arise in different research areas.

What you will learn from this chapter

- How to use data simulation effectively to determine the appropriate sample sizes for common test statistics (two-sample *t*-test, Pearson's correlation coefficient, χ^2-test) and a variety of experimental designs.
- The stopping-rule principle and whether recruiting participants or adding samples to a study until a fixed upper bound is reached or statistical significance is obtained gives valid and unbiased results.
- How to simulate how long a time series needs to be before we can confidently conclude that there is a significant trend in the data.

4.1 Simple group comparisons

One of the simplest statistical tests, and perhaps the most familiar one, is the two-sample *t*-test. We use the two-sample *t*-test to test whether the difference between the means of two normally-distributed random variables is significantly different from zero.[3] Although there is an analytical solution to the problem of determining the necessary sample size, given an estimate of the standardised effect size and the acceptable probability of type-I and type-II errors (Figure 3.3 in Chapter 3, §3.4), it is still useful to demonstrate that the analytical function and simulations yield the same estimates of necessary sample sizes.

As a first example, suppose we want to be able to tell whether two groups whose means differ by $\Delta = 5$ units and for which the pooled standard deviation (s) of our measurements equals 4 are significantly different from one another. We then ask, "How many samples do we need to measure for each group?" We assume implicitly that we will test for significant differences at the critical level of $\alpha = 0.05$, and that we want to find this difference in 80% of the cases (i.e., have a power $1 - \beta = 0.8$).

Following Senn (2007), we can compute the necessary sample size as

$$N = 2(Z_{\alpha/2} + Z_{\beta})^2 \sigma^2 / \Delta^2, \tag{4.1}$$

where α is the acceptable probability of a type-I statistical error (routinely set at $\alpha = 0.05$); β is the acceptable probability of a type-II statistical error (routinely set at $\beta = 0.2$, which implies statistical power = $1 - \beta = 0.8$); $Z_{\alpha/2}$ is the value of the normal distribution that cuts off the upper-tail probability at $\alpha/2$ (which equals 1.96 for $\alpha = 0.05$); Z_{β} is the value of the normal distribution that cuts off the lower-tail probability at β (which equals 0.84 if

[3] In this description, we use the common, albeit informal, phrasing. To ensure that we all agree on what a two-sample *t*-test actually is, here are the elements of its more formal definition. "Normally distributed" means that the observed data Y can be treated as if they were a random sample of a random variable y drawn from a Gaussian (i.e., "normal") distribution, whose parameter μ is a function of some predictor(s) X. In the case of a *t*-test, X is an indicator for the two groups, whereas in a linear regression X is a predictor. For completeness, therefore, we should write $Y \sim \mathcal{N}(\mu = a + bX, \sigma = c)$, with a, b, and c being parameters of the model. In common practice, however, b either is assumed to be equal to zero or the function is subsumed within μ and we write $Y \sim \mathcal{N}(\mu, \sigma = c)$.

Code Box 4.1: Calculating the sample size needed for a reliable two-sample *t*-test (analytical solution)

```
power.t.test(delta=5, sd=4, sig.level=0.05, power=0.8)

|      Two-sample t-test power calculation
|
|               n = 11.09423
|           delta = 5
|              sd = 4
|       sig.level = 0.05
|           power = 0.8
|     alternative = two.sided
|
| NOTE: n is number in *each* group
```

$\beta = 0.2$); σ is the pooled standard deviation (assumed here equal to 4 and also to be the same for the two groups); and Δ is the expected difference between groups. Using these values in Equation 4.1 gives the sample size N needed *for each* group:

$$N = [2(1.96 + 0.84)^2 \times 4]/5^2 = 10.03. \tag{4.2}$$

Doing this calculation in R (Code Box 4.1), we get:

The analytical formula (Equation 4.1) and the R function (Code Box 4.1) give approximately, but not exactly, the same result (10 versus 11 individuals needed per group). Why? Most importantly, the value for σ in Equation 4.1 is actually the standard deviation of the sample that is assumed to be observed *after* the experiment has been run. But in practice, we only have an estimate of σ based on previous observations. Second, the values for Z in Equation 4.1 are from look-up tables of Z scores for the standard normal distribution, whereas the `power.t.test()` function in R is computing directly (via numerical integration) the Z scores for the t-distribution.

Third, `power.t.test()` uses Cochran's formula for computing sample size, $N = \frac{Z^2 p(1-p)}{\epsilon^2}$, where Z is the Z-score as in Equation 4.1 corresponding to the desired level of confidence; p is the proportion of the sample population for which the attribute of interest is present; and ϵ is the absolute precision of p. `power.t.test()` first estimates the necessary values of Z, p, and ϵ from the values of α, β, Δ, and σ input by the user (see also Sugden et al. 2000), then solves for the minimum sample size N.

Finally, Equation 4.1 is accurate only for "large" sample sizes, and, as is clear from the result of the computation, this is a "small-sample" problem.[4]

We now have enough information to estimate the necessary sample size using simulation. Our goal is clear: we want to know the necessary sample size for which we will find a

[4] To add further confusion, the `power.t.test()` function is one of at least two sample-size functions available in R. `power.t.test()` is part of R's basic `stats` package. An alternative, `pwr.t.test()`, is available in the contributed `pwr` library. It uses similar formulas and computations as `power.t.test()`. Both use Cochran's formula with comparable estimates of z, p, and ϵ, and the `uniroot()` function to estimate via numerical integration the Z scores of the tails of the t-distribution. However, as inputs to `uniroot()`, `pwr.t.test()` uses a precision two orders of magnitude finer than `power.t.test()`. The results differ only in the fifth decimal place ($N = 11.09423$ in `power.t.test()` and $N = 11.09420$ in `pwr.t.test()`).

Code Box 4.2: Using simulation to determine the sample size needed for a reliable two-sample *t*-test (*part 1*)

```
# Assumptions
 delta=5; sigma=4; alpha=0.05; power=0.8
# Initial run:
 N <- 10
 y <-  rep(c(0, delta), each=N) + rnorm(2*N, mean=delta, sd=sigma)
 x <- rep(c("A", "B"), each=N)
 t.test(y ~ x, var.equal=T)

|   Two Sample t-test
|
| data:  y by x
| t = -4.0443, df = 18, p-value = 0.0007609
| alternative hypothesis: true difference in means between group A and
| group B is not equal to 0
| 95 percent confidence interval:
|  -8.386943 -2.652333
| sample estimates:
| mean in group A mean in group B
|        4.859346       10.378984
```

significant difference (if it exists) with $p < \alpha = 0.05$ between two groups whose means differ by 5 units and for which the pooled standard deviation of the data equals 4 (alternatively, a "large" standarised effect size equal to $\Delta/\sigma = 1.25$).

For the simulation, we draw $2N$ data points from the normal distribution $y \sim \mathcal{N}$ ($\Delta = 5, \sigma = 4$) and run a *t*-test comparing two halves of the simulated data (Code Box 4.2).

We then repeat this simulation 1,000 times for different values of N and count, for each value of N, in how many cases the p-value of the t-test is ≤ 0.05 (Code Box 4.3). Our required sample size, for a power of 0.8, is when this count reaches 80%, or 800/1,000 simulations. By repeating this for different values of N and fixing the random seed to allow for reproducibility, we satisfyingly obtain a result consistent with theory and similar to that estimated by Equation 4.1 and by both `R` power functions: the necessary sample size per group is somewhere between 10 and 12 (Figure 4.1).

The same simulation also could be used with different significance levels, for one-tailed tests, for non-parametric two-sample tests such as the Mann-Whitney U-test, and for interactions between predictors (Royston & Sauerbrei 2014).

4.2 How many data points do we need for a simple correlation?

Competing with the *t*-test for the simplest statistical procedure is the estimation of Pearson's correlation coefficient for two variables X_1 and X_2:

$$r = \frac{\sum_{i=1}^{N}(X_{1i} - \bar{X}_1)(X_{2i} - \bar{X}_2)}{\sqrt{\sum_{i=1}^{N}(X_{1i} - \bar{X}_1)^2 \cdot \sum_{i=1}^{N}(X_{2i} - \bar{X}_2)^2}}. \tag{4.3}$$

Code Box 4.3: Using simulation to determine the sample size needed for a relaible two-sample *t*-test (*part 2*)

```
set.seed(123)
prop.less.5percent <- 1
for (i in 3:20){
  N <- i
  Ps <- replicate(1000, {
    y <-  rep(c(0, delta), each=N) + rnorm(2*N, mean=delta, sd=sigma)
    x <- rep(c("A", "B"), each=N)
    t.test(y ~ x, var.equal=T)$p.value
  })
  prop.less.5percent[i] <- sum(Ps < 0.05)
}
```

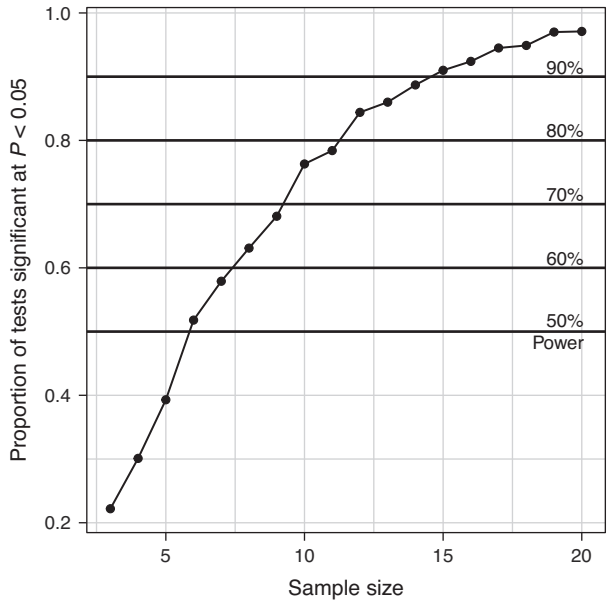


Figure 4.1 Simulation of proportion of significant *t*-tests for different sample sizes ($\Delta = 5$, sd $= 4$). Horizontal lines indicate different power levels. For power = 80%, the required sample size is just above 11.

A common question when planning to collect data is to ask how many observations we need to reliably detect a correlation? Schönbrodt & Perugini (2013) concluded that the estimate of *r* stabilises at around 250 data pairs—many more than most disciplines would consider necessary for such a simple statistic. Schönbrodt & Perugini (2013) reached this conclusion using simulation, providing a very nice example for us to look at.

Before doing the simulations, however, we first clearly define "reliable" and "stable." Once we have criteria for reliability and stability, we can reproduce and discuss Schönbrodt & Perugini's (2013) simulations and analysis. Last, we re-do the simulations using an alternative method that ensures that different samples are independent of one another and compare the results of our simulations to those of Schönbrodt & Perugini (2013).

4.2.1 Reliability and stability

Crucial for the simulation is a definition of what we deem "reliable." By reliable, we mean that we should get the same (or acceptably close to the same) result (e.g., value of r) in repeated experiments (which may include simulations). What is "acceptably close to the same"? Schönbrodt & Perugini (2013) introduced the concept of a "corridor of stability," which is the interval $[\hat{r} - w, \hat{r} + w]$ around $\hat{r}$, the estimated value of r, that is considered by the researcher to be "the same" because it is within acceptable bounds of uncertainty (i.e., $\pm w$). Schönbrodt & Perugini (2013) considered r to be stable once it passed the "point of stability": the sample size beyond which *no* value of $\hat{r}$ falls outside the corridor of stability.[5]

4.2.2 Simulation

The algorithm used by Schönbrodt & Perugini (2013) in their simulation was:

1. Simulate a population of 1,000,000 data pairs from a bivariate normal distribution with a given (known) correlation ρ.
2. Draw 1,000,000 bootstrap samples from this population of size $N_{\max} = 1{,}000$.
3. For each bootstrap, start with the first 20 data pairs and compute the observed correlation $\hat{r}$.
4. Then add the next data pair to the set and recompute $\hat{r}$.
5. Continue adding data pairs until all 1,000 data pairs have been used.
6. For each such sequence of $\hat{r}$ values, find the point of stability: the number of data pairs beyond which all values of $\hat{r}$ fall within the corridor of stability.
7. For their simulations, Schönbrodt & Perugini (2013) considered three values of w to define the corridor of stability: $w = 0.1, 0.2$, and 0.3.

Note that each successive bootstrapped sample is a non-independent sequence of increasing sample sizes, since each longer sample always includes the preceding, smaller samples. We relax this assumption in §4.2.3.

Code Box 4.4 shows the results of a single run of the simulation and its estimate of the point of stability.

We then use a for loop to run the same simulation for many bootstraps (Code Box 4.5; we ran 10,000, which took several minutes):

Figure 4.2 illustrates the results of these simulations, including the point of stability and the corridor of stability. The true correlation coefficient in the simulated population equals 0.301, whereas the estimates have a slight positive bias ($\hat{r}_{N=1000} = 0.316 \pm 0.0276$ [SD]). This simulation clearly illustrates that increasing N decreases the variability of the correlation estimate, and that at $N \approx 250$ the 95% confidence interval indeed matches the corridor of stability.

4.2.3 An alternative approach

When we first read the paper by Schönbrodt & Perugini (2013), we liked it for several reasons. First, it drove home the point that even to reliably identify a simple correlation between two variables, we need far more data points than many researchers often have resources to collect. Second, it used a simple simulation method that could be extended easily to other types of statistical analysis. Finally, it was a succinct and well-written paper that did not over-interpret its findings.

[5] This corridor has an eerie resemblance to a confidence interval. Since it is set by the analyst, rather than computed from the data, it can be interpreted as the target CI for r.

Code Box 4.4: Simulating the reliability and stability of Pearson's *r* (*part 1*)

```
#Load required libraries
 library(mvtnorm)

# Set parameters
 rho = 0.3    # Set desired correlation
 N = 20:1000  # Range of sample sizes to look at
 w = 0.1      # Width of the corridor of stability

# Simulation
 population <- rmvnorm(1e5, mean=c(0,0), sigma=matrix(c(1, rho, rho, 1), 2, 2))
 boots <- population[sample(1000, replace=T), ]
 cors <- NA
 for (i in N) cors[i] <- cor(boots[1:i,1], boots[1:i,2])
 (POS <- max(which((cors > (0.3+w) | cors < (0.3-w)))))

| [1] 94
```

Instead of downloading and using their R code for this simulation (which they laudably provided on the journal's web page), we programmed it ourselves. As can be seen, their results and ours are essentially identical. However, while re-implementing their algorithm, we wondered whether their approach of extending the initial small dataset by adding new data to it is a reasonable one. What would happen if instead we re-drew an entirely new dataset for each sample size? These two different ways of simulating the samples from which we compute $\hat{r}$ have fundamentally different meanings and interpretations.

Schönbrodt & Perugini's (2013) approach implied that we would have collected k data pairs and then gone out and collected another pair, so that we ended up with the previous k pairs plus one new pair. Clearly, adding one new data pair should not make a large difference in the computation of $\hat{r}$ (i.e., $\hat{r}_k \approx \hat{r}_{k+1}$). In contrast, if we decided to collect independently k or $k+1$ data points, we would go out once (or twice) and get these different datasets, rather than successively increasing our sample size. Would this change in sampling also change the point of stability? By sampling each set independently, the lines would not be trajectories of an expanding sample. Rather, for each sample of size N, we would have B bootstrap estimates. Finally, why should we first create the population and then draw the bootstrapped samples from it (as Schönbrodt & Perugini 2013 did) instead of drawing each sample directly from the underlying distribution (see also §2.5)?

We used simulation to examine these alternatives (Code Box 4.6). Note that these alternative simulations also were much simpler. For each sample size, we simply drew B samples (equivalent to the bootstraps from the simulated population in the last simulation). The results of this simulation suggested a much later point of stability (Figure 4.3). Perhaps more importantly, the estimates of r were unbiased ($\hat{r}_{N=1000} = 0.301 \pm 0.058$ [SD]).

4.2.4 Conclusions

The lesson from our comparison is that details of simulations can make big differences. It does matter whether successive samples are considered to be an expansion of what we already have or are new samples, and whether we simulate a population and then sample from it or repeatedly sample from the underlying distribution. Our point here is not to say

Code Box 4.5: Simulating the reliability and stability of Pearson's *r* (*part 2*)

```
# This code uses the "population" object from Code Box 4.4

# Load required libraries
 library(mvtnorm)

# Set parameters
 rho = 0.3    # Set desired correlation
 N = 20:1000  # Range sample sizes to look at
 w = 0.1      # Width of the corridor of stability
 B = 1e4      # Number of simulations

# Simulations
 set.seed(123)
 boots <- replicate(B, population[sample(1000, replace=T), ]) # Array: 1000 x 2
     x B
 cors.mat <- matrix(NA, nrow=max(N), ncol=B)
 for (i in 1:B){
   cors <- NA    # set up empty vector
   for (j in N) cors[j] <- cor(boots[1:j, 1, i], boots[1:j, 2, i])
   cors.mat[,i] <- cors
   rm(cors)
 }
 cors.mat <- cors.mat[-c(1:19),] # remove N=1:19
 POSvec <- apply(cors.mat, 2, \(x) max(which((x > (rho+w) | x < (rho-w)))) )
     + 19  # COS
 POSvec[which(is.infinite(POSvec))] <- 20 # runs always within COS: set to N=20
 summary(POSvec)

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
|     1.0    34.0    79.0   122.6   162.0   981.0

  CIs <- apply(cors.mat, 1, quantile, c(0.025, 0.975))  # Compute confidence
       interval (and plot it)
```

that one approach is correct and the other is wrong. Rather, it is crucial to specify what the simulation is supposed to address and use that specification to clearly define the data-generating model. Expanding a sample will stabilise $\hat{r}$ *for that sample* at ≈ 250 observations, but repeatedly sampling a dataset with 250 data points will *not* reliably yield values of $\hat{r}$ within the corridor of stability (Figure 4.3).

4.3 Is "recruit until significant" problematic?

Some experiments are designed explicitly to collect and analyse data sequentially. Such experiments are common in clinical trials, where patients are treated one-by-one and additional participants may be added to the trial from a waiting list. Sequential data collection also is common in many other fields. For example, astronomers record data on the sun's

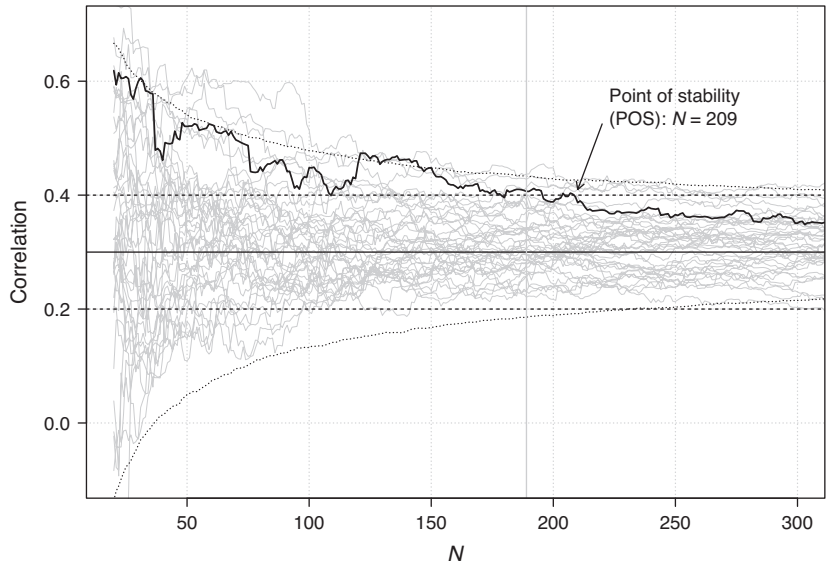


Figure 4.2 Evolution of the estimated correlation coefficient $\hat{r}$ as the sample on which it is based is extended from 2 to N. Note that each line represents a bootstrap sample with its own point of stability. The corridor of stability (with $w = 0.1$) is indicated by the dashed horizontal line. The dotted lines are 95% confidence intervals of the 10,000 bootstrap samples.

corona one solar eclipse at a time. Conservation biologists repeatedly and sequentially capture cheetahs or other animals, record their weight and age, release them, and often capture the same animals again. In studies like these, we should ask, "For how long should we continue to collect data?" Should we keep sampling *until* we have 100 data points? Or until our hypothesised effect is significant? Note that these questions conceptually are similar to those addressed in the simulation study by Schönbrodt & Perugini (2013) that we presented in the last section (Figure 4.2 and Code Boxes 4.4 and 4.5).

4.3.1 The stopping-rule principle

The stopping-rule principle is that either approach—"sampling up to a fixed (and predefined) number of individuals" or "sample until statistical significance is obtained"—should yield equivalent parameter estimates and *p*-values, and therefore either approach is valid and can be used (Fletcher 2019). However, stopping rules have been criticised for different reasons. The most intuitive critique is that, following the Law of Small Numbers (a.k.a. "hasty generalisation"), a small dataset may exhibit a pattern that disappears with larger sample sizes. Thus, stopping a study as soon as the first statistically significant pattern is observed will lead to biased estimates. You should not be surprised that we can use simulation to investigate the effect of a stopping rule on parameter estimates and *p*-values.

4.3.2 A case-study: Puerperal fever deaths

As a case study, we simulated an experiment to quantify the positive effect of hand-washing on the reduction in puerperal ("childbirth") fever deaths. This effect was first observed by Ignaz Semmelweis in the mid-nineteenth century (Semmelweis 1859; Jay 1999). The guiding question, and anticipated final figure, is whether the estimated rate of infection is biased. We also ask how the results differ between using a predetermined number of participants versus a stopping rule.

Code Box 4.6: An alternative approach to simulating the stability of Pearson's *r*

```
# Load required libraries
 library(mvtnorm)

# Set parameters
 rho = 0.3    # set desired correlation
 N = 20:1000  # range of sample sizes to look at
 w = 0.1      # width of the corridor of stability
 B = 1e4      # number of simulations

# Simulations
 set.seed(123)
 cors.mat2 <- matrix(NA, nrow=max(N), ncol=B)
 for (i in N){
   cors.mat2[i,] <- replicate(B, {
     X <- rmvnorm(i, mean=c(0,0), sigma=matrix(c(1, rho, rho, 1), 2, 2))
     cor(X)[1,2]
   })
 }
 cors.mat2 <- cors.mat2[-c(1:19),] # remove N=1:19
 POSvec2 <- apply(cors.mat2, 2, \(x) max(which((x > (0.3+w) | x < (0.3-w)))) )
      + 19
 summary(POSvec2)

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
|   280.0   540.0   635.0   646.7   746.0   981.0

 CIs2 <- apply(cors.mat2, 1, quantile, c(0.025, 0.975))
```

We take this simulation to a modern setting by envisaging recruiting at most 1,000 patients per treatment: doctors not washing their hands (the control, as was the norm in 1850) versus washing their hands between bedside visits (the treatment). In the stopping-rule setting, we stop the experiment when the proportion of infected patients significantly differs between the control and treatment groups at $p < 0.05$ as evaluated by a χ^2-test.[6]

There are, as usual, different ways to simulate such a study. Because we want to compare a pre-determined number of observations with the stopping rule, we first run a full simulation of 2,000 patients allocated, a pair at a time, to each of the two treatments. Then, we identify the number of patients at which the test was first observed to be significant. Finally, we compare the death rate estimated at that point to the death rate estimated from the full sequence estimates, and the true death rate (as defined a priori for the simulation; Code Box 4.7). We used 0.18 and 0.01, respectively, as the true death rates, which were reported by Semmelweis for puerperal fever in the 1850s.

The stopping rule—terminate the experiment when $p \leq 0.05$—ended the experiment after 110 pairs of patients and estimated the death rates as 0.16 and 0.018 for, respectively,

[6] We employ a χ^2-test here because we compare binomial data: the number of individuals that lived or died in the control treatment versus the number of individuals that lived or died in the hand-washing treatment.

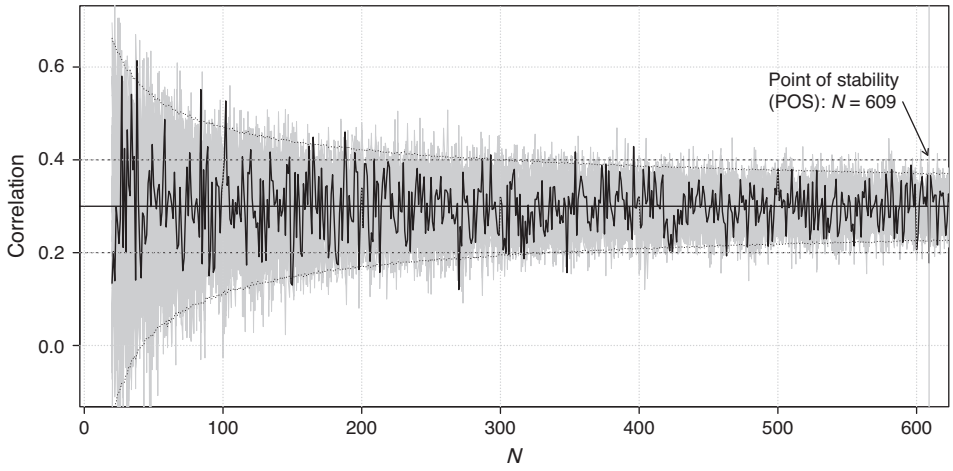


Figure 4.3 The estimated correlation coefficient $\hat{r}$ as a function of sample size N, where each sample was drawn independently (Code Box 4.6). As in Figure 4.2, there are 40 lines, but now they blur into each other, as they zigzag up and down (one of them is emphasised here in black). The corridor of stability (when $w = 0.1$) is indicated by the dashed horizontal line. The dotted lines are 95% confidence intervals of the 10,000 bootstrap samples. Note that sample size for the point of stability is more than twice as large as that obtained in the sequential simulations (Code Box 4.5 and in Schönbrodt & Perugini 2013) and the limit of the x-axis here is twice that of the previous figure.

the control and hand-washing treatments (Figure 4.4).[7] The results look reasonable, but to be sure, we repeated the simulation many times, and for computational efficiency, admitted patients in groups of 20 (Code Box 4.8). Further, when we report the results of this second simulation, we report not only the estimated death rates when the sequence was stopped but also the estimated death rates from the full sequences.

On average, the stopping rule terminated the experiment after only 36 pairs of patients, yielding estimated death rates of 0.157 and 0.0032, respectively (Figure 4.5). For the control treatment, the estimate is reasonably close to the true value, but for the hand-washing treatment, the estimate is too low by a factor of 3. The intuitive reason for this biased result is that given a low death rate, too few deaths occurred in only 36 patients to yield a reliable estimate. However, the complete sequences of $N = 1{,}000$ individuals yielded unbiased estimates of the death rates (0.181 and 0.010) that were very close to the true values (0.18 and 0.01).

Finally, we may wonder whether the bias is a function of the difference between control and treatment death rates. Because of the effects of sample size on statistical power (§4.1), smaller effect sizes (i.e., between-treatment differences in death rates) should lead to less biased estimates and longer sequences before a significant difference between treatments is found.[8] To test this, we simulated a range of death rates for the hand-washing treatment, from the low value of 0.01 up to the same value (0.18) as the control (Code Box 4.9). Here, the results are illustrated more clearly in a table than a figure (Table 4.1).

[7] The `suppressWarnings()` function in Code Box 4.7 hides the warnings about `NA`s that occurred during the computation of treatment death rates when no deaths were observed. In those cases, the table `Tr` does not have two entries and hence returns `NA` when indexed at position 2. For the same reason we replace all `NA`s with zeroes after the simulation.

[8] Strictly speaking, the decrease in bias with larger datasets or longer sequences is true in all cases only for simulated data. It will also hold for real datasets, but only if the samples taken from observational or experimental studies are themselves unbiased. If the sample collection was biased, the results will be biased no matter how large the dataset (an illustrative example is in Daru et al. 2018).

Code Box 4.7: A single simulation of the puerperal fever experiment

```
# Set parameters
 pC <- 0.18 # probability of death in controls
 pT <- 0.01 # probability of death in treatment (washing hands)

# Simulation
 set.seed(1234)
 Ns <- seq(10, 1000, by=50)
 simu1res <- matrix(NA, nrow=length(Ns), ncol=4)
 colnames(simu1res) <- c("N", "p-value", "control rate", "treatment rate")
 for (i in seq_along(Ns)){
   control <- sample(c(0,1), size=Ns[i], prob=c(1-pC, pC), replace=T)
   treat <- sample(c(0,1), size=Ns[i], prob=c(1-pT, pT), replace=T)
   Co <- table(control)
   Tr <- table(treat)
   simu1res[i,] <- c("N"=Ns[i], "p-value"=suppressWarnings(chisq.test(cbind(Co,
       Tr))$p.value), "control rate"=Co[2]/sum(Co), "treatment rate"=Tr[2]/sum(
       Tr))
 }
 simu1res[is.na(simu1res)] <- 0

# Now identify N when first time p<0.05:
 firstSignif <- min(which(simu1res[, 2] < 0.05))
 simu1res[firstSignif, ]

|              N        p-value   control rate treatment rate
|   1.100000e+02   4.351208e-04   1.636364e-01   1.818182e-02
```

4.3.3 Conclusions

There are several things to notice in Table 4.1. First, the smaller the difference between pC (death rate in the controls) and pT (death rate in the hand-washing treatment), the longer the sequence before the stopping rule terminated the experiment. Second, the bias in the estimate computed based on the stopped sequence was at least one order of magnitude larger than for the full sequence (unless there was no difference between pC and pT). Third, the estimates for the control treatment also appear to be biased when the stopping rule was used. Fourth, even when there was no difference between control and treatment (i.e., both pC and $pT = 0.18$; last line in Table 4.1), a spurious significant difference nonetheless was detected after ≈ 400 pairs of patients (but not so in the full dataset: $p = 0.056$). This illustrates that the stopping rule affects not only estimates of treatment effects but also p-values and assessments of statistical significance.

We conclude from our simulations that a stopping rule is problematic and cannot be considered to be unbiased. Indeed, current statistics textbooks in medicine (e.g., Riffenburgh & Gillen 2020) provide guidance on how to overcome problems with sequential analysis. Correcting the p-values in these kinds of experiments has been a subject of research since Wald (1945). Still, many researchers are unaware of these other methods and continue to regularly fall into the sequential analysis trap.

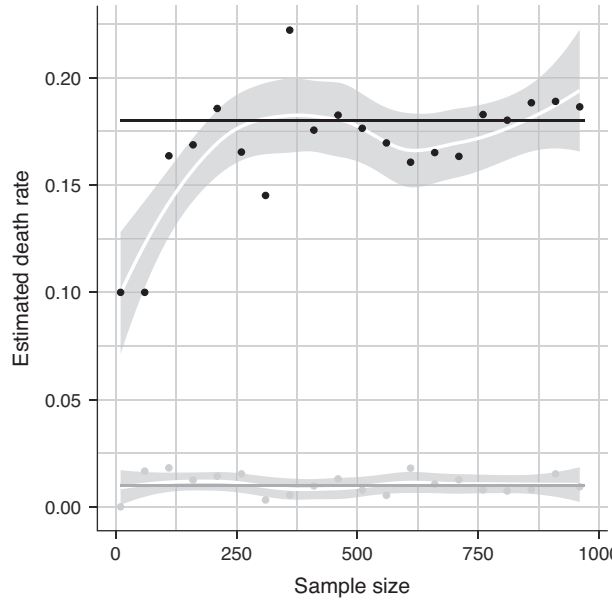


Figure 4.4 Results of a single run of the puerperal fever simulation (Code Box 4.7), with dots representing estimates of death rates as sample size increases (black: control; grey: hand-washing treatment). Loess smooths have been added to better visualise the patterns in the data. It takes a minimal sample size of around 250 to get estimates close to the truth (horizontal lines), and stopping the experiment before that minimal sample size is reached will yield biased results. Beyond this minimal sample size, intermediate analyses appear not be biased.

4.4 How long does a time series have to be?

Climatological and ecological data suggest that the Earth is warming at a rate unseen in over 400 million years (e.g., Foster et al. 2017), while the climate is becoming increasingly chaotic (e.g., Fuwape et al. 2020), biodiversity is declining precipitously (e.g., Díaz et al. 2019), and novel zoonotic diseases are appearing with catastrophic consequences for human health and society (e.g., Salata et al. 2019). How long do researchers need to monitor these trends before we have confidence that there really are significant changes (i.e., trends) in the data (e.g., Karpov et al. 2010; Arkilanian et al. 2020; Valdez et al. 2023)?

4.4.1 A case-study: Monitoring atmospheric carbon dioxide

In January 1959, when Charles David Keeling started measuring the concentration of atmospheric carbon dioxide ($[CO_2]$) every month on the summit of Mauna Loa (Harris 2010), he may have asked himself, "How long do I have to make these measurements until I detect a significant trend in the data?" To answer this question, we cast ourselves back to 1961, i.e., after data had been collected for only two years, giving only 24 monthly data points. Our past selves wonder how much longer we will have to continue our measurements before a trend emerges and our estimate of a trend in increasing $[CO_2]$ becomes significant and unbiased.

This time-series example has several features that make it a bit more challenging and realistic than our two previous examples of stopping rules. In the spirit of the previous chapter (see Figure 3.1), we start by specifying what the final figure should be. Since in 1961 we do not know the strength of the trend in $[CO_2]$, we want to explore several linear trends,

Code Box 4.8: Complete simulation of the puerperal fever experiment

```
# Objects pC and pT are the same as were used in Code Box 4.7

set.seed(1234)
Ns <- seq(20, 1000, by=20)
simus <- replicate(100, {
  simu1res <- matrix(NA, nrow=length(Ns), ncol=4)
  for (i in seq_along(Ns)){
    control <- sample(c(0,1), size=Ns[i], prob=c(1-pC, pC), replace=T)
    treat <- sample(c(0,1), size=Ns[i], prob=c(1-pT, pT), replace=T)
    Co <- table(control)
    Tr <- table(treat)
    simu1res[i,] <- c("N"=Ns[i], "p-value"=suppressWarnings(chisq.test(cbind(Co,
         Tr))$p.value), "control rate"=Co[2]/sum(Co), "treatment rate"=Tr[2]/
        sum(Tr))
  }
  simu1res[is.na(simu1res)] <- 0
  # Identify N when first time p<0.05:
  firstSignif <- min(which(simu1res[, 2] < 0.05))
  c(simu1res[firstSignif, ], simu1res[nrow(simu1res),])
})
rowMeans(simus)

| [1] 3.600000e+01 7.408239e-03 1.574167e-01 3.208333e-03 1.000000e+03
| [6] 1.901584e-31 1.810400e-01 1.009000e-02
```

in this case between 0 and 0.2 ppm/month.[9] For each value, we want to know how long we have to run our monthly monitoring before we can reliably detect a significant trend in the data. Thus, our final figure will have sample size on the x-axis, the proportion of significant trends on the y-axis, and different lines representing different trend strengths.

4.4.2 Estimating the parameters for the simulation

Next, we need to identify which specific features of this time series we want to include. The $[CO_2]$ data vary seasonally because of the flush of leaves of deciduous trees in the spring in the northern hemisphere (Barichivich et al. 2013), exhibit strong temporal autocorrelation, and are noisy. In the absence of literature data (i.e., in 1961), we need to estimate all these features—trend, seasonal cycles, and noise—from only the two years of data we have already (Figure 4.6).

We first fit a sine wave to the data to represent seasonality (Code Box 4.10):[10]

$$f(\text{month}) = a \sin\left(\frac{2\pi}{12}\text{month} + b\right) + c \tag{4.4}$$

[9] Our mental time-machine also let us fit a linear regression to the data from 1959 until 1997, which yielded an estimated increasing trend in $[CO_2]$ of 0.11 ppm/month.

[10] More sophisticated functions are, of course, possible, but with only two observation periods they would overfit the data.

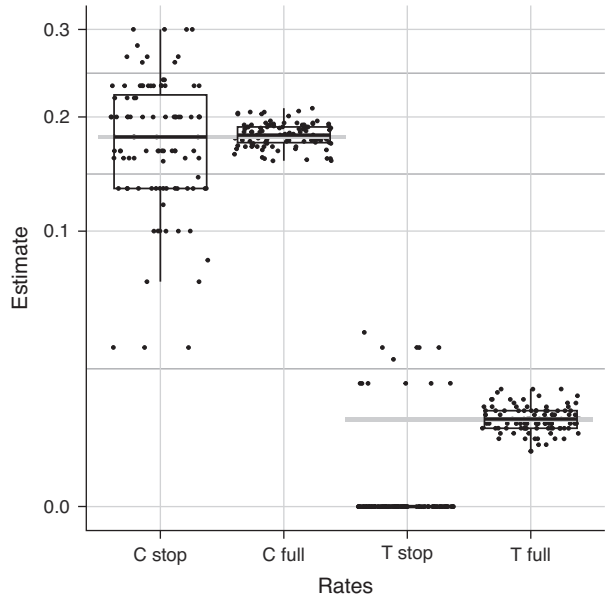


Figure 4.5 Estimates of death rates for control (C) and hand-washing treatment (T), from data generated with a stopping rule (stop) or the full 990 sample pairs. Grey lines indicates true values. The mass of black dots for estimate = 0 for the data simulated with a stopping rule indicates that no deaths occurred in the hand-washing (control) treatment in many of the "stopped" datasets.

The sine-wave function needs to be rescaled (so that 12 months represent the period of 2π), shifted upwards (so that the mean is not equal to zero; the parameter c in Code Box 4.10), and optimally centred on the data (parameter b), and its amplitude fitted to the observations (parameter a). The resulting fit is rather satisfactory (Figure 4.6, grey line).

The residuals from this fit give us an indication of the noise in the data. But we cannot really assume that we have seen all the noise in the data from only 24 data points, so to be on the safe side, we multiply the noise estimate from the residuals by a factor of 1.5.[11] We also can use the residuals to estimate the range and strength of temporal autocorrelation (Code Box 4.11). The results suggest a lag-1 autocorrelation, which can be simulated as described in Code Box 2.11 in Chapter 2.

However, in the $[CO_2]$ data, we have both a signal from the month and temporal autocorrelation and noise to accommodate. A simple way to simulate this is to make the noise temporally autocorrelated.[12] Now we have all ingredients for a first simulation of, say, eight years ahead of the two years with which we started (Code Box 4.12).

4.4.3 Simulating and analyzing trends

The next step is to add a trend to these simulated data. A simple way to add in a trend is to make the mean of the noise term a function of time. A more interesting question is

[11] This is rather arbitrary, but not uncommon in engineering, where it is known as the "factor of safety" (e.g., Duncan 2000). But it is only a guess, at best a guesstimate that must do until we have good data to estimate it.

[12] There are faster ways to do this in R, including dedicated functions such as `tseries::arima.sim`. But here, we want to illustrate the details of the simulation rather than resort to having to trust a function that does it all for us. See also the discussion of R functions to compute statistical power in footnote 4 in this chapter and §2.4 in Chapter 2.

Code Box 4.9: Expanded complete simulations of the puerperal fever experiment

```
pC <- 0.18
pT <- seq(0.01, 0.18, by=0.01)

set.seed(123)
Ns <- seq(10, 1000, by=20)

manysimus <- matrix(NA, nrow=length(pT), ncol=8)
for (j in seq_along(pT)){
  simus <- replicate(100, {
    simu1res <- matrix(NA, nrow=length(Ns), ncol=4)
    for (i in seq_along(Ns)){
      control <- sample(c(0,1), size=Ns[i], prob=c(1-pC, pC), replace=T)
      treat <- sample(c(0,1), size=Ns[i], prob=c(1-pT[j], pT[j]), replace=T)
      Co <- table(control)
      Tr <- table(treat)
      simu1res[i,] <- c("N"=Ns[i], "p-value"=suppressWarnings(chisq.test(cbind(
          Co, Tr))$p.value), "control rate"=Co[2]/sum(Co), "treatment rate"=Tr
          [2]/sum(Tr))
    }
    simu1res[is.na(simu1res)] <- 0
    # Identify N when first time p < 0.05:
    firstSignif <- min(which(simu1res[, 2] < 0.05))
    c(simu1res[firstSignif, ], simu1res[nrow(simu1res), ])
  })
  manysimus[j,] <- rowMeans(simus, na.rm=T)
}
colnames(manysimus) <- c("Nstop", "pstop", "Cstop", "Tstop", "Nfull", "pfull", "
    Cfull", "Tfull")
manysimus <- cbind.data.frame("TtrueRate"=pT, manysimus)
out <- cbind(manysimus[, c(9, 1, 3,4)], "bias stopping"=manysimus[,4]-manysimus
    [,9], manysimus[, c(7,8)], "bias full"=manysimus[,8]-manysimus[,9])
```

how to analyze the data, whether they are simulated or real. When a seasonal component is present, the sensitivity for detecting a trend is increased when we account for the seasonal component. One flexible way to do so is by means of a cyclic spline. Our final general additive model (GAM) thus has two components: a seasonal cycle and a trend (Code Box 4.13); we assume that the trend is linear. Once we have fitted the model to the data, we will explore whether the residuals exhibit temporal autocorrelation, which also will necessitate including an autoregressive component in the model.

This long set of code and output shows that the GAM detected a significant linear trend (0.026 ppm per month), but its residuals were autocorrelated. The general additive mixed model (GAMM) used to accommodate the lag-1-autocorrelation (AR-1) yielded an almost identical estimate for the trend (0.025 ppm per month), but with a standard error twice as large, and thus a larger (but still significant) *p*-value.[13]

[13] The fit of the autocorrelation model can be accessed using `summary(fgamm$lme)`, which shows an estimate for `rho` = 0.60, i.e., close to the true value, 0.632.

Table 4.1: Effect of reducing difference in death rates between control and hand-washing treatments on estimation bias. pC and pT refer to rates of death in controls and hand-washing treatments, respectively; pT varied from 0.01 to 0.18 while pC was fixed equal to 0.18 in all of the simulations. The "hats" indicate that the values of pC and pT were estimated using the stopping rule or the full dataset. $N = 990$ for the full sequence.

pT_{true}	N_{stop}	$\widehat{pC}_{\text{stop}}$	$\widehat{pT}_{\text{stop}}$	bias $\widehat{pT}_{\text{stop}}$	$\widehat{pC}_{\text{full}}$	$\widehat{pT}_{\text{full}}$	bias $\widehat{pT}_{\text{full}}$
0.01	38	0.180	0.005	−0.005	0.183	0.010	−0.0003
0.02	42	0.185	0.008	−0.012	0.180	0.020	0.0002
0.03	51	0.193	0.018	−0.012	0.182	0.030	0.0004
0.04	50	0.208	0.024	−0.016	0.181	0.040	−0.0003
0.05	58	0.209	0.034	−0.016	0.180	0.050	0.0005
0.06	65	0.207	0.036	−0.024	0.180	0.060	0.0004
0.07	74	0.214	0.049	−0.021	0.180	0.070	0.0003
0.08	75	0.223	0.057	−0.023	0.180	0.081	0.0014
0.09	92	0.211	0.065	−0.025	0.179	0.091	0.0010
0.10	96	0.219	0.072	−0.028	0.181	0.101	0.0008
0.11	114	0.214	0.078	−0.032	0.181	0.109	−0.0011
0.12	133	0.216	0.087	−0.033	0.179	0.120	−0.0003
0.13	142	0.222	0.102	−0.028	0.178	0.131	0.0005
0.14	193	0.211	0.105	−0.035	0.181	0.140	0.0003
0.15	234	0.216	0.114	−0.036	0.182	0.150	0.0002
0.16	321	0.209	0.136	−0.024	0.181	0.162	0.0015
0.17	369	0.195	0.156	−0.014	0.179	0.170	−0.0001
0.18	421	0.190	0.180	−0.000	0.183	0.180	−0.0002

4.4.4 Simulating and analyzing many replicates

A single simulation of a single trend is, of course, not enough. We need to replicate this single run to get an estimate of the proportion of simulations that give us estimates of trends that differ significantly ($p \leq 0.05$) from zero. Then we need to do this for durations of our data that range from 1 to 8 years. And finally, we should do this for trends that differ in their strengths.

Code Box 4.14 recaps the minimal code to get the p-value for the trend in a single run and Code Box 4.15 illustrates how to repeat this simulation many times.

We repeat this simulation 100 times and compute the mean p-value (Code Box 4.15).

Next, we add into our simulations different lengths of the time series yielding a mean p-value per year of additional monitoring (Code Box 4.16).

The output in the last line of Code Box 4.16 shows that the mean p-value for the trend comes down with every year of monitoring, hitting the 0.025-threshold after 9 years.[14]

Finally, we repeat this step for different trends, from no trend to 0.2 ppm/month.[15] Without parallelisation (which we do not employ to prevent technical details from cluttering the code), the computation now takes quite a while (dozens of minutes).

[14] We assume that a researcher would want to detect trends in positive or negative directions, hence apply a two-sided test. Thus, the threshold for detecting a positive trend at the $\alpha = 0.05$ level for a two-sided test should be divided by 2; hence $\alpha = 0.25$.

[15] Recall from note 9 in this chapter that the observed trend in increasing $[CO_2]$ for the first 10 years was 0.11 ppm/month.

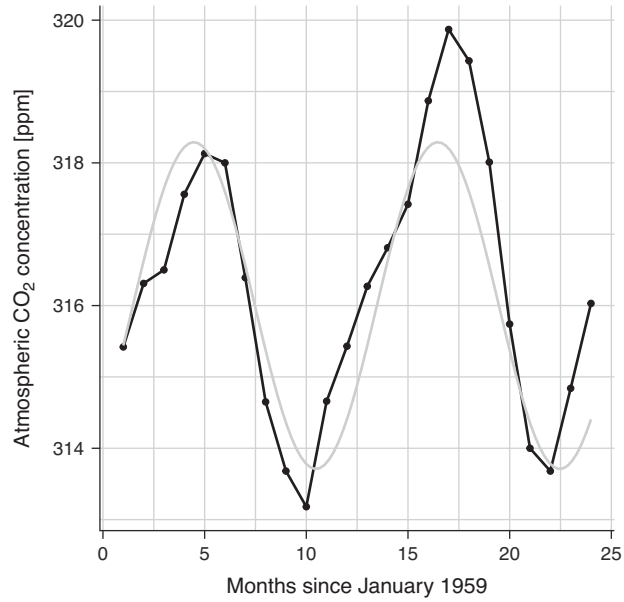


Figure 4.6 The first 24 monthly measurements of atmospheric [CO_2] on Mauna Loa. The fitted grey sine wave represents the seasonal variation in [CO_2] that results from carbon uptake by the forests of the northern hemisphere.

Code Box 4.10: Fitting seasonality to two years of CO_2 data

```
# For this example, we use the "co2" dataset in the R datasets package
 dats <- data.frame("months"=1:24, "month"=rep(1:12, times=2), CO2=co2[1:24])
 fnls <- nls(CO2 ~ a * sin(2*pi/12 * month + b) + c, start=list(a=1, b=0, c=316),
       data=dats)
 summary(fnls)

| Formula: CO2 ~ a * sin(2 * pi/12 * month + b) + c

| Parameters:
|  Estimate Std. Error  t value Pr(>|t|)
| a   2.2917     0.2570    8.916 1.39e-08 ***
| b  -0.7677     0.1122   -6.845 9.12e-07 ***
| c 316.2867     0.1818 1740.216  < 2e-16 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.8904 on 21 degrees of freedom
```

From the plot of the results (with the x-axis on a square-root scale, as we can expect the detection to scale with $\sqrt{N}$), we see that we can expect a trend of 0.01 ppm/month to take another seven years to be reliably detectable, whereas a trend ≥ 0.05 ppm/month would be detectable with the two years of data we have already (Figure 4.7).

Code Box 4.11: Computing noise and temporal autocorrelation

```
noise <- sd(residuals(fnls)) * 1.5 # Factor of safety: 1.5
noise

| [1] 1.276205
5

acf(residuals(fnls), plot=F)

| Autocorrelations of series 'residuals(fnls)', by lag
|
10 |     0      1      2      3      4      5      6      7      8      9
|  1.000  0.632  0.079 -0.194 -0.003  0.392  0.526  0.276 -0.192 -0.428

|    10
| -0.290
15 |  rho <- acf(residuals(fnls), plot=F)$acf[2]
```

Code Box 4.12: Simulating noise and temporal autocorrelation

```
ms <- 1:96 # Months since start
m <- rep(1:12, len=length(ms)) # Month of the year

#Simulation
5 set.seed(123)
noises <- rnorm(1, 0, noise)
for (i in 2:length(ms)) noises[i] <- rho*noises[i-1] + rnorm(1, 0, noise)
acf(noises, plot=F)[0:8]

10 | Autocorrelations of series 'noises', by lag
|
|      0      1      2      3      4      5      6      7      8
|  1.000  0.618  0.369  0.285  0.152  0.124  0.081  0.012 -0.075

15 co2hat <- 2.29*sin(2*pi/12*ms -0.77) + 316 + noises
```

4.5 Improving estimates: Is the experiment powerful enough?

After several hundred years of using the scientific method in many disciplines, we can certainly claim to have rough ideas about the causes and effects of many processes that we observe in the world around us. Scientists also have developed mathematical or computer models to forecast future dynamics of processes such as the weather, economic growth of a country, material fluxes among economic sectors, carbon fluxes in ecosystems, spread of a virus, development of cancerous cells, and the probability of a meteor impacting Earth along with the size of the resulting crater. The parameters in many of these models typically have some amount of uncertainty and reducing this uncertainty is the goal of many lab and field

Code Box 4.13: Simulating and analyzing trends

```
# Load required libraries
 library(mgcv)

 noises <- rnorm(1, 0, noise)
 for (i in 2:length(ms)) noises[i] <- rho*noises[i-1] +
                            rnorm(1, mean=0.01*ms[i], noise)
 co2hat <- 2.29 * sin(2*pi/12 * ms -0.77) + 316 + noises
 dats10 <- rbind.data.frame(dats, cbind("months"=25:(24+length(ms)), "month"=m,
     "CO2"=co2hat))

# Fit a GAM
 fgam <- gam(CO2 ~ s(month, k=12) + months, data=dats10)
 summary(fgam)

| Family: gaussian
| Link function: identity
|
| Formula:
| CO2 ~ s(month, k = 12) + months
|
| Parametric coefficients:
|              Estimate Std. Error  t value Pr(>|t|)
| (Intercept) 3.154e+02  2.610e-01 1208.492  < 2e-16 ***
| months      2.570e-02  3.748e-03    6.858 3.83e-10 ***
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| Approximate significance of smooth terms:
|            edf Ref.df     F p-value
| s(month) 4.247   5.27 28.68  <2e-16 ***
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| R-sq.(adj) =  0.614   Deviance explained = 63.1%
| GCV = 2.1122  Scale est. = 2.0023    n = 120

 acf(residuals(fgam)) # plot (not shown) reveals lag-1 autocorrelation

# Add an AR1 autocorrelation term:
 fgamm <- gamm(CO2 ~ s(month, k=12) + months, correlation=corAR1(form =
     ~ months), data=dats10)
 summary(fgamm$gam)

| Family: gaussian
| Link function: identity
|
| Formula:
```

```
| CO2 ~ s(month, k = 12) + months
|
| Parametric coefficients:
|              Estimate Std. Error t value Pr(>|t|)
| (Intercept) 3.154e+02  4.935e-01 639.112  < 2e-16 ***
| months      2.531e-02  7.043e-03   3.594 0.000485 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Approximate significance of smooth terms:
|            edf Ref.df     F p-value
| s(month) 5.657  5.657 13.34  <2e-16 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| R-sq.(adj) =  0.611   Scale est. = 1.9608    n = 120
```

Code Box 4.14: Simulation: one run, one year, one trend

```
# Set parameters
 N <- 1 # Number of additional years of monitoring
 trend <- 0.01
 months  <- 1:(12*(2+N)) # months
 moty <- rep(1:12, len=length(months)) # month of the year

# Simulation
 set.seed(123)
 CO2 <- round(2.29*sin(2*pi/12*months -0.77) + 316 + rnorm(length(months),
      0.01*months, noise), 2)
 dats <- data.frame(months, moty, CO2)
 fgamm <- gamm(CO2 ~ s(moty, k=12) + months, correlation=corAR1(form = ~ months),
       data=dats)
 summary(fgamm$gam)$p.table[2,4]

| [1] 0.000485212
```

observations and experiments. For any of these sampling efforts or experiments, we should ask whether the expected results will actually provide adequate or sufficient data to reduce the uncertainty and improve the existing estimate. In other words, will the new data increase our knowledge as we have encoded it in the model?

Improving existing estimates fundamentally is not too different from planning de novo experiments, except that we can quantify the current knowledge more easily. Recall that in Bayesian statistics this current knowledge is called prior knowledge and is quantified in a parameterised distribution—the prior—whose expected value is the current estimate. As we briefly outlined in §3.3.1 of Chapter 3, new measurements will shift the expected value or reshape the prior into the posterior (i.e., the distribution(s) of the parameter(s) after the

Code Box 4.15: Simulation: many runs, one year, one trend

```
set.seed(123)
simu.one <- replicate(100, {
  CO2 <- round(2.29*sin(2*pi/12*months -0.77) + 316 + rnorm(length(months),
      0.01*months, noise), 2)
  dats <- data.frame(months, moty, CO2)
  fgamm <- gamm(CO2 ~ s(moty, k=12) + months, correlation=corAR1(form =
      ~ months), data=dats)
  summary(fgamm$gam)$p.table[2,4]
  })

mean(simu.one)

| [1] 0.13
```

Code Box 4.16: Simulation: many runs, many years, one trend

```
set.seed(123)
Ns <- 1:10 # number of years
simu.res01 <- NA
for (i in seq_along(Ns)){
  simu.one <- replicate(100, {
    months  <- 1:(12*(2+Ns[i])) # months
    moty <- rep(1:12, len=length(months)) # month of the year
    CO2 <- round(2.29*sin(2*pi/12*months -0.77) + 316 + rnorm(length(months),
        0.01*months, noise), 2)
    dats <- data.frame(months, moty, CO2)
    fgamm <- gamm(CO2 ~ s(moty, k=12) + months, correlation=corAR1(form =
        ~ months), data=dats)
    summary(fgamm$gam)$p.table[2,4]
  })
  simu.res01[i] <- mean(simu.one)
}
round(simu.res01, 2)

|  [1] 0.44 0.37 0.34 0.25 0.24 0.12 0.07 0.04 0.01 0.01
```

experiment has been completed). Simulating the effect of new measurements can thus reveal what the outcome of such new measurements will be and how precise they have to be to improve the estimate to a pre-specified degree.

Sometimes such new measurements surprise us; they do not improve the prior because the new measurements lie in its tails. In such cases, the resulting posterior will be wider (less precise) than the prior. Under such conditions, simulations would not have helped, as we normally would assume (and build into our simulations) that the new measurements follow (roughly) the same distribution as the old ones.

Code Box 4.17: Simulation: many runs, many years, many trends

```
set.seed(123)
trends <- seq(0, 0.2, len=21) # Range of possible simulated trends
Ns <- 1:10 # Number of years
simu.mat <- matrix(NA, nrow=length(Ns), ncol=length(trends))
for (j in seq_along(trends)){ # This takes quite a few minutes!
  for (i in seq_along(Ns)){
    simu.one <- replicate(1000, {
      months  <- 1:(12*(2+Ns[i])) # months
      moty <- rep(1:12, len=length(months)) # month of the year
      CO2 <- round(2.29*sin(2*pi/12*months -0.77) + 316 + rnorm(length(months),
          trends[j]*months, noise), 2)
      dats <- data.frame(months, moty, CO2)
      fgamm <- gamm(CO2 ~ s(moty, k=12) + months, correlation=corAR1(form =
          ~ months), data=dats)
      summary(fgamm$gam)$p.table[2,4]
    })
    simu.mat[i,j] <- mean(simu.one) # Across the 1000 simulations per trend and
        year
  }
}
```

In sum, we suggest that the central rationale for planning experiments or surveys is, *it is not worth doing unless there is a fair chance of getting a good estimate.* The point of all the simulations we have used in this chapter and the previous one to study effect sizes, stopping rules, and *p*-values is to determine whether the study is worth doing, given existing financial, personal, temporal, technical, or other constraints. Because any simulation must be tailored to the research question at hand, to close this chapter we present a few additional examples that should help you to think about how such simulations could be done even though we do not provide the code for running them.

4.5.1 A case study: The mass of a W boson

To start, we consider the recent analysis of decades-old data from the long-decommissioned Tevatron particle accelerator that were collected to improve estimates of the mass of the W bosons (CDF Collaboration 2022).[16] The W bosons are elementary particles of the Standard Model of physics and exist briefly during nuclear decay. For example, a neutron decays into a proton and a W^- boson, and then the W^- boson itself decays further into an electron and neutrino. Until 2022 the prior distribution of measurements of the mass of the W bosons (whether W^+ or W^-) was 80,379 ±12 MeV (Zyla et al. 2020). Subsequent analysis of the Tevatron data yielded a new value of 80,433 MeV and greater precision (±9 MeV; CDF Collaboration 2022). What would the posterior distribution be (Figure 4.8)?

[16] The W bosons may be positively or negatively charged, denoted, respectively, W^+ and W^-. Both have the same mass, $\frac{1}{2}vg$, where g is the SU(2) gauge coupling and v is the Higgs vacuum expectation value.

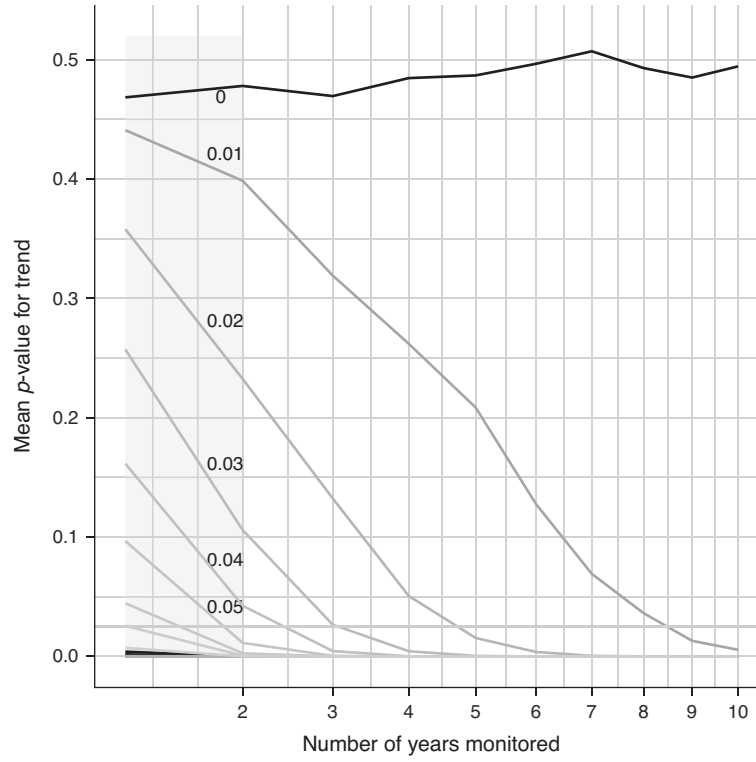


Figure 4.7 Simulated mean *p*-values for trend detection. Lines represent different strengths of the simulated trend in $[CO_2]$ increase (in ppm/month). The horizontal line indicates the critical threshold for a two-tailed test of significant trend. The shaded area covers the initial two years of data already monitored. Note that the *x*-axis is on a square-root scale.

The 2022 analyses caused ripples in physics communities. If they were correct, the Tevatron data would be 7 standard deviations greater than the theoretical value (80,360 MeV).[17] As is often the case, however, when measurements threaten to overturn an extremely well-tested physical theory, we are still waiting to declare the death of the Standard Model. In fact, in 2023, the newest ATLAS measurements yielded an estimate of 80,360 ±16 MeV, in line with the Standard Model (grey-dashed distribution in Figure 4.8).[18] How might you design a simulation to explore what kind of observations would be needed to kill off the Standard Model or maintain it on life support?

4.5.2 Dormouse monitoring

In several European countries, populations of the hazel dormouse (*Muscardinus avellanarius*), a small, squirrel-like rodent, are declining rapidly. As a result, the dormouse has legal protection and monitoring its local population sizes is done by local environmental agencies or NGOs. Such monitoring involves setting up nest boxes and checking them several times per

[17] This value is dependent on the mass of the top quark, which, in turn, is linked back to the Higgs vacuum expectation value, which links back to the W bosons. To be consistent, the Standard Model allows only for very specific mass of the W bosons for a given top-quark mass. The value used here is the one most compatible with the measurements before and after the Tevatron analysis, as indicated in CDF Collaboration (2022).

[18] These data were unpublished when we were writing this book, but are available from the ATLAS Collaboration (2024).

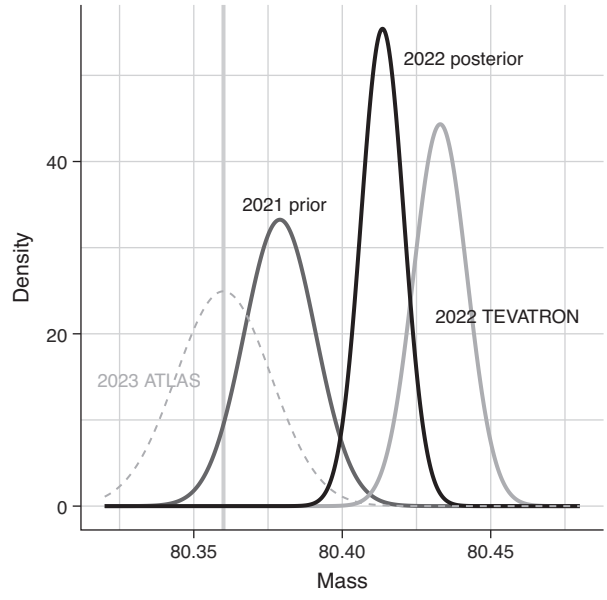


Figure 4.8 The 2022 Tevatron estimates of the mass of the W± boson (light grey) placed in the context of previous measurements (dark grey) and the theoretically expected value (vertical grey line). The black curve is the posterior derived from prior estimates and the Tevatron data. The dashed distribution represents the expected value and variance of the latest measurements made by the ATLAS experiment of the Large Hadron Collider in 2023.

year to see if dormice have taken up residence in them (see, e.g., Goodwin et al. 2017). We should ask how many nest boxes should be monitored at a potentially suitable site and how many such sites should be monitored to determine the rate of dormouse population change.

A simulation of such monitoring efforts would have to consider many factors that affect the precision of estimates of population trends (for an extensive review, see Kéry & Royle 2015, 2020). These include:

Detection probability: Even if a hazel dormouse is present at a location, it may not breed and hence not use the nest boxes. Or, it may have started breeding but its litter was preyed upon and hence it abandoned the nest box after a short while.

Selection probability: In a suitable environment, the hazel dormouse prefers a natural tree cavity to a nest box, whereas in a degraded habitat it may have no alternative to a nest box.

Stochastic population dynamics: In a short-lived mammal such as a dormouse, year-to-year variation in breeding success may yield fluctuating population sizes, making trends more difficult to detect.

Monitoring frequency: Recording nest-box use every week requires a sustained commitment by volunteers. Would it be sufficient to visit them only once a month or twice per season?

Variation in habitat suitability across sites: Populations may be more vulnerable in habitats of marginal quality or small area. Should the monitoring stratify the sampling by habitat quality or should it focus on habitats of particular quality? If the focus is on population strongholds, it may take longer to detect strong trends there as opposed to in marginal habitats. However, trends observed in population strongholds might be more informative about the state of the species in general. In sum: is the aim to detect *any* change, or to estimate the total population size as unbiasedly as possible?

Borrowing strength from other species: The hazel dormouse is not the only vulnerable species in deciduous woodlands. Habitat degradation may affect these other species in a similar way. Can monitoring the population sizes (or occurrence) of other rodent species inform trends in the hazel dormouse?[19]

The situation is not helped by the need for a relatively complicated statistical analysis, which involves at a minimum a detection model and a hierarchical Bayesian model. As a result, each simulation runs extremely slowly (Kéry & Royle 2015, 2020).

Because such data often are collected by volunteers recruited through citizen- or community-science initiatives, it could be problematic to communicate simulation results, as they may give the impression that there is no point in volunteering unless there were, say, 500 sites to be monitored. In practice, however, power analysis or sample-size simulations are rarely done in advance because legal frameworks, management plans, and individuals' interest in charismatic species ensure that some data will be collected regardless of their effect on subsequent analysis and estimation. It would be more useful if the official agencies supporting the management of threatened or declining species explored in advance the efficacy of proposed monitoring programs and determined whether the minimal effort required is affordable.

4.5.3 Planet 9

Our solar system has eight planets, plus a range of dwarf planets.[20] Innumerable smaller bodies also circle the sun. Those beyond Neptune, the farthest planet in our solar system, are called transneptunian objects, and three of them (the Sednoids) are on an extremely oval orbit around the sun.[21] The similarity in the Sednoids' orbit could be explained by the gravitational effect of a yet-to-be-discovered new planet, provisionally called "Planet 9." If Planet 9 exists, the orbits of the planets and the Sednoids could be used to estimate its mass and orbit. Indeed, such calculations exist (Batygin & Brown 2016; Batygin et al. 2019) and have led to as-yet unsuccessful efforts to detect Planet 9.

But telescope time is precious. There are only few large telescopes on Earth that are sensitive enough to have a fleeting chance of detecting a celestial body on a poorly known orbit and as far away as Planet 9 is estimated to be. Should a telescope-operating project allocate time to staring into empty space in the whimsical hope of a breakthrough discovery? Simulations could give us a hint: How realistic would it be to detect Planet 9 (should it exist) for a given amount of telescope time? Such simulations need to consider a substantial number of uncertainties, from the angle, eccentricity, and perihelion of the orbit itself, to the chance of detecting a tiny dark object against a dark background. Finally, the simulations need to account for the fact that Earth-based telescopes can only access a small portion of the sky at any given point in time.

At present, astronomers have argued for different telescope networks (Rice & Laughlin 2019, which includes simulations) or recalculated potential positions from deflections of meteoroid trajectories (Socas-Navarro 2023). These discussions and calculations are

[19] This topic is covered in the statistical and ecological literature under the keyword "joint species distribution modelling" (Ovaskainen & Abrego 2020).

[20] The demotion in 2006 of Pluto from planet to dwarf planet by the International Astronomical Union was covered extensively in the media, largely because some people attached sentimental value to Pluto-the-planet. The part of the definition of "planet" that got Pluto removed is relevant also for Planet 9: a planet has to have cleared the neighbourhood around its orbit, i.e., be the sole large body in that orbit.

[21] Technically, the perihelion of the Sednoids is > 60 times the earth-to-sun distance. The other known transneptunian objects are on more circular paths of < 50 earth-to-sun distances.

aimed at reducing the large uncertainties of the existence of Planet 9 before astronomers are even willing to commit themselves to simulations of the orbit and position of Planet 9 itself.

4.5.4 Experimental designs for impact assessment

To be able to evaluate a technological or management activity or bring a new product to market, we need to quantify its potential effects and impact. Whether we are evaluating the efficacy of quarantine or lockdown during a pandemic, introducing a new drug or pesticide, building a new power plant, altering teaching materials, or changing rules on agricultural subsidies, a scientific investigation of the effect of any such actions requires a comparison with a reference. There are different types of references, and here we are mainly concerned with measurable ones, not counterfactual hypothetical ones.[22]

For an impact assessment, three types of references are common (e.g., Wauchope et al. 2021):

Before-after: The state after the action is compared to the state before. The main problem here is that for longer-term effects, the system would change anyway, so that the "before" state is not a valid control situation anymore. For example, even if a sore throat is treated with a medication (the action), the before-state also would change over time, so that after several days many sore throats would be back to normal even without the action. Thus, even though there is a perceived effect of the action, this may be an incorrect attribution to the medication, because in many cases the sore throat would have cleared itself up without any therapeutic interventions.

Control-impact: The state after the action is compared to a control system that was not acted upon at the same time. The main problem here is that a control can be difficult to identify, particularly after the action. For example, the effect of water warmed by a power plant on a fish population in the source river versus changes in fish populations in a nearby river without a cooling tower is problematic. The two rivers may differ in many other characteristics (e.g., discharge rate, geological substrate, catchment land cover). So, if the observed difference in population sizes was in fact the result of differences in fertiliser inputs in the two rivers, the size of a smaller fish population in the power-plant river could be attributed incorrectly to an effect of the power plant.

Before-after control-impact: Combining the two, we get a "before-after control-impact" (BACI) design that controls for changes over time in the control individuals or sites (before-after for the control), and compares that with the changes observed in the impacted ones (before-after for the impact). If the sites were different before the impact, then we also would be able to account for those differences after the impact. Although there are still some assumptions in a BACI design, such as that the different individuals or sites would have similar trends in the absence of an impact, it is the de facto standard for studying and assessing environmental and other impacts through time.

We can do more than logically evaluate these different types of designs. We can simulate an impact and analyse it based on each of them (e.g., Christie et al. 2019). Obviously, how problematic a BA or CI design is, relative to a full BACI design, depends on how much their assumptions are violated. What their simulations showed, however, is that the

[22] In climate impact research and other fields, the entire world is the case study and there is no control or reference planet. Hence any change observed or predicted is typically compared against a hypothetical world without the action. For example, a world with observed increased CO_2 emissions would be compared with one with constant atmospheric [CO_2] (IPCC 2023), the latter being the counterfactual.

mis-attribution (i.e., getting it wrong) can be much higher for both BA and CI, and BACI generally fares much better.

An example of these different types of impact assessment designs is in the evaluation of the effect of light reflectors along roadsides. These reflectors aim to reflect the light of car headlights onto a wider stretch of the roadside verge and reduce the number of deer and other animals hit by cars. Brieger et al. (2016) used meta-analysis to differentiate between the three designs and determine whether the claims of reflector producers that their products reduce wildlife-vehicle encounters were, in fact, valid.

For the before-after case, wildlife crossings and approaching cars were monitored by wildlife cameras on some rural road stretches for a year. Then, all street posts along the associated stretches of roads were fitted with light reflectors and the monitoring continued for another year. The problem with this design is that there could have been different weather in the second year, causing the deer to behave differently, or the population could have suffered strong hunting pressure and there were simply fewer deer to attempt road crossings, or the local car drivers could have started avoiding the road due to wildlife collisions, or a different crop could have grown across the road, …. There is no way of knowing which of the many things that differed between the two years of the study caused the observed effect. Although BA designs were far and away the most common ones used by researchers, Brieger et al. (2016) found no differences in wildlife-vehicle interactions before or after the reflectors were installed.

For the control-impact case, a few road segments were fitted with light reflectors while other segments served as controls. Wildlife crossings were monitored on all segments. However, unless road segments are purposefully matched in their characteristics and then randomly assigned to control or impact, variability in local environmental conditions will add substantial noise to the study: roads may be straight or curvy, have forest on one or both sides, be close or far from settlements, go up or downhill, …. CI designs were less common than BA designs and came tantalisingly close (but not close enough) to showing a significant decrease in wildlife-vehicle interactions when reflectors were installed.

Finally, BACI studies combined the previous designs and monitored wildlife crossing for some time before fitting half of the road segments with light reflectors. This controlled for changes over time (e.g., weather) and differences in local starting conditions. Although the BACI studies had the largest effect size (around 1, largely due to the lower variance than BA and CI designs), because there were so few of them ($N = 9$), the apparent effects must be taken with a pinch of salt. Indeed, the analysis combining all studies and all designs, and accounting for design and publication type (scientific journal, report, or other) revealed no significant effect of adding light reflectors on wildlife-vehicle interactions.

4.5.5 Conclusions

The number of possible designs of an experiment or survey are legion (Fisher 1925; Mead 1988; Gregoire & Valentine 2008; Lohr 2009; Hoshmand 2017). All the examples in this chapter make clear that we can benefit substantially from consolidating previous knowledge or carrying out pilot studies to get an idea of effect sizes, noise, and feasibility of our proposed research.

The medical literature offers examples of how rigorous pre-planning and study simulations resulted in the double-blind randomised clinical trials becoming the unchallenged gold standard (e.g., Bothwell & Podolsky 2016). Similarly, opinion polls aiming to be representative now have to have a substantial number of interviewees to ensure a reasonable chance of being informative (Moon 1999; Chang & Krosnick 2009; Lohr 2009; Silver 2012).

4.6 Summary

- Data simulations show that the sample sizes needed to reliably conclude that there is an effect of an experimental treatment or that two variables are correlated with one another can be much larger than expected, especially if effects or correlations are not large or strong.
- The use of stopping rules for experiments or long-term observations generally gives unreliable and biased results, and can incorrectly imply that apparent trends are statistically significant.
- The length of a time series collected in long-term experiments or monitoring programs may need to be quite long before the trend can be determined to be significant. The actual length needed depends on the strength of the trend, its seasonality, and the noise in the data.
- There are many different designs for experiments, monitoring programs, and surveys. The choice of design will always benefit from consolidating previous knowledge and carrying out pilot studies to get an idea of effect sizes, noise, and feasibility of our proposed research. Using this information in data simulations can help us decide if a study is worth doing, given available resources, time, and labor.

References

Arkilanian, A. A., Clements, C. F., Ozgul, A., & Baruah, G. (2020). Effect of time series length and resolution on abundance- and trait-based early warning signals of population declines. *Ecology*, 101(7), e03040. doi: 10.1002/ecy.3040.

ATLAS Collaboration (2024). Measurement of the W-boson mass and width with the Atlas detector using proton-proton collision at $\sqrt{s} = 7$ TeV. arXiv: 2403.15085v1 [hep-ex]. doi: 10.48550/arXiv.2403.15085.

Barichivich, J., Briffa, K. R., Myneni, R. B., et al. (2013). Large-scale variations in the vegetation growing season and annual cycle of atmospheric CO_2 at high northern latitudes from 1950 to 2011. *Global Change Biology*, 19(10), 3167–3183. doi: 10.1111/gcb.12283.

Batygin, K., Adams, F. C., Brown, M. E., & Becker, J. C. (2019). The planet nine hypothesis. *Physics Reports*, 805, 1–53. doi: 10.1016/j.physrep.2019.01.009.

Batygin, K. & Brown, M. E. (2016). Evidence for a distant giant planet in the solar system. *The Astronomical Journal*, 151(2), 22. doi: 10.3847/0004-6256/151/2/22.

Bothwell, L. E. & Podolsky, S. H. (2016). The emergence of the randomized, controlled trial. *New England Journal of Medicine*, 375(6), 501–504. doi: 10.1056/NEJMp1604635.

Brieger, F., Hagen, R., Vetter, D., et al. (2016). Effectiveness of light-reflecting devices: A systematic reanalysis of animal-vehicle collision data. *Accident Analysis and Prevention*, 97, 242–260. doi: 10.1016/j.aap.2016.08.030.

CDF Collaboration (2022). High-precision measurement of the W boson mass with the CDF II detector. *Science*, 376(6589), 170–176. doi: 10.1126/science.abk1781.

Chang, L. & Krosnick, J. A. (2009). National surveys via RDD telephone interviewing versus the internet: Comparing sample representativeness and response quality. *Public Opinion Quarterly*, 73(4), 641–678. doi: 10.1093/poq/nfp075.

Christie, A. P., Amano, T., Martin, P. A., et al. (2019). Simple study designs in ecology produce inaccurate estimates of biodiversity responses. *Journal of Applied Ecology*, 56(12), 2742–2754. doi: 10.1111/1365-2664.13499.

Daru, B. H., Park, D. S., Primack, R. B., et al. (2018). Widespread sampling biases in herbaria revealed from large-scale digitization. *New Phytologist*, 217(2), 939–955. doi: 10.1111/nph.14855.

Díaz, S., Settele, J., Brondízio, E. S., et al. (2019). Pervasive human-driven decline of life on earth points to the need for transformative change. *Science*, 366(6471), eaax3100. doi: 10.1126/science.aax3100.

Duncan, J. M. (2000). Factors of safety and reliability in geotechnical engineering. *Journal of Geotechnical and Geoenvironmental Engineering*, Volume 126(4), 307–316. doi: 10.1061/(ASCE)1090-0241(2000)126:4(307).

Fisher, R. A. (1925). *Statistical Methods for Research Workers*. Edinburgh, Scotland, UK: Oliver and Boyd.

Fletcher, S. C. (2019). Stopping rules as experimental design. *European Journal for Philosophy of Science*, 9(2), 29. doi: 10.1007/s13194-019-0252-x.

Foster, G. L., Royer, D. L., & Lunt, D. J. (2017). Future climate forcing potentially without precedent in the last 420 million years. *Nature Communications*, 8(4), 14845. doi: 10.1038/ncomms14845.

Fuwape, I., Oluyamo, S., Rabiu, B., & Ogunjo, S. (2020). Chaotic signature of climate extremes. *Theoretical and Applied Climatology*, 139(1–2), 565–576. doi: 10.1007/s00704-019-02987-6.

Goodwin, C. E. D., Hodgson, D. J., Al-Fulaij, N., et al. (2017). Voluntary recording scheme reveals ongoing decline in the United Kingdom hazel dormouse *Muscardinus avellanarius* population. *Mammal Review*, 47(3), 183–197. doi: 10.1111/mam.12091.

Gregoire, T. G. & Valentine, H. T. (2008). *Sampling Strategies for Natural Resources and the Environment*. New York: Chapman & Hall/CRC.

Harris, D. C. (2010). Charles David Keeling and the story of atmospheric CO_2 measurements. *Analytical Chemistry*, 82(19), 7865–7870. doi: 10.1021/ac1001492.

Hoshmand, R. (2017). *Design of Experiments for Agriculture and the Natural Sciences*, 2nd edition. New York: Chapman and Hall/CRC.

IPCC (2023). *Climate Change 2023: Synthesis Report. A Report of the Intergovernmental Panel on Climate Change*. Geneva, Switzerland: IPCC.

Jay, V. (1999). Ignaz Semmelweis and the conquest of puerperal sepsis. *Archives of Pathology & Laboratory Medicine*, 123(7), 561–562. doi: 10.5858/1999-123-0561-ISATCO.

Karpov, K. A., Bergen, M., Geibel, J. J., et al. (2010). Prospective (a priori) power analysis for detecting changes in density when sampling with strip transects. *California Fish and Game*, 96(1), 69–81.

Kéry, M. & Royle, J. A. (2015). *Applied Hierarchical Modeling in Ecology: Analysis of Distribution, Abundance and Species Richness in R and BUGS. Volume 1: Prelude and Static Models*. Academic Press.

Kéry, M. & Royle, J. A. (2020). *Applied Hierarchical Modeling in Ecology: Analysis of Distribution, Abundance and Species Richness in R and BUGS: Volume 2: Dynamic and Advanced Models*. Cambridge, UK: Academic Press.

Lohr, S. L. (2009). *Sampling: Design and Analysis*, 2nd edition. North Scituate, MA, USA: Duxbury Press.

Mead, R. (1988). *The Design of Experiments: Statistical Principles for Practical Applications*. Cambridge, UK: Cambridge University Press.

Moon, N. (1999). *Opinion Polls: History, Theory and Practice*. Manchester, UK: Manchester University Press.

Ovaskainen, O. & Abrego, N. (2020). *Joint Species Distribution Modelling: With Applications in R*. Cambridge, UK: Cambridge University Press.

Rice, M. & Laughlin, G. (2019). The case for a large-scale occultation network. *The Astronomical Journal*, 158(1), 19. doi: 10.3847/1538-3881/ab21df.

Riffenburgh, R. H. & Gillen, D. L. (2020). *Statistics in Medicine*, 4th edition. London, UK: Academic Press.

Royston, P. & Sauerbrei, W. (2014). Interaction of treatment with a continuous variable: Simulation study of power for several methods of analysis. *Statistics in Medicine*, 33(27), 4695–4708. doi: 10.1002/sim.6308.

Salata, C., Calistri, A., Parolin, C., & Palù, G. (2019). Coronaviruses: A paradigm of new emerging zoonotic diseases. *Pathogens and Disease*, 77(9), ftaa006. doi: 10.1093/femspd/ftaa006.

Schönbrodt, F. D. & Perugini, M. (2013). At what sample size do correlations stabilize? *Journal of Research in Personality*, 47(5), 609–612. doi: 10.1016/j.jrp.2013.05.009.

Semmelweis, I. F. (1983 [1859]). *The Etiology, Concept, and Prophylaxis of Childbed Fever, Translated and Edited, With an Introduction, by K. Codell Carter*. Madison, WI, USA: University of Wisconsin Press.

Senn, S. (2007). Determining the sample size. In *Statistical Issues in Drug Development*, 2nd edition (pp. 195–2012). Chichester, UK: John Wiley & Sons.

Silver, N. (2012). *The Signal and the Noise: Why So Many Predictions Fail—But Some Don't*. New York, NY, USA: Penguin Press.

Socas-Navarro, H. (2023). A candidate location for Planet Nine from an interstellar meteoroid: The messenger hypothesis. *The Astrophysical Journal*, 945(1), 22. doi: 10.3847/1538-4357/acb817.

Sugden, R. A., Smith, T. M. F., & Jones, R. P. (2000). Cochran's rule for simple random sampling. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 62(4), 787–793. doi: 10.1111/1467-9868.00264.

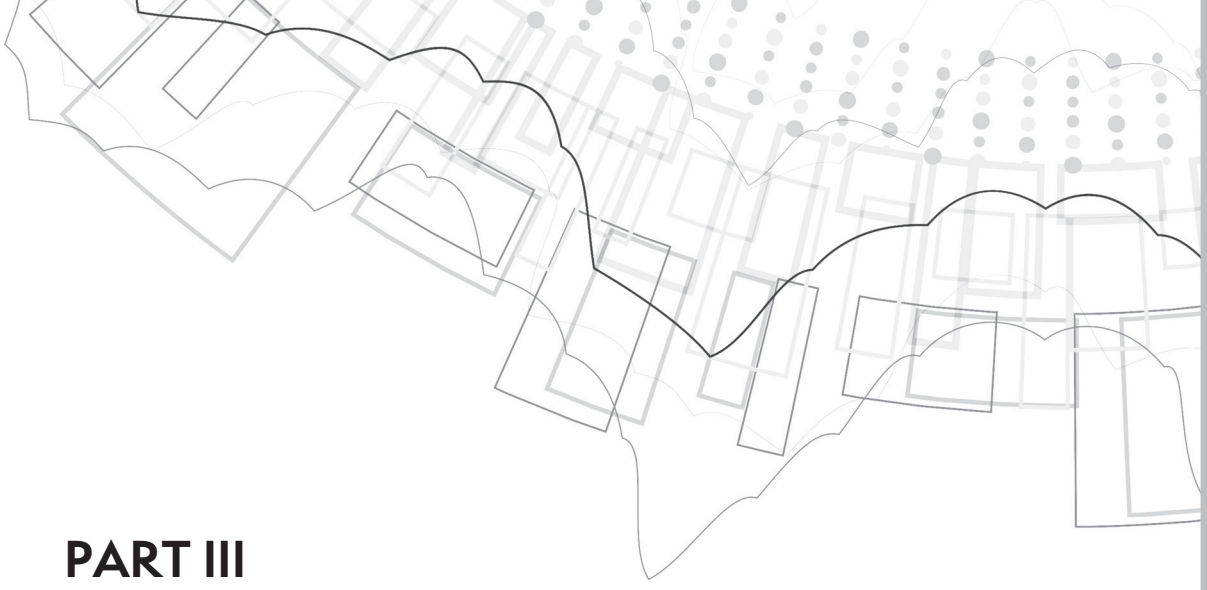
Treder, M., Müller, M., Fellner, L., et al. (2023). Defined exposure of honey bee colonies to simulated radiofrequency electromagnetic fields (RF-EMF): Negative effects on the homing ability, but not on brood development or longevity. *Science of the Total Environment*, 896, 165211. doi: 10.1016/j.scitotenv.2023.165211.

Valdez, J. W., Callaghan, C. T., Junker, J., et al. (2023). The undetectability of global biodiversity trends using local species richness. *Ecography*, 2023(3), e06604. doi: 10.1111/ecog.06604.

Wald, A. (1945). Sequential tests of statistical hypotheses. *Annals of Mathematical Statistics*, 16(2), 117–186. doi: 10.1214/aoms/1177731118.

Wauchope, H. S., Amano, T., Geldmann, J., et al. (2021). Evaluating impact using time-series data. *Trends in Ecology & Evolution*, 36(3), 196–205. doi: 10.1016/j.tree.2020.11.001.

Zyla, P. A., Barnett, R. M., Beringer, J., et al. (2020). Review of particle physics. *Progress of Theoretical and Experimental Physics*, 2020(8), 083C01. doi: 10.1093/ptep/ptaa104.



PART III

Post mensuram: Simulations in statistical analysis

Unless we have had formal courses in mathematics, we are likely to apply statistics with a certain amount of craftsmanship, guided by implicit assumptions, rules-of-thumb, our own and others' experiences, and schools of thinking (e.g., Bayesian versus frequentist versus machine learning). But even with extensive background in mathematics, we cannot always analytically derive an answer to simple questions such as, "How skewed can the data be before a *t*-test becomes invalid?" or "Will a threshold effect ruin my GLM?" Furthermore, many statistical rules-of-thumb are simply folklore, rather than derived from reliable data or analysis.

The three chapters in this part are the essential core of this book. Checking the importance of key assumptions (Chapter 5), critically examining received statistical wisdom (Chapter 6), and testing for potential effects and error propagation resulting from stringing together several steps of an analysis into a workflow (including imputation of missing values, estimating parameters and their confidence intervals, and hypothesis testing and inference; Chapter 7) require more complex and involved simulations than presented in other parts of the book.[1] At the same time, however, such simulations efficiently and cost-effectively provide us ways to find out for ourselves whether the words of wisdom passed on by statistical gurus actually are correct *for the situations at hand.*

[1] Workflows also should be tested a priori. However, missing data only show up after measurement, and correlations between predictors may not be known in advance. Hence, we expect the workflow to emerge, at least to some extent, from the analytical needs of a specific data set.

5

Assumptions: Is that one important?

Every statistical method entails certain assumptions about the samples and the data. Familiar assumptions include randomness or representativeness of the samples, independence of observations, and homogeneity of variances. For common statistical tests, there are many assertions about their reliability or robustness in the face of violation of some of these assumptions (discussed in this chapter) and rules-of-thumb (the topic of the next chapter) about how far one can push their boundaries. For many more statistical tests, however, the importance of such rules and assertions are untested. In particular, machine-learning approaches that may be approximately correct frequently are imbued with robustness properties that have not been examined or have no mathematical basis.

When assumptions are violated, the effect on analytical results is unpredictable. If assumptions are violated in a specific way (e.g., predictable bias), simulations can help us understand, for example, whether such violations increase type-I or type-II error rates (Table 3.1 in Chapter 3). In this chapter, we show how to use simulations to explore the consequences of some of the more common—and often violated—assumptions. For each example, we first identify the key assumption(s) in need of attention and testing and then describe an illustrative case study. Although the precise details vary for examining the effects of violating a specific assumption, the underlying approach is the same for each of them:

1. Fit the original model.
2. Run model diagnostics to identify which assumption(s) appear to be violated.
3. Simulate a dataset similar to the original, but which violates the assumption(s) in a defined way. Fit the same model as was fit to the original (empirical) dataset and compare model estimates with the known (i.e., simulated) truth.
4. Repeat the last step many times, particularly if the interest is in the rate of false positives or negatives.

What you will learn from this chapter

- The effects of violating the assumption of normality in the dependent variable in a linear regression model.
- The effects of inaccuracy or imprecision in measurements of predictor variables in a linear regression model, and of unintended values for experimental treatments in a regression context.
- The effect of violating the homoscedasticity-of-variance assumption in ANOVA.
- Why critical values for type-I errors need not always be equal to 0.05.

- The importance of the assumption of conditional independence among observations used in hierarchical mixed models.
- How correlation structure affects extrapolating outside the sample range.

5.1 Linear regression requires the data to be normally distributed

A basic linear regression model makes four assumptions:

1. The relationship is linear in the parameters.[1]
2. The data were collected using a valid sampling method and the observations are independent of one another.
3. The variance of the data is homogeneous and does not change with changes in the predictor variable.
4. The data (in this case, the residuals of fitting the linear model) follow a normal distribution.

In this section, we focus on the fourth assumption—that the data follow a normal distribution. We take up the third assumption—homogeneity of variance (a.k.a. homoscedasticity)—later in this chapter (§5.4).

5.1.1 A case study: Progress in pottery

Archaeologists digging for cultural artefacts have unearthed innumerable shards of pottery. The observed thickness of the shards is thought to indicate the proficiency of the potter and parallel improvements in materials and technology for firing clay; thinner, lighter pottery requires preparation of more homogeneous clay and sustained higher temperatures in the kiln. Assuming temporal progress in potters and their craft, one could hypothesise that the thickness of pottery should decline through time. Indeed, Daszkiewicz et al. (2019) found that the thickness of pottery shards of funerary urns in a cemetery in northern Poland that was used from approximately 150 BCE to 450 CE declined by about 3 mm over this 600-year period. They also found that the range of thickness values declined by nearly a factor of 5 (from $>$ 10 mm to $\approx$ 2 mm) over the same interval. Ignoring for now possible temporal autocorrelation in the data (but see §2.4 in Chapter 2), a test of the temporal decline in thickness hypothesis could be tested with a linear regression model predicting observed shard thickness from the year to which its production can be reliably dated.

5.1.2 Testing for normality using simulation

Many statistical tests are available to assess whether our data meet the assumption of normality: the most commonly-used ones are the Shapiro-Wilk, Anderson-Darling, Cramér-von Mises, Kolmogorov-Smirnov, and Lilliefors tests (we introduced the last two in §2.3.2 in Chapter 2). However, most of these tests have been criticised because they have unacceptable type-I error rates for common sample sizes: when the sample size is too small it is almost impossible to identify the underlying distribution, whereas for very large samples there are always likely to be some aberrant values and the tests fail to identify the correct distribution. Thus, some statisticians have advised that examination of diagnostic plots

[1] This is not the same as saying the relationship modelled with a linear regression fits a straight line to the data. The model $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2$ fits a parabola to the data but is still "linear in the parameters" β_i, meaning all terms in the model are additive.

Code Box 5.1: Testing for normality of residuals for a modest-sized dataset once

```
N <- 113
set.seed(123)
years <- runif(N, -300, 400)
thickness <-  6.71  - 0.0043 * years + rnorm(N, 0, 2)
fm <- lm(thickness ~ years, data=dats)
shapiro.test(residuals(fm))

|   Shapiro-Wilk normality test
|
| data:  residuals(fm)
| W = 0.99073, p-value = 0.6433
```

of the residuals (residuals versus fitted values) is a better determinant of normality than the *p*-values returned by normality tests (e.g., Crawley 2007; Gelman & Hill 2007; Harrell 2015).

In the dataset published by Daszkiewicz et al. (2019) there are 113 shards. We first want to determine if the data are normally distributed, but after using one of the aforementioned tests, we also want some assurance that we are in the "Goldilocks zone" of enough—but not too many—observations to have confidence in our conclusion.

In the general case of testing for normality, we want to assess two different questions: (1) Does the normality test reliably identify normally distributed data? (2) Does the normality test correctly identify non-normally distributed data? As always with data simulations, we explore these questions for a specific dataset and its sample size, and do not aim to draw general conclusions.

QUESTION 1: DOES THE NORMALITY TEST RELIABLY IDENTIFY NORMALLY DISTRIBUTED DATA?

We simulated data corresponding to the data in Daszkiewicz et al. (2019) using the following steps (Code Boxes 5.1 and 5.2):

1. Draw a "small" sample (size = 113) of years X_i from the uniform random variable $x \sim \mathcal{U}(-300, 400)$ (where negative values indicate years BCE and positive values indicate years CE) and simulate the corresponding shard thicknesses $Y = \{Y_1, Y_2, Y_3, \ldots, Y_N\}$ from the model $y \sim \mathcal{N}(\mu = \beta_0 + \beta_1 X, \sigma^2)$, with $\beta_0 = 6.71$ mm, $\beta_1 = -0.0043$ mm/yr,[2] and variance $\sigma^2 = 2$.[3]
2. Fit the linear model $\hat{Y} = \beta_0 + \beta_1 X$ to the dataset and save the residual values $\{(Y_i - \hat{Y}_i)\}$.
3. Compute the Shapiro-Wilk statistic on the residuals of the model.

[2] The values for β_0 and β_1 were determined by noting that the pottery becomes 3-mm thinner over the simulated 700-year-period, starting at 8 mm. So, $8 - 3/700(\text{year} + 300)$ gives the values for the slope and intercept parameters of the simulated distribution.

[3] We note that Daszkiewicz et al. (2019) only had five values for year X_i, corresponding to the approximate midpoints of the Pomeranian culture (ca. 450 BCE), Late Pre-Roman Iron Age (ca. 300 BCE), Early Roman period (ca. 50 CE), Roman period (ca. 100 CE), and Late Roman period (ca. 400 CE). We "eyeballed" the standard deviation from their Figure 21.

Code Box 5.2: Testing for normality of residuals for a modest-sized datasets many times

```
set.seed(123)
N=113
Ps <- replicate(1000, {
    years <- runif(N, -300, 400)
    thickness <-  6.71  - 0.0043*years + rnorm(N, 0, 2)
    dats <- data.frame("thickness"=thickness, "year"=years)
    fm <- lm(thickness ~ years, data=dats)
    shapiro.test(residuals(fm))$p.value
})
sum(Ps < 0.05)
```

```
[1] 54
```

4. Repeat 1,000 times and determine how often the test indicates significant deviance from normal.

For the 1,000 simulations, we counted the number of significant deviations from normality (Code Box 5.2).

The result—54 out of 1,000—is is very close to the 5% ideal. So, the answer to question 1 is that indeed the Shapiro-Wilk normality test has the nominal error rate (specificity), if the data are indeed normally distributed.

QUESTION 2: WILL THE TEST CORRECTLY IDENTIFY NON-NORMALLY DISTRIBUTED DATA?

To answer this question, we use uniform noise with a range akin to what the data in Daszkiewicz et al. (2019) suggested, and again repeat the simulation 1,000 times and count the number of significant deviations from normality (Code Box 5.3):

The Shapiro-Wilk test again had high sensitivity, identifying 99.7% of the uniform residuals as non-normal.

Out of curiosity, we explored whether this result was specific to the size of the dataset and asked: would a small ($N = 10$), non-normal dataset also correctly be recognised (Code Box 5.4)?

Indeed, the initial result depended on the size of the dataset and the Shapiro-Wilk test did not recognise correctly that the small sample was drawn from a uniform, not a normal distribution.

In sum, we can say that for a regression model for the kind of data provided by Daszkiewicz et al. (2019), a Shapiro-Wilk test correctly identifies normality and non-normality. For much smaller datasets, however, non-normality may look too similar to normality and may not be reliably detected.[4] Whether that invalidates the application of linear regression to data that violate the assumption of normality is, however, a different question (see also §2.3.4 in Chapter 2).

[4] You could determine the approximate range of sample sizes by running the simulation in Code Box 5.4 with different values of N.

Code Box 5.3: Testing for normality of non-normal residuals for a modest-sized datasets many times

```
set.seed(123)
N <- 113
thickness <-  6.71  - 0.0043 * years + runif(N, -3, 3)
fm <- lm(thickness ~ years, data=dats)
shapiro.test(residuals(fm))
```

```
|         Shapiro-Wilk normality test
|
| data:  residuals(fm)
| W = 0.96414, p-value = 0.003989
```

```
Ps <- replicate(1000, {
  years <- runif(N, -300, 400)
  thickness <-  6.71  - 0.0043*years + runif(N, -3, 3)
  dats <- data.frame("thickness"=thickness, "year"=years)
  fm <- lm(thickness ~ years, data=dats)
  shapiro.test(residuals(fm))$p.value
})
sum(Ps < 0.05)
```

```
| [1] 997
```

Code Box 5.4: Testing for normality of non-normal residuals for a small dataset many times

```
set.seed(123)
N=10
Ps <- replicate(1000, {
  years <- runif(N, -300, 400)
  thickness <-  6.71  - 0.0043*years + runif(N, -3, 3)
  dats <- data.frame("thickness"=thickness, "year"=years)
  fm <- lm(thickness ~ years, data=dats)
  shapiro.test(residuals(fm))$p.value
})
sum(Ps < 0.05)
```

```
| [1] 33
```

5.2 Regression models also assume that errors in predictor variables are negligible or unimportant

A second, and much less discussed, assumption of linear regression is that the values of the predictor are measured without error. That is, although the response Y is assumed to have been drawn from some distribution—often the normal (or Gaussian) distribution (§5.1), but other regression models can be used to fit data with different distributions, such as the binomial, Poisson, etc.—no distributional assumptions are made about the predictors X, since their values are considered to be fixed and measured without any error.

In reality, this is rarely the case. Errors in predictor variables arise in the same way as errors in the response variable: measurement error (e.g., inaccuracy or imprecision in GPS coordinates of a location), temporal variability of the target (e.g., "daily" values of blood pressure in patients measured at different times of day), or variability within things grouped into one type (e.g., median income brackets or housing costs). Such errors in predictor variables have important consequences for statistical inference. In a nutshell, because error or noise in Y is represented explicitly by distributional assumptions (§5.1), it has no effect on the point estimate of regression parameters, $\hat{\beta}$. But because X is assumed to be noise-free, any errors in the predictors will affect effect estimates of $\hat{\beta}$ and they do so in a particular way: they pull the estimates towards zero. This phenomenon is known as either regression dilution or attenuation (MacMahon et al. 1990; Frost & Thompson 2000), and was already a "well-known" issue in the 1970s (Griliches & Ringstad 1970, p. 368). The regression approaches that tackle this problem are called error-in-variable models (e.g., Fuller 1987; Hausman 2001; Schennach 2004; Stoklosa et al. 2015; Bartlett & Keogh 2018). The linear regression model with Gaussian errors in both observations and predictor variables is also called a model II regression or a major axis regression model (e.g., Sokal & Rohlf 1995; Legendre & Legendre 2012).

5.2.1 A case study: The impact of immigration on wages

A common expectation of economic theory is that increased immigration leads to a surplus of workers and a consequent reduction in overall wages (Samuelson 1964). One oft-cited estimate is that a 10% increase in supply of workers through immigration reduces wages at a regional or national level by 3–4% (Borjas 2003). However, a careful review of the literature on this topic is surprisingly equivocal (Aydemir & Borjas 2011). Many studies have found no effect of immigration on wages and some have found a weak negative effect, but only a very few have found the expected strong negative effect. Aydemir & Borjas (2011) argued that the apparent variation in effects of immigration on wages may be a consequence of regression dilution because the district- or county-level data used commonly in labour-market studies mask the variability that exists within each labour cell, which they define as the total sample size divided by the number of labour markets.[5] Using household-level (i.e., micro-census) data, Aydemir & Borjas (2011) showed this to be the case and corrected the dilution effect.

5.2.2 Simulation reveals a consistent regression-dilution effect

We use simulation to illustrate the consistent regression-dilution effect and show that it has a consistent effect (see also Brakenhoff et al. 2018; Boulesteix et al. 2020). We base our simulations on the data reported by Aydemir & Borjas (2011) and compare our results with their corrected estimate of the effect of immigration on wages (Code Box 5.5). We find that

[5] In their analysis of the Canadian national labour market, there were 240 labour cells: five education groups × eight experience groups × six cross-sectional (temporal) surveys.

Code Box 5.5: Simulating regression dilution

```
set.seed(43210)
immi <- runif(500, 0.2, 0.4) # Proportion of immigrants
slope <- -0.1
lgwage <- .1 * mean(immi) + slope * immi + rnorm(length(immi), sd=0.005) #
    intercept to center values on 0
coef(lm(lgwage ~ immi))

| (Intercept)        immi
|  0.03002892 -0.10189632

x <- immi + rnorm(length(immi), sd=.05) #add a small amount of noise to the
    predictor
coef(lm(lgwage ~ x))

| (Intercept)            x
|     0.01610     -0.05484
```

adding a small amount of noise ($\sigma = 0.05$) to the predictor variable (`immi` in Code Box 5.5) reduced the estimate for the effect of immigration on wages by 50%, from -0.1 to -0.05.

We can use the same code (Code Box 5.6) to explore the effects of different amounts of noise and verify that more noise leads to smaller (more shrunken) estimates of the slope (Figure 5.1).

ADDITIONAL GUIDANCE FOR ERROR-IN-VARIABLE MODELS

Errors in predictor variables are problematic for a wide range of parametric and non-parametric analyses, and also for machine-learning models. Solving this problem is straightforward only for the case of a simple linear regression. In this case the corrected estimate for $\hat{\beta}_x$ is $\hat{\beta}_x^* = \hat{\beta}_x \left(1 + \frac{\sigma_x^2}{\sigma_b^2}\right)$, where σ_x^2 is the measurement error on the predictor and σ_b^2 is the variance of underlying noise-free values (Frost & Thompson 2000).[6]

Pierre Legendre, who developed the `R` package `lmodel2` for analysing data with Gaussian errors in both observations and predictors, provided a concise set of recommendations for deciding whether to use an ordinary least squares (OLS) regression model (that assumes no error in predictors) or a major axis regression model (Legendre 2022). In brief, Legendre recommended using OLS if the magnitude of the error on $Y \gg 3\times$ the magnitude of the error on X; if the joint error distribution of X and Y is not bivariate normal and cannot be suitably transformed to be bivariate normal; or if the purpose of the model is not to infer the relationship between X and Y but only to predict Y from X. In all other cases, some form of major axis regression should be used.

Describing and analysing all the solutions to the general problem of errors in predictor variables would take a book in itself (e.g., Fuller 1987). For non-linear, survivorship, or hierarchical models, and for models with non-Gaussian data or error terms, the solutions are much more complex. In general, the key idea in all cases is to estimate the unknown (latent)

[6] Of course, we normally have no knowledge of σ_b^2, as we only observe the values *with* error.

Code Box 5.6: Simulating regression dilution, *continued*

```
dilut05 <- as.data.frame(t(replicate(20, coef(lm(lgwage ~ I(immi + rnorm(length(
    immi), sd=.05)))))))   # Very little noise: sd = 0.05
dilut1 <- as.data.frame(t(replicate(20, coef(lm(lgwage ~ I(immi + rnorm(length(
    immi), sd=.1)))))))    # Twice as much noise: sd = 0.1
colnames(dilut1) <- colnames(dilut05) <- c("Intercept", "slope")

summary(dilut05)   # For small noise: sd = 0.05

|    Intercept           slope
|  Min.   :0.01574   Min.   :-0.06093
|  1st Qu.:0.01621   1st Qu.:-0.05854
|  Median :0.01672   Median :-0.05684
|  Mean   :0.01674   Mean   :-0.05696
|  3rd Qu.:0.01711   3rd Qu.:-0.05518
|  Max.   :0.01798   Max.   :-0.05345

 summary(dilut1)   # For more noise: sd = 0.1

|    Intercept            slope
|  Min.   :0.007034   Min.   :-0.02905
|  1st Qu.:0.007198   1st Qu.:-0.02710
|  Median :0.007489   Median :-0.02568
|  Mean   :0.007556   Mean   :-0.02589
|  3rd Qu.:0.007903   3rd Qu.:-0.02441
|  Max.   :0.008360   Max.   :-0.02391
```

true value X^* of each observation (or measurement) of X and then do the actual regression with the estimated values X^* (Fuller 1987; Stoklosa et al. 2015). But the devils therein really are hiding in the details.

5.3 The intended, rather than the realised, manipulation is an admissible predictor variable

In the previous section, we introduced the idea that errors in predictor variables bias estimators in regression models such as ordinary least squares models that assume that predictor variables are measured without error. A similar issue arises when not all continuous or ordinal treatment levels are as accurate as intended or expected. For example, we might set the temperature of our incubators to 8°C, but we actually record daily temperatures between 6 and 10°C. The question then is, should we be using the *intended* value of the predictor variable—here, 8°C—in our model, or the *realised* temperatures of 6–10°C in our regression model?[7] If the realised value randomly fluctuates around the intended value, we are in the exact same position as in the previous section. However, when the realised

[7] We assume in the rest of this section that the actual measurements of the predictor variables are measured without error.

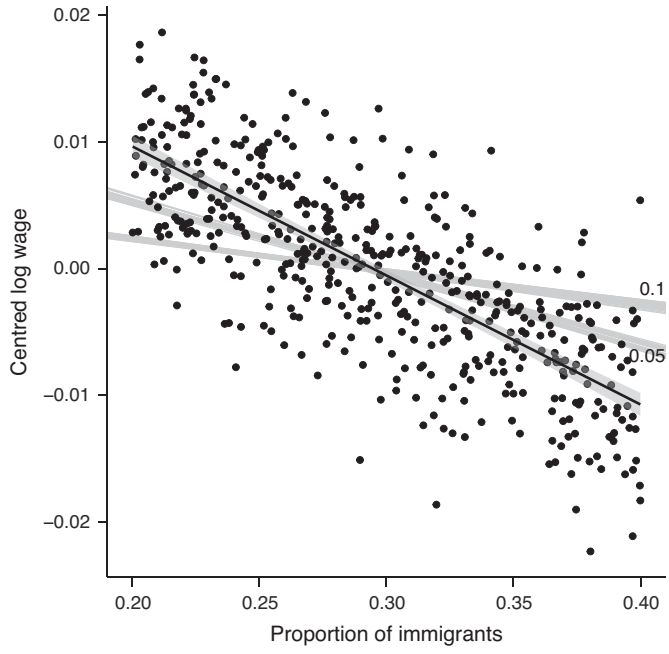


Figure 5.1 Effect of errors in the predictor variables on the estimate of the regression slope. The simulated data are the black symbols and the true (simulated) relationship between the proportion of immigrants in the workforce and wages (centred on log_{10}-transformed data) is indicated by the black regression line. The two sets of the 20 simulated replicates to which Gaussian noise was added to the predictor variable, with $\sigma = 0.05$ or $\sigma = 0.1$, are shown as as grey lines.

value systematically deviates towards lower or higher values, we have a different situation at hand.

5.3.1 A case study: Effects of species diversity on ecosystem function

Ecologists do field experiments to explore the effect of the number of species (species "richness" or "diversity") on various measures of ecosystem "functions" such as productivity, nutrient cycling, or weed infestation (e.g., Tilman & Downing 1994). The results of these experiments typically are presented as a graph with plant species richness (S) on the x-axis. Yet, the plotted values of S are (without any exception of which we are aware) the intended S (i.e., the number of species that were planted in the experiment), not the realised S. The realised value of S often is different from the intended S because, despite the best efforts of the researchers, monocultures become invaded by weeds and high-diversity plots inevitably lose some of their species over the course of the experiment.

Weisser et al. (2017) synthesised 15 years of data from the Jena Experiment, Europe's well-known grassland experiment on the effects of plant species richness on various ecosystem functions. This synthesis provides the background for the following simulation study to answer the question: Does it matter whether the intended or realised species richness is used in the analysis? Our discussion in the previous section of errors in predictor variables suggests that it does matter, but we need to determine for this specific situation whether a regression-dilution effect is actually important (Legendre & Legendre 2012; Legendre 2022).

Code Box 5.7: The relationship between intended and realised species richness

```
jena <- read.csv("../data/JenaExperiment/487_3_data.csv", stringsAsFactors = T)
    # (Local copy)
jena22 <- jena[jena$year==2022 & jena$season=="Total",]
with(jena[jena$year==2022 & jena$season=="Total",], table(sowndiv, realizedSR))
```

```
|         realizedSR
| sowndiv  1  2  3  4  5  6  7  8  9 10 11 12 29 30 33 38
|       1 14  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
|       2  0 16  0  0  0  0  0  0  0  0  0  0  0  0  0  0
|       4  1  5  8  2  0  0  0  0  0  0  0  0  0  0  0  0
|       8  0  1  1  2  1  6  2  3  0  0  0  0  0  0  0  0
|       16 0  0  0  0  0  1  0  0  2  6  3  2  0  0  0  0
|       60 0  0  0  0  0  0  0  0  0  0  0  0  1  1  1  1
```

```
cor(jena22[, "sowndiv"], jena22[,"realizedSR"])
```

```
| [1] 0.980799
```

```
Sint <- jena22$sowndiv
Sreal <- jena22$realizedSR
cor(Sint, Sreal)
```

```
| [1] 0.9674996
```

5.3.2 Simulating changes in species number and their effect on ecosystem function

The intended species richness levels in the Jena Experiment were 1, 2, 4, 8, 16, and 60 species of plants, but the synthesis states that these levels were maintained only in 46 of the 60 plots, i.e., 23% had unintended values. We first analyse the relationship between the intended and the realised species richness data from the experiment (Code Box 5.7).[8] We then use these data to simulate the effects of the realised species richness on ecosystem function (Code Box 5.8).

The correlation between intended and realised S is very high (indeed, it was reported as "closely related" by Weisser et al. 2017). So maybe we shouldn't expect too much of an effect of using one or the other as predictor. Note that none of the plots gained species, not even the monocultures. This may be more related to the fact that richness was assessed after the plots were weeded, not before. In other words, we observe a systematic but non-constant bias in the realised S.

Following the meta-analysis of Cardinale et al. (2006), species-richness effects can be described best by a Michaelis-Menten equation with parameters $Y_{\max} = 2 \pm 0.5$ and $K = 1 \pm 0.5$. We included Gaussian noise in our model ($\sigma = 15\%$ of $Y_{\max}$, based on ocular

[8] We are indebted to Anne Ebeling, University of Jena, Germany, for making these available as dataset #487 on https://jexis.uni-jena.de.

Code Box 5.8: Effect of species richness on ecosystem function

```
K=1 # +/- 0.5 95%CI
Ymax=2 # +/- 0.5 95%CI   # Cardinale et al. 2006
effect.real <- Ymax*Sreal/(K+Sreal) + rnorm(82, 0, 0.15*Ymax)
```

estimation from Figure 18 of Weisser et al. 2017). We used these parameter values to simulate some ecosystem function Y (Code Box 5.8).

5.3.3 Analysis of a single simulation

Having simulated an effect of S on Y, we next compared analyses of Y as functions of intended (S_{int}) or realised (S_{real}) species richness (Code Box 5.9). To do so we fit a Michaelis-Menten function to the data to estimate its two parameters:[9]

The difference between using the two models is not large and both models seem to under- or overestimate parameters in a similar way (whose true values, which we defined in Code Box 5.8, are 2 and 1, respectively).

We use the code in Code Box 5.10 to plot the observed change in species numbers (Figure 5.2, left)[10] and the simulated and fitted effects of S_{int} and S_{real} on ecosystem function Y (Figure 5.2, right).[11]

5.3.4 Analysis of multiple simulations

As you might expect, we need to repeat the simulations many times to investigate whether and by how much the estimated effect of species richness, represented by the values of Y_{max} and K, differs between S_{int} and S_{real} (Code Box 5.11).

The results of the full set of simulations suggest a very small effect of using intended rather than realised species richness, with their means being very similar (Figure 5.3). However, and not unexpectedly, using S_{int} resulted in biased estimates of both parameters, whereas using S_{real} gave unbiased parameter estimates.

The results of these simulations caution us not to jump to conclusions about "problematic" analyses. Our intuition, combined with the discussion of regression dilution in §5.2, would have suggested a strong effect of uncertainty in plant species richness and should have led us to using the realised predictor values, not the intended ones, in our analyses. With the amount of noise that was actually observed in the field, however, what appeared to be a problem turned out to be somewhat statistically biased but largely irrelevant in the "real world."[12] Without simulation, it would have been difficult to be convinced of that.

[9] We did this using the non-linear least square regression function (`nls()` in `R`), to which we provided the mathematical function of the Michaelis-Menten kinetic equation ($f_{MM}(S) = \frac{Y_{\text{max}}S}{K+S}$) and starting values for the optimiser as a named list.

[10] Note that, in line 4 of Code Box 5.10, the `ifelse`-statement serves to plot only arrows for those points that actually change in the S-value. That leads to some arrows not being drawn, as reported by the warning message.

[11] In the commands for the second plot in Code Box 5.10, the syntax `\(x)` on lines 15–17 represents an "anonymous" function, i.e., it is defined and computed "on the fly." The longer version of this command would be `function(x)`.

[12] Indeed, because the observed variance in Y was not three-fold greater than the observed variance in S, Legendre's rule-of-thumb of using an OLS model in this case was borne out.

Code Box 5.9: A single comparison of the effects of intended and realised species richness on ecosystem function

```
# Fitting the model based on intended species richness
 summary(fmJ <- nls(effect.real ~ Ymax*Sint/(K+Sint), start=list("Ymax"=2,
     "K"=1)))

| Formula: effect.real ~ Ymax * Sint/(K + Sint)
|
| Parameters:
|      Estimate Std. Error t value Pr(>|t|)
| Ymax  1.97958    0.06203  31.911  < 2e-16 ***
| K     1.07626    0.15255   7.055 6.19e-10 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.2651 on 78 degrees of freedom

# Fitting the model based on actual (realized) species richness
 summary(fmC <- nls(effect.real ~ Ymax*Sreal/(K+Sreal), start=list("Ymax"=2,
     "K"=1)))

| Formula: effect.real ~ Ymax * Sreal/(K + Sreal)
|
| Parameters:
|      Estimate Std. Error t value Pr(>|t|)
| Ymax  2.09909    0.07021  29.899  < 2e-16 ***
| K     1.12253    0.14547   7.716 3.33e-11 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.2501 on 78 degrees of freedom
```

5.4 ANOVA requires homoscedasticity

Analysis of Variance (ANOVA) has two important assumptions: that the data are drawn from a normal distribution around the means of each group (as in linear regression: §5.1); and that the variances in these groups are equal (e.g., Underwood 1997; Potvin 2001; Quinn & Keough 2002). The latter condition is also known as homoscedasticity. Many authors have asserted that homoscedasticity is more important than normality for reliable interpretation of the results of an ANOVA because heteroscedasticity (inequality of variances) can affect the *F*-statistic (Rogan & Keselman 1977; Coombs et al. 1996; Underwood 1997). We set up and initially sketched out this problem in §2.2.2 of Chapter 2.

The *F*-statistic (a.k.a. *F*-ratio) calculated in an ANOVA is the ratio of explained to residual mean squared errors (MS):

$$F = \frac{\mathrm{MS}_{\mathrm{explained}}}{\mathrm{MS}_{\mathrm{residual}}} = \frac{\mathrm{SS}_{\mathrm{effect}}/\mathrm{df}_{\mathrm{effect}}}{\mathrm{SS}_{\mathrm{residuals}}/\mathrm{df}_{\mathrm{residuals}}}, \tag{5.1}$$

Code Box 5.10: Plot of the difference between models with intended or realised species richness

```
# Plot the change in species numbers (Sint --> Sreal)
 dats <- cbind.data.frame(Sreal, Sint, effect.real, effect.int)
 ggplot(dats) +
   geom_segment(aes(x=ifelse(Sint==Sreal, NA, Sint), y=effect.int, xend=Sreal,
        yend=effect.real), arrow = arrow(length=unit(.15, "inches"), angle=15,
        type="closed"), colour="grey80", linejoin="mitre") +
   geom_point(aes(y=effect.int, x=Sint), colour="grey50") +
   geom_point(aes(y=effect.real, x=Sreal)) +
   scale_x_continuous(trans='log2') +
   labs(x = "species richness", y = "ecosystem function") +
   theme_minimal()

# Plot the simulated (''true'') and fitted effects of Sint and Sreal on
    ecosystem function Y
 ggplot(dats) +
   geom_point(aes(y=effect.int, x=Sint), colour="grey50") +
   geom_point(aes(y=effect.real, x=Sreal)) +
   geom_function(fun = \(x) 2*x/(1+x), linetype=1) +
   geom_function(fun = \(x) coef(fmJ)[1]*x/(coef(fmJ)[2]+x), colour = "grey70",
        linetype=2, size=1) +
   geom_function(fun = \(x) coef(fmC)[1]*x/(coef(fmC)[2]+x), colour = "black",
        linetype=2, size=1) +
   scale_x_continuous(trans='log2') +
   labs(x = "species richness", y = "ecosystem function") +
   annotate("point", x=12, y=c(.6, .7), col=c("grey70", "black")) +
   annotate("text", x=13, y=c(.7, .6), label=c("realised richness", "intended
        richness"), hjust=0, col=c("black", "grey60")) +
   theme_minimal()
```

where df is the appropriate degrees of freedom and SS is the sum of squared deviations from the mean. The *F*-distribution, from which the significance of the ANOVA is computed, assumes that the ratio of the mean squared errors (Equation 5.1) is representative for all groups. If one group has much larger variances than the others, then the residual sum of squares will be dominated by the other groups and estimates of the within-group variance will be too small. As a result, the estimated *F*-values will be too large. The effect of deviation from homoscedasticity was evaluated through simulation by Wilcox et al. (1986), who showed that a 4:1 ratio of standard deviations approximately doubled the type-I error rate (from 0.05 to 0.1).

5.4.1 A case study: Differences in butterfat content among cattle breeds

Dairy products such as milk, butter, and cheese are a mainstay of people's diets worldwide, and consumption of high-fat dairy products are forecast to continue increasing at a steady rate for the near future (OECD/FAO 2020). Because production of dairy products from cattle contributes to global climate change via land-use changes and methane emissions, there are

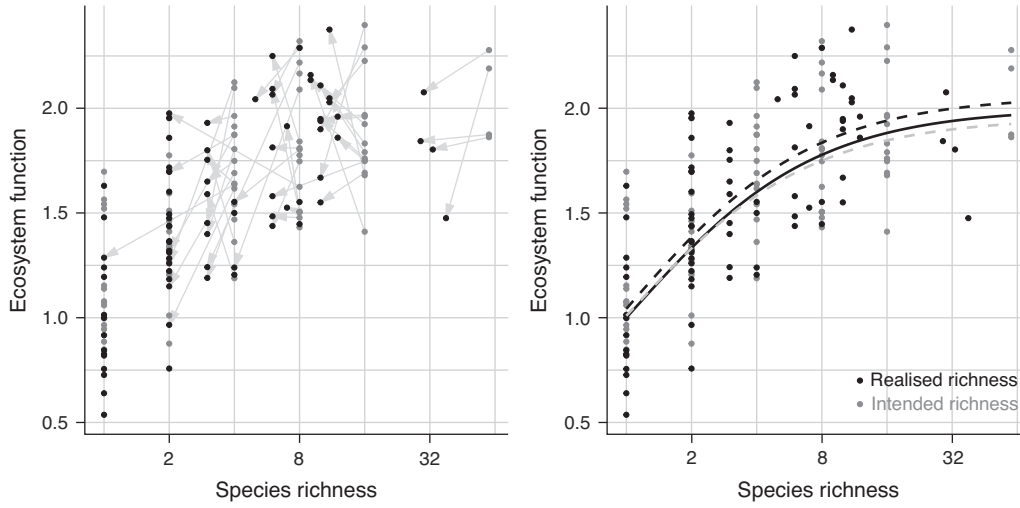


Figure 5.2 Simulated effect of plant species richness on some ecosystem function. Points and dashed lines indicate intended (grey) and realised (black) values of *S*; the solid line is the simulated truth. *Left:* During the experiment, the intended species richness changed for ≈ 20% of the plots, as indicated by the arrows. *Right:* Fitting a curve to the intended, rather than realised, species richness gave similar results, although the latter overestimated the effect of *S* on *Y* more than the former underestimated it.

incentives to increase the intensity of production by increasing the butterfat content of raw milk. Butterfat content varies by cattle breed, and perhaps unsurprisingly, there is a wealth of dairy data available to be mined (Hutchins & Hueth 2023).

We examined a classic dataset on the butterfat content in five cattle breeds extracted from Canadian records and published by Sokal & Rohlf (1995). The data are butterfat percentages in milk sampled at random from 10 mature (≥ 5 years old) and 10 two-year-old cows from each of five cattle breeds. We are interested in testing whether there are significant differences in butterfat production among breeds. Figure 5.4 illustrates the data for each breed with their age classes combined. Because the predictor variable (breed) is categorical, we use an ANOVA to test the null hypothesis of no difference in butterfat content among breeds.

5.4.2 Testing the effect of homoscedasticity using simulation

We address two questions with our simulations. First, is the overall ANOVA robust to the substantial heteroscedasticity apparent in the data (Figure 5.4)? And, second, does this heteroscedasticity affect post hoc comparisons between specific breeds? Note that in both cases we can, and probably should, use methods that can handle heteroscedasticity, such as generalised least squares and mixed-effect models. The point we make with this simulation is thus one of demonstrating an effect, not optimising the analysis.

QUESTION 1: IS THE ANOVA ROBUST TO SUBSTANTIAL HETEROSCEDASTICITY?

We compared the *F*-value from the original data with simulations in which all five breeds have the same (mean) variances (Code Box 5.12). The *F*-value calculated for the observed heteroscedastic data, while slightly lower than that generated by simulations using homoscedastic data, is still far greater than 5.0, the critical value of $F_{4,95}$ (Figure 5.5). We conclude that the result of the overall ANOVA was not qualitatively affected by the heteroscedasticity in the data.

Code Box 5.11: Multiple simulations of the effects of intended and realised species richness on ecosystem function

```
# Set parameters
 K=1
 Ymax=2
 N = 1000
 Sint <- rep(c(1,2,4,8,16,60), times=c(14, 16, 16, 16, 14, 4) ) # From jena487

# Simulation
 set.seed(123)
  simu.results <- data.frame("Ymax.real"=rep(NA, N), "K.real"=NA, "Ymax.int"=NA,
      "K.int"=NA)
 for (i in 1:N){
     # Generate Sreal-values (from Code Box 5.7):
     Sreal.simu <-  c(sample(c(1,2), 14, prob=c(0.9, 0.1), replace=T), #1
                      sample(c(2,3,4), 16, prob=c(.85, .1, .05), replace=T), #2
                      sample(c(1,2,3,4), 16, prob=c(1/16, 5/16, 1/2, 1/8),
                          replace=T), #4
                      sample(2:8, 16, prob=c(1,1,2,1,6,2,3)/16, replace=T), #8
                      sample(6:12, 14, prob=c(1,0,0,2,6,3,2)/14, replace=T), #16
                      sample(c(29,30,33,38), size=4, prob=rep(1/4, 4), replace=T)
                          ) #60
     # Compute real effect:
     effect.real.simu <- Ymax*Sreal/(K+Sreal.simu) + rnorm(length(Sreal.simu), 0,
          .15*Ymax)
     # Fit MM-kinetics, once on real, once on intended S:
     fmC <- nls(effect.real.simu ~ Ymax * Sreal.simu / (K + Sreal.simu), start=
         list("Ymax"=2, "K"=1))
     fmJ <- nls(effect.real.simu ~ Ymax * Sint/(K + Sint), start=list("Ymax"=2
          , "K"=1))
     simu.results[i,] <- c(coef(fmC), coef(fmJ))
     rm(Sreal.simu, fmC, fmJ)
 }
 summary(simu.results)
```

```
|    Ymax.real         K.real          Ymax.int         K.int
|  Min.   :1.750   Min.   :0.5382   Min.   :1.670   Min.   :0.4506
|  1st Qu.:1.940   1st Qu.:0.8740   1st Qu.:1.821   1st Qu.:0.7321
|  Median :1.996   Median :0.9934   Median :1.868   Median :0.8338
|  Mean   :1.999   Mean   :0.9975   Mean   :1.870   Mean   :0.8508
|  3rd Qu.:2.059   3rd Qu.:1.1148   3rd Qu.:1.919   3rd Qu.:0.9619
|  Max.   :2.210   Max.   :1.5272   Max.   :2.078   Max.   :1.4374
```

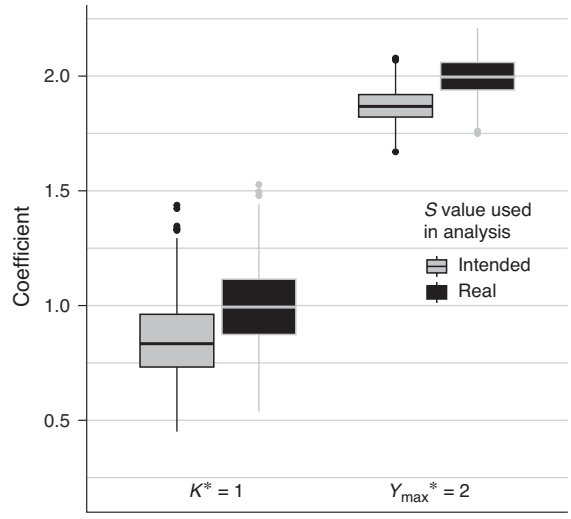


Figure 5.3 Estimates of K and Y_{max} for 1,000 simulated Jena experiments (true values are indicated by a * below the boxes). Realised S estimates the true values without bias, while using intended S yields underestimations.

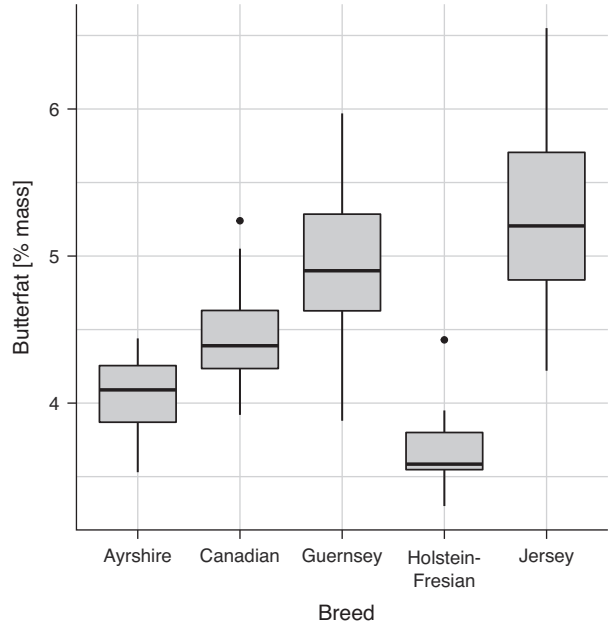


Figure 5.4 Butterfat (in % of mass) from five breeds of cattle. Note the different variances, which Levene's test identifies as significantly heteroscedastic ($p = 0.016$). In lieu of an ANOVA, one could analyse these data using Welch's (1951) generalisation of the t-test for groups with unequal variances or a non-parametric Kruskal-Wallis test.

Code Box 5.12: Overall effect of heteroscedasticity on a test of significance using ANOVA

```
#Load required libraries
 library(faraway) # For butterfat dataset
 library(car) # For Levene's test

 fm <- aov(Butterfat ~ Breed, data=butterfat)
 summary(fm)
|             Df Sum Sq Mean Sq F value Pr(>F)
| Breed        4  34.32   8.580    49.8 <2e-16 ***
| Residuals   95  16.37   0.172
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

 fligner.test(Butterfat ~ Breed, data=butterfat)

|   Fligner-Killeen test of homogeneity of variances
|
| data:  Butterfat by Breed
| Fligner-Killeen:med chi-squared = 12.364, df = 4, p-value = 0.01484

car::leveneTest(fm)

| Levene's Test for Homogeneity of Variance (center = median)
|       Df F value  Pr(>F)
| group  4  3.1914 0.01661 *
|       95
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

 with(butterfat, tapply(Butterfat, Breed, var))

|         Ayrshire         Canadian         Guernsey Holstein-Fresian
|       0.06806316       0.13360289       0.23357895       0.06701553
|
|           Jersey
|       0.35918816

 mean(with(butterfat, tapply(Butterfat, Breed, sd)))

| [1] 0.3935806

# One simulation
 set.seed(123)
 butterfat$Sim <- rnorm(100, mean=rep(with(butterfat, tapply(Butterfat, Breed,
     mean)), each=20), sd=0.394)
 fmSim <- aov(Sim ~ Breed, data=butterfat)

# Multiple simulations
```

```
set.seed(234)
fmList <- list()
for (i in 1:1000){
  butterfat$Sim <- rnorm(100, mean=rep(with(butterfat, tapply(Butterfat, Breed,
      mean)), each=20), sd=0.394)
  fmList[[i]] <- aov(Sim ~ Breed, data=butterfat)
}
Ps <- sapply(fmList, \(x) summary(x)[[1]]$"Pr(>F)"[1])
Fs <- sapply(fmList, \(x) summary(x)[[1]]$F[1])
mean(Fs)
```

```
[1] 57.30481
```

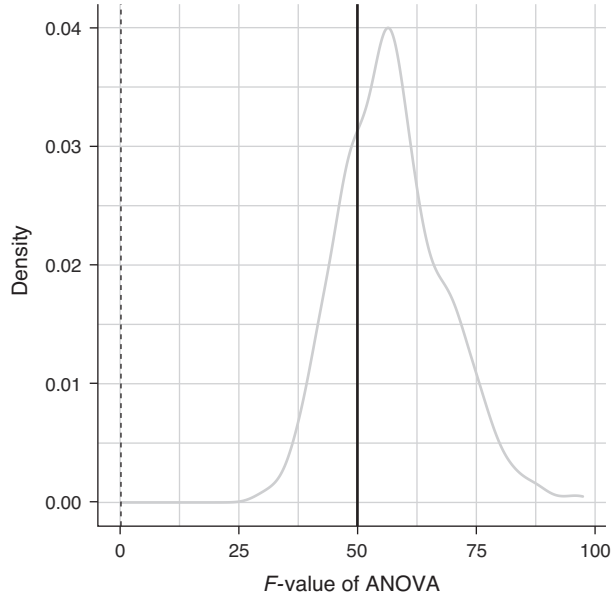


Figure 5.5 *F*-values of ANOVA testing for the effect of cattle breed on butterfat content. The solid vertical line is the *F*-value calculated for the observed, heteroscedastic data (Figure 5.4) and the distribution (grey curve) shows the *F*-values for simulated homoscedastic data (Code Box 5.12). The dashed vertical line at 5 is the critical value for $F_{4,95}$ for an acceptable type-I error rate ($\alpha = 0.05$).

Whether or not heteroscedasticity in a given dataset quantitatively affects statistical results depends on both its size and the differences among groups. The simulation here shows that such an effect can be substantial. It also suggests using the mean of the simulated *F*-values as a better estimate of the correct variance ratio, which can then be used to identify the corresponding *p*-value of the appropriate *F*-distribution.[13]

[13] In this case, the code `pf(57.3, 4, 95, lower.tail=F)` yields a value of $1.6 \cdot 10^{-24}$, rather than `pf(49.8, 4, 26, lower.tail=F)`$= 1.65 \cdot 10^{-22}$, for the model assuming homoscedasticity.

Code Box 5.13: Effect of heteroscedasticity on pairwise comparisons following ANOVA

```
TukeySim <- sapply(fmList, \(x) TukeyHSD(x)$Breed[1, 4])
summary(TukeySim)

|     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
5 | 0.000000 0.002323 0.025565 0.124206 0.128577 1.000000

sum(TukeySim > 0.05) / length(TukeySim)

| [1] 0.409
```

QUESTION 2: DOES HETEROSCEDASTICITY AFFECT POST HOC COMPARISONS?

We used the same simulation to compare the butterfat of the Canadian and Ayrshire breeds, which, using the observed heteroscedastic data, had been identified as being significantly different ($P = 0.038$; Tukey's HSD test).

Using simulation, however, the summary of pairwise-difference tests indicated a very skewed distribution (with the mean being much larger than the median), and, on average, no significant difference between the two breeds (Code Box 5.13). Indeed, in 409 out of the 1,000 simulations, p-values for this comparison were greater than 0.05, suggesting that heteroscedasticity led to an incorrect conclusion that there was a significant difference in butterfat content between Ayrshire and Canadian cattle breeds.

In conclusion, heteroscedasticity can be, but need not be, problematic for ANOVA. With several groups, the results of the overall ANOVA often are reasonably robust for large differences among groups, but heteroscedasticity can result in incorrect p-values for specific post-hoc comparisons between groups.[14]

5.5 Multiple testing and the inflation of false positives

One of the issues that frequently arises is that of testing of multiple hypotheses or doing multiple post hoc comparisons (Dunn 1961). The first thing to realise is that even when any individual statistical test has a nominal false positive rate of 0.05 (the conventional critical value for significance tests), conducting many such tests substantially increases the probability of finding at least one statistically significant result. For example, after 10 independent tests, the probability of a false positive is not 0.05, but 0.4, an order of magnitude greater.

The mathematical reason for this result is straightforward. The probability of a false positive for k tests is

$$p = 1 - (1 - \alpha)^k, \tag{5.2}$$

where α is the probability of a false positive and usually set equal to 0.05 (i.e., the probability of committing a type-I statistical error; see Table 3.1 in Chapter 3). This rate of detecting false positives across many independent statistical tests is called the family-wise error rate.

Where does Equation 5.2 come from? The probability of obtaining a true positive from a statistical test is 1 – the probability of false positive, α, e.g., $1 - 0.05$ (if our acceptable critical

[14] Because of the heteroscedasticity in the data, this conclusion would not differ if we used a priori (predefined) contrasts instead of post hoc (after the fact) comparisons among means. However, if you are not convinced, run the simulations in Code Box 5.13 using a priori constrasts.

Code Box 5.14: Multiple testing

```
tenner <- sample(size=10, c("different","same"), prob=c(0.05, 0.95), replace=T)
any(tenner == "different")
```

```
| [1] FALSE
```

Code Box 5.15: Multiple testing, *continued*

```
set.seed(123)
    simu10er <- replicate(10000, {
    tenner <- sample(size=10, c("different","same"), prob=c(0.05, 0.95), replace=T)
    any(tenner == "different")
})
sum(simu10er)/length(simu10er)
```

```
| [1] 0.3977
```

value of committing a type-I statistical error is set at $\alpha = 0.05$). For k such independent tests, each has this same probability, so the probability of k true positives is $(1-\alpha)^k$. Subtracting this value from 1 gives the probability of k false positives: $1-(1-\alpha)^k$. Thus, for $k=10$ independent tests and an acceptable type-I error rate $\alpha = 0.05$, the family-wise error rate of 10 false positives is $p = 1-(1-0.05)^{10} = 0.401$.

5.5.1 Adjusting statistical tests to account for family-wise error rates

We are interested in determining the appropriate critical value of α for an individual test that would result in a family-wise error rate equivalent to the 5% type-I error rate that we normally accept for a single test.

First, we use simulation to confirm the mathematical derivation (Equation 5.2). For example, if our individual statistical test is a t-test of two groups of observations drawn from the same distribution, there should be no significant difference between the two groups. Because the t-test has nominal coverage and should have a 5% type-I error rate (see §2.3.4 in Chapter 2), it should give the correct result of no difference between the groups (we call this result "same"; the alternative is "different") in 95% of the cases. For the simulation we draw 10 times from "same" and "different", with (incorrect) "different" showing up with a probability of 0.05. If any of these ten draws is "different", we have discovered a false-positive (Code Box 5.14).

By repeating this simulation many times and seeing how often we discover a (non-existent) difference, we can estimate the family-wise error rate (Code Box 5.15).

With a 5% nominal coverage, we would expect to get, on average, one false positive in every 20 such simulations. In fact, the result of the simulation, 39.7%, is very close to the 40.1% calculated using the analytical solution (Code Box 5.15), and both are eight times higher than the expected, and acceptable, type-I error rate. Therefore, if we do find ourselves needing to do multiple independent hypothesis tests on the same dataset, we need to adjust the acceptable type-I error rate α for each individual test so that the overall family-wise error rate is our usual 5%.

Code Box 5.16: Multiple testing, *continued*

```
set.seed(1234)
FWER <- NA
j=1
for (i in 10^seq(-4, -1, by=0.1)) {# Log-scaled sequence
    simu10er <- replicate(1000, {
        tenner <- sample(size=10, c("different","same"), prob=c(i, 1-i), replace
            =T)
        any(tenner == "different")
    })
    FWER[j] <- sum(simu10er)/length(simu10er)
    j <- j + 1 # count up for each loop
}
FWER

| [1] 0.000 0.001 0.001 0.002 0.005 0.003 0.003 0.011 0.011
| [10] 0.007 0.013 0.016 0.017 0.024 0.032 0.029 0.036 0.049
| [20] 0.064 0.085 0.089 0.128 0.157 0.158 0.226 0.280 0.318
| [30] 0.378 0.490 0.558 0.660
```

The simplest adjustment is the Bonferroni correction, which sets a new critical value $\alpha' = \alpha/k$ for k independent tests (for further discussion of the Bonferroni correction and other adjustments, see Dunn 1961). So, for 10 independent tests, our adjusted critical value would be $\alpha' = 0.05/10 = 0.005$ and only tests that returned a p-value ≤ 0.005 would be considered to be statistically significant.

To understand the Bonferroni correction further, we revisit the multiple testing problem for different values of α. First, we can solve this analytically for a desired family-wise error rate = 0.05:

$$\begin{aligned} 0.05 &= 1 - (1-\alpha)^k \\ -(0.05-1) &= (1-\alpha)^k \\ 0.95^{1/k} &= 1-\alpha \\ \alpha' &= 1 - 0.95^{1/k}. \end{aligned} \tag{5.3}$$

Thus, for 10 comparisons, our the new critical value α' would be equal to $1 - 0.95^{1/10} = 0.0051$.[15]

We could use simulation to verify the analytical solution and to explore the outcome for different initial values of α (Code Box 5.16; results in Figure 5.6).

Although we plotted the results of our simulation on a logarithmic scale (Figure 5.6) to make the different values of α' more legible, on a linear scale the curve would be almost

[15] This value, 0.0051, is larger (by 0.0001) and therefore less stringent than the Bonferroni correction of 0.005. The derivation shown in Equation 5.3 is credited to Zbyněk Šidák (1967). But in practice, the 1/100% difference rarely matters and the Bonferroni correction is quicker to compute and easier to remember.

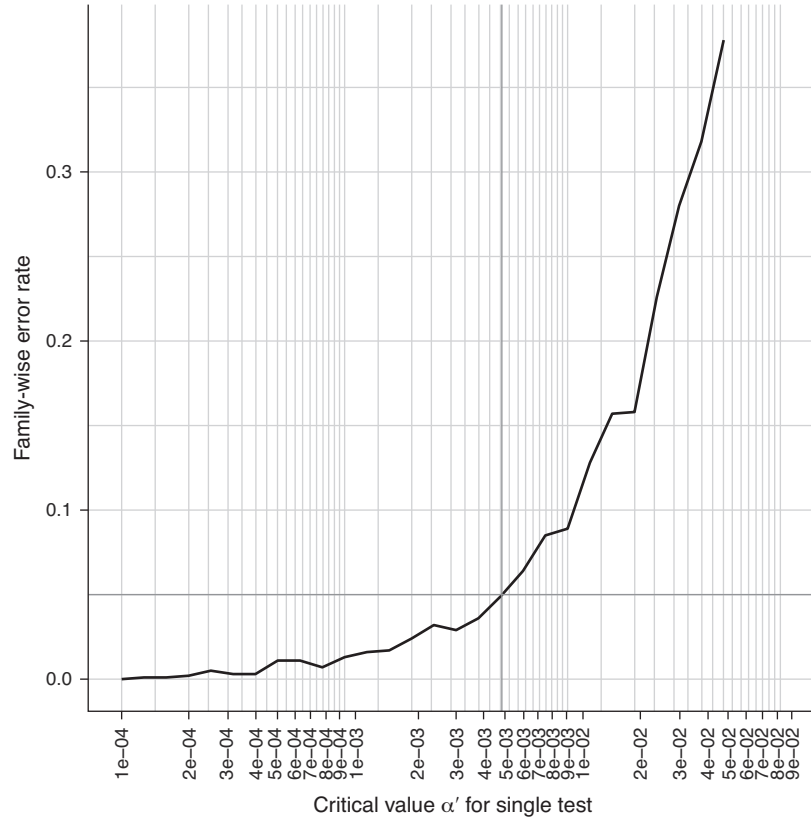


Figure 5.6 Family-wise error rate for $k = 10$ independent tests for different single-test α values. The horizontal line is the conventionally accepted type-I error rate = 0.05 and the vertical line is the single-test adjusted critical value $\alpha' = 0.0051$ (i.e., a family-wise error rate of 0.05). Note that the x-axis is logarithmically scaled.

a straight line; the adjustment per test is nearly constant, as would be expected from the Bonferroni correction. Recall from Figure 3.3 in Chapter 3 that statistical power (1 – the probability of committing a type-II statistical error) declines non-linearly with a decrease in the critical value α. Thus, adjusting the acceptable family-wise error rate to any $\alpha' < \alpha = 0.05$ will reduce the power to detect an effect of our experiment, if the effect in fact exists. The alternative to adjusting for the family-wise error rate that does not affect statistical power as strongly is to adjust for the false discovery rate (Benjamini & Hochberg 1995).

5.5.2 Adjusting for the false discovery rate

The false discovery rate (FDR) is defined as the proportion of significant tests that are wrong. The logic behind this approach is that only significant outcomes (i.e., rejections of the null hypothesis) are relevant for the error rate, and so we only need to adjust for those, not for the total number of tests (as is done by the Bonferroni correction).[16] The simplest FDR correction, called the Benjamini-Hochberg procedure, has three steps. First, sort the p-values resulting from the multiple tests in ascending order. Second, for the m tests at $\alpha = 0.05$, compute $k\alpha/m$ for $k \in \{1, \ldots, m\}$. Third, compare those values to the observed and sorted

[16] Note that if *all* the possible null hypotheses being tested are true, then the FDR equals the family-wise error rate (FWER). But if only some of the null hypotheses are true, then FDR < FWER.

p-values and find the value of k for which all the values to the left are smaller than the p-values, i.e., $\max_k\{\forall\, k\colon\, p_{(k)} \leq k\alpha/m\}$. The conclusion is that only those values that satisfy the aforementioned condition are actually significant (Code Box 5.18).

5.5.3 A case study: The root of happiness

Money alone does not make us happy, but what else does? Before we can find out and identify therapies to make us happier, we need to define a measure of happiness (e.g., Diener 2000). Regrettably, happiness is not something researchers simply can measure with a blood test or an MRI scan. Rather, researchers use self-reported answers to formal questionnaires to determine how happy individuals are and the correlates of their happiness (e.g., Rastelli et al. 2021). From these questionnaires, different indices and scales of happiness have been developed, but there are no universally accepted measures of it (Diener 2000). Thus, psychological studies typically employ a range of happiness scores and test many or all of them using the same dataset. This approach is liable to a dramatic increase in the risk of committing a type-I error.

Rastelli et al. (2021) reported a mindfulness-based therapeutic intervention and investigated its effect on an individual's self-assessment of happiness and well-being. The authors used six different questionnaires, which together included a total of 22 measures of happiness, and analysed the before-after comparisons data using one-way ANOVA (see also the discussion of BACI designs in §4.5.4 of Chapter 4). Using Equation 5.2, the expected probability of finding a significant effect (at $p < 0.05$), even if there is no effect, is $1 - (1 - 0.05)^{22} = 0.676$.

FAMILY-WISE ERROR RATES VERSUS FALSE DISCOVERY RATES: A SIMULATION

If our goal for these 22 tests is a type-I error with a family-wise error rate of 0.05, we might use a Bonferroni correction and use a new critical value $\alpha' = 0.05/22 = 0.0023$. However, this small value of α' would reduce our statistical power to detect an effect among the 29 participants in Rastelli et al.'s (2021) study to zero. Using simulation, we illustrate how using the FDR instead of the family-wise error rate improves on the Bonferroni correction.

First, we will show that the probability of any of them being significant is, indeed, 0.68—the family-wise error rate for 22 t-tests of simulated data with no differences between groups. We then show that a procedure to control for this based on the FDR leads to family-wise error rates at the desired level of 0.05. Finally, we simulate an additional effect into 10% of the replicates and show that, perhaps unsurprisingly, this will lead (without correction) to an even higher family-wise error rate, which again is correctly controlled for using an FDR approach.

SIMULATING THE FAMILY-WISE ERROR RATE

We can easily re-purpose our earlier code (Code Boxes 5.14–5.16) to simulate the family-wise error rate for $N = 29$ participants and $k = 22$ independent tests of a true null hypothesis (Code Box 5.17).

In 1,000 simulations (Code Box 5.17) 67.8% of them reported at least one of the 22 measures as being significant, despite our having coded the null as true ("not significant") in all cases. This is essentially identical to the analytical solution of 67.6%.

CORRECTING FOR FALSE DETECTION RATE

We turn the three-step algorithm of the Benjamini-Hochberg procedure described in §5.5.2 into code (Code Box 5.18).

Using the Benjamin-Hochberg procedure, we recover the approximately nominal level of $\alpha = 0.05$, suggesting that it is reliable when there is no effect of the treatment.

Code Box 5.17: Multiple testing without effect: The problem

```
# A single simulation
 N <- 29
 k <- 22
 onesim <- replicate(k, {
   sim <- matrix(rnorm(2*N), ncol=2)
   t.test(sim)$p.value
 })

# Count if ANY p<0.05 (family-wise significance)
 any (onesim < 0.05)

| [1] FALSE

# Multiple simulations
 set.seed(321)
 simus22 <- replicate(1000, {
   onesim <- replicate(k, {
     sim <- matrix(rnorm(2*N), ncol=2)
     t.test(sim)$p.value
   })
   any (onesim < 0.05)
 })
 mean(simus22)

| [1] 0.678
```

USING THE FDR WORKS WELL EVEN WHEN THERE ARE FALSE NULLS

Finally, we want to see how the FDR algorithm works in the presence of a real effect in, say, 10% of the cases (i.e., we know the null hypothesis is false for 10% of the individuals; the value of `prop.true` we define in Code Box 5.19).

The result of this simulation—the number of significant effects (`mean(simusFDR true)`)—may be unexpected. We found a significant effect in 15.3% of the replicates, not the 10% that we defined in the `prop.true` term in our simulation. Why? The number of significant effects is the sum of the actual number true effects (which we set at 10%) *plus* the 5% error accepted as our critical α-level.

We report the proportion of false significances—5 out of 15%—as the FDR: one-third of our significant findings are false (the output 0.346 from the computation in the last line of code). This FDR (34%) is a function of the proportion of true effects (i.e., the prevalence of true effect; here, 10%). Further, the rarer a true effect, the larger the proportion of false discoveries. In contrast, as the true effect increases, the FDR decreases.[17] This phenomenon, known as the "base rate fallacy" (Bar-Hillel 1980), is independent of the problem of multiple

[17] For example, setting `prop.true` equal to 0.5 (i.e., 50%) results in an FDR close to 0.05.

Code Box 5.18: Multiple testing without effect: Benjamini-Hochberg procedure

```
# A single simulation
 N <- 29
 k <- 22
 onesim <- replicate(k, {
   sim <- matrix(rnorm(2*N), ncol=2)
   t.test(sim)$p.value
 })

# Sort in ascending order
 sort(onesim)

|  [1] 0.2058209 0.2392423 0.2725505 0.3077748 0.3133248 0.3165211 0.3752846
|  [8] 0.3994587 0.4002993 0.4033543 0.4049102 0.4284226 0.4537670 0.5914760
| [15] 0.6000132 0.7124927 0.7492348 0.8783858 0.9132764 0.9342046 0.9347411
| [22] 0.9954170

# Find the largest k such that P_k < k / m * alpha (m=22, alpha=0.05)
 ks <- sort(onesim) < (1:22 / 22 * 0.05)
 signifs <- max(0, which(ks == TRUE)) # Largest k
 signifs > 0

| [1] FALSE

# Multiple simulations
 k <- 22
 set.seed(321)
 simusFDR <- replicate(1000, {
   onesim <- replicate(k, {
     sim <- matrix(rnorm(2*N), ncol=2)
     t.test(sim)$p.value
   })
   ks <- sort(onesim) < (1:k / k * 0.05)
   signifs <- max(0, which(ks == TRUE)) # largest of ks
   signifs > 0
 })
 mean(simusFDR)

| [1] 0.041
```

testing. That is to say: the principle of correcting for the false-discovery rate works, within the limits set by the base rate.[18]

[18] The base rate fallacy also is known as the prosecutor's fallacy because when the base rate is very low, diagnostic tests such as facial identification or DNA testing that have a false positive rate that is very low but still greater than the base rate imply that the guilt of a given individual is much more likely to be false than prosecutors (or jurors or judges) assert it to be based on the sensitivity of the tests (Thompson & Schumann 1987).

Code Box 5.19: Multiple testing with some real effect: Benjamini-Hochberg procedure

```
# A single simulation
prop.true <- 0.1
N <- 29
k <- 22
effect <- FALSE
if (runif(1) < prop.true) effect <- TRUE # Add a true effect?
if (effect) { # Real effect
  onesim <- replicate(k, {
    sim <- matrix(rnorm(2*N), ncol=2)
    sim[,2] <- sim[,2] + 2
    t.test(sim)$p.value
  })
} else{ # No effect
  onesim <- replicate(k, {
    sim <- matrix(rnorm(2*N), ncol=2)
    t.test(sim)$p.value
  })
}
ks <- sort(onesim) < (1:k / k * 0.05)
signifs <- max(0, which(ks == TRUE)) # largest k
signifs > 0
```

```
[1] TRUE
```

```
# Multiple simulations
prop.true <- 0.1
k <- 22
set.seed(321)
simusFDRtrue <- replicate(1000, {
  effect <- FALSE
  if (runif(1) < prop.true) effect <- TRUE
  if (effect) { # real effect
    onesim <- replicate(k, {
      sim <- matrix(rnorm(2*N), ncol=2)
      sim[,2] <- sim[,2] + 2
      t.test(sim)$p.value
    })
  } else{
    onesim <- replicate(k, {
      sim <- matrix(rnorm(2*N), ncol=2)
      t.test(sim)$p.value
    })
  }
  ks <- sort(onesim) < (1:k / k * 0.05)
  signifs <- max(0, which(ks == TRUE)) # Largest k
  signifs > 0
```

```
})
 mean(simusFDRtrue)
50
| [1] 0.153

# False discovery rate
 (sum(simusFDRtrue) - prop.true * 1000) / sum(simusFDRtrue)
55
| [1] 0.3464052
```

5.6 Hyper-distributions in mixed-effect models are normal

Mixed-effect models are used when individual observations are not independent, but instead are grouped within higher-order subjects or strata. The most familiar example may be the randomised block design, in which individuals subjected to different experimental treatments are first assigned to groups that share particular characteristics (e.g., sex or gender, neighbourhood of residence, elevation). Then, the different treatments are assigned to individuals within the blocks.

There can be multiple levels of different groups. For example, in evaluating student performance in schools, the assessment could be made for a random sample (or all) of the pupils in a class, for several classes in a school, for several schools in a district, for several districts in a country, and across countries. In such a highly nested setup, we would expect pupils within the same class to be more alike than pupils from different classes in the same district. The pupils in the same class are taught by the same teachers, their parents are likely to have similar socioeconomic status, and they live in similar urban or rural environments. In this case, the class would be a grouping factor within which observations of individual pupils are considered to be independent of one another.[19]

In (generalised) linear mixed-effect models (LMMs), which are the most commonly used form of mixed-effect models (e.g., Gelman & Hill 2007), the distribution of estimates for each parameter is assumed to be normal. Continuing with the above example, the mean performance per class is assumed to follow a normal distribution across all schools in a district. The mean and standard deviation of this district-level normal distribution are called "hyper-parameters," because they are not at the level of the observation, but at a "higher" level.[20]

The Central Limit Theorem ensures that *as the number of groups goes to infinity* the distributions of the estimates of the data and the hyper-parameters will be normally distributed. However, in most real cases, the number of groups (strata) is small and the lower-level estimates may deviate substantially from the assumed Gaussian distribution. Such cases can be accommodated using dedicated software packages for Bayesian data analysis (e.g., de Valpine et al. 2017; Pantel & Becks 2023; Stan Development Team 2023) that allow users to specify the distribution(s) of the observations and the hyper-parameters, but such specifications rarely are available in standard turn-key implementations of mixed-effect models.

[19] This is called "conditional independence," as within a class there is no way of predicting the performance of one pupil from that of another, but across schools or districts knowledge of the class *does* allow us to make a better-than-guessing prediction. Thus, only *conditional on knowing the class*, observations are independent.

[20] Note that there can be multiple, hierarchical levels of hyper-parameters, e.g., one of classes per school, of schools per district, of district per country. In LMMs, the distributions of all of these hyper-parameters are assumed to be normal.

Can we get away with ignoring normality assumptions of the hyper-parameters or do we need to bulk up for heavy Bayesian lifting?

5.6.1 A case study: Comparing students within classes within schools within localities within countries

The Programme for International Student Assessment (PISA) is an initiative to measure and compare schooling performance in a growing number of countries around the world.[21] Since 1990, PISA has used a standardised assessment instrument to evaluate the ability of 15-year-olds to use their knowledge and skills in reading, mathematics, and science to meet real-world challenges. All students in participating classes in schools, states or provinces, and countries take the PISA test. Because the data are hierarchical, it is possible to compare test results among individuals, their classes, their schools, their states or provinces, and their countries. PISA and the OECD are most interested in comparisons among countries.[22] In contrast, individual governments of countries with federal education systems, such as those countries within the European Union, are most concerned about province- or state-level test results. In countries like the United States, where curricula are controlled at the state or county level, comparisons among counties or school districts will be most relevant. Any or all of these comparisons can be done using mixed-effect models in which class, school, district, county, state or province, or country—whichever levels are below the one of interest—are included as nested random factors.

A LINEAR MIXED-EFFECT MODEL FOR STUDENT PERFORMANCE

An LMM (assuming Gaussian distributions) has the following structure:

$$Y = \beta X + \delta Z + \varepsilon, \tag{5.4}$$

where Y is the column vector of student scores; X is the matrix of fixed predictors such as age, affluence, or gender; and the effects of X on Y are determined by the regression coefficients β. The matrix Z represents the random effects—in our example, these are the categorical levels of class, school, state or province, and country—whose effects on Y are modelled by the coefficients δ. X and Z are organised differently; blocks of similar entries in X do not coincide with such blocks for Z. Finally, ε is the column vector of residuals, which are normally distributed with mean equal to zero.

Another important feature of this analysis is the hierarchically nested structure: classes are nested within schools (and schools within provinces, etc.), so that we expect classes within one school to be more similar to one another than to classes from another school. In the analysis, therefore, this nested structure leads us to separately estimate, for example, "the average school performance in province A" from the individual performance of "School 21" in province A. Using an LMM, we can estimate the average performance for each province (random effect), assuming a normal distribution of estimates for the schools in that district (effects in the level "below" province).

[21] PISA is organised, but not financed, by the Organisation for Economic Co-operation and Development (OECD) and includes the 38 OECD member countries and others that are not members. See https://www.oecd.org/pisa/ for more details.

[22] For example, in the 2018 assessment, students in Singapore and some non-random provinces in China scored highest in reading, higher than the top four OECD countries—Estonia, Canada, Finland, and Ireland. Singapore and China also had the highest proportion of students—16%—who tested at the highest level of mathematics proficiency. The OECD country-wide average for this metric was only 2.4% (OECD 2019).

5.6.2 Simulation and estimation

Simulating hierarchical data is really difficult. Among other challenges, we need to keep track of the various effects that contribute to estimates in the higher level. Fortunately, there are R packages that handle the bookkeeping after we specify the nested data structure and associated parameter values. Even those specifications, however, require concerted focus and attention.

Our model structure is hierarchically organised as follows (from coarsest to finest scale): At the level of the `province`, there is some variation; at the `school` level, there is random variation around province-level variation; at the `class` level, there is random variation around `school`-level variation *plus* an effect of `affluence` (as students are grouped by school neighbourhoods); and finally, at the level of the individual student, there is again variation around the affluence-affected class-level variation:

$$\begin{aligned} \text{province} &\sim \mathcal{N}(\mu = 0, \sigma^2 = 4), \\ \text{school} &\sim \mathcal{N}(\mu = 0 + \text{province}, \sigma^2 = 6), \\ \text{class} &\sim \mathcal{N}(\mu = 1 \cdot \text{affluence} - 1 + \text{school}, \sigma^2 = 8), \\ \text{student test score} &\sim \mathcal{N}(\mu = 0 + \text{class}, \sigma^2 = 10). \end{aligned} \tag{5.5}$$

In specifying these distributions, we have assigned different variances to let us easily check whether our LMM estimates them correctly.

USE `SIMSTUDY` TO GENERATE THE DATA

We use the R package `simstudy` for our simulations. To use the package, we (1) define the setup; (2) generate the data; and (3) add in the next level down. Code Box 5.20 illustrates how we generated a dataset of schools with two attributes: the province-level contribution to the test score and the number of schools in each of 16 provinces. Each attribute is generated in its own line using the `defData` function. Only then are the actual data generated (using the `genData()` function):

Next, we simulate the data for the next level down—the schools within the provinces (Code Box 5.21). The syntax must deal with three constraints: (1) we want to add these data to the existing province-level data; (2) we want to include a predictor (`affluence`); and (3) we want to nest the schools in provinces (in `simstudy`, nestedness is called "clustering"). We also need to simulate the number of classes within each `school` (just as we simulated the number of schools within provinces in Code Box 5.20).

Our updated dataset (Code Box 5.21) now has schools per province (from Code Box 5.20), a predictor `affluence` that varies at the level of the `school`, and random effects for `province` and `school`. We now add class- and student-level elements, which also include simulating the effect of `affluence` (with fixed effect slope $\beta_1 = 1$ and intercept $\beta_0 = -1$; Code Box 5.22). This results in a simulated hierarchical dataset of 30,027 observations with a single fixed predictor (`affluence`) and all assumptions met for normality of the hyper-parameters of random effects (from Equation 5.5).

FIT THE LINEAR MIXED MODEL

Finally, we fit an LMM to the simulated data and see whether we can successfully recover the parameter settings we had defined (Code Box 5.23).

The results we need are scattered throughout the output in Code Box 5.23. The random effect of student-level variance is in `Residual` (`Variance` = $\hat{\sigma}^2 = 10$), `class`-level in `idClass:(idSchool:idProvince)` ($\hat{\sigma}^2 = 8$), `school`-level in

Code Box 5.20: Simulate hierarchical data: Top level

```
# Load required libraries
 library(simstudy)

# Use simstudy functions to generate simulated data
 gen.province <- defData(varname="Province", dist = "normal", formula = 0,
     variance = 4, id = "idProvince")
 gen.province <- defData(gen.province, varname = "nSchools", dist = "
     noZeroPoisson", formula = 20) # The attribute: the mean number of schools
     according to the distribution given in dist
 exaProvince <- genData(16, gen.province) #generates 16 provinces with average of
      20 schools per province
 exaProvince

|     idProvince   Province nSchools
|  1:          1  0.1086747       21
|  2:          2 -1.0327173       22
|  3:          3 -0.9864984       17
|  4:          4 -1.9128482       18
|  5:          5  0.6186097       28
|  6:          6  0.5677275       22
|  7:          7  2.3358999       28
|  8:          8  2.3845485       20
|  9:          9  1.3703121       10
| 10:         10 -0.4375186       14
| 11:         11  0.4863021       13
| 12:         12 -2.3754602       16
| 13:         13 -0.4530499       18
| 14:         14  1.1428510       20
| 15:         15 -0.3635563       16
| 16:         16 -0.9428277       19
```

`idSchool:idProvince` ($\hat{\sigma}^2 = 6$), and `province`-level in `idProvince` ($\hat{\sigma}^2 = 2$). Higher-level estimates are of lower precision than lower-level ones. The intercept and slope of the fixed effect (`affluence`) are also estimated (-1 and 1, respectively).

REPEAT THE SIMULATION MULTIPLE TIMES

We repeat and run many such simulations by wrapping the entire set of data-generating and data-adding code within a `replicate` function (Code Box 5.24); their definitions (`dataDef`) are carried over from Code Boxes 5.20 and 5.21. This code takes quite some time to run per replicate, so we only run 100 simulations here (estimates stabilise very soon, so 100 is fine here).[23]

Across these 100 runs, all estimates are unbiased, as we would expect. Now let us rock the boat and question our assumption.

[23] If you want feedback during the run to assure yourself that the simulations have not frozen or escaped into an infinite loop, uncomment the line with the command `print("one run almost done")`.

Code Box 5.21: Simulate hierarchical data: Next level

```
gen.school <- defDataAdd(varname = "School", dist = "normal", formula = "
    Province", variance = 6)
gen.school <- defData(gen.school, varname = "affluence", dist = "normal",
    formula = 2, variance=1) # Attribute: GDP
gen.school <- defDataAdd(gen.school, varname = "nClasses", dist =
    "noZeroPoisson", formula = 4) # Attribute: number of classes
# Make nested data, and assign ID to schools:
exaSchool <- genCluster(exaProvince, "idProvince", numIndsVar = "nSchools",
    level1ID = "idSchool")
exaSchool <- addColumns(gen.school, exaSchool)
exaSchool
|      idProvince   Province nSchools idSchool     School affluence nClasses
|   1:          1  0.1086747       21        1 -1.6430294 0.8320365        2
|   2:          1  0.1086747       21        2  1.5825982 1.0693938        5
|   3:          1  0.1086747       21        3 -2.4912792 2.0378512        3
|   4:          1  0.1086747       21        4 -2.1669522 2.1138947        6
|   5:          1  0.1086747       21        5  5.6560951 2.6600879        5
|  ---<snip>---
| 298:         16 -0.9428277       19      298  0.3076717 1.7686657        9
| 299:         16 -0.9428277       19      299 -2.1415854 0.6090572        4
| 300:         16 -0.9428277       19      300  2.8193575 1.0705190        8
| 301:         16 -0.9428277       19      301 -2.0043043 1.1293052        3
| 302:         16 -0.9428277       19      302 -5.9422585 2.0785889        5
```

5.6.3 Testing the assumption of normality

To evaluate the effect of non-normal hyper-parameters, we opt for a (technically) simple option: mixing two normal distributions. In this case, we generate `class` values that are not the effect of Equation 5.5, but rather are composed of two normal distributions:

$$\begin{aligned} \text{class1} &\sim \mathcal{N}(\mu = 1 \cdot \text{affluence} - 1 + \text{school}, \sigma^2 = 8), \\ \text{class2} &\sim \mathcal{N}(\mu = 1 \cdot \text{affluence} + 7 + \text{school}, \sigma^2 = 8), \\ \text{class} &= 0.8 \cdot \text{class1} + 0.2 \cdot \text{class2}. \end{aligned} \tag{5.6}$$

Equation 5.6 results in a non-normal distribution, including some "extreme" values (Figure 5.7).[24] When fitting our LMM to the simulated data with this new distribution used to define the random effect of `class`, we expect that only its variance estimate will differ from that of the previous simulation (Code Box 5.24) and there should be no difference in the estimated effect of `affluence`.

[24] Simulation-based model diagnostics that we discuss in §8.4 in Chapter 8 can be used to ascertain that the simulated random effect is indeed violating normality assumptions. The minimal code to do so is this:

```
library(DHARMa)
simres <- simulateResiduals(fm)
testUniformity(simres, plot=F),
```

which yields a highly significant departure from normality (Kolmogorov-Smirnov $D = 0.0293$, $p < 2.2 \times 10^{-16}$).

Code Box 5.22: Simulate hierarchical data: Last two levels

```
gen.class <- defDataAdd(varname = "Class", dist = "normal", formula = "affluence
    *1 - 1 + School", variance = 8) # Simulates the effect of affluence
gen.class <- defDataAdd(gen.class, varname = "nStudents", dist =
    "noZeroPoisson", formula = 22) # Attribute: number of students
# Make nested data, and assign ID to schools:
exaClass <- genCluster(exaSchool, "idSchool", numIndsVar = "nClasses", level1ID
    = "idClass")
exaClass <- addColumns(gen.class, exaClass)
exaClass

|       idProvince   Province nSchools idSchool     School affluence
|    1:          1 -0.8926784       18        1 -1.8986035  0.913733
|    2:          1 -0.8926784       18        1 -1.8986035  0.913733
|    3:          1 -0.8926784       18        2 -2.8048763  1.209667
|    4:          1 -0.8926784       18        2 -2.8048763  1.209667
|    5:          1 -0.8926784       18        2 -2.8048763  1.209667
|   ---<snip>---
| 1353:         16 -0.6046312       17      327 -0.2267791  1.110640
| 1354:         16 -0.6046312       17      328 -2.9631912  2.490425
| 1355:         16 -0.6046312       17      328 -2.9631912  2.490425
| 1356:         16 -0.6046312       17      329 -2.9936852  2.873294
| 1357:         16 -0.6046312       17      329 -2.9936852  2.873294
|
|    nClasses    idClass      Class nStudents
|           2          1 -4.5583235        13
|           2          2 -1.0449016        22
|           2          3 -0.6064108        25
|           2          4 -3.5110352        28
|           2          5 -2.4700639        11
| ---<snip>---
|           2       1353  2.9471031        23
|           2       1354  1.8326211        24
|           2       1355 -0.7854414        19
|           2       1356 -4.3657932        15
|           2       1357 -5.4971245        21

# Effect of student and ID:
gen.student <- defDataAdd(varname="test", dist = "normal", formula = "0 +
    Class", variance = 10)
simStudent <- genCluster(exaClass,cLevelVar="idClass", numIndsVar = "nStudents",
     level1ID = "idStudent")
simStudent <- addColumns(gen.student, simStudent)
simStudent

|        idProvince   Province nSchools idSchool    School affluence
|     1:          1 -0.8926784       18        1 -1.898604  0.913733
|     2:          1 -0.8926784       18        1 -1.898604  0.913733
```

```
|     3:          1 -0.8926784       18        1 -1.898604  0.913733
|     4:          1 -0.8926784       18        1 -1.898604  0.913733
|     5:          1 -0.8926784       18        1 -1.898604  0.913733
|   ---<snip>---
| 30023:         16 -0.6046312       17      329 -2.993685  2.873294
| 30024:         16 -0.6046312       17      329 -2.993685  2.873294
| 30025:         16 -0.6046312       17      329 -2.993685  2.873294
| 30026:         16 -0.6046312       17      329 -2.993685  2.873294
| 30027:         16 -0.6046312       17      329 -2.993685  2.873294
|
| nClasses    idClass     Class nStudents idStudent      test
|        2          1 -4.558324        13         1 -2.950039
|        2          1 -4.558324        13         2 -3.962354
|        2          1 -4.558324        13         3 -7.694957
|        2          1 -4.558324        13         4 -4.422856
|        2          1 -4.558324        13         5 -5.527798
|   ---<snip>---
|        2       1357 -5.497124        21     30023 -3.589710
|        2       1357 -5.497124        21     30024 -4.731218
|        2       1357 -5.497124        21     30025 -3.802707
|        2       1357 -5.497124        21     30026 -2.997374
|        2       1357 -5.497124        21     30027 -3.643415
```

To update our simulation, we replace the value of the variable `Class` generated by the `gen.class()` function within the `defDataAdd()` function in Code Box 5.22 and update its effect on `affluence` through each of the normal distributions within the bimodal distribution (variables `Class1` and `Class2`). We then re-run the simulations 100 times (Code Box 5.25).

The results show that the non-normal random effect of `class` does not bias estimates (Figure 5.8). As a consequence of changing the distribution of `class`, the estimate of the intercept and the variance of `class` are both larger (0.6 and 18.2, respectively) than those estimated by the simulations using normally distributed data.[25] As we had hoped, the estimated effect of `affluence` was not affected by the non-normality of the distribution of `class`.

5.7 Correlations among predictors are the same outside the range of the observed data

As Douglas Adams quipped: "The most misleading assumptions are the ones you don't even know you're making" (Adams & Carwardine 1992, p. 95). One of these is that making a prediction from any statistical model assumes (statistical) stationarity. This assumption

[25] The correct analytical value for the intercept and variance of a non-normal `class` variable can be computed from a Gaussian mixture distribution. The `intercept` for the non-normal distribution in Figure 5.8 is $\mu = 0.8 \cdot -1 + 0.2 \cdot 7 = 0.6$. The corresponding variance (i.e., the `class` variable in Figure 5.8) is then computed as $0.8 \cdot (8 + (-1)^2) + 0.2 \cdot (8 + 7^2) - \mu^2 = 18.24$. Thus, both values are correctly estimated in the mixed-effect model.

Code Box 5.23: Analyse the simulated hierarchical data

```
# Load required libraries
library(lmerTest)   # For the lmer() function

summary(fm <- lmer(test ~ affluence + (1|idProvince/idSchool/idClass), data=
    simStudent))
5
| Linear mixed model fit by REML. t-tests use Satterthwaite's method
| ['lmerModLmerTest']
| Formula: test ~ affluence + (1 | idProvince/idSchool/idClass)
|    Data: simStudent
10 |
| REML criterion at convergence: 158946.3
|
| Scaled residuals:
|     Min      1Q  Median      3Q     Max
15 | -3.8813 -0.6629  0.0099  0.6573  4.1741
|
| Random effects:
|  Groups                        Name        Variance Std.Dev.
|  idClass:(idSchool:idProvince) (Intercept)  7.246   2.692
20 |  idSchool:idProvince           (Intercept)  6.310   2.512
|  idProvince                    (Intercept)  1.860   1.364
|  Residual                                   10.099   3.178
| Number of obs: 30027, groups:  idClass:(idSchool:idProvince), 1357;
| idSchool:idProvince, 329; idProvince, 16
25 |
| Fixed effects:
|             Estimate Std. Error       df t value Pr(>|t|)
| (Intercept)  -1.4764     0.4918  41.2538  -3.002  0.00454 **
| affluence     1.1344     0.1652 318.4749   6.866 3.46e-11 ***
30 | ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Correlation of Fixed Effects:
|           (Intr)
35 | affluence -0.639
```

means that the process(es) that gave rise to the observed data will not change over time or space.[26]

[26] In geology and palaeontology, this assumption underlies the concept of Uniformitarianism: the natural laws and processes that we observe in the present day have always worked the same way in all locations in the past and will continue to work the same way everywhere and anytime in the future. This idea, commonly traced to Charles Lyell's *Principles of Geology* (1837), has a much longer history. Notably, nearly 100 years earlier, the philosopher David Hume wrote that "all inferences from experience suppose, as their foundation, that the future will resemble the past, and that similar powers will be conjoined with similar sensible qualities. If there be any suspicion that the

Code Box 5.24: Simulate and analyse hierarchical data multiple times

```
sims <- replicate(100, {
  exaProvince <- genData(16, gen.province)
  exaSchool <- genCluster(exaProvince, "idProvince", numIndsVar = "nSchools",
      level1ID = "idSchool")
  exaSchool <- addColumns(gen.school, exaSchool)
  exaClass <- genCluster(exaSchool, "idSchool", numIndsVar = "nClasses",
      level1ID = "idClass")
  exaClass <- addColumns(gen.class, exaClass)
  simStudent <- genCluster(exaClass, cLevelVar="idClass", numIndsVar =
      "nStudents", level1ID = "idStudent")
  simStudent <- addColumns(gen.student, simStudent)
  # print("one run almost done")
  fm <- lmer(test ~ affluence + (1|idProvince/idSchool/idClass), data=simStudent
      )
  c(as.data.frame(VarCorr(fm))[,4], fm@beta)
})
rownames(sims) <- c(as.data.frame(VarCorr(fm))[,1], "intercept", "affluence")
summary(t(sims))

|  idClass:(idSchool:idProvince) idSchool:idProvince   idProvince
|  Min.   :6.972                 Min.   :4.520       Min.   :1.130
|  1st Qu.:7.706                 1st Qu.:5.523       1st Qu.:3.039
|  Median :7.961                 Median :5.923       Median :4.038
|  Mean   :7.975                 Mean   :5.974       Mean   :4.230
|  3rd Qu.:8.229                 3rd Qu.:6.406       3rd Qu.:5.009
|  Max.   :9.107                 Max.   :7.863       Max.   :9.518
|
|  Residual         intercept         affluence
|  Min.   : 9.839   Min.   :-2.6313   Min.   :0.6414
|  1st Qu.: 9.967   1st Qu.:-1.5671   1st Qu.:0.8935
|  Median :10.017   Median :-0.9865   Median :1.0322
|  Mean   :10.013   Mean   :-1.0734   Mean   :1.0139
|  3rd Qu.:10.071   3rd Qu.:-0.5927   3rd Qu.:1.1452
|  Max.   :10.204   Max.   : 0.1730   Max.   :1.3063
```

The "process" behind the data in a statistical model is represented by the predictors. Stationarity means two things. First, that the parameters and moments of their distributions do not change. Second, the correlation structure among the predictors also does not change. That is, stationarity implies that the joint distribution of the predictors does not change in time or space.[27] In this final section of the chapter, we put these assumptions to the test.

course of nature may change, and that the past may be no rule for the future, all experience becomes useless, and can give rise to no inference or conclusion" (Hume 1748, p. 65). See Scott (1963) for a thorough review and critique of Uniformitarianism.

[27] Even though we may not have no way of describing the underlying predictor-generating multivariate distribution(s), it is commonly assumed that the predictors are multivariate normal. That embedded assumption is

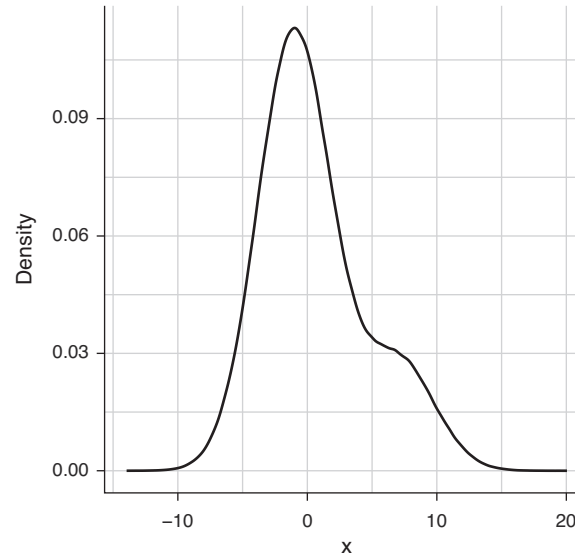


Figure 5.7 80/20-mixture of two normal distributions (from Equation 5.6) used to simulate the non-normal hyperparameters for class. The standard deviation is $\sqrt{8}$ for both of the normal distributions, but the means are -1 and 7, respectively.

We ask, when we use a statistical model fitted to data from some spatiotemporal range to another one, do we have to worry about changes in the marginal distributions of predictors or the correlations among them?

Because changes in the marginal distributions can occur only when we extrapolate beyond the range of the predictors used for fitting the model, the effect of these changes depends on whether the functional relationship between Y and X changes outside the fitted range. If it continues along the same straight line, then we don't have to worry about this assumption. But if it does not, the problem may be substantial. For example, if the current pH range (7.4–8.5) of seawater declines below its current minimum of ≈ 7.4 because of ocean acidification, it is likely that pH effects on corals, molluscs, and other shelled organisms will change non-linearly and unpredictably, simply because marine organisms have not been exposed to pH < 7.5 for at least 550 million years (Halevy & Bachan 2017).

Even if the range of none of the predictors changes, if the statistical correlation among them does, the results for model predictions can be substantial (Dormann et al. 2013). One example of where such changes in correlation structure are observable and where they make for difficult forecasts is in estimating the potential distribution of an introduced species. The important climatic variables that determine the distributional limits of a species in its native range are difficult to determine from observational data because climatic variables are highly correlated with one another. For example, when a place has large differences in temperature between summer and winter, it inevitably will have a cold winter. When annual rainfall is low, it also will be low in summer. Cold places also tend to be wetter places.

also questionable, as some of the underlying distributions may be uniform, log-normal, Poisson, etc. Such mixed multivariate distributions are generally not amenable to analytical solutions, but they can be simulated.

Code Box 5.25: Simulate and analyse hierarchical data multiple times: Non-normal random effect

```
# Modify the previous code for non-normal random effect of Class
 gen.class <- defDataAdd(varname = "Class1", dist = "normal", formula =
     "affluence*1 - 1 + School", variance = 8)
 gen.class <- defDataAdd(gen.class, varname = "Class2", dist="normal", formula =
      "affluence*1 + 7 + School", variance = 8)
 gen.class <- defDataAdd(gen.class, varname="Class", formula="Class1 | 0.8 +
     Class2 | 0.2", dist="mixture")
 gen.class <- defDataAdd(gen.class, varname = "nStudents", dist =
     "noZeroPoisson", formula = 22)

# Simulate and analyse
 sims2 <- replicate(100, {
  exaProvince <- genData(16, gen.province)
  exaSchool <- genCluster(exaProvince, "idProvince", numIndsVar = "nSchools",
       level1ID = "idSchool") # ...
  exaSchool <- addColumns(gen.school, exaSchool)
  exaClass <- genCluster(exaSchool, "idSchool", numIndsVar = "nClasses",
       level1ID = "idClass")
  exaClass <- addColumns(gen.class, exaClass)
  simStudent <- genCluster(exaClass, cLevelVar="idClass", numIndsVar =
       "nStudents", level1ID = "idStudent")
  simStudent <- addColumns(gen.student, simStudent)
#  print("one run almost done")
  fm <- lmer(test ~ affluence + (1|idProvince/idSchool/idClass), data=simStudent
       )
  c(as.data.frame(VarCorr(fm))[,4], fm@beta)
 })
 rownames(sims2) <- c(as.data.frame(VarCorr(fm))[,1], "intercept", "affluence")
 summary(t(sims2))

|  idClass:(idSchool:idProvince) idSchool:idProvince   idProvince
|  Min.   :16.17                 Min.   :3.900       Min.   :0.8232
|  1st Qu.:17.71                 1st Qu.:5.353       1st Qu.:2.9700
|  Median :18.22                 Median :5.995       Median :3.8420
|  Mean   :18.22                 Mean   :6.041       Mean   :4.1227
|  3rd Qu.:18.85                 3rd Qu.:6.606       3rd Qu.:4.8873
|  Max.   :20.06                 Max.   :9.751       Max.   :9.1261

|  Residual         intercept         affluence
|  Min.   : 9.796   Min.   :-1.1866   Min.   :0.6067
|  1st Qu.: 9.955   1st Qu.: 0.2658   1st Qu.:0.8576
|  Median : 9.998   Median : 0.6544   Median :1.0000
|  Mean   :10.005   Mean   : 0.6145   Mean   :0.9867
|  3rd Qu.:10.074   3rd Qu.: 0.9420   3rd Qu.:1.1081
|  Max.   :10.216   Max.   : 2.2586   Max.   :1.4741
```

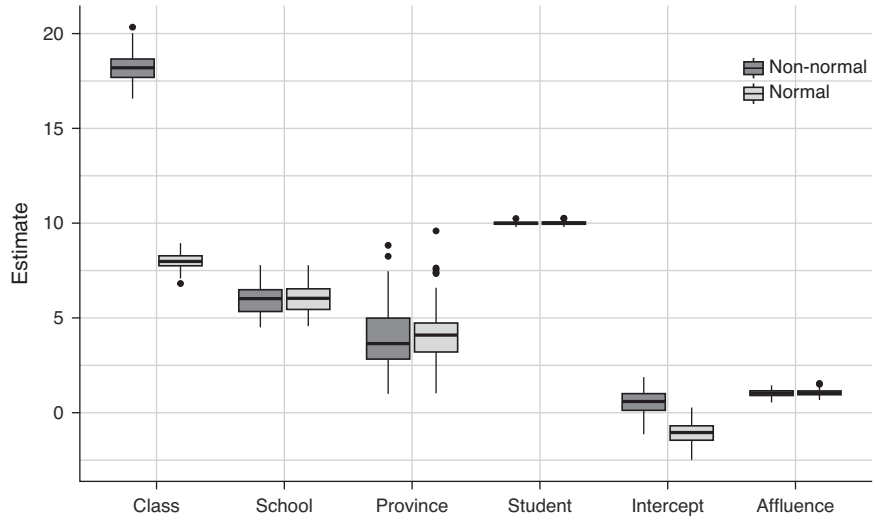


Figure 5.8 Estimates for variances of the random effects (class, school, province, student) and regression parameters (intercept, effect of affluence) for data meeting assumptions of normality (lighter grey) and those with a non-normal distribution (darker grey) for class.

5.7.1 A case study: Niche shifts of the fire ant

The red imported fire ant (*Solenopsis invicta*), or "fire ant" for short, is native to central South America, but it has been introduced into various tropical and subtropical regions around the world (Cover 2023). In the southern United States, it is well established and spreading rapidly. Since the climate there is very different from the climate of its home range in the Amazon and Cerrado, it is reasonable to ask whether the climatic niche of the introduced population is different from the climatic niche of the native population (Fitzpatrick et al. 2007).

A common way to visualise niche shifts is to compare the environmental conditions in the two different locations using a Principal Component Analysis (PCA) biplot (Figure 5.9; e.g., Broennimann et al. 2007; González-Moreno et al. 2014). Figure 5.9 illustrates that the climatic "space" in the native (grey) and introduced (black) ranges of *Solenopsis invicta* differ, suggesting a niche shift of the fire ant in its introduced range. One way such a niche shift could have occurred was suggested by Fitzpatrick et al. (2007): fire ants from the southern edge of the native population (subtropical in the southern hemisphere) were introduced to the United States, where they established in a climatic environment similar to that found in their native peripheral range. These colonists then evolved and expanded further northward (more temperate in the northern hemisphere) into different climatic environments.

Alternatively, the climatic attributes are simply different in southern and northern America, so that any random set of data collected in them would show up as separate point clouds. In other words, if North and South America have different correlations among climatic indices, then they should occupy different positions in their joint PCA space. Figure 5.10 shows that this is indeed the case. Thus, a comparison of any random set of points (one from each of the two continents) would suggest, perhaps erroneously, a niche shift when plotted on the PCA under this characterisation.

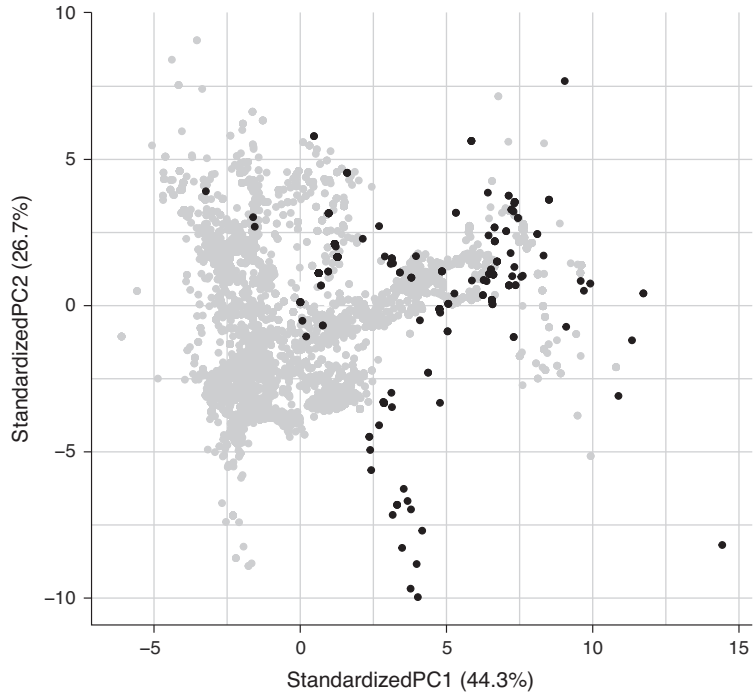


Figure 5.9 Principal Component Analysis of climatic data recorded at locations where the fire ant *Solenopsis invicta* has been collected. Points indicate collections in the ant's native South America (grey dots) and its introduced range in the United States (black dots).

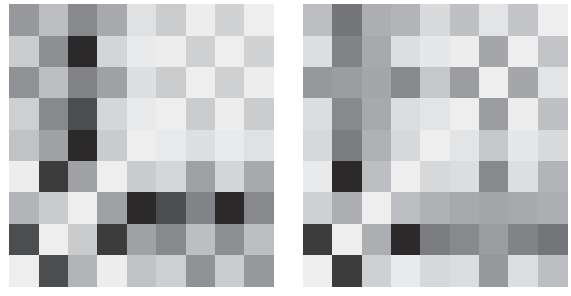


Figure 5.10 Correlation structure (using Spearman's correlation coefficient) of climatic variables at fire ant locations in the United States (left) and South America (right) for nine variables representing temperature (`bio1–bio4` in WorldClim notation) and precipitation characteristics (`bio5–bio9`). Very light and very dark indicate strong positive and negative correlations, respectively. The diagonal from bottom left to top right illustrates the correlation between one variable and itself, which is always equal to 1.

5.7.2 Demonstrating a spurious niche shift

We use simulation to show how random sampling of climatic information from the two regions may incorrectly suggest a niche shift (Code Box 5.26). In this first simulation, we plot the distribution of the simulated data only along the first principal axis (the *x*-axis in Figure 5.9). The two regions of the world separate well along the first principal axis (Figure 5.11), which is mostly determined by annual mean temperature, annual temperature

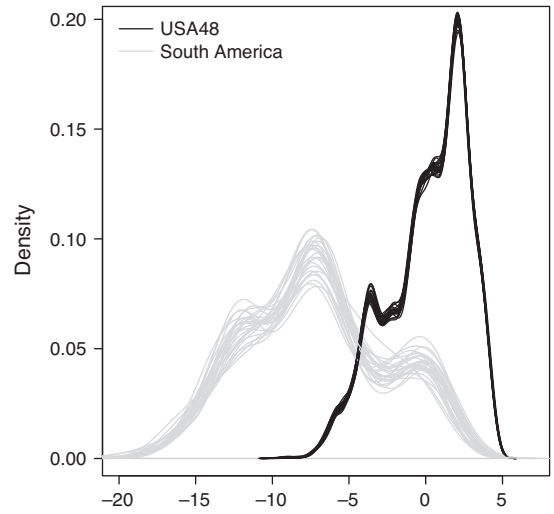


Figure 5.11 Distribution of scores of Principal Component 1 in climate space for random data points from the 48 contiguous states in the United States (dark grey) and South America (light grey). Despite some overlap, the two regions are clearly distinguished by the correlation structure of their climatic variables (Figure 5.10).

range, and annual precipitation (`bio1`, `bio7`, and `bio12`, respectively). Interpreting it solely as a temperature effect, this would mean that the South American continent is warmer (a negative loading of temperature on PC-1 yields very negative values for high temperatures) than the 48 contiguous states of the United States.

5.7.3 Where would a virtual species invade?

The simulation (Code Box 5.26) and its results (Figure 5.11) do not prove that a niche shift has not occurred, only that it need not have occurred to get the result illustrated in Figure 5.9. We need another way to determine whether the observed shift in the PCA is an effect only of warmer South America, only of changed correlation structure, or some combination of the two.

We start down the road of answering this question with a thought experiment. First, a species with a predefined climatic niche is placed into both South America and the United States. Then, we model its true range in South America and forecast where it should occur, according to our analysis, in the United States. Since the niche of our simulated species is defined to be the same in both parts of the world, the model should forecast nicely. Because of the high correlation among climatic predictor variables, the model may identify the wrong predictors, i.e., not those that cause the distribution but those that are highly correlated with the causal variables. If the correlation among the predictors in the forecast region (United States) is the same as in the predicted region (South America), the forecast should be accurate. On the other hand, if we see a discrepancy between the forecast and the simulated true distribution, we can infer that our result is an effect of a difference in the correlation structure among the predictors.

We run with this thought experiment and simulate a species with a predefined climatic niche, i.e., a fixed relationship between where the species occurs and the local climate. Then, we analyse its distribution in South America and project its distribution into the United States. This allows for a direct assessment of the quality of this forecast, because, from the inputs to our simulation, we know the true relationship between climate and occurrence and hence the true distribution of that virtual species in the United States.

As our simulations should be based on the data at hand, we use the fire-ant occurrence data from South America to fit a GLM and then use part of the fitted effects to simulate a virtual fire ant (Code Box 5.27).[28]

From the long list of effects, we used only three of the quadratic effects with significantly negative quadratic terms (so that we get the commonly-observed hump-shaped relationship between a climatic variable and the probability of occurrence; Lu & Jetz 2023) to simulate the niche of the virtual fire ant: `bio1` (mean annual temperature), `bio10` (mean temperature of warmest quarter), and `bio11` (mean temperature of coldest quarter). We also chose the intercept so that the median value equalled zero before transforming the data into the range [0, 1] (Code Box 5.28).

At the end of Code Box 5.28 we have two sets of presence-absences: `preds`, which are predictions from the random forest model; and `yUSA`, which are the simulated true occurrences. Mapping these two variables shows us the discrepancy between the two (Figure 5.12).

Of course we could quantify this difference and run it repeatedly, but the point to make is clearly visible already: forecasting to a new site with somewhat different correlation structure could get you anything. More extensive and generalizable simulations have shown this phenomenon to be attributable to the correlation structure, not to the prevalence or position of a species on a climatic gradient or such ecological interpretation (Dormann et al. 2013; Larson et al. 2014; Lo Parrino et al. 2023). Indeed, even proponents of using PCA to identify a niche shift have since revised their position (Broennimann et al. 2012; Petitpierre et al. 2012). Independent of the niche itself, model transfer to sites with a different correlation structure may well mean misery and a lack of transferability of model conclusions (Yates et al. 2018; Ploton et al. 2020).

5.8 Summary

In brief:

- Violating the assumption of normality of the dependent variable (or its residuals) in a linear regression model can have a range of effects on parameter estimation and statistical inference (e.g., *p*-values).
- The effects of inaccuracy, imprecision, or unintended values of predictor variables in a linear regression model are likely to affect estimation and inference, but only if the effects are much larger than the variance in the dependent variables.
- Heteroscedacity may quantitatively affect the inferences about estimated ANOVA parameters, but it need not make a qualitative difference. However, heteroscedasticity can strongly affect the conclusions drawn from a priori contrasts or post hoc multiple comparisons among means.
- Multiple tests on the same dataset can dramatically inflate the false positive rate, but this problem is solvable by adjusting the critical value (α) used for inference.
- Violating the assumption of conditional independence among observations used in hierarchical mixed models can quantitatively or qualitatively change parameter estimates, but the new estimates are not likely to be biased.

[28] Note, however, that in this example we have more information than the "presence-only" data normally used for species distribution modelling (Araújo et al. 2019)—we use background points as absences—and also, contra standard practice, do not include interactions among predictors in the GLM; standardise the predictors; remove highly correlated predictors; calibrate the pseudo-probability predictions; or use probabilities instead of binary classes. But we did follow Liu et al. (2005) in using the overall predicted prevalence as a reasonable threshold for converting predictions into presences.

Code Box 5.26: Random location PCA of climatic variables from Bioclim

```
# Load required libraries
 library(raster)
 library(sp)
 library(maps)

# Get data (only down to 60 degrees south latitude)
 r <- getData("worldclim", var="bio", res=10)

# Maps and projections
 mapa <- map("world", regions = c("Brazil", "Uruguay", "Argentina", "Suriname", "
      French Guiana", "Colombia", "Venezuela", "Bolivia", "Ecuador", "Chile", "
      Paraguay", "Peru", "Guyana"), fill=T, ylim=c(-60,20))
 mapb <- maptools::map2SpatialPolygons(mapa, IDs=mapa$names, proj4string=CRS("+
      proj=longlat"))
 mapc <- map("state", fill=T) # only 48!
 mapd <- maptools::map2SpatialPolygons(mapc, IDs=mapc$names, proj4string=CRS("+
      proj=longlat"))
 randomNativepoints <- spsample(mapb, n=324, type="random")
 randomUSpoints <- spsample(mapd, n=13553, type="random")

# Combine the two datasets:
 randomPoints <- randomNativepoints
 randomPoints@coords <- rbind(randomNativepoints@coords, randomUSpoints@coords)
 randomPoints@bbox <- apply(cbind(randomNativepoints@bbox, randomUSpoints@bbox),
      1, range)

# Principal Components Analysis (PCA)
# Simulation
 set.seed(123)
 PC1s <- replicate(25, {
   randomNativepoints <- spsample(mapb, n=324, type="random")
   randomUSpoints <- spsample(mapd, n=13553, type="random")
   randomPoints <- randomNativepoints
   randomPoints@coords <- rbind(randomNativepoints@coords, randomUSpoints@coords)
   randomPoints@bbox <- apply(cbind(randomNativepoints@bbox, randomUSpoints@bbox)
        , 1, range)
   climAllRandomLocations <- extract(r, randomPoints)
   randomPCA <- prcomp(na.omit(climAllRandomLocations), scale=T)
   randomPCA$x[,1]
 })
```

- Extrapolation should be restricted to data with the same correlation structure as that of the training sample.

However, the succinctness of these conclusions hides important nuances that can be appreciated better using simulations.

Code Box 5.27: Virtual fire ants in South America and the United States

```
# Load required libraries
 library(rgbif)

# Get occurrence data from Global Biodiversity Information Facility (GBIF)
 gbif_fireant <- occ_search(scientificName = "Solenopsis invicta", hasCoordinate
     = TRUE, limit=200000)
 fireant_coords <- gbif_fireant$data[ , c("decimalLongitude", "decimalLatitude",
     "occurrenceStatus", "continent", "coordinateUncertaintyInMeters", "
     institutionCode", "references", "occurrenceRemarks")]
 climAllFireAntLocations <- extract(r, fireant_coords[ , c("decimalLongitude", "
     decimalLatitude")])
 missing.index <- unique(which(is.na(climAllFireAntLocations), arr.ind=T)[,1])
 indexSA <- which(gbif_fireant$data[-missing.index,]$continent=="SOUTH_AMERICA")

# Simulations and analysis of South American and USAmerican distribution
 set.seed(123)
 randomNativePoints <- spsample(mapb, n=length(indexSA), type="random")
 systematicUSPoints <- spsample(mapd, n=length(indexUSA), type="regular")
 climAllSARandomLocations <- extract(r, randomNativePoints)
 climAllUSASystematicLocations <- extract(r, systematicUSPoints)
 datsSA <- na.omit(data.frame("PAfireant"=rep(c(0,1), each=length(indexSA)),
     rbind(climAllSARandomLocations, climAllFireAntLocations[indexSA,])))
 datsUSA <- na.omit(data.frame("PAfireant"=rep(c(0,1), times=c(nrow(
     climAllUSASystematicLocations), length(indexUSA))), rbind(
     climAllUSASystematicLocations, climAllFireAntLocations[indexUSA,])))

# Analyse to get an idea which predictors are important
 f <- as.formula(paste("PAfireant ~ ", paste0("bio", 1:19, collapse=" + "), "+",
     paste0("I(bio", 1:19, "^2)", collapse=" + ")))
 fm <- glm(f, data=dats, family=binomial)
 fmstep <- step(fm, trace=F)
 summary(fmstep)

| Call:
| glm(formula = PAfireant ~ bio1 + bio2 + bio3 + bio4 + bio5 +
|     bio9 + bio10 + bio12 + bio13 + bio14 + bio16 + bio17 + I(bio1^2) +
|     I(bio2^2) + I(bio4^2) + I(bio5^2) + I(bio9^2) + I(bio10^2) +
|     I(bio11^2) + I(bio12^2) + I(bio13^2) + I(bio14^2) + I(bio15^2) +
|     I(bio16^2) + I(bio19^2), family = binomial, data = dats)
|
| Deviance Residuals:
|     Min       1Q   Median       3Q      Max
| -3.3003  -0.1116   0.0000   0.3652   3.7121
|
| Coefficients:
|               Estimate Std. Error z value Pr(>|z|)
| (Intercept) -4.999e+01  1.784e+01  -2.802 0.005085 **
```

```
| bio1         -1.951e+00  4.741e-01  -4.115 3.87e-05 ***
| bio2         -3.847e-01  1.269e-01  -3.032 0.002430 **
| bio3          1.666e-01  1.028e-01   1.621 0.105007
| bio4         -9.532e-03  5.100e-03  -1.869 0.061622 .
| bio5         -1.143e+00  3.532e-01  -3.236 0.001214 **
| bio9          1.189e-01  6.586e-02   1.806 0.070923 .
| bio10         3.504e+00  7.905e-01   4.432 9.33e-06 ***
| bio12         3.886e-02  8.151e-03   4.767 1.87e-06 ***
| bio13         2.691e-01  6.395e-02   4.207 2.58e-05 ***
| bio14        -2.947e-01  6.277e-02  -4.696 2.66e-06 ***
| bio16        -1.070e-01  2.732e-02  -3.918 8.92e-05 ***
| bio17         4.385e-02  1.544e-02   2.839 0.004521 **
| I(bio1^2)     6.467e-03  1.338e-03   4.834 1.34e-06 ***
| I(bio2^2)     1.082e-03  5.011e-04   2.159 0.030837 *
| I(bio4^2)     7.194e-07  2.490e-07   2.889 0.003865 **
| I(bio5^2)     1.783e-03  5.529e-04   3.225 0.001259 **
| I(bio9^2)    -3.403e-04  1.934e-04  -1.760 0.078452 .
| I(bio10^2)   -7.142e-03  1.465e-03  -4.876 1.09e-06 ***
| I(bio11^2)   -2.182e-03  6.132e-04  -3.558 0.000374 ***
| I(bio12^2)   -9.670e-06  2.332e-06  -4.146 3.38e-05 ***
| I(bio13^2)   -5.123e-04  1.429e-04  -3.584 0.000338 ***
| I(bio14^2)    7.338e-04  2.863e-04   2.563 0.010374 *
| I(bio15^2)    5.377e-04  3.025e-04   1.778 0.075475 .
| I(bio16^2)    5.564e-05  2.146e-05   2.593 0.009519 **
| I(bio19^2)    5.681e-06  2.496e-06   2.276 0.022848 *
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| (Dispersion parameter for binomial family taken to be 1)
|
|     Null deviance: 896.93  on 646  degrees of freedom
| Residual deviance: 319.03  on 621  degrees of freedom
| AIC: 371.03
|
| Number of Fisher Scoring iterations: 9
```

For example, some assumptions are clearly stated in statistical textbooks when introducing a method; examples include homogeneity of variances and normality (or other distribution) of the response. When these assumptions are violated, all bets are off for the consequences for estimation and inference (p-values). They may be anything between irrelevant and dramatic, but we can use simulations effectively to see where on this continuum we are likely to land for a specific analysis.

More problematic are the assumptions that are stated more abstractly or in strange words. For example, the stationarity assumption in the context of time-series or spatial analysis means that the underlying model does not change over time or in space, respectively. But when we use a temporally explicit model in which coefficients are allowed to vary over

Figure 5.12 Predicted distribution of a virtual fire ant based on a prediction of a model fitted to data in its native range (*left*) and the true climatically-determined distribution (*right*).

Code Box 5.28: Virtual fire ants in South America and the United States, *continued*

```
# Load required libraries
 library(ranger)

# Simulation of the true distribution
 set.seed(123)
 ySA <-  plogis(with(datsSA[,-1], -150.2 + .6* bio1 + 1.2*bio10 + -.003 * bio10^2
       - 0.7*bio11 + 0.001 * bio11^2))
 yUSA <-  plogis(with(data.frame(datsUSA[,-1]), -150.2 + .6* bio1 + 1.2*bio10 +
      -.003 * bio10^2 - 0.7*bio11 + 0.001 * bio11^2))

 fr <- ranger(as.factor(ySA>mean(ySA)) ~., data=datsSA)
 preds2USA <- predict(fr, data=datsUSA)$predictions
```

time, what does stationarity mean? In this case, we at least assume that the variance is still constant and that the autocorrelation is unchanged, unless we also explicitly make these a function of time. This is but one example that illustrates that some assumptions can be relaxed if they are incorporated as features in a model. Keep in mind that doing so may lead the analysis into deep waters, much more complicated model structures, and more tedious programming.[29]

What can be annoying is that we might get told off for violating an "obvious" assumption that we have never heard of (vide Adams & Carwardine 1992). This happened to us in the case of extrapolation of niches in geographic space (§5.7): since our model represents $P(Y \mid X)$, an implicit assumption is that the prediction will be a new X' that is similar to the original X in all aspects, including the correlation among X', spatial resolution of X, measurement error on each X'_i, and so forth. If X' fails to comply with these implicit assumptions, it could have either no effect or ruin the entire prediction. Diagnostics and simulations for any observed deviations ought to be a required step for any extrapolation study.

[29] And in the end you might end up becoming a Bayesian!

References

Adams, D. & Carwardine, M. (1992). *Last Chance to See*. New York: Ballantine Books.

Araújo, M. B., Anderson, R. P., Barbosa, A. M., et al. (2019). Standards for distribution models in biodiversity assessments. *Science Advances*, 5(1), eaat4858. doi: 10.1126/sciadv.aat4858.

Aydemir, A. & Borjas, G. J. (2011). Attenuation bias in measuring the wage impact of immigration. *Journal of Labor Economics*, 29(1), 69–112. doi: 10.1086/656360.

Bar-Hillel, M. (1980). The base-rate fallacy in probability judgments. *Acta Psychologica*, 44(3), 211–233. doi: 0.1016/0001-6918(80)90046-3.

Bartlett, J. W. & Keogh, R. H. (2018). Bayesian correction for covariate measurement error: A frequentist evaluation and comparison with regression calibration. *Statistical Methods in Medical Research*, 27(6), 1695–1708. doi: 10.1177/0962280216667764.

Benjamini, Y. & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B (Methodological)*, 57(1), 289–300. doi: 10.1111/j.2517-6161.1995.tb02031.x.

Borjas, G. J. (2003). The labor demand curve is downward sloping: Reexamining the impact of immigration on the labor market. *The Quarterly Journal of Economics*, 118(4), 1335–1374. doi: 10.1162/003355303322552810.

Boulesteix, A.-L., Groenwold, R. H. H., Abrahamowicz, M., et al. (2020). Introduction to statistical simulations in health research. *British Medical Journal Open*, 10(12), e039921. doi: 10.1136/bmjopen-2020-039921.

Brakenhoff, T. B., van Smeden, M., Visseren, F. L. J., & Groenwold, R. H. H. (2018). Random measurement error: Why worry? An example of cardiovascular risk factors. *PLoS ONE*, 13(2), e0192298. doi: 10.1371/journal.pone.0192298.

Broennimann, O., Fitzpatrick, M. C., Pearman, P. B., et al. (2012). Measuring ecological niche overlap from occurrence and spatial environmental data. *Global Ecology and Biogeography*, 21(4), 481–497. doi: 10.1111/j.1466-8238.2011.00698.x.

Broennimann, O., Treier, U. A., Müller-Schärer, H., et al. (2007). Evidence of climatic niche shift during biological invasion. *Ecology Letters*, 10(8), 701–709. doi: 10.1111/j.1461-0248.2007.01060.x.

Cardinale, B. J., Srivastava, D. S., Duffy, J. E., et al. (2006). Effects of biodiversity on the functioning of trophic groups and ecosystems. *Nature*, 443(7114), 989–992.

Coombs, W. T., Algina, J., & Oltman, D. O. (1996). Univariate and multivariate omnibus hypothesis tests selected to control type I error rates when population variances are not necessarily equal. *Review of Educational Research*, 66(2), 137–179. doi: 10.3102/00346543066002137.

Cover, S. (2023). *Solenopsis invicta*. https://www.antwiki.org/wiki/Solenopsis_invicta. Accessed 13 July 2023.

Crawley, M. J. (2007). *The R Book*. Chichester, West Sussex, UK: John Wiley & Sons.

Daszkiewicz, M., Łuczkiewicz, P., Kleeman, J., & Kuzioła, A. (2019). What shall we put in the grave? Archaeometric analyses of ceramics from a late Pre-Roman, Roman and Migration period cemetery in Malbork-Wielbark, northern Poland. *Praehistorische Zeitschrift*, 94(2), 414–453. doi: 10.1515/pz-2019-0018.

de Valpine, P., Turek, D., Paciorek, C. J., et al. (2017). Programming with models: Writing statistical algorithms for general model structures with NIMBLE. *Journal of Computational and Graphical Statistics*, 26(2), 403–417. doi: 10.1080/10618600.2016.1172487.

Diener, E. (2000). Subjective well-being: The science of happiness and a proposal for a national index. *American Psychologist*, 55(1), 34–43. doi: 10.1037/0003-066X.55.1.34.

Dormann, C. F., Elith, J., Bacher, S., et al. (2013). Collinearity: A review of methods to deal with it and a simulation study evaluating their performance. *Ecography*, 36(1), 27–46. doi: 10.1111/j.1600-0587.2012.07348.x.

Dunn, O. J. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293), 52–64. doi: 10.1080/01621459.1961.10482090.

Fitzpatrick, M. C., Weltzin, J. F., Sanders, N. J., & Dunn, R. R. (2007). The biogeography of prediction error: Why does the introduced range of the fire ant over-predict its native range? *Global Ecology and Biogeography*, 16(1), 24–33. doi: 10.1111/j.1466-822x.2006.00258.x.

Frost, C. & Thompson, S. G. (2000). Correcting for regression dilution bias: Comparison of methods for a single predictor variable. *Journal of the Royal Statistical Society A*, 163(2), 173–189. doi: 10.1111/1467-985X.00164.

Fuller, W. A. (1987). *Measurement Error Models*. New York, NY, USA: Wiley.

Gelman, A. & Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge, UK: Cambridge University Press.

González-Moreno, P., Diez, J. M., Richardson, D. M., & Vilà, M. (2014). Beyond climate: Disturbance niche shifts in invasive species. *Global Ecology and Biogeography*, 24(3), 360–370. doi: 10.1111/geb.12271.

Griliches, Z. & Ringstad, V. (1970). Error-in-the-variables bias in nonlinear contexts. *Econometrica*, 38(2), 368–370. doi: 10.2307/1913020.

Halevy, I. & Bachan, A. (2017). The geologic history of seawater pH. *Science*, 355(6329), 1069–1071. doi: 10.1126/science.aal4151.

Harrell, F. E. (2015). *Regression Modeling Strategies—with Applications to Linear Models, Logistic and Ordinal Regression, and Survival Analysis*, 2nd edition. New York, NY, USA: Springer.

Hausman, J. (2001). Mismeasured variables in econometric analysis: Problems from the right and problems from the left. *Journal of Economic Perspectives*, 15(4), 57–67. doi: 10.1257/jep.15.4.57.

Hume, D. (1748). *Philosophical Essays Concerning Human Understanding*. London, UK: A. Millar.

Hutchins, J. & Hueth, B. (2023). 100 years of data sovereignty: Cooperative data governance and innovation in US dairy. *Applied Economic Perspectives and Policy*, 45(3), 1551–1576. doi: 10.1002/aepp.13339.

Larson, E. R., Gallagher, R. V., Beaumont, L. J., & Olden, J. D. (2014). Generalized "avatar" niche shifts improve distribution models for invasive species. *Diversity and Distributions*, 20(11), 1296–1306. doi: 10.1111/ddi.12233.

Legendre, P. (2022). lmodel2: Model II regression, version 1.7-3. https://cran.r-project.org/package=lmodel2.

Legendre, P. & Legendre, L. (2012). *Numerical Ecology*. Amsterdam, The Netherlands: Elsevier.

Liu, C., Berry, P. M., Dawson, T. P., & Pearson, R. G. (2005). Selecting thresholds of occurrence in the prediction of species distributions. *Ecography*, 3(3), 385–393. doi: https://doi.org/10.1111/j.0906-7590.2005.03957.x.

Lo Parrino, E., Falaschi, M., Manenti, R., & Ficetola, G. F. (2023). All that changes is not shift: Methodological choices influence niche shift detection in freshwater invasive species. *Ecography*, 2023(2), e06432. doi: 10.1111/ecog.06432.

Lu, M. & Jetz, W. (2023). Scale-sensitivity in the measurement and interpretation of environmental niches. *Trends in Ecology & Evolution*, 38(5), 554–567. doi: 10.1016/j.tree.2023.01.003.

Lyell, C. (1837). *Principles of Geology*, 5th edition. London, UK: John Murray.

MacMahon, S., Peto, R., Cutler, J., et al. (1990). Blood pressure, stroke, and coronary heart disease. Part 1, Prolonged differences in blood pressure: Prospective observational studies corrected for the regression dilution bias. *The Lancet*, 335(8692), 765–774. doi: 10.1016/0140-6736(90)90878-9.

OECD (2019). *PISA 2018 Results: Combined Executive Summaries, Volume I, II & III*. Paris, France: Organisation for Co-operation and Development.

OECD/FAO (2020). *OECD-FAO Agricultural Outlook 2020–2029*. Rome and Paris: FAO and OECD.

Pantel, J. H. & Becks, L. (2023). Statistical methods to identify mechanisms in studies of eco-evolutionary dynamics. *Trends in Ecology & Evolution*, 38(8), 760–772. doi: 10.1016/j.tree.2023.03.011.

Petitpierre, B., Kueffer, C., Broennimann, O., et al. (2012). Climatic niche shifts are rare among terrestrial plant invaders. *Science*, 335(6074), 1344–1348. doi: 10.1126/science.1215933.

Ploton, P., Mortier, F., Réjou-Méchain, M., et al. (2020). Spatial validation reveals poor predictive performance of large-scale ecological mapping models. *Nature Communications*, 11(11), 4540. doi: 10.1038/s41467-020-18321-y.

Potvin, C. (2001). ANOVA: Experiments layout and analysis. In J. Gurevitch & S. M. Scheiner (Eds.), *Design and Analysis of Ecological Experiments*, 2nd edition (pp. 63–76). Oxford, UK: Oxford University Press.

Quinn, G. P. & Keough, M. J. (2002). *Experimental Design and Data Analysis for Biologists*. Cambridge, UK: Cambridge University Press.

Rastelli, C., Calabrese, L., Miller, C., et al. (2021). The art of happiness: An explorative study of a contemplative program for subjective well-being. *Frontiers in Psychology*, 12, 600982. doi: 10.3389/fpsyg.2021.600982.

Rogan, J. C. & Keselman, H. J. (1977). Is the ANOVA *F*-test robust to variance heterogeneity when sample sizes are equal? An investigation via a coefficient of variation. *American Educational Research Journal*, 14(4), 493–498. doi: 10.3102/00028312014004493.

Samuelson, P. A. (1964). *Economics*, 6th edition. New York, NY, USA: McGraw-Hill.

Schennach, S. M. (2004). Estimation of nonlinear models with measurement error. *Econometrica*, 72(1), 33–75. doi: 10.1111/j.1468-0262.2004.00477.x.

Scott, G. H. (1963). Uniformitarianism, the uniformity of nature, and paleoecology. *The New Zealand Journal of Geology and Geophysics*, 6(4), 510–527. doi: 10.1080/00288306.1963.10420063.

Sokal, R. R. & Rohlf, F. J. (1995). *Biometry*, 3rd edition. New York, NY, USA: W. H. Freeman and Company.

Stan Development Team (2023). *Stan Modeling Language Users Guide and Reference Manual, 2.33*. NumFOCUS, Austin, TX, USA.

Stoklosa, J., Daly, C., Foster, S. D., et al. (2015). A climate of uncertainty: Accounting for error in climate variables for species distribution models. *Methods in Ecology and Evolution*, 6(4), 412–423. doi: 10.1111/2041-210X.12217.

Thompson, W. C. & Schumann, E. L. (1987). Interpretation of statistical evidence in criminal trials: The prosecutor's fallacy and the defense attorney's fallacy. *Law and Human Behavior*, 11(3), 167–187. doi: 10.1007/BF01044641.

Tilman, D. & Downing, J. A. (1994). Biodiversity and stability in grasslands. *Nature*, 367(6461), 363–365. doi: 10.1038/367363a0.

Underwood, A. J. (1997). *Experiments in Ecology: Their Logical Design and Interpretation Using Analysis of Variance*. Cambridge, UK: Cambridge University Press.

Ŝidák, Z. K. (1967). Rectangular confidence regions for the means of multivariate normal distributions. *Journal of the American Statistical Association*, 62(318), 626–633. doi: 10.1080/01621459.1967.10482935.

Weisser, W. W., Roscher, C., Meyer, S. T., et al. (2017). Biodiversity effects on ecosystem functioning in a 15-year grassland experiment: Patterns, mechanisms, and open questions. *Basic and Applied Ecology*, 23, 1–73. doi: 10.1016/j.baae.2017.06.002.

Welch, B. L. (1951). On the comparison of several mean values: An alternative approach. Biometrika, 38(3/4), 330–336. doi: 10.1093/biomet/38.3-4.330.

Wilcox, R. R., Charlin, V. L., & Thompson, K. L. (1986). New Monte Carlo results on the robustness of the ANOVA F, W and F* statistics. *Communications in Statistics—Simulation and Computation*, 15(4), 933–943. doi: 10.1080/03610918608812553.

Yates, K. L., Bouchet, P. J., Caley, M. J., et al. (2018). Outstanding challenges in the transferability of ecological models. *Trends in Ecology & Evolution*, 33(10), 790–802. doi: 10.1016/j.tree.2018.08.001.

6

Folklore: Is that rule-of-thumb true or useful?

After testing and determining the robustness of our statistical assumptions (Chapter 5), we are ready to move on to use simulations to explore rules-of-thumb and other forms of expert advice for statistical analyses.[1] We do not argue against rules-of-thumb, recipes, or statistical guidelines. In fact, some are extremely helpful to avoid common errors, particularly to the target audience of this book: domain scientists and practitioners who use statistics in their day-to-day work but are not necessarily card-carrying statisticians (see, e.g., van Belle 2008; Good & Hardin 2012; Gelman et al. 2020; Quinn et al. 2021). What we want to demonstrate is that whenever a rule-of-thumb seems counterintuitive, strange, or too good to be true, simulations can help us determine whether it is useful or not.

For example, in the early days of mixed-effect modelling, we routinely were told to treat random effects as fixed effects unless we had at least seven different treatment levels (e.g., patients, experimental units, or years). Other well-trod but unsubstantiated assertions are that multiple regression models are problematic or untrustworthy when correlations (r) between predictors are > 0.7 and that time series should always be detrended before further analysis. Do these rules make sense? And now, as methods for working with Big Data are proliferating rapidly, it has become common knowledge that machine-learning methods are much more robust to a range of assumptions than are classical parametric analyses. Are they really?

Taking on statistical folklore and tackling expert advice is challenging and can involve complex simulations. In this chapter, we subject a number of common rules-of-thumb to investigation via simulation.

[1] A "rule-of-thumb" is a roughly practical "method or procedure derived from practice or experience, rather than theory or scientific knowledge." The Oxford English Dictionary (OED Online 2023) identifies its first occurrence in the literature in James Durham's posthumously published *Heaven upon Earth* (Durham 1685, ii. 217): "Many profest [professed] Christians are like to foolish builders, who build by guess and by rule of thumb." Durham (1622–1658) was a moderate minister in the Church of Scotland who is perhaps best remembered for his *Commentary upon the Book of Revelation* (Durham 1658) and his plea—dictated as he lay dying and published a year later (Durham 1659)—for peace and unity in the Scottish church (Holfelder 2004). There seems to be no historical basis for the linguistic folklore that the "rule-of-thumb" refers to the maximum allowable width of a stick that may be used to beat one's spouse (Kelly 1994).

What you will learn from this chapter

- Whether model selection improves interpretation.
- How well common rules-of-thumb for regression and ANOVA models hold up to scrutiny: variable selection to deal with collinearity; the acceptability of categorising continuous predictor variables; the minimum number of levels needed to consider an ordinal independent variable a continuous one; and using Monte Carlo simulation to handle heteroscedasticity and variable precision among observations.
- The consequences of detrending time-series data before analysing them.
- Whether machine-learning models really are more robust than classical parametric models, and hence obviate the need for statistical rules-of-thumb.

Through all these examples, we seek to glean the grains of truth within the folklore and banish the chaff to the sands of time. In doing so, the examples in this chapter highlight the spirit of this book: don't take our word for it, find out for yourself.

6.1 Model selection does not always improve interpretation

When several predictors are plausible, and the number of observations is relatively small, it has become common practice to use stepwise model selection to home in on those predictors most relevant for "explaining"—or better, "representing"—the data (Heinze et al. 2018; Tredennick et al. 2021). The goal with stepwise model selection is to identify a subset of predictors that we can use in a relatively simple and easily interpretable model (i.e., simple and interpretable when compared with the full model that includes all the possible predictors and their plausible interactions). Stepwise model selection is similar to climbing or pruning a tree. We can step forwards, successively adding terms as we climb the tree and stopping when we have a good enough representation of it. Alternatively, we can work backwards from the complete tree, picking a branch and pruning it down until we arrive at the essential treeness of the model. Finally, we can do both simultaneously. In none of these three cases, however, are we likely to scan the entire model space.

Dredging the statistically feasible model space for the best-fitting models (a.k.a. best subset regression: Hosmer et al. 1989) has become a common procedure among analysts in many scientific disciplines, but it is generally shunned by statisticians (including those writing packages and programming functions for it: Bartoń 2023; see also Burnham & Anderson 2002; Wasserstein & Lazar 2016 and additional discussion in Chapter 3, §3.5.1).[2] Whether we step through our data and models or dredge them in the hopes of finding glints of gold among the tailings, we have been led to expect that the end result will be the same: an interpretable model with reasonable predictive power and a low rate of false positives (type-I errors; Table 3.1 in Chapter 3). Is this a sound expectation?

6.1.1 A case study: Military coups in sub-Saharan Africa

The causes of military coups are varied, but are they idiosyncratic? Some countries in sub-Saharan Africa have seen more than their fair share of military coups since independence from their various colonial powers, whereas other countries have been spared the continual transitions from one military dictatorship to another (e.g., Roessler 2011; Houle & Bodea 2017; IISS 2021).[3] The 47 sub-Saharan countries differ in many aspects, but to understand

[2] In the notes for the `dredge` function in the `R` package `MuMIn`, Bartoń (2023) writes that "users should keep in mind the hazards that a 'thoughtless approach' of evaluating all possible models poses. Although this procedure is in certain cases useful and justified, it may result in selecting a spurious 'best' model, due to the model selection bias."

[3] The `africa` dataset in the `R` package `faraway` (Faraway 2022) refers to these as "oligarchies."

political dynamics better and develop general theories of political (in)stability, we would like to try to identify one or two dominant correlates of the frequency of military coups.[4]

6.1.2 Analysing the data with a simple GLM

Code Box 6.1 gives a summary of the data and the results of a first attempt to model them with a GLM.

The full GLM has far too few residual degrees of freedom (18) for the residuals to provide a meaningful test of the fitted effects. Model simplification yields a smaller and more interpretable model, with a main effect of years under military dictatorship and the interaction of a country's size and the number of elections as predictors.[5]

6.1.3 How does stepwise regression compare with dredging for the best subset?

Although dredging (i.e., best subset regression), like other questionable research practices, is generally reviled, we apply it to the `africa` dataset simply to illustrate that there are several more models of similar quality as the one shown in Code Box 6.1 resulting from stepwise model selection. The results of the dredging (Code Box 6.2) gives us the first inkling that there are reasonable competitors to the "best" model.

Dredging all models with five or fewer predictors takes a few minutes and scans fits of over 130 million combinations of predictor combinations, storing (in this case) only the 561 best models.

Satisfyingly, the best model identified by the stepwise selection (Code Box 6.1) is also number one on the list of dredged models (Code Box 6.2). However, using the rule-of thumb that a delta $AIC_c \approx 2$ identifies comparable models (Burnham & Anderson 2002), the penultimate output column (`delta`) in Code Box 6.2 indicates that there are at least three other models of roughly similar fits. These three models use different predictors (number of parties `prt` and population size `ppn`) from the best model. This variability in model structure suggests that the simple model that emerged from the stepwise model selection may blind us to alternative model structures of similar quality.

The value in the last output column, `weight`, is the Akaike weight of each model, which can be considered to be the probability that said model is the best model, given the data and the set of candidate models (Burnham & Anderson 2001; Wagenmakers & Farrell 2004). We also can use the normalised ratio of two weights, $\frac{w_{\text{model A}}}{w_{\text{model A}}+w_{\text{model B}}}$, to determine the probability that one model should be preferred over another. In this case, the first model (number 4164, $w = 0.200$) is 58% more likely than the second model (4172, $w = 0.141$).

6.1.4 Does model selection result in a structurally correct model?

We now return to the question that motivated this section: does model selection aid interpretation? To be a useful basis for interpretation, the selected model needs to be

[4] We ignore the variable `numregim` (numbers of regime types) in the `africa` dataset to allow the full model to be able to be fit. This variable is the weakest predictor of coup frequency and its omission does not affect our conclusions.

[5] Note that we imputed 11 values for `pollib` (an index of political liberalisation: 0 = no civil rights for political expression; 1 = limited civil rights for expression but the right to form political parties exists; 2 = full civil rights) and `pctvote` (percent voting in the last election) so as to not lose 11 countries from our analysis. Since we will be using simulated responses later, this is not a concern here. However, in a proper analysis of the original data, we also would quantify the effect of imputation (Nielsen 2003; Nakagawa & Freckleton 2011). For further discussion of imputation of missing values, see §7.3 in Chapter 7.

Code Box 6.1: An initial model of coup frequency in sub-Saharan Africa

```
# Load required libraries
 library(faraway) # Includes the africa dataset
 library(mice)   # For imputation of missing values

#Summarise the dataset
 summary(africa) # (Note: variable "numregim" omitted from output)

|    miltcoup       oligarchy          pollib         parties
| Min.   :0.000   Min.   : 0.000   Min.   :0.000   Min.   : 0.00
| 1st Qu.:0.000   1st Qu.: 0.000   1st Qu.:1.000   1st Qu.:10.00
| Median :1.000   Median : 1.000   Median :2.000   Median :13.00
| Mean   :1.404   Mean   : 4.447   Mean   :1.667   Mean   :15.96
| 3rd Qu.:2.000   3rd Qu.: 9.000   3rd Qu.:2.000   3rd Qu.:19.00
| Max.   :6.000   Max.   :18.000   Max.   :2.000   Max.   :62.00
|                                 NA's   :5
|    pctvote           popn              size           numelec
| Min.   : 0.00   Min.   :  0.067   Min.   :   0.5   Min.   : 0.000
| 1st Qu.:18.90   1st Qu.:  1.450   1st Qu.:  33.0   1st Qu.: 4.000
| Median :28.95   Median :  5.600   Median : 274.0   Median : 6.000
| Mean   :31.88   Mean   : 10.953   Mean   : 516.7   Mean   : 6.191
| 3rd Qu.:43.04   3rd Qu.: 11.450   3rd Qu.: 813.0   3rd Qu.: 8.500
| Max.   :77.40   Max.   :113.800   Max.   :2506.0   Max.   :14.000
| NA's   :6

# Impute the few missing values:
 africa.imp <- complete(mice(africa, m=5, method="pmm"))

#Fit and summarise the GLM
 fm <- glm(miltcoup ~ (oligarchy + pollib + parties + pctvote + popn + size
      + numelec)^2 + I(oligarchy^2), data=africa.imp, family=poisson)
 summary(fm)

| Deviance Residuals:
|     Min        1Q    Median        3Q       Max
| -1.11087  -0.48782  -0.09472   0.20821   1.69267
|
| Coefficients:
|                     Estimate Std. Error z value Pr(>|z|)
| (Intercept)       -3.331e+00  2.891e+00  -1.153   0.2491
| oligarchy          3.551e-01  2.590e-01   1.371   0.1703
| pollib             1.639e+00  1.892e+00   0.866   0.3864
| parties            1.538e-01  2.030e-01   0.758   0.4487
| pctvote            4.424e-02  8.110e-02   0.545   0.5854
| popn              -1.396e-01  3.530e-01  -0.395   0.6925
| size              -1.194e-02  6.627e-03  -1.802   0.0715 .
| numelec           -2.302e-01  3.119e-01  -0.738   0.4606
| oligarchy:pollib  -1.169e-01  1.252e-01  -0.934   0.3501
```

```
| oligarchy:parties  2.302e-03  6.766e-03   0.340   0.7337
| oligarchy:pctvote -6.123e-04  4.147e-03  -0.148   0.8826
| oligarchy:popn    -3.960e-03  8.468e-03  -0.468   0.6401
| oligarchy:size    -1.659e-05  1.975e-04  -0.084   0.9331
| oligarchy:numelec  1.364e-02  1.738e-02   0.785   0.4325
| pollib:parties    -6.830e-02  8.378e-02  -0.815   0.4150
| pollib:pctvote    -1.996e-02  3.676e-02  -0.543   0.5872
| pollib:popn       -1.378e-01  1.494e-01  -0.923   0.3563
| pollib:size        3.066e-03  1.965e-03   1.561   0.1186
| pollib:numelec     2.685e-02  1.467e-01   0.183   0.8548
| parties:pctvote   -3.362e-03  3.094e-03  -1.087   0.2772
| parties:popn       1.209e-03  9.047e-03   0.134   0.8937
| parties:size       1.241e-04  1.108e-04   1.120   0.2626
| parties:numelec    5.428e-03  1.505e-02   0.361   0.7184
| pctvote:popn       1.043e-02  6.523e-03   1.600   0.1097
| pctvote:size      -2.490e-05  5.943e-05  -0.419   0.6752
| pctvote:numelec    4.868e-03  5.306e-03   0.917   0.3590
| popn:size          2.074e-04  1.158e-04   1.791   0.0732 .
| popn:numelec      -2.172e-02  2.140e-02  -1.015   0.3101
| size:numelec       6.488e-04  4.726e-04   1.373   0.1698
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| (Dispersion parameter for poisson family taken to be 1)
|
|     Null deviance: 93.971  on 46  degrees of freedom
| Residual deviance: 15.598  on 18  degrees of freedom
| AIC: 147.28

# Stepwise regression (both backward and forward); model selection with BIC
 fmstep <- step(fm, trace=F, k=log(47))
 summary(fmstep)

| Deviance Residuals:
|     Min       1Q   Median       3Q      Max
| -1.5167  -0.9743  -0.4870   0.5643   2.0006
|
| Coefficients:
|               Estimate Std. Error z value Pr(>|z|)
| (Intercept)  -7.366e-01  4.509e-01  -1.633  0.10237
| oligarchy     1.552e-01  2.391e-02   6.490  8.6e-11 ***
| size         -1.792e-03  6.547e-04  -2.738  0.00619 **
| numelec       2.905e-02  5.194e-02   0.559  0.57590
| size:numelec  1.987e-04  6.681e-05   2.974  0.00294 **
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| (Dispersion parameter for poisson family taken to be 1)
|
```

```
|     Null deviance: 93.971  on 46  degrees of freedom
| Residual deviance: 42.263  on 42  degrees of freedom
| AIC: 125.94 #Actually, BIC, since k was specified and not equal to 2
```

Code Box 6.2: Dredging the correlates of unrest in sub-Saharan Africa

```
# Load required libraries
 library(MuMIn) # For multi-model inference

# Dredge the full model from Code Box 6.1
 options(na.action = "na.fail") # Interrupt and fail if there are missing values
 fd <- dredge(fm, m.lim=c(0,5)) # Fit the model (takes a few minutes)

| Fixed term is "(Intercept)"

 subset(fd, delta < 4)

| Global model call: glm(formula = miltcoup ~ (oligarchy + pollib
| + parties + pctvote + popn + size + numelec)^2, family = poisson,
| data = africa.imp)
| ---

# Model selection table; Models ranked by AICc(x)
# Row names are the model numbers, which range from 1 to > 3 million
    combinations of predictors
# Note that beause there are 19 columns, the table is broken into three parts

|            (Int)     nml    olg      prt     pct      pll       ppn
| 4164      -0.7366 0.02905 0.1552
| 4172      -1.0250 0.01410 0.1624          0.01041
| 4168      -0.8266 0.01926 0.1506 0.009048
| 4196      -0.6969 0.02612 0.1498                            0.003871
| 16777292  -2.4350 0.10300 0.1707          0.03154
| 4292      -0.6163 0.01146 0.1170
| 16415     -0.8686         0.1841 0.020220 0.03123 -0.59360
| 135236    -0.7857 0.02992 0.1607
| 4180      -0.6990 0.02850 0.1545                  -0.01843
| 16777307  -1.1270         0.1481          0.03658 -0.41640
| 134217828 -1.0740 0.12230 0.1751                            -0.062870
| 16777295  -2.3320         0.1582 0.017020 0.03991
| 31        -0.3677         0.1117 0.022920 0.01531 -0.61910
| 4         -1.2570 0.10570 0.1334
|
|                 siz nml:olg   nml:siz   olg:pct    olg:siz    pct:siz
| 4164      -0.001792         0.0001987
| 4172      -0.002102         0.0002338
| 4168      -0.001708         0.0001944
```

```
| 4196      -0.001955         0.0002112
| 16777292   0.001086                                          -4.444e-05
| 4292      -0.001726 0.00665 0.0001868
| 16415                                  -0.002344
| 135236    -0.001591         0.0001890           -1.423e-05
| 4180      -0.001775         0.0001962
| 16777307   0.001119                                          -4.438e-05
| 134217828 -0.001188
| 16777295   0.001384                                          -4.989e-05
| 31
| 4
|
|             ppn:siz df  logLik  AICc delta weight
| 4164                 5 -57.972 127.4  0.00  0.200
| 4172                 6 -57.004 128.1  0.70  0.141
| 4168                 6 -57.485 129.1  1.66  0.087
| 4196                 6 -57.667 129.4  2.03  0.073
| 16777292             6 -57.757 129.6  2.21  0.066
| 4292                 6 -57.760 129.6  2.21  0.066
| 16415                6 -57.859 129.8  2.41  0.060
| 135236               6 -57.934 130.0  2.56  0.056
| 4180                 6 -57.968 130.0  2.63  0.054
| 16777307             6 -58.066 130.2  2.82  0.049
| 134217828 6.906e-05  6 -58.087 130.3  2.87  0.048
| 16777295             6 -58.263 130.6  3.22  0.040
| 31                   5 -59.720 130.9  3.50  0.035
| 4                    3 -62.418 131.4  3.99  0.027
```

structurally correct—containing (at least most of) the true predictors—and numerically accurate—yielding accurate and unbiased parameter estimates. We concern ourselves here with structural correctness and defer the question of accuracy of the parameter estimates to the next section (§6.2). We ask, does stepwise model selection reliably identify the true model structure (Burnham & Anderson 2001)?

WHAT IS "MODEL STRUCTURE"?

To determine if stepwise model selection can identify the true model structure, we first need to define what we mean by model structure. There are two common measures of model structure, to which we add two more. In common usage, model structure may be measured by either its true or its false positive rate. The true positive rate is the proportion of models that correctly assigns a significant effect to a true model predictor (i.e., those with effects significantly different from zero). In contrast, the false positive rate is the proportion of wrongly identified, spurious predictors for which the effect is estimated to be $\neq 0$ when in fact it equals zero. To these two measures, we add S- and M-errors (Gelman & Carlin 2014). The S-error rate is the proportion of significant effects that are estimated to have the wrong sign, whereas the M-error rate is the proportion of significant effects whose estimates are off by one or more orders of magnitude (Gelman & Carlin 2014; and see also §9.3.4 in Chapter 9).

Code Box 6.3: A single simulation of a model predicting military coups in sub-Saharan Africa

```
# Load required libraries
 library(faraway) # For the africa dataset
 library(mice) # For imputation of missing values

 africa.imp <- complete(mice(africa, m=5, method="pmm")) #imputation
 attach(africa.imp) #for ease of reading the code

 set.seed(123)
  coups <- rpois(nrow(africa.imp), lambda=exp(-.7 + .15*oligarchy -.0018*size
       + 0.03*numelec + .0002*numelec*size))
  dats <- cbind.data.frame(coups, africa.imp)

# Stepwise model selection
 fst <- step(ffull <- glm(coups ~ (oligarchy + pollib + parties + pctvote + popn
      + size + numelec)^2, data=dats, family=poisson, maxit=100), trace=F)
```

For all four measures, the reference model is the full model: the one with all the terms that is fitted before doing model selection. Especially for small datasets, the full model may have far too few degrees of freedom to have a chance of finding any effect (i.e., the full model is considered to be underpowered). In such cases the true positive rate of the full model would be low. By reducing the number of predictor variables, model selection should increase power and consequently increase the true positive rate.

SIMULATIONS OF MODEL SELECTION AND ITS EFFECT ON THE TRUE POSITIVE RATE

We start by simulating a single new `africa` dataset, fitting the full model, and running a stepwise regression (Code Box 6.3).

We then extract the *p*-values and the coefficient estimates, define the true parameters, and compute, for this single run, the error rates in which we are most interested (Code Box 6.4). These rate computations are simple, but we need to pay careful attention to which parameter is extracted from the relevant objects.

Next, we run this simulation many times (Code Box 6.5). To simplify the code a bit further, we first wrote a function to compute the error rates and called this function within the replicated simulation.

The results of our simulations are quite clear (Figure 6.1). The full model does very poorly in detecting actual effects (true positives), but at least it also has a low rate of returning spurious effects (false positives). Its S- and M-error rates are nearly zero. In contrast, (stepwise) model selection does detect the true predictors in 10% of the cases, but even more often the significant predictors are falsely identified (the false positive rate exceeds the true positive rate). S-errors seem not to be a problem for this particular stepwise model, but M-errors do occur (see also Stewart et al. 2023). The problems for inference introduced by model selection are well known (e.g., Leeb & Pötscher 2005), and there is a rich literature on how to possibly correct for introduced biases (reviewed recently in Kuchibhotla et al. 2022). In many disciplines, however, model selection still incorrectly precedes inference. Our simulation only serves to highlight this issue once again.

Code Box 6.4: Estimating error rates for simulation

```
# Matrix of estimates:
 coef.mat <- matrix(NA, ncol=28, nrow=2)
 colnames(coef.mat) <- names(coef(ffull)[-1]) # kick out intercept!
 coef.mat[1,] <- coef(ffull)[-1]
 coef.mat[2, colnames(coef.mat) %in% names(coef(fst)[-1])] <- coef(fst)[-1]

# Matrix of p values:
 p.mat <- matrix(NA, ncol=28, nrow=2)
 colnames(p.mat) <- names(coef(ffull)[-1]) # kick out intercept!
 p.mat[1,] <- summary(ffull)$coefficients[-1, 4]
 p.mat[2, colnames(p.mat) %in% names(coef(fst)[-1])] <-
     summary(fst)$coefficients[-1, 4]

# Analysis of the matrix of coefficients:
 truth <- coef(ffull)[-1] # Without intercept
 truth[] <- 0
 truth[c("oligarchy", "size", "numelec", "size:numelec")] <- c(.15, -.0018, .003,
      .0002)

# True positives:
 realIndex <- which(truth != 0)
 (sum(p.mat[1, realIndex] < 0.05))/28  # True positive rate

| [1] 0

 (sum(p.mat[2, realIndex] < 0.05))/28  # True positive rate

| [1] 0.03571429

# False positive rate:
 sum(p.mat[1, (1:28)[-realIndex]] < 0.05)/28 # Error rate

| [1] 0.03571429

 sum(p.mat[2, (1:28)[-realIndex]] < 0.05, na.rm=T)/28 # Error rate

| [1] 0.3214286

# Signifcant only:
 signifs <-  (p.mat < 0.05 ) * matrix(truth != 0, nrow=2, ncol=28, byrow=T)

# Sign error (S-error): only for significant findings!!
 sum(sign(coef.mat[1, signifs[1,]==1]) != sign(truth[signifs[1,]==1]))

| [1] 0

 sum(sign(coef.mat[2, signifs[2,]==1]) != sign(truth[signifs[2,]==1]), na.rm=T)
```

```
| [1] 0

# Order-of-magnitude error (M-Error): only for significant findings!!
 ratio1 <- truth / coef.mat[1, signifs[1,]==1]
 sum(abs(ratio1) > 10 | abs(ratio1) < .1)

| [1] 0

 ratio2 <- truth[signifs[2,]==1] / coef.mat[2, signifs[2,]==1] # Later need to
     subtract 1 for the (trivial) intercept!
 sum(abs(ratio2) > 10 | abs(ratio2) < .1, na.rm=T)

| [1] 0
```

Code Box 6.5: Simulation for repeated computation of error rates

```
# First we write a function for computing error rates
# We are using coef.mat, p. mat, and truth from Code Box 6.4
# Note that the number of columns of both matrices and length of truth vector
    must be the same!

 eval.full.and.step <- function(p.mat, coef.mat, truth){
   realIndex <- which(truth != 0)
   truePosRateFull <- (sum(p.mat[1, realIndex] < 0.05))/28  # True positive rate
   truePosRateStep <- sum(p.mat[2, realIndex] < 0.05, na.rm=T)/28  # True
       positive rate
   falsePosRateFull <- sum(p.mat[1, (1:28)[-realIndex]] < 0.05)/28 # Error rate
       (false positives)
   falsePosRateStep <- sum(p.mat[2, (1:28)[-realIndex]] < 0.05, na.rm=T)/28 #
       Error rate

   # Signifcant ones only:
    signifs <-  (p.mat < 0.05 ) * matrix(truth != 0, nrow=2, ncol=28, byrow=T)
   # Compute S- and M-error rates only for significant findings!
    if (sum(signifs[1,], na.rm=T) != 0){
     SignErrorFull <- sum(sign(coef.mat[1, signifs[1,]==1]) != sign(truth[signifs
         [1,]==1])) / sum(signifs[1,], na.rm=T)
     ratio1 <- truth[signifs[1,]==1] / coef.mat[1, signifs[1,]==1]
     MagnErrorFull <- sum(abs(ratio1) > 10 | abs(ratio1) < .1)  /  sum(signifs
         [1,], na.rm=T) # Wrong order of magnitude
    } else {
     SignErrorFull <- 0
     MagnErrorFull <- 0
    }
    if (sum(signifs[2,], na.rm=T) != 0){
     # S-error
```

```
    SignErrorStep <- sum(sign(coef.mat[2, signifs[2,]==1]) != sign(truth[signifs
        [2,]==1]), na.rm=T) / sum(signifs[2,], na.rm=T)
    # M-error
    ratio2 <- truth[signifs[2,]==1] / coef.mat[2, signifs[2,]==1]
    MagnErrorStep <- sum(abs(ratio2) > 10 | abs(ratio2) < .1, na.rm=T) / sum(
        signifs[2,], na.rm=T)
   } else {
    SignErrorStep <- 0
    MagnErrorStep <- 0
   }

  return(c(truePosRateFull, truePosRateStep, falsePosRateFull, falsePosRateStep
      , SignErrorFull, SignErrorStep, MagnErrorFull, MagnErrorStep))

} #end of the eval.full.and.step function

# Structural error, many runs
 truth <- coef(ffull)[-1] # without intercept
 truth[] <- 0
 truth[c("oligarchy", "size", "numelec", "size:numelec")] <- c(.15, -.0018, .003,
      .0002)

 sims <- replicate(1000, { # 1000 simulations
   coups <- rpois(nrow(africa.imp), lambda=exp(-.7 + .15*oligarchy -.0018*size +
       0.03*numelec + .0002*numelec*size))
   dats <- cbind.data.frame(coups, africa.imp)
   fst <- step(ffull <- glm(coups ~ (oligarchy + pollib + parties + pctvote + popn
       + size + numelec)^2, data=dats, family=poisson, maxit=100), trace=F)

  # Matrix of estimates:
   coef.mat <- matrix(NA, ncol=28, nrow=2)
   colnames(coef.mat) <- names(coef(ffull)[-1]) # Remove the intercept!
   coef.mat[1,] <- coef(ffull)[-1]
   coef.mat[2, colnames(coef.mat) %in% names(coef(fst)[-1])] <- coef(fst)[-1]

  # Matrix of p-values:
   p.mat <- matrix(NA, ncol=28, nrow=2)
   colnames(p.mat) <- names(coef(ffull)[-1]) # Remove the intercept!
   p.mat[1,] <- summary(ffull)$coefficients[-1, 4]
   p.mat[2, colnames(p.mat) %in% names(coef(fst)[-1])]  <- summary(fst)
       $coefficients[-1, 4]

   eval.full.and.step(p.mat, coef.mat, truth)
 })

 colMeans(sims)

| [1] 0.008714286 0.068714286 0.037571429 0.194392857 0.022166667
|     0.123000000 0.057500000 0.164500000
```

```
colnames(sims) <- c("truePosRateFull", "truePosRateStep", "falsePosRateFull",
    "falsePosRateStep", "SignErrorFull", "SignErrorStep", "MagnErrorFull",
    "MagnErrorStep")
apply(sims, 2, summary)
```

```
|         truePosRateFull truePosRateStep falsePosRateFull
| Min.        0.000000000      0.00000000       0.00000000
| 1st Qu.     0.000000000      0.03571429       0.00000000
| Median      0.000000000      0.07142857       0.00000000
| Mean        0.008714286      0.06871429       0.03757143
| 3rd Qu.     0.000000000      0.10714286       0.03571429
| Max.        0.142857143      0.14285714       0.50000000
|
|         falsePosRateStep SignErrorFull SignErrorStep
| Min.           0.0000000    0.00000000         0.000
| 1st Qu.        0.1071429    0.00000000         0.000
| Median         0.1785714    0.00000000         0.000
| Mean           0.1943929    0.02216667         0.123
| 3rd Qu.        0.2857143    0.00000000         0.000
| Max.           0.6428571    1.00000000         1.000
|
|         MagnErrorFull MagnErrorStep
| Min.           0.0000     0.0000000
| 1st Qu.        0.0000     0.0000000
| Median         0.0000     0.0000000
| Mean           0.0575     0.1645000
| 3rd Qu.        0.0000     0.3333333
| Max.           1.0000     1.0000000
```

6.1.5 Model selection improves interpretability: Is it folklore?

The false positive rates will be specific to the setting, including the number of observations, the strength of the effects, the complexity of the model, and the degree of correlation among the predictors (for more on the last item, see §6.2). Because we did not make any attempt to find a particular dataset that would falsify this rule-of-thumb, we regard this outcome as a representative one and conclude that model selection using stepwise regression increases the probability of spurious effects (i.e., type-I errors). This has also been found in a range of studies in other fields, especially when sample sizes are relatively small ($\ll 300$; e.g., Emiliano et al. 2014; Chen et al. 2019; Frankot et al. 2023).

Finally, we cannot always use model selection to improve interpretability of an otherwise complicated full model (Tredennick et al. 2021). Although some true predictors will come through, some false predictors often appear to be statistically significant and may be erroneously interpreted as having an important predictive effect.[6] In short: the rule-of-thumb that model selection improves interpretability is more folklore than truth.

[6] Taken to an extreme, we can easily formulate a model with only one predictor. For example, rainfall (which varies both spatially and temporally) can "explain" any number of global patterns, including economic production (Kotz et al. 2022). Such a finding remains a correlation without a causal mechanism, a short story that to be convincing would need other lines of supporting evidence.

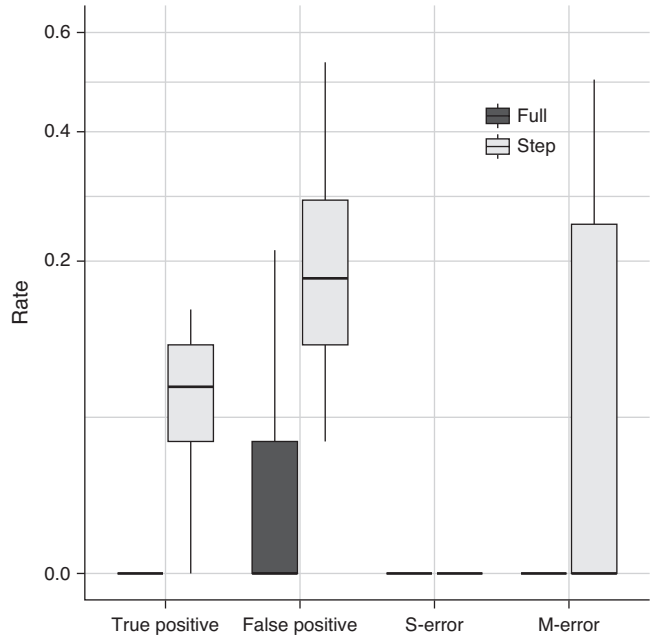


Figure 6.1 Error rates for full and stepwise simplified regression of simulated data on military coups in sub-Saharan Africa. Model selection improves detection of real effects (true positives) at the expense of an increased rate of identifying spurious effects (false positives) and occasional errors in magnitude (M-errors).

6.2 Selecting one of two correlated predictors does not mitigate collinearity in regression and machine learning

We are embarrassed to note that our most-cited paper (Dormann et al. 2013) is cited largely for a single number: 0.7. Indeed, we used simulations to show that if the predictor variables X of the standard regression model $Y = X\beta + \varepsilon$ are strongly correlated ($|r| \geq 0.7$), the parameter estimates β are unreliable. Further, models that include strongly correlated predictor variables yield poor predictions if the models differ in their correlation structures. However, a single number was not the main point of our paper. Rather, the key conclusions were two different ones: (1) that extrapolation to a *different* correlation structure is problematic (see also §5.7 in Chapter 5); and (2) all methods investigated suffered from this problem—not only classical GLMs, but also machine-learning and regularised methods (albeit to different extents). We emphasise these two points here, along with their importance for model predictions.

Specifically, in this section we are concerned with the issue of what to do when we identify two highly correlated predictors and whether model predictions change if we simply decide to exclude one of the two correlated predictors from the model.[7] We want to drive home the point that the rule-of-thumb of excluding one of two highly correlated predictors and then interpreting the retained variable as the predictor is simply wrong. To do so, we compare models in which the correlated variables are included in models in which one of the variables has been removed. To illustrate that the problem is independent of model classes

[7] Although we explicitly scorned this practice, it seems to be the main approach used by many researchers, including those who cite our paper as a reference. In contrast, using variable-reduction methods such as PCA to combine many correlated variables into a single representative one is done much less frequently, even though we encouraged that approach (Dormann et al. 2013).

(e.g., GLM and machine learning), we show analyses and simulations using both a simple linear model and a random forest model.

6.2.1 A case study: Plant species richness in the Appalachian Mountains

The species richness of plant in the Appalachian Mountains is partly a result of the characteristics of the soils on which the plants live (Bondell & Reich 2008; Wheeler & Peet 2016). A correlation matrix of the concentrations of elements present in the soil illustrates clearly that they are not independent of one another (Code Box 6.6).

The highest correlation (0.98) is between [Ca] and [Mg], but two other correlation coefficients also exceed |0.7| (between [K] and [Mg], and between [P] and [Na]).

6.2.2 Simulation and modelling

We now simulate a response, the number of plant species, *S*, as a function of only [Ca]. We then fit three models to these data: one model with [Ca] *and* [Mg] and all the other soil variables as predictors (full model), and one model each with only one of either [Ca] *or* [Mg] as a predictor, in addition to all the other soil variables (Code Box 6.7). We fit the models using standard linear regression and a random forest. We expect that model parameters in the full regression model will have substantial standard errors for the parameter estimates of [Ca] and [Mg], but we are open to what importance the random forest will attach to them. In contrast, in each of the reduced (single-factor) regression and random-forest models, we expect [Ca] and [Mg], respectively, to be extremely important, even though only [Ca] is the true predictor in our simulated dataset.

INITIAL RESULTS

In the full regression model, the effects of [Ca] and [Mg] were imprecisely and inaccurately estimated as 0.24 ± 1.1 (correct value $= 0.5$) and 0.20 ± 1.3 (true value $= 0$), respectively. Similarly, the full random forest model assigned high importance to both [Ca] and [Mg] (1.02 and 0.62, respectively). This indicates that neither the linear regression model nor the random forest model can separate the effects of these two strongly correlated predictors.

Running the reduced models with either [Ca] or [Mg] shifts the full effect of [Ca] onto [Mg] when [Ca] is not in the model. The linear model estimate for [Mg] in the [Mg]-only model is 0.47 ± 0.35, and random forest assigns the same importance to [Mg] (1.02) in the [Mg]-only model as it does to [Ca] in the [Ca]-only model (1.06). In sum, in both the linear regression and the random forest models, excluding one of a pair of highly correlated predictor variables shifts its contribution onto the remaining one of the pair.

MULTIPLE SIMULATIONS

Since we would not know the true causal relationship in a real dataset, we might select the variable of a pair of strongly correlated ones that "explains" *Y* better. We can use many runs of our simulation (Code Box 6.7) to see how often the models select the wrong variable. Across many simulations, we can count, for example, how often [Mg] has a higher importance in a random forest model than [Ca] (Code Box 6.8). If a model gets this right for our simulated data, [Ca] should have a value of 100, and [Mg] of 0. The more this value goes to 50:50, the less able the model is to pick the correct predictor.[8] Alongside the ratio, we store the standardised coefficient estimates of the three linear models.

[8] At this point it becomes a bit more important which implementation of random forest is used. We opted to use the `ranger` package because the routine implemented in C++ is much faster than the pure R implementation.

Code Box 6.6: The Appalachian soil data

```
# Read in the data
 soil <- read.csv("soil.csv") # Local copy of original data from Supplemenatry
      Material in Bondell & Reich (2008)
 soil <- as.data.frame(scale(soil[,-c(1,2,3,16)])) # Excludes some redundant
      columns and scales the data to get standarised regression coefficients in
      lm output

 round(cor(soil), 2)    # Compute pairwise correlations for all variables
```

	Ca	Mg	K	Na	P	Cu	Zn	Mn
Ca	1.00	0.98	0.62	0.64	0.47	0.61	0.29	0.16
Mg	0.98	1.00	0.71	0.67	0.51	0.63	0.37	0.23
K	0.62	0.71	1.00	0.36	0.22	0.29	0.15	0.13
Na	0.64	0.67	0.36	1.00	0.80	0.43	0.56	0.24
P	0.47	0.51	0.22	0.80	1.00	0.15	0.62	0.34
Cu	0.61	0.63	0.29	0.43	0.15	1.00	0.29	0.16
Zn	0.29	0.37	0.15	0.56	0.62	0.29	1.00	0.49
Mn	0.16	0.23	0.13	0.24	0.34	0.16	0.49	1.00
HumicMatter	0.06	-0.03	-0.31	0.14	0.08	0.21	-0.18	0.01
Density	0.35	0.32	0.22	-0.15	-0.25	0.21	-0.10	0.00
pH	0.41	0.40	0.24	-0.23	-0.26	0.36	-0.13	-0.01
ExchAc	-0.31	-0.29	-0.19	0.32	0.41	-0.35	0.26	0.11

	HumicMatter	Density	pH	ExchAc
Ca	0.06	0.35	0.41	-0.31
Mg	-0.03	0.32	0.40	-0.29
K	-0.31	0.22	0.24	-0.19
Na	0.14	-0.15	-0.23	0.32
P	0.08	-0.25	-0.26	0.41
Cu	0.21	0.21	0.36	-0.35
Zn	-0.18	-0.10	-0.13	0.26
Mn	0.01	0.00	-0.01	0.11
HumicMatter	1.00	-0.61	-0.25	0.31
Density	-0.61	1.00	0.63	-0.68
pH	-0.25	0.63	1.00	-0.93
ExchAc	0.31	-0.68	-0.93	1.00

Code Box 6.7: Simulating *S* as a function of [Ca] and three-fold analysis

```
# Load required libraries
 library(ranger) # For random-forest model

# Create a simulated dataset of number of plants (S) as a function of [Ca]
 set.seed(123)
 soil$S <- round(with(soil, 15 + .5*Ca + rnorm(nrow(soil), mean=0, sd=.2)))

# Full linear regression model
```

```
 summary(lm(S ~ Ca + Mg + K + Na + P + Cu + Zn + Mn + HumicMatter + Density + pH
      + ExchAc, data=soil))
10
| Residuals:
|      Min       1Q   Median       3Q      Max
| -0.42344 -0.10118  0.00452  0.06353  0.44353
|
15 | Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept) 15.15000    0.08045 188.318 3.14e-14 ***
| Ca           0.24430    1.09438   0.223    0.830
| Mg           0.20489    1.25387   0.163    0.875
20 | K            0.06787    0.23131   0.293    0.778
| Na          -0.17747    0.21473  -0.826    0.436
| P            0.11629    0.19130   0.608    0.562
| Cu          -0.32288    0.19884  -1.624    0.148
| Zn           0.26093    0.17610   1.482    0.182
25 | Mn          -0.12054    0.12723  -0.947    0.375
| HumicMatter  0.23793    0.33477   0.711    0.500
| Density      0.18071    0.30401   0.594    0.571
| pH          -0.11676    0.28373  -0.412    0.693
| ExchAc      -0.15752    0.29858  -0.528    0.614
30 | ---

# Full random forest model
 sort(importance(ranger(S ~ Ca + Mg + K + Na + P + C Cu + Z Zn + Mn +
      HumicMatter + Density + pH + ExchAc, data=soil, importance="impurity")),
      decreasing=T)

35 |          Ca          Mg      ExchAc          pH           K
|  1.01852823  0.61848245  0.42473856  0.36523790  0.29371608
|
|          Cu     Density          Zn           P          Na
|  0.24477317  0.21628317  0.17316933  0.16992470  0.10478916
40 |
| HumicMatter          Mn
|  0.06376668  0.04649058

# Reduced linear regression model with Ca but without Mg
45 summary(lm(S ~ Ca + K + Na + P + Cu + Zn + Mn + HumicMatter + Density + pH +
      ExchAc, data=soil))

| Residuals:
|      Min       1Q   Median       3Q      Max
| -0.42371 -0.10185  0.00406  0.06424  0.44299
50 |
| Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept) 15.15000    0.07540 200.938 4.21e-16 ***
```

```
| Ca           0.41516    0.30272   1.371   0.2075
| K            0.08896    0.17987   0.495   0.6342
| Na          -0.16852    0.19459  -0.866   0.4117
| P            0.12321    0.17483   0.705   0.5010
| Cu          -0.30189    0.14223  -2.123   0.0666 .
| Zn           0.25725    0.16369   1.572   0.1547
| Mn          -0.11041    0.10413  -1.060   0.3199
| HumicMatter  0.20711    0.25920   0.799   0.4473
| Density      0.15397    0.24014   0.641   0.5393
| pH          -0.09959    0.24701  -0.403   0.6974
| ExchAc      -0.15262    0.27841  -0.548   0.5985
| ---

# Reduced random forest model with Ca but without Mg
 importance(ranger(S ~ Ca + K + Na + P + Cu + Zn + Mn + HumicMatter + Density +
     pH + ExchAc, data=soil, importance="impurity"))

|         Ca           K          Na           P          Cu
| 1.06115175  0.37305372  0.15748217  0.32253616  0.40237272
|         Zn          Mn HumicMatter     Density          pH
| 0.22484311  0.09524508  0.09954993  0.20974180  0.35573631
|     ExchAc
| 0.47752058

# Reduced linear regression model with Mg but without Ca
 summary(lm(S ~ Mg + K + Na + P + Cu + Zn + Mn + HumicMatter + Density + pH +
     ExchAc, data=soil))

| Residuals:
|      Min       1Q   Median       3Q      Max
| -0.42099 -0.09831 -0.01947  0.07310  0.43631
|
| Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept) 15.15000    0.07552 200.608 4.27e-16 ***
| Mg           0.47232    0.34741   1.360   0.2110
| K            0.05062    0.20467   0.247   0.8109
| Na          -0.18308    0.20019  -0.915   0.3872
| P            0.11246    0.17886   0.629   0.5470
| Cu          -0.35235    0.13960  -2.524   0.0356 *
| Zn           0.27077    0.16005   1.692   0.1292
| Mn          -0.13681    0.09791  -1.397   0.1999
| HumicMatter  0.29246    0.21490   1.361   0.2106
| Density      0.22882    0.20125   1.137   0.2884
| pH          -0.13647    0.25312  -0.539   0.6045
| ExchAc      -0.16710    0.27737  -0.602   0.5636
| ---

# Reduced random forest model with Mg but without Ca
```

```
 sort(importance(ranger(S ~ Mg + K + Na + P + Cu + Zn + Mn + HumicMatter +
      Density + pH + ExchAc, data=soil, importance="impurity")), decreasing=T)

|          Mg      ExchAc          pH           K           P
|  1.02370597  0.50144811  0.39724393  0.39459045  0.31047709
|          Cu          Zn     Density          Na HumicMatter
|  0.28821793  0.24863169  0.22513302  0.22450562  0.11724270
|          Mn
|  0.08987017
```

Code Box 6.8: Many simulations: Average importance with/out selection

```
#Load required libraries
 library(ranger) For random forest models

 set.seed(123)
 importances.simu <- replicate(1000, { # 1000 Replicates
   soil$S <- round(with(soil, 15 + .5*Ca + rnorm(nrow(soil), mean=0, sd=.2)))
   c(coef(lm(S ~ Ca + Mg + K+Na+P+Cu+Zn+Mn+HumicMatter+Density+pH+ExchAc, data=
       soil))[c("Ca", "Mg")],
     coef(lm(S ~ Ca + K + Na + P + Cu + Zn + Mn + HumicMatter + Density + pH +
         ExchAc, data=soil))["Ca"],
     coef(lm(S ~ Mg + K + Na + P + Cu + Zn + Mn + HumicMatter + Density + pH +
         ExchAc, data=soil))["Mg"],
     importance(ranger(S ~ Ca + Mg + K + Na + P + Cu + Zn + Mn + HumicMatter +
         Density + pH + ExchAc, data=soil, importance="impurity"))[c("Ca", "Mg
         ")],
     importance(ranger(S ~ Ca + K + Na + P + Cu + Zn + Mn + HumicMatter +
         Density + pH + ExchAc, data=soil, importance="impurity"))["Ca"],
     importance(ranger(S ~ Mg + K + Na + P + Cu + Zn + Mn + HumicMatter +
         Density + pH +  ExchAc, data=soil, importance="impurity"))["Mg"]
   )
 })

# Linear regression model: Ca > Mg?
 sum(importances.simu[1,] > importances.simu[2,])

| [1] 639

# Random Forest model: Ca > Mg?
 sum(importances.simu[5,] > importances.simu[6,])

| [1] 562
```

Code Box 6.9: Average importance of Ca and Mg in random forests (full model)

```
# Ca vs Mg in the full model (columns 5 and 6)
 mean(importances.simu[5,] /  (importances.simu[5,] +importances.simu[6,]))

| [1] 0.5071435 # For [Ca]

 mean(importances.simu[6,] /  (importances.simu[5,] +importances.simu[6,]))

| [1] 0.4928565 # For [Mg]

# Ca vs Mg in single models (columns 7 and 8)
 mean(importances.simu[7,] /  (importances.simu[7,] +importances.simu[8,]))

| [1] 0.5047745 # For [Ca]

 mean(importances.simu[8,] /  (importances.simu[7,] +importances.simu[8,]))

| [1] 0.4952255 # For [Mg]
```

In 64% of 1,000 linear regression models and 56% of the 1000 random forest models, [Ca] was correctly picked as more influential than [Mg]. For the random-forest model, the probability of picking the correct model is not different from tossing a fair coin and seeing it land heads up. Although the linear model is substantially better, it is still not good.

Scaling the importance values from the random forest models allows us to average them for the three different random forest models (Code Box 6.9).

This 50:50 split should suffice to convince us that random forest models cannot identify the correct predictor from two highly correlated ones.

6.2.3 Bias and precision of parameter estimates in the regression model

To hammer the final nail into this coffin, we also look at the distribution of parameter estimates for the linear regression model. Because of the outcome of our initial simulation (Code Box 6.7), we expect that multiple simulations of the full model should give us unbiased but very imprecise parameter estimates. In contrast, we expect that both the [Ca]- and [Mg]-only models should have much more precise parameter estimates, but only the parameter estimates of the [Ca]-only model should be accurate; the parameter estimates of the [Mg]-only model should be strongly biased. These expectations were met (Figure 6.2). Note in particular that the [Mg]-only model (thin grey line in Figure 6.2) also estimates its effect as 0.5, i.e., the same as the [Ca]-only model, simply because the model cannot allocate any weight to the missing [Ca].

The change in width of the distribution is referred to as variance inflation of the standard error of the parameter estimate (e.g., Fox & Monette 1992). The full model is approximately unbiased and its wide distribution of parameter estimates represents the logical uncertainty as to which predictor is actually causal (Morrissey & Ruxton 2018).

6.2.4 Is "pick one" to solve collinearity folklore?

The logical problem of identifying the correct predictor from two highly correlated ones cannot be solved, even for a small, uncomplicated problem. With strongly correlated predictors,

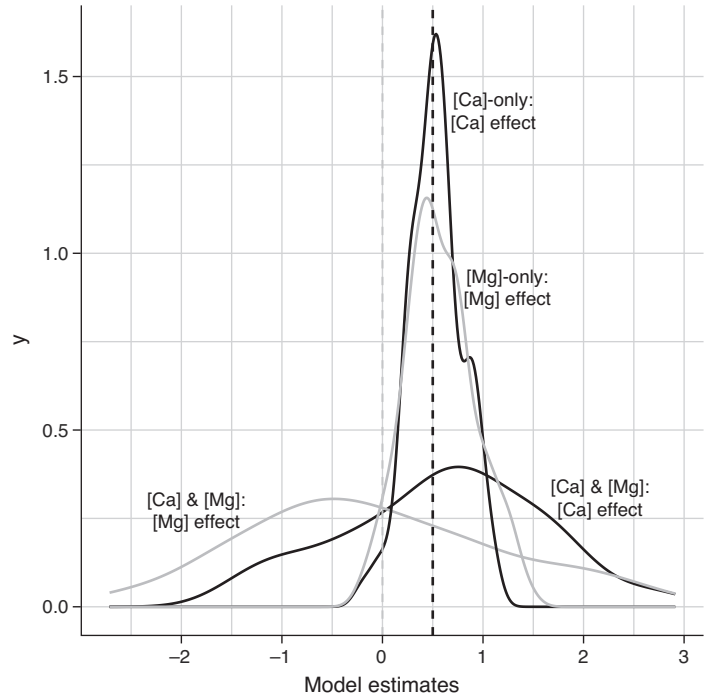


Figure 6.2 Distribution of linear model estimates for 1,000 simulated datasets. Estimates for the effect of predictors [Ca] (black) and [Mg] (grey) in a joint regression (heavy lines, wide distributions) and in separate models (thin lines, narrow distributions). True values are indicated by the vertical dashed black (for [Ca]) and grey (for [Mg]) lines. The estimates of the joint regression are very uncertain yet approximately unbiased, whereas the separate estimates are biased and much more precise than is warranted.

classical linear regression may, on average, yield unbiased parameter estimates, but the estimates are so imprecise as to be effectively useless. Random forest models are not magic wands that can make the problems of multicollinearity in a regression model disappear (contra Oppel et al. 2009). If a random forest is a good representative of machine-learning models in general, the latter are no more a panacea than is traditional regression for dealing with multicollinearity (Dormann et al. 2013).

A useful way of thinking about this logical problem was formulated by McElreath (2020). In an analysis of a basketball player's abilities and performance, we might include the average length of their legs or the length of only one leg, but not the lengths of both the left leg and the right leg. This is because the length of one leg adds no more information than is already included in the length of the other leg. Any statistical tool is agnostic as to the meaning of the predictors, and hence it cannot evaluate whether [Mg] is like the second leg of the soil or if it is a meaningful and informative second predictor. The higher the correlation between two predictors, the more they look to the model like two legs.

6.3 It is not OK to categorise continuous predictor variables

Many predictor variables are continuous. Common examples include age, income, time, and wavelengths of light. Predictors like these, and many others, are often grouped into categories, such as infants, children, teenagers, adults, and seniors. We do this because it appears to make it much easier to understand the effect of a category like "child" or "adult"

on, say, cognitive ability, than to try to articulate a difference in the same trait between a 14- and a 15-year-old.

From a statistical standpoint, however, continuous variables such as age should not be split into categories (e.g., Altman & Royston 2006). The arbitrary position of any such categorical boundary and the increase in the number of fitted parameters (one for each category, as opposed to a single slope parameter for the continuous predictor) is a high price to pay for a seemingly more intuitive interpretation. Worse, what seems to be a universal human desire of putting continuous features into boxes can lead to confirmation bias by creating the illusion of conformity within, and dissimilarity among, boxes. As Dawkins (2011) famously decried in his essay "The tyranny of the discontinuous mind," two people, one just above and the other just below a categorical "poverty line" likely are indistinguishable in every other aspect. Similarly, failing an exam with 49% of the points is really no worse than passing with 50% (Shepard 2019) and $p = 0.049$ (statistically significant) is not really that different from a $p = 0.051$ (not statistically significant). Nonetheless, making either of these distinctions can lead to radically different decisions or outcomes.

6.3.1 A case study: The Birkenhead drill and the sinking of the RMS *Titanic*

The romantic, and from a male point of view, chivalrous notion of saving women and children first in an emergency situation is known as the Birkenhead drill.[9] The Birkenhead drill has no legal status in any law book (Delap 2006) and has been applied inconsistently in famous tragic incidents. Here, we are interested in statistically evaluating whether people in charge actually have followed the drill when it really mattered.

DID A GREATER PROPORTION OF CHILDREN SURVIVE THE SINKING OF THE *TITANIC*?

Consider the disastrous sinking of the RMS *Titanic* in 1912. Although mortality was substantial among adult men (around 80%) and much less among women (around 20%), it is not obvious whether children aboard the *Titanic* benefited from the Birkenhead drill. For the *Titanic* disaster we have data to explore this question (Figure 6.3). The data suggest a substantial drop in survival for girls (somewhere below $\approx$ 18 years old), but a linear and small increase of survival for young boys.

Before we can determine whether children sailing on the doomed *Titanic* benefited from the Birkenhead drill, we need to define what we mean by children. On a sinking boat, no one checks birth certificates, so were children identified by the youthful impression they projected? Does childhood refer to the perceived helplessness of a person or to their pre-adolescent status? Should, in a statistical analysis, the category "child" be truncated at, for example, 6, 14, 18, or 21 years (representing, respectively, pre-school, working, legal, and marital age in the early twentieth century)? Depending on the age at which we draw the line between children and adults we may get very different answers to the question of whether children benefited from the Birkenhead drill on the *Titanic* (Code Box 6.10).

The simple tabulations in Code Box 6.10 illustrate that delimiting the end of childhood at age six leads to a higher estimate of mortality of children than adults, whereas considering all individuals less than 18 years old to be children would lead us to conclude

[9] The naming of the Birkenhead drill comes from Rudyard Kipling's lyrical interpretation, in his 1893 poem "Soldier an' Sailor Too," of the 1852 sinking of HMS *Birkenhead*:

> To take your chance in the thick of a rush, with firing all about,
> Is nothing so bad when you've cover to 'and, an' leave an' likin' to shout;
> But to stand an' be still to the Birken'ead drill is a damn tough bullet to chew,
> An' they done it, the Jollies – 'Er Majesty's Jollies – soldier an' sailor too!

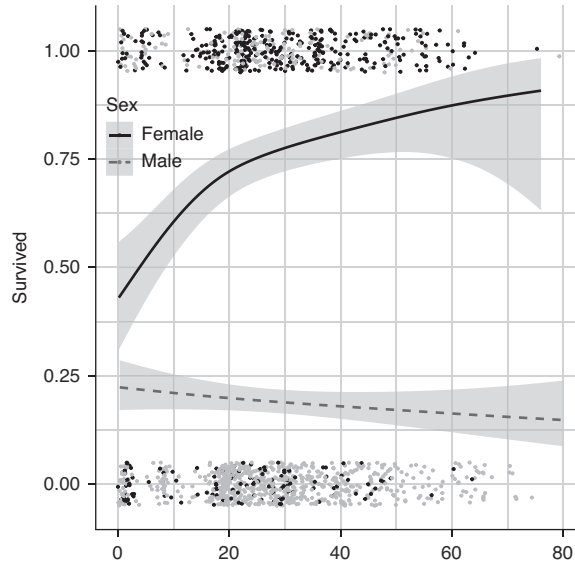


Figure 6.3 Survivourship of male (grey points) and female passengers (black points) on the RMS *Titanic* as a function of their age. Lines are smooth regressions (fit with a GAM) and associated 95% confidence intervals. Raw data for survival (1) and death (0) plotted are jittered to illustrate more clearly the number of observations available.

that a greater percentage of children survived than did adults. This type of arbitrary category-dependent result does not instil confidence in the robustness of this analysis of the effect of the Birkenhead drill.

The more general question this raises is how often and under what circumstances will categorisation of a continuous variable lead to wrong conclusions relative to treating continuous variables as continuous? Since we want to ensure that we know the truth and can trust the results of our analyses, we resort to data simulation.

SIMULATING THE EFFECT OF CATEGORISING AGE

As a first step, we simulate a *Titanic*-like dataset, with only additive effects of sex and age. This is a very simple case, but if categorisation leads to loss of information here, it surely will not be better in more complex situations. Next, we threshold childhood age at any year between 1 and 21 and fit a GLM with the categorical age as the predictor variable (Code Box 6.11). We also fit a GLM with age as a continuous predictor (as in Figure 6.3). One informative result would be the proportion of correctly detected significant effects of age on survival. Another would be the additional prediction error resulting from categorisation across all ages relative to the prediction error of a continuous model.

This single simulation suggested a loss of information incurred by categorising the continuous predictor `age` (Figure 6.4), but we need to run many such simulations.

In our multiple simulations (Code Box 6.12), we counted the proportion of correct significant results and quantified the error of the fitted model to the entire dataset.[10]

The results for counting significant values (stored in the first dimension of the `sims` array) and mean absolute difference (MAD; `sims[2,,]`) are clear (Figure 6.5). The type-II

[10] For unbounded errors, the root mean square error (RMSE) would be the appropriate measure of error. In this case, where prediction is bounded between 0 and 1, and the observation is either 0 (dead) or 1 (alive), the mean absolute difference (MAD) is a bit more intuitive: predicting a value of 0.3 and observing a 0 (or a 1) intuitively is an error of 0.3 (or 0.7, respectively). The RMSE and MAD are similar in our simulations, and the choice of the error metric typically will not greatly affect the conclusion.

Code Box 6.10: Two ways of categorising age of children on the Titanic

```
# Load required libraries
 library(effects) # Includes the Titanic data
 library(mice) # For imputation of missing values
 library(Hmisc) # Includes the cut2 function to cut a continuous numeric
     variable into discrete intervals

# Load and summarise the data
 data(TitanicSurvival)
 summary(TitanicSurvival)

| survived      sex           age          passengerClass
| no :809   female:466   Min.   : 0.1667   1st:323
| yes:500   male  :843   1st Qu.:21.0000   2nd:277
|                        Median :28.0000   3rd:709
|                        Mean   :29.8811
|                        3rd Qu.:39.0000
|                        Max.   :80.0000
|                        NA's   :263

 Ti <- complete(mice(TitanicSurvival, method="pmm", print=F)) # Impute missing
     values
 Ti$survived <- ifelse(Ti$survived == "yes", 1, 0) #Make survivorship numeric
 ageCat <- cut2(Ti$age, cuts=c(0, 6, 100)) # Define "children" to be < 6 years
     old
 table(Ti$survived, ageCat) #Tabulate the data (raw numbers)

|   ageCat
|    [  0,  6) [  6,100]
|  0        81       728
|  1        44       456

 table(Ti$survived, ageCat) / matrix(tapply(Ti$survived, ageCat, length), nrow
     =2, ncol=2, byrow=T) #Tabulate proportions

|   ageCat
|    [  0,  6) [  6,100]
|  0 0.6480000 0.6148649
|  1 0.3520000 0.3851351

 ageCat <- cut2(Ti$age, cuts=c(0, 18, 100)) #Repeat for "children" to be <18
     years old
 table(Ti$survived, ageCat) / matrix(tapply(Ti$survived, ageCat, length), nrow
     =2, ncol=2, byrow=T)

|   ageCat
|    [  0, 18) [ 18,100]
|  0 0.5921053 0.6234968
|  1 0.4078947 0.3765032
```

Code Box 6.11: Simulate an effect of age and model survivorship as a function of categorical or continuous age

```
# Load required libraries
 library(Hmisc) # Includes the cut2 function

# Simulate a dataset
 isMale <- ifelse(Ti$sex == "male", 1, 0)
 theAge <- Ti$age
 theSex <- Ti$sex
 survSim <- rbinom(1309, prob=plogis(0 - 1*isMale + 0.01*theAge), size=1)

# Model with age as a continuous variable
 fcont <- glm(survSim ~ theAge, family=binomial) # Age is a continuous predictor
 pCont <- summary(fcont)$coefficients[2,4]
 fitCont <- predict(fcont, type="response")
 ellCont <- sum(dbinom(survSim, prob=fitCat, size=1, log=T))
 madCont <- mean(abs(survSim - fitCont))

# Model with childhood categorised at different values from 1 to 21.
 ageCuts <- 1:21
 p.vec <- NA # Vector to collect p values
 ellCat.vec <- madCat.vec <- NA
 for (i in 1:21){ # Cut the end of childhood at different ages
   ageCat <- cut2(theAge, cuts=c(0, ageCuts[i], 100))
   fcat <- glm(survSim ~ ageCat, family=binomial)
   p.vec[i] <- summary(fcat)$coefficients[2,4]
   fitCat <- predict(fcat, type="response")
   ellCat.vec[i] <- sum(dbinom(survSim, prob=fitCat, size=1, log=T))
   madCat.vec[i] <- mean(abs(survSim - fitCat))
   rm(fcat, ageCat)
 }
```

error rate (false negatives) is always higher for the categorised data and the error in model fit also is consistently higher.

As usual, this simulation only represents a specific question and does not allow for general inference of the effect of categorisation. It is this kind of observation, however, that is behind the statement that "categorisation leads to loss of information," which is obvious in one way (replacing a linear relationship by two horizontal lines) and quantifiable in the way presented here.

The effect of categorisation may differ when more than two categories are created from a continuous variable. On one hand, because more categories corresponds more closely to a continuous variable, having them should lead to a better model fit and lower type-II error rate. On the other hand, however, one additional parameter needs to be fit for each category, reducing the residual degrees of freedom and making the model more complex with each category added.

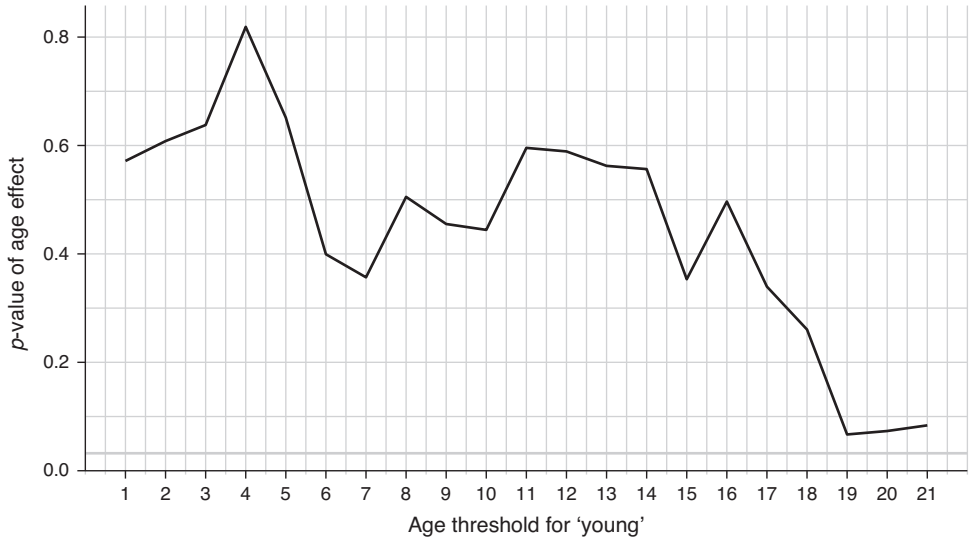


Figure 6.4 Example of a single simulation on the significance (i.e., $p \leq 0.05$) of age when it is categorised into two classes (child and non-child) with different cut points (from 1 to 21). The thick grey horizontal reference line is the *p*-value for the effect of a continuous value for age.

6.3.2 Is categorisation OK?

Statisticians are clear in their overall recommendation: "Avoid categorising continuous variables and predicted values at all costs."[11] This viewpoint is pervasive and generally applied in biomedical research (e.g., Royston et al. 2006), but much less so elsewhere. Ecologists and psychologists, for example, have been slow to accept this insight, even though simulations repeatedly have helped to make the point more clearly (e.g., Kreyling et al. 2018). We encourage you to do similar simulations before you categorise your own continuous data.

6.3.3 Avoid categorising in manipulative experiments, too

In the preceding discussion and case study, we implicitly assumed that the data were derived from observations, not manipulative experiments. There are not, as far as we know, any tests of the utility of the Birkenhead drill in which a researcher pre-selected individuals for inclusion as children and then proceeded to sink a ship. In many fields, however, manipulative experiments are the norm and it is routine to apply treatments in discrete treatment levels. Examples include studying the growth of plants when the concentration of CO_2 in the growth chamber or the forest is either low or high, where "low" is the current (ambient) atmospheric [CO_2] and "high" is much higher—e.g., double the ambient value (U.S. DOE 2020). Similarly, in early (Phase I or II) clinical trials of new drugs, experiments are run to determine the range of doses (quantity of the drug delivered to a patient at a given time) that are either optimally effective or least harmful. Guidance for designing such dose-ranging studies specifies selecting one dose, or at most three doses, for comparative trials in addition to a control (e.g., a placebo group, low-dose group, medium-dose group, and a high-dose

[11] Quote of Frank Harrell provided by Andrew Gelman (2017). There are other quotes there well worth further reflection. His reasoning, together with lots of simulations and references, is detailed at Harrell (2022).

Code Box 6.12: Multiple simulations of continuous and categorical effects of age on survivorship

```
# Simulate 100 replicates
 set.seed(123)
 sims <- replicate(100, {
   # Simulate data and set up vectors for output
    survSim <- rbinom(1309, prob=plogis(0 - 1*isMale + 0.02*theAge) , size=1)
    p.vec <- NA
    ellCat.vec <- madCat.vec <- NA
   for (i in 1:21){ # Cut the end of childhood at different ages
     ageCat <- cut2(theAge, cuts=c(0, ageCuts[i], 100))
     fcat <- glm(survSim ~ ageCat, family=binomial)
     p.vec[i] <- summary(fcat)$coefficients[2,4]
     fitCat <- predict(fcat, type="response")
     ellCat.vec[i] <- sum(dbinom(survSim, prob=fitCat, size=1, log=T))
     madCat.vec[i] <- mean(abs(survSim - fitCat))
     rm(fcat, ageCat, fitCat)
   }
   # Continuous model
    fcont <- glm(survSim ~ theAge, family=binomial)
    pCont <- summary(fcont)$coefficients[2,4]
    fitCont <- predict(fcont, type="response")
    ellCont <- sum(dbinom(survSim, prob=fitCat, size=1, log=T))
    madCont <- mean(abs(survSim - fitCont))

  # Output results
   isSignificant <- ifelse(c(p.vec, pCont) < 0.05, 1, 0)
   MAD <- c(madCat.vec, madCont)
   rbind(isSignificant, MAD)
 })

# Summaries
 rowMeans(sims[1,,])

# The first 21 values are from the categorical model, the last is the continuous
      model
# A larger proportion of significant values is better

|  [1] 0.12 0.60 0.62 0.91 0.93 0.90 0.90 0.90 0.92 0.95 0.95 0.93 0.94 0.99
| [16] 0.92 0.91 0.93 0.90 0.84 0.89 0.91
| [22] 1.00

 rowMeans(sims[2,,])

# The first 21 values are from the categorical model, the last is the continuous
      model
# A lower type-II error rate is better
```

```
45 |  [1] 0.4981847 0.4968234 0.4966001 0.4951637 0.4950236 0.4951191 0.4950487
   |  [8] 0.4949499 0.4949379 0.4947086 0.4946778 0.4944717 0.4945023 0.4944828
   | [15] 0.4945693 0.4948539 0.4949116 0.4949465 0.4953232 0.4952985 0.4948004
   | [22] 0.4897562
```

group), even though individual responses to different drug doses are likely to be nonlinear (Schmidt 1988; Ting 2006).

In the last fifty years or so, many agricultural and ecological researchers have suggested alternatives to factorial designs that are better suited to identifying nonlinear responses to underlying gradients (e.g., Nelder 1962; Antonovics & Fowler 1985; Inouye 2001; Cottingham et al. 2005; Gotelli & Ellison 2012; Kreyling et al. 2018). Unreplicated designs, also known as gradient designs or response-surface designs, have only one replicate (observation) for each of many levels along one or more environmental gradients (Inouye 2001; Cottingham et al. 2005; Kreyling et al. 2018). Fan designs, which are used to examine simultaneously the effects of planting density and the number of different species planted in a polyculture, can have multiple replicates for each combination of density and local species diversity (Nelder 1962; Antonovics & Fowler 1985).

The strength of experimental designs with discrete treatment groups (i.e., factorial designs) is that they allow for adequate replication of individuals in each treatment group, estimation of expected values of responses to treatment *and* their variances, and the ability to test for differences among groups using ANOVA and either a priori contrasts or post hoc tests for multiple comparisons among means (e.g., Fisher 1925; Underwood 1997; Kohout & Norwood 1981). However, results from factorial experiments cannot determine the shape of the response to a continuous predictor variable (Figure 6.6). This is a particular concern if the response is nonlinear, as it is expected to be in dose-response curves or along environmental gradients, and if such responses are used in models to define courses of treatments for life-threatening diseases or models forecasting ecological responses to environmental changes (e.g., Kreyling et al. 2018).

Using detailed simulation studies, Kreyling et al. (2018) found that unreplicated gradient designs better identified the three underlying response surfaces they tested (saturating Michaelis-Menten, unimodal [hump-shaped], and linear responses) than did replicated (factorial) designs, even if the latter had many levels. Their simulations showed that this conclusion held at the maximum number of sampling locations, regardless of the amount of stochasticity and the shape of the underlying response surface. Although it is possible to add replication at each level of a gradient experiment (Cottingham et al. 2005), Kreyling et al. (2018) suggest that this would be an unnecessary use of scarce resources. It would be better to increase the number of locations along the gradient used in the experiment rather than add replicates at a given location. If space is not limiting, however, hexagonal fan designs give the best of both worlds—replication and dense sampling along one or two gradients. The statistical properties of fan designs are well established (Nelder 1962) and R code is available to optimize their layout under a range of situations (Rozins et al. 2020).

Gradient designs do not appear to be used in clinical trials. Among the many new adaptive designs (sensu Bowalekar 2011) for clinical trials that are being developed and implemented, at least one allows for changing doses during the trial to better identify ranges of dose responses and improve patient welfare (Ajmera et al. 2021). The current drive towards individualised or precision medicine might provide new opportunities for designing

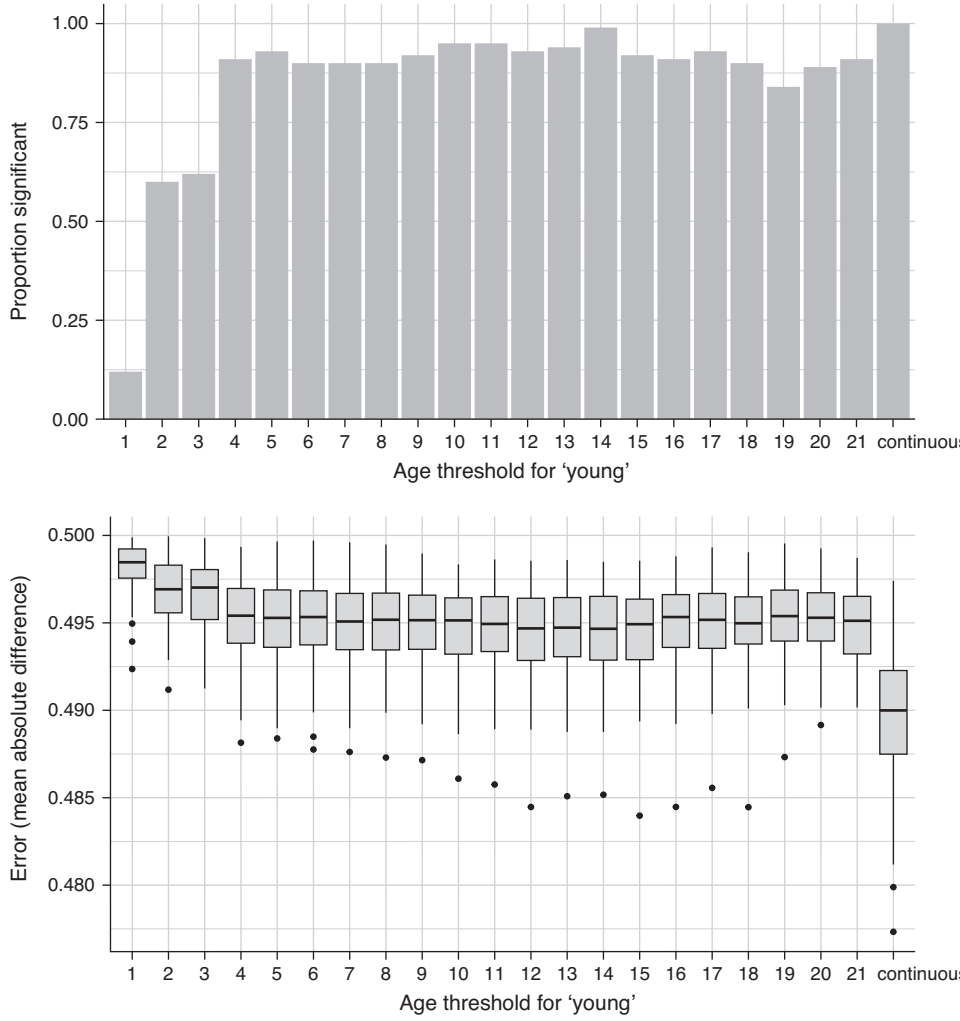


Figure 6.5 True positive rate (top) and mean absolute difference error (bottom) for different age cut-offs for "children" relative to adults, and for the model treating age as a continuous variable.

gradient-based clinical trials that could lead to new advances in therapeutics. However, any new design and especially its implementation will require careful planning, new methods of organisation and oversight of the experimental trials, detailed attention to novel ethical issues, and development of appropriate statistical methods to analyse the results (Ajmera et al. 2021). Simulations certainly will be invaluable for the latter.

6.4 Use Monte Carlo simulation when data are heteroscedastic

All data are not equal. One datum may be more reliable than another because, for example, measuring tools with different accuracies or precisions were used for different observations or because different sample sizes were used to calculate different estimates of the saved value. One way to handle such heterogeneity (i.e., the variability in accuracy or precision of individual observations) is to use weighted regression, in which each datum is assigned a weight

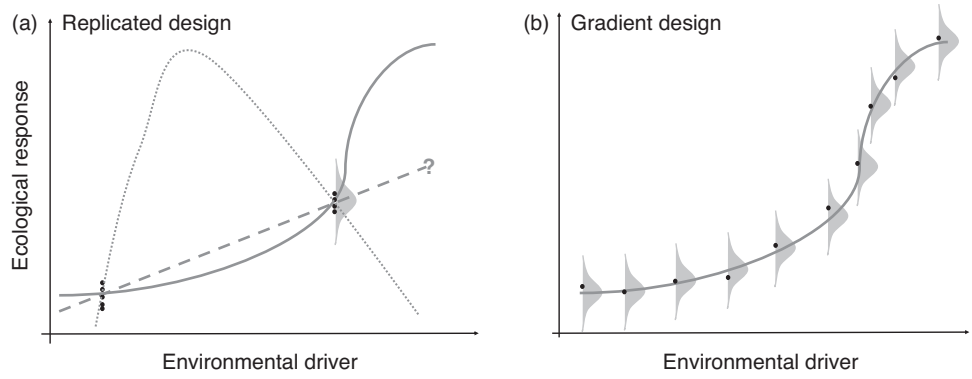


Figure 6.6 Visualisation of a key difference between (a) a replicated experiment with high-precision estimates of means and variances for responses to two factors and (b) an unreplicated experiment with very low precision in specific estimates but the ability to identify nonlinear responses to an underlying gradient. In both panels, the light grey bell-curves represent the probability distributions that could be used for random draws in a simulation study. Modified from Figure 1 of Kreyling (2018).

proportional to the precision (where precision here means the inverse of the variance of each observation).[12]

A more general, folkloric way of handling variable precision in the response is to draw a value for each datum at random from the distribution described by the mean and standard error of the datum, run the analysis, and repeat this procedure many times. Since random values are drawn, this method is called Monte Carlo simulation.[13] The resulting distribution of estimates can then be assumed to represent the heteroscedasticity of the underlying data.

6.4.1 A case study: An apparent decline in the airborne fraction of CO_2

Van Marle et al. (2022b) reported a somewhat surprising recent decrease in the "airborne fraction" of carbon dioxide: the proportion of CO_2-emissions that remain in the atmosphere (Fig. 6.7). They deduced that the "combined land-ocean sink has been able to grow at least as fast as anthropogenic emissions," which certainly would imply the existence and operation of a much appreciated, if as yet poorly understood, mechanism to counteract anthropogenically-caused climatic change. The key statistical approach used by van Marle et al. (2022b) was a trend analysis of a new dataset quantifying the airborne fraction of greenhouse gas emissions induced by land-use and land-cover changes (their Figure 3).[14]

Normally, such data would be analysed for a trend using regression (possibly with a term for temporal autocorrelation, which we here ignore for simplification). Since van Marle et al. (2022b) argued that their dataset was heteroscedastic and they provided a different error estimate for each data point, the folkloric approach is a Monte Carlo analysis.

As there is uncertainty in the estimated airborne fraction, van Marle et al. (2022b) added noise (proportional to the variance observed in the data) to create 10,000 simulations. Each simulation was then analysed separately; the proportion of simulations yielding a negative

[12] Not all statistical approaches allow such a weighting, however. Sometimes we want to have both a weight and an independent variance, otherwise inference may be affected (Chambers 1992).

[13] Monte Carlo simulation is not restricted to dealing with variance of data points. It is used even more frequently for understanding and quantifying error propagation in models with uncertain parameters (see also footnote 17 in Chapter 2, and §7.7.2 in Chapter 7).

[14] We omit here the complementary steps of smoothing, filtering, and bootstrapping.

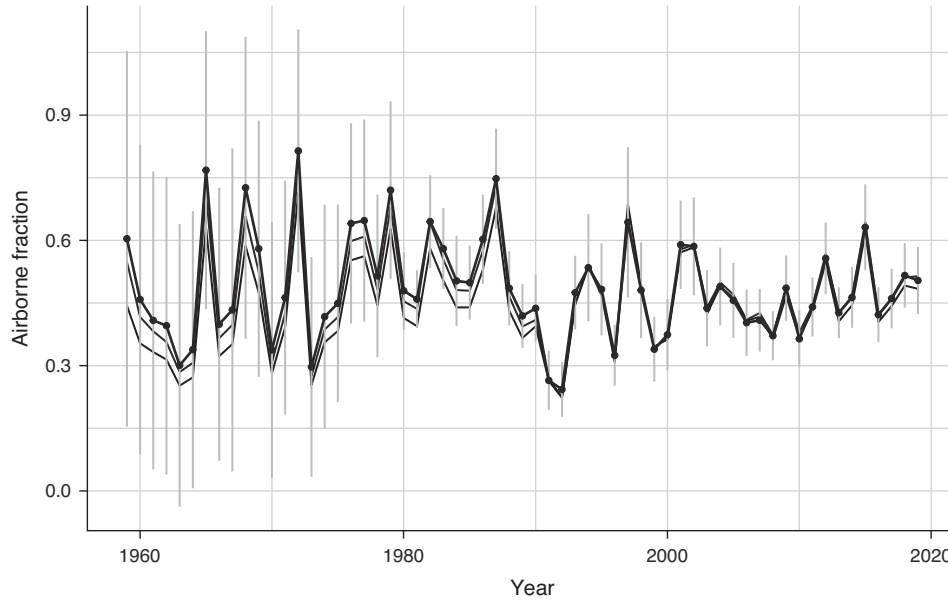


Figure 6.7 Airborne fraction of CO_2-emissions according to van Marle et al. (2022b). The data we analyse here are the black dots that are connected by the thick black line. The nearly identical thinner lines are from two previous datasets. One is from Houghton & Nassikas (2017) and the other is from the Global Carbon Budget Office (2023) (as archived by van Marle et al. 2022a). The Monte Carlo analysis was motivated by the error bars (95% confidence intervals) shown for each data point.

trend was the key output of their analysis. Note that this is different from a statistical test; indeed van Marle et al. (2022b) reported the proportion of negative estimates rather than overall statistical significance of the regression (the slope of which was not significantly different from zero: $p = 0.38$).

Bennedsen et al. (2023) criticised van Marle et al. (2022b) on the grounds that adding noise in the aforementioned way did not provide a valid reference test distribution, because it was added to a dataset with a trend. Instead, Bennedsen et al. (2023) asserted that it would be better (and thus, "correct") to use simulation to construct a test distribution from a dataset that does not have a trend and then compare the observed trend statistic with it.[15]

In this section, we examine both approaches because it is instructive to identify the different choices made by the two sets of authors. Our reanalysis of these data also illustrates clearly that simulation has its own significant challenges. Finally, we include a section clarifying the Bayesian element of such an analysis because van Marle et al. (2022b) confusingly interpreted results from their frequentist analysis in a Bayesian way.

6.4.2 Analysing the time series with a linear regression

We start by analysing the time series of the airborne fraction of CO_2 using a linear regression (Code Box 6.13). To avoid the normality assumption of ordinary least-squares regression,

[15] Bennedsen et al. (2023) also added a breakpoint model to incorporate into the model the fall of the Iron Curtain and the end of the Cold War in 1989 and to show that all trends in the airborne fraction were increasing. But that is not the point here. However, it was the point that led van Marle et al. to retract their 2022 paper (van Marle et al. 2023).

Code Box 6.13: Standard modelling of the trend in the airborne fraction of CO_2

```
# Load required libraries
 library("readxl") # Allows for direct import from the Excel files provided by
      van Marle et al. (2022a)
 library(Kendall) # Includes the MannKendall function

# Read in the data (local copy)
 dats <- read_excel("../Data/vanMarle2023/
      Marle_et_al_Nature_AirborneFraction_Datasheet.xlsx", sheet=6, skip=6)
 colnames(dats) <- c("year", "AFnew", "AFnew.error", "AFcorrnew", "AFcorrnew.
      error", "AFHN", "AFHN.error", "AFHNcorr", "AFHNcorr.error", "AFGCP",
      "AFGCP.error", "AFGCPcorr", "AFGCPcorr.error")

# Linear model
 summary(fm <- lm(AFnew ~ year, data=dats))

| Call:
| lm(formula = AFnew ~ year, data = dats)
|
| Residuals:
|       Min        1Q    Median        3Q       Max
| -0.242278 -0.071943 -0.009592  0.087686  0.312866
|
| Coefficients:
|               Estimate Std. Error t value Pr(>|t|)
| (Intercept)  2.0980367  1.8151366   1.156    0.252
| year        -0.0008096  0.0009126  -0.887    0.379
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.1255 on 59 degrees of freedom
| Multiple R-squared:  0.01317, Adjusted R-squared:  -0.00356
| F-statistic: 0.7872 on 1 and 59 DF,  p-value: 0.3786

# Mann-Kendall trend test to avoid the assumption of normality
 MannKendall(dats$AFnew)

| tau = -0.0295, 2-sided pvalue =0.74154
```

we also tested for trend using the Mann-Kendall trend test (Mann 1945; Kendall 1975), as did van Marle et al. (2022b).

The uncertainty around the trend estimate was as large as the estimate itself and neither the ordinary linear regression model nor the Mann-Kendall trend test revealed any significant trend. An autocorrelation plot of the residuals also showed no temporal autocorrelation.[16]

[16] Use the R command `acf(residuals(fm), type="partial")` to create this plot.

Code Box 6.14: Estimating the probability of a negative trend in the airborne fraction of CO_2

```
pnorm(0, mean=coef(fm)[2], sd=summary(fm)$coefficients[2,2])
```

```
| [1] 0.8125188
```

To quantify the probability of a negative trend, we computed the area under the distribution of the slope estimates for which the estimate was negative (Code Box 6.14):

The result of this calculation is that 81% of the area under the distribution of slope estimates includes negative estimates. Van Marle et al. (2022b) used this conclusion to infer a decline in the airborne fraction of CO_2.

6.4.3 Simulating error on the data

Van Marle et al. (2022b) also computed the uncertainty attached to each value of the airborne fraction (error bars on each point in Figure 6.7). They then investigated whether this error affected the proportion of negative trends computed in Code Box 6.14. To do so, they added noise to the observed data, where the noise was drawn from a normal distribution with standard deviation equal to the uncertainty estimate. In this way, they generated 10,000 simulated datasets, which were then analysed one by one (Code Box 6.15).

The simulations with added noise yielded a nearly identical proportion of negative slopes, 82%, as did the ordinary least-squares regression (Code Box 6.14). This suggests that the variable error on data points had no substantial effect in this analysis.[17]

6.4.4 Comparing an observed trend to a reference distribution

Bennedsen et al. (2023) argued that the negative simulations of noisy trends should not be summed, but rather that the observed data should be compared to no-trend simulations—the reference null hypothesis. We note that this approach does not actually address the conclusion of van Marle et al. (2022b). This is because Bennedsen et al. (2023) were testing for statistical significance, not quantifying the negatively-valued proportion. After reproducing Bennedsen et al.'s (2023) results, however, we will rework their approach to directly address van Marle et al.'s (2022b) conclusion.

GENERATE SIMULATED DATASETS WITHOUT A TREND

The first step is to generate simulated datasets that lack any trend. Although there is no statistically significant trend in the data (as shown in Code Box 6.13), there still is some apparent trend. We can remove this by using the residuals of the ordinary least-squares regression as the response variable.

ESTIMATION

The estimation procedure is identical to that used by van Marle et al. (2022b): add Monte Carlo noise to the data and compute the slope estimates (Code Box 6.16).

[17] The traditional way to handle the per-datum uncertainty is a weighted regression, with 1/variance as the weight: `summary(lm(AFnew ~ year, data=dats, weights = 1/AFnew.error^2))`. The result for the slope estimate, -0.00062 ± 0.00099, $p = 0.531$, is slightly shallower than that estimated using an unweighted regression. The negative part of the distribution from the weighted regression, 79.3%, also is slightly less than the 82.3% found using an unweighted regression (last line of Code Box 6.15).

Code Box 6.15: Monte Carlo analysis of the trend in the airborne fraction of CO_2

```
# Single simulation with Monte Carlo noise added
 dats$AFprime <- dats$AFnew + rnorm(nrow(dats), mean=0, sd=dats$AFnew.error)
 summary(lm(AFprime ~ year, data=dats))

| Call:
| lm(formula = AFprime ~ year, data = dats)
|
| Residuals:
|      Min       1Q   Median       3Q      Max
| -0.35747 -0.07988 -0.01758  0.08689  0.30778
|
| Coefficients:
|              Estimate Std. Error t value Pr(>|t|)
| (Intercept) 4.870e-01  2.203e+00   0.221    0.826
| year        3.250e-07  1.107e-03   0.000    1.000
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.1523 on 59 degrees of freedom
| Multiple R-squared:  1.46e-09,  Adjusted R-squared:  -0.01695
| F-statistic: 8.613e-08 on 1 and 59 DF,  p-value: 0.9998

# 10000 simulations; store estimate of the slope
 set.seed(123)
  slopes <- replicate(10000, {
    dats$AFprime <- dats$AFnew + rnorm(nrow(dats), mean=0, sd=dats$AFnew.error)
    summary(lm(AFprime ~ year, data=dats))$coefficients[2,1]
  })

# Proportion of negative slopes:
 mean(slopes< 0)
| [1] 0.8232
```

Unsurprisingly, when adding Gaussian noise to the no-trend data, the estimate of the slope (trend) equals 0.

Note that the simulation in Code Box 6.16 is the inverse of that in Code Box 6.15. Now, instead of comparing the simulated slopes to 0, we compare the simulated zeros to the real data. The more negative the observation, the fewer simulations are larger. So, we use the proportion of estimated slopes that are larger than the observation as the proportion of negative slopes (Code Box 6.17).

The result is exactly the same: 82% of trend-free slopes are larger than the observed; meaning only 18% would be in line with no trend but still yielding an estimate as low as the observed.

STOP AND THINK

At this point it is useful to visualise what we have done and achieved so far (Figure 6.8). We:

Code Box 6.16: Analysis of simulated data with no trend in the airborne fraction of CO_2

```
# The data here are the residuals of the linear regression
 dats$AFresids <- residuals(lm(AFprime ~ year, data=dats))
# Add noise, model, and summarise
 dats$AFresidsprime <- dats$AFresids + rnorm(nrow(dats), mean=0,
      sd=dats$AFnew.error)
 summary(lm(AFresidsprime ~ year, data=dats))

| Call:
| lm(formula = AFresidsprime ~ year, data = dats)
|
| Residuals:
|      Min       1Q   Median       3Q      Max
| -0.53829 -0.09530 -0.01145  0.09723  0.56105
|
| Coefficients:
|               Estimate Std. Error t value Pr(>|t|)
| (Intercept) -0.6813905  3.1674784  -0.215    0.830
| year         0.0003345  0.0015924   0.210    0.834
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.219 on 59 degrees of freedom
| Multiple R-squared:  0.0007474, Adjusted R-squared:  -0.01619
| F-statistic: 0.04413 on 1 and 59 DF,  p-value: 0.8343

# Simulate 10000 times:
 set.seed(123)
  resid.ps <- replicate(10000, {
   dats$AFresidsprime <- dats$AFresids + rnorm(nrow(dats), mean=0, sd=dats$AFnew.
       error)
   summary(lm(AFresidsprime ~ year, data=dats))$coefficients[2,1]
 })

 mean(resid.ps)

| [1] 5.64352e-06
```

1. modelled the density of the slope estimates using an ordinary least squares model (Code Box 6.13);
2. determined the density of the slope estimates from the initial simulations using Monte Carlo analysis (Code Box 6.15);
3. determined the density of the slope estimates from the simulations without a trend (Code Box 6.16).

Code Box 6.17: Estimating the probability of a negative trend in the airborne fraction of CO_2

```
mean(resid.slopes > -0.0008096)
```

```
| [1] 0.8249
```

Figure 6.8 encapsulates the idea that the analysis of van Marle et al. (2022b) and the suggestion of Bennedsen et al. (2023) give essentially equivalent results. Neither detects a significant trend and both suggest about 80% of the estimates to be negative.[18]

6.4.5 Bayesian analysis

The critique by Bennedsen et al. (2023) was motivated by the fact that van Marle et al. (2022b) found a negative trend in the airborne fraction in CO_2, whereas most previous studies had found a positive trend. The correct way to include such prior information is to use Bayes' Theorem (see also §3.3.1 in Chapter 3):

$$P(\theta|\text{data}) = P(\text{data}|\theta)P(\theta)/P(\text{data}). \tag{6.1}$$

The two alternative datasets presented by van Marle et al. (2022b) (thin lines in Figure 6.7) have a mean trend of $(0.001 + 0.0002)/2 = 0.0006$.[19] Eyeballing their standard deviation as SD = 0.0005 gives a pooled standard deviation of $\sqrt{\left(\frac{0.0005}{2}\right)^2 + \left(\frac{0.0005}{2}\right)^2} = 0.00035$.[20] Using Bayes' Theorem, we multiply the distribution of the slopes (i.e., the likelihood) from the ordinary least-squares analysis (dashed distribution in Figure 6.8) by this prior distribution to get the posterior distribution.

In detail, $P(\text{data}|\theta)$ is the estimated slope from the ordinary least-squares model analysis, and $P(\theta)$ is the prior, i.e., how much credence we give to an estimate. Their product, normalised by the integral over all values of x, is the posterior:

$$P(\theta|\text{data}) = \frac{\mathcal{N}(x;\mu_1,\sigma_1^2)\cdot\mathcal{N}(x;\mu_2,\sigma_2^2)}{\int_{-\infty}^{\infty}\mathcal{N}(u;\mu_1,\sigma_1^2)\cdot\mathcal{N}(u;\mu_2,\sigma_2^2)du}. \tag{6.2}$$

There often is no analytical solution to Equation 6.2 and this expression needs to be evaluated numerically. To find the mean (which in this case is also where the function has its maximum, as the resulting function is a normal distribution), we use optimisation (Code Box 6.18).[21] The negative proportion of this new distribution is the integral from $-\infty$ to 0.

[18] The interpretation of the grey curve is much more straightforward than the interpretation of the black one. Imagine that the slope estimate had been very negative. Then the black reference line would be farther to the left, and the area under the curve to its right would be larger. This area indicates how likely it is to get such a slope estimate without a trend in the data. The larger that area, the more likely it is that the estimate is not equal to zero, or, in other words, that more simulations also would have found a negative value. It is understandable that van Marle et al. (2022b) used their approach, which led to the more intuitively understandable grey line in Figure 6.8.

[19] Figure 3 in van Marle et al. (2022b) represents slopes per decade, whereas ours are per year. Thus, our values are 0.1 times those shown in van Marle et al. (2022b).

[20] Because this SD was based only on two studies, we increased it in our prior by giving it twice the variability (SD = 0.001) to allow for a substantially wider range of possible trend estimates. You can easily rework these analyses with other values for the SD to see their effects on the results.

[21] For a non-symmetric distribution, we can find the median by minimising the difference between the integral over this function and 0.5.

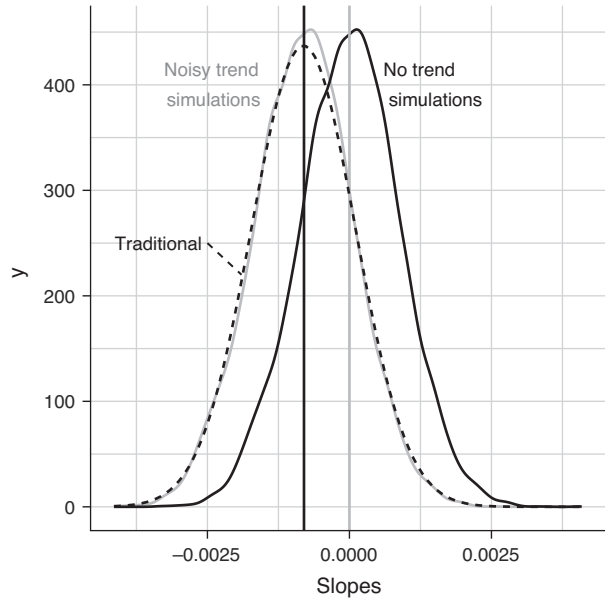


Figure 6.8 Distribution of slope estimates for the trend in the airborne fraction of CO_2 computed in three different ways. The grey curve is equivalent to the estimates obtained from the analysis of van Marle et al. (2022b), simulating uncertainty in the airborne fraction and the resulting uncertainty of slope estimates. The vertical grey reference line identifies a slope equal to zero. The dashed distribution is from an ordinary least-squares analysis without simulation. The black curve is from the simulation of no-trend, as suggested by Bennedsen et al. (2023); the black vertical reference line is the observed slope (−0.0008). Note the near-perfect symmetry of these two simulations: the area to the left of the grey vertical line under the grey curve is the same as the area to the right of the black vertical line under the black curve.

Thus, by using the prior to formally take into account the results of two previous studies, the proportion of the simulations exhibiting a negative trend is around 60%, substantially lower than the 82% reported by van Marle et al. (2022b).

Alternatively, and in the spirit of simulations, we can look at this last step slightly differently. The prior distribution quantifies how much we believe[22] an estimate. For example, we might consider an estimate of −50 or 2 to be completely untrustworthy, whereas a value of 0.001 may well seem plausible. In this way, we can now give each of the 10,000 simulations a plausibility weight and compute their weighted average (Code Box 6.19). Figure 6.9 illustrates how using plausibility weights in the prior moves the centre of the distribution of the estimates much closer to zero. Thus, our estimate for the posterior mean from the 10,000 simulations is similar to the previous solution and increases in precision (relative to the prior) with even more simulations.[23]

CONCLUSION

We conclude that van Marle et al. (2022b) analysed their data in a way that seems appropriate. Their conclusions are unaffected by Bennedsen et al.'s (2023) critique and reanalysis of the data. Van Marle et al.'s (2022b) interpretation of the "posterior" of their otherwise frequentist

[22] Yes, this is the correct Bayesian term; not to be confused with other meanings of the word.

[23] Note that the sum of weights is very large, and we need to make that value represent our 10,000 values. This is why we divide by `10000/sum(weights)` = 0.0058. When plotting the prior based on the density function, we also need to multiply this by the average bin area, which is the scaled weights times 10,000 replications, or 1.946.

Code Box 6.18: Bayesian analysis of the trend in airborne fraction of CO_2

```
# Define the prior, the likelihood, and the normalisation constant
 numerator <- function(x) dnorm(x, mean = -0.0008, sd = 0.0009) * dnorm(x, mean
     =0.0006, sd=0.001)
 denominator <- integrate(numerator, -Inf, Inf)

# Compute the posterior and determine its mean (= its maximum)
 post <- function(x) numerator(x)/denominator$value
 optimize(post, interval=c(-.1, .1), maximum=T)

| $maximum
| [1] -0.0001746192

# Calculate the proportion of the distribution that is negative:
 integrate(post, -Inf, 0)

| 0.6023086 with absolute error < 2e-08
```

Code Box 6.19: Bayesian analysis with plausibility weights of the trend in airborne fraction of CO_2

```
# Load required libraries
 library(ggplot2)

# Simulate weights
 weights <- dnorm(slopes, mean=0.0006, sd=0.001)
 sum(weights * slopes) / sum(weights)

| [1] -0.0001946087

# Create the plot
 toplot <- data.frame(slopes, weights=dnorm(slopes, mean=0.0006, sd=0.001),
     ones=1)
 ggplot(toplot, aes(slopes)) +
    geom_histogram(fill="grey30", alpha = 0.3) +  # slopes
    geom_histogram(aes(slopes, weight=weights*0.0058), fill="grey30", alpha
        = 0.5) + # weighted slopes
    geom_function(fun = \(x) 1.946*dnorm(x, mean=0.0006, sd = .001), colour
         = "black", linetype=1) +
    geom_vline(xintercept=0, linetype=1, colour="black", size=.8) +
    theme_minimal()
```

analysis is, however, somewhat surprising. Our full Bayesian analysis of the data with an informed prior suggests that van Marle et al. (2022b) overstate their results by at least 25%.

The most severe blow to van Marle et al.'s (2022b) conclusions, however, is the second point of criticism by Bennedsen et al. (2023), which we do not model here. Using a breakpoint model that corrects for the observed, albeit brief, drop in CO_2-emissions after

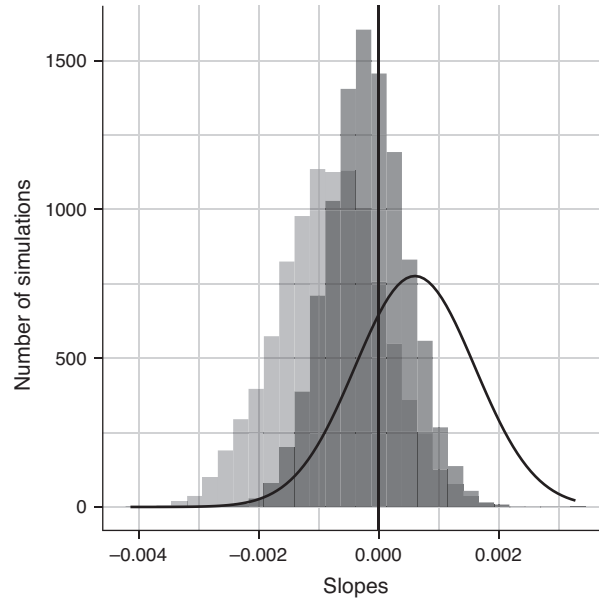


Figure 6.9 The original slope estimates (light grey) are shifted towards 0 (vertical black line) as they are weighted by our prior belief (black curve; density rescaled to 10,000 simulations), yielding the posterior estimates (darker grey).

the fall of the Iron Curtain in 1989, Bennedsen et al. (2023) showed that there is no cause for optimism. In fact, the airborne fractions went up before and after the short decline in CO_2-emissions co-incident with the end of the Cold War.

6.4.6 Should we use Monte Carlo simulation for heteroscedastic data?

Monte Carlo simulation (sensu Sawilowsky 2003) is an important technique in the toolbox of statistical analyst. In this case study, both van Marle et al. (2022b) and Bennedsen et al. (2023) appropriately used Monte Carlo simulations to represent the heteroscedasticity in the observations and to make their (equivalent) points. Monte Carlo simulation works well if we can assume that each datum and its variance are independent. However, when data points are non-independent, Monte Carlo samples should be drawn jointly from a multivariate distribution representing this dependence. For example, Monte Carlo simulation is inefficient when used for process models with uncertain parameters (see §7.7.2 in Chapter 7 for more discussion of this topic).

6.5 Time series should not be detrended by default

Time series are common in many different fields of science (e.g., meteorology, economics, botany, medicine, geology, oceanography), yet they are challenging to analyse using basic statistical tools that rely on independent observations. By independent, we mean that every observation can be considered as a new draw from some distribution, e.g., $y \sim \mathcal{N}(\mu, \sigma)$. When such independence is violated, standard maximum likelihood estimators (including least squares) are invalid because the product of the joint probability densities for any two (or more) observations is no longer equal to simply the product of the individual densities.[24]

[24] That is, $P(A, B) = P(A) \cdot P(B)$ if and only if $P(A \cap B) = 0$.

Individual observations in a time series are rarely independent. Rather, they are correlated with one another, as observations made more closely in time are more likely to be similar than those made further apart.[25] A simple example of an autocorrelated time series was given in Equation 2.1 in Chapter 2; we repeat it here for convenience:

$$Y_t = \beta_0 + \beta_1 Y_{t-1} + \varepsilon_t. \tag{6.3}$$

How should we model and analyse such autocorrelated data?

6.5.1 Detrending and differencing

Many introductory statistics texts recommend that the first step in modelling and analysing time-series data is to detrend them. Detrending means removing (i.e., subtracting) any long-term trend from all data points, so that, ceteris paribus, there is no correlation between any pair of detrended observations Y'_t and Y'_{t+k} (in the case of Equation 6.3, $k = 1$). For example, we assume that the trend can be modelled as

$$\mu_t = at + b, \tag{6.4}$$

where a and b are parameters that are estimated from the data. By subtracting this trend from the data ($Y'_t = Y_t - \hat{\mu}_t$), we are left with a set of i.i.d residuals $Y' \sim \mathcal{N}(0, \sigma^2)$ that can be analysed using, say, ordinary least-squares regression. Code Box 6.20 and Figure 6.10 illustrate this ideal outcome of detrending. An alternative and similarly recommended approach is to compute the lag-1 differences (i.e., $\Delta Y = Y_t - Y_{t-1}$), and then analyse the differences ΔY rather than Y itself (Code Box 6.20; Figure 6.10).

These recommendations are unfortunate because the ceteris paribus assumption is almost always violated. For example, if Y is the price of a commodity and we analyse how Y increases over time, then analysing the differences is simply plotting inflation: it tells us that (and perhaps when) commodities become more expensive through time, but not how expensive they actually are. Which question would we rather answer? Similarly, the effect of a predictor X also can change over time. For example, coffee prices had virtually no effect on inflation measures 100 years ago because the price of coffee was not included in the consumer price index. Now it is, and rising coffee prices contribute to inflation. A time-series model should have enough flexibility to fit different effects for different points in time; such models are called (temporally) variable coefficient models.

Second, there is no single best practice for detrending, and different methods return different and often incompatible results. Some use linear regression and then analyse the residuals ("linear detrending"), whereas others use splines or locally estimated means using GAMs or empirical mode decomposition (Huang et al. 1998). Some software packages even return a large set of detrended time series and let the user decide which one to use. This is the default in the `R` package `dplR` (Bunn 2008), which is the standard package for analysing tree-ring data and reconstructing past climates from them.[26] As a tree matures, tree rings become

[25] Similar problems occur with spatial and phylogenetic data (Shumway & Stoffer 2006; Cressie & Wikle 2011; Ives 2022, and see also §2.4 in Chapter 2). For spatial data, observations that are separated by only short distances tend to be more similar than observations that are separated by larger distances. Phylogenetic autocorrelation refers to the idea that species (other taxa, or other things whose relationships-by-descent can be specified) that are more closely related to one another should share more characteristics than species that are more distantly related to one another. Note that evolutionary relationships are also temporal ones—closely-related species diverged from a common ancestor more recently than distantly-related ones.

[26] The description of the detrend `series()` function in `dplR` refers to detrending as a "dark art" and goes on to say that "automating detrending and not evaluating each series individually is folly. Remember, frustration over

Code Box 6.20: A toy example of detrending or differencing a simulated time series

```
# Set the time frame
 time <- seq(1901,2000,1) #Create the time frame

# Define the linear trend
 a <- 0
 b <- 2
 mu <- a + b*t

# Simulate the time series with normal error
 set.seed(12345)
 y <- mu + rnorm(length(time),0,10)

# Detrend the time series
 y.detrended <- y - mu

# Difference the time series
 y.diff <- diff(y, lag=1)
```

ever smaller, simply because the diameter of a tree increases and a constant growth rate leads to the new growth being distributed over a larger circumference. Detrending removes such age-related growth from the analysis—and anything else that changed over the lifetime of that tree (for alternative approaches to dendrochronological analysis that do not require detrending, see Ellison et al. 2014; Schofield et al. 2016).

Third, and perhaps most importantly, detrending removes trends. So, it must not be used if the trend itself is of interest. Detrending a time series of steady increases in prices will yield a flat line; that does not mean there was no change in price! On the other hand, if the focus of the analysis is only seasonal or short-term patterns, or the reference is what economists call a "buy-and-hold" strategy that would follow the trend, then removing the trend may be useful.

There are many statistical tools that can handle and estimate the degree of non-independence in time series (e.g., Shumway & Stoffer 2006; Cressie & Wikle 2011). These tools allow us to directly analyse and model the actual dataset and ask more detailed questions not only about trends but also about seasonal cycles or short-term patterns.

6.5.2 A case study: Abrupt environmental change in a lake

The clarity of the water in a lake is an indicator of how healthy it is. Clear lakes are oligotrophic—relatively low in dissolved nutrients, high in dissolved oxygen, and usually with large populations of healthy fish—whereas murky lakes are eutrophic—relatively high in dissolved nutrients, low in dissolved oxygen, and with fewer fish in them. The switch from an oligotrophic state to a eutrophic state can occur very abruptly, and detecting the change-point between these two states has been the focus of a great deal of research (e.g., Carpenter 2003; Contamin & Ellison 2009).

A standard way to assess water clarity is to measure the depth at which a 30-cm diameter, black-and-white disk (a Secchi disk; Cialdi & Secchi 1865) that is lowered into the water

detrending is the number one cause of dendros [i.e., dendrochronologists, or tree-ring scientists] going to live as hermits in the tall-grass prairie, where there are no trees to worry about."

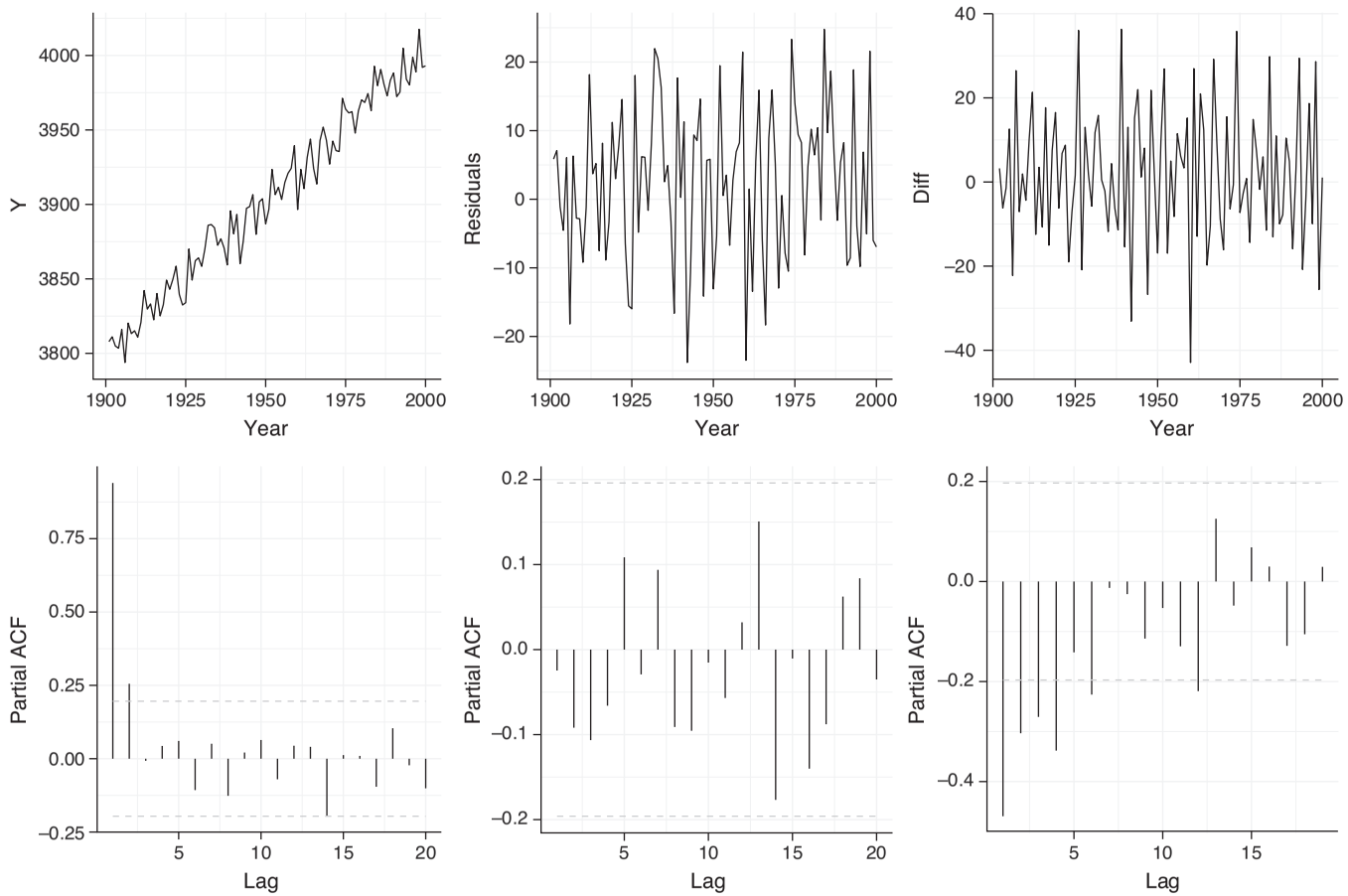


Figure 6.10 The outcome of detrending or differencing a time series. The top row shows (left-to-right) the raw time series simulated with the code in Code Box 6.20 along with the simulated trend line; the residuals (observed − actual trend); and the lag-1 differences. The plots in the lower row are the partial autocorrelation plots corresponding to the time-series, residual, and first difference plots. There is significant autocorrelation for lag $k = 1$ for the time series. This autocorrelation disappears in the residuals, but not in the first differences.

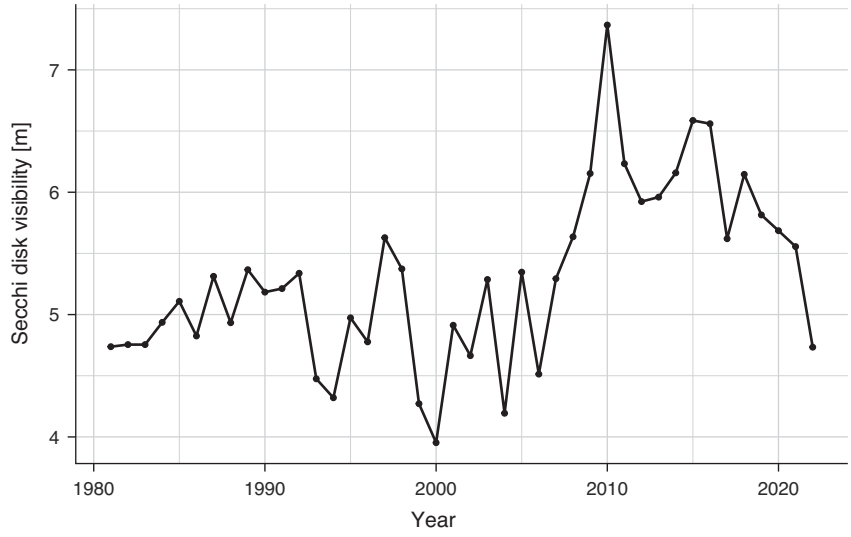


Figure 6.11 Depth at which a Secchi disk disappears from surface view in a real lake. These data are annual means of regular measurements that have been taken for > 40 years at 4–6-week intervals during ice-free conditions at Trout Lake, Wisconsin, USA (Magnuson et al. 2022).

from a boat becomes undetectable from the surface. This depth is called the Secchi depth. The time series of Secchi depth measurements can be analysed to identify when the shift from oligotrophic to eutrophic (or vice-versa) conditions occurs. An example of such data are shown in Figure 6.11.

We ask whether or not detrending data such as these improves the detection of any underlying shift (anomaly) in the Secchi depth. In the simplest case, such a change would be manifested as a significant change in the mean Secchi depth at a particular point in time. Such changes are detectable using change-point methods (e.g., Dakos et al. 2012).

SIMULATION

We simulated a time series of normalised Secchi depth values (i.e., range = [0,1]), to which we added a change after half of its length. We then subtracted the jump from the first half of the time series and added it to its second half (Code Box 6.21, Figure 6.12).

We analysed this initial simulation using standard detrending methods (as in the R package `dplR`) and change-point detection algorithms (as implemented by Killick et al. 2022 in the `changepoint` package).

In this initial analysis, the change-point function did not detect a change-point in the raw data. When the algorithms were applied to the detrended datasets, only the age-dependent spline detected a change point, and it did so at the correct position (26 = year 1996). The other five methods applied to the detrended data did not detect a change point.

We complicated this simulation in two ways. First, as usual, we replicated the simulation (Code Box 6.23). Then, we ran the same simulations 100 times for each of 20 different jump sizes between 0 and 1 (Code Box 6.24). All simulations were then examined for change points in the same way as the initial simulation (Figure 6.13).

In the first case, in which we always used the same magnitude of the change (i.e., the `jump`), the change-point algorithm detected a change in the raw time series for all 100 simulations (Code Box 6.23). Among the six algorithms applied to the detrended time-series,

Code Box 6.21: Simulating a time series with non-linear trend and change in mean

```
# Set the time span
 time <- 1970:2021

# Write a short function to normalise time series to [0, 1]
 rangit <- function(x) (x - min(x)) / (max(x) - min(x))

# Simulation
 set.seed(1234)
 jump <- .5 # jump size
 lightextinction <- rangit(sapply((time-1970)/100, \(x) -1.2*x^5 + .5*x^4 -3*x^3
     + 2.75*x^2 -1*x + .21) + rnorm(length(time), 0, .1) )
 lightextinction <- lightextinction + rep(c(-jump, jump), each=26) + jump
```

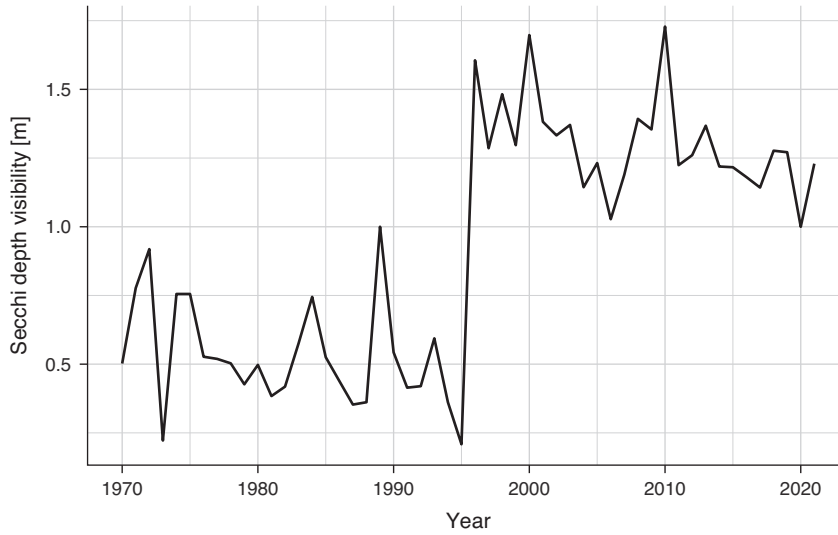


Figure 6.12 Simulated Secchi depth visibility over time, with a clear imposed signal after 26 years (jump size 0.5 m).

only the age-dependent spline performed similarly well. In all cases when a change-point was detected, it was detected at the correct location (position 26).

When we applied the same method to jumps of different sizes, the age-dependent spline detrending was able to detect correctly the change-point for smaller jump-sizes than the algorithm applied to the raw data. All other detrending algorithms performed more poorly (Figure 6.13).

6.5.3 Should we detrend data before analysing or modelling them?

This short example of a real problem—change-point detection—and an analysis of its corresponding simulation suggest that detrending a time series makes sense when the question of interest does not involve the trend itself. This type of time series, however, is atypical because it has a trend but not any predictable daily, seasonal, or annual cycles. Indeed, because Secchi depths in temperate lakes usually are not measured all year (because lakes freeze in the

Code Box 6.22: Detrending the simulated time series and looking for a change-point

```
# Load required libraries
 library(changepoint)
 library(dplR)

# Convert simulated data from Code Box 6.21 to a time-series object, detrend,
    and examine for change points.
 ts.raw <- lightextinction # Convert the simulated data (a vector) to a time
     -series object
 ts.det <- detrend.series(ts.raw)[,-4] # Remove AR, as it causes NAs

# Find change-points for both raw and detrended time series
 chpts <- apply(cbind(ts.raw, ts.det), 2, cpt.mean)
 sapply(chpts, function(x) x@cpts[x@cpts != 52])

|  $ts.raw
| numeric(0)
|
| $Spline
| numeric(0)
|
| $ModNegExp
| numeric(0)
|
| $Mean
| numeric(0)
|
| $Friedman
| numeric(0)
|
| $ModHugershoff
| numeric(0)
|
| $AgeDepSpline
| [1] 26
```

winter) or are measured at relatively long intervals (here, monthly or more), the data are adequate only to assess the trend and potential change-points.[27]

Analysis and inference of time series with both trends and cycles can give very different results about whether and how the data are detrended. As an example, consider the data on the number of passengers flying every month within the USA from 2002 to

[27] Statistical education of many ecologists emphasises the need to ensure that data points are independent of one another. Hence, much effort is made to ensure that data collected through time are *uncorrelated*! Such data are amenable to testing for trends, but not for periodicity. It also turns out that most change-point detection algorithms require knowledge of the autocorrelation function (Dakos et al. 2012), so ecological data such as those shown in Figures 6.11 and 6.12 are inadequate for reliable detection of change-points (Bestelmeyer et al. 2011).

Code Box 6.23: Repeated simulations and change-point detection; 100 simulations for one value for `jump`

```
# Repeated simulations of one jump
 set.seed(123)
 jump <- 1 # % jump
 chpts.simu <- replicate(100, { # Simulate 100 times
   lightextinction <- rangit(sapply((time-1970)/100, \(x) -1.2*x^5 + .5*x^4
        -3*x^3 + 2.75*x^2 -1*x + .21) + rnorm(length(time), 0, .1) )
   lightextinction <- lightextinction + rep(c(-jump, jump), each=26) + jump
   ts.raw <- lightextinction
   ts.det <- detrend.series(ts.raw)[,-4] # Remove AR, as it causes NAs
   chpts <- apply(cbind(ts.raw, ts.det), 2, cpt.mean) # Change-point detection
   sapply(chpts, function(x) x@cpts[x@cpts != 52])
 })

# Compute number of times a changepoint was detected:
 apply(chpts.simu, 1, \(x) length(unlist(x)))

|        ts.raw        Spline     ModNegExp          Mean      Friedman
|           100             0            80            80             0
| ModHugershoff  AgeDepSpline
|            80           100
```

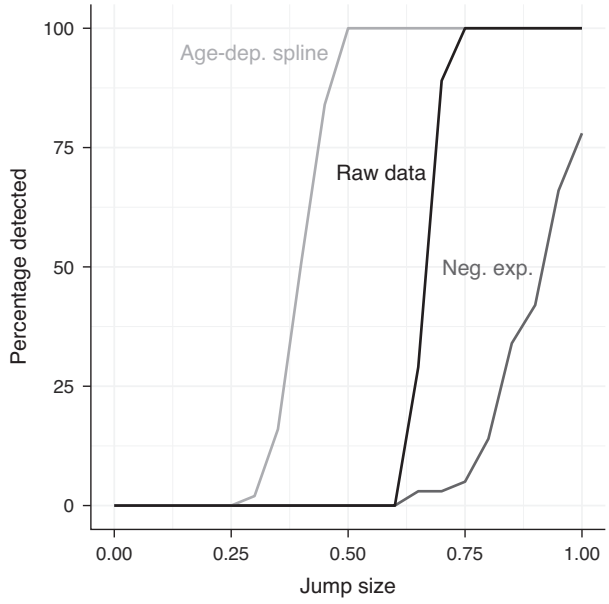


Figure 6.13 Improved detection of simulated change-point after detrending. As the simulated size of the jump in the change-point increases (x-axis), the probability of detection (y-axis) against the background noise also increases. The change-point detection algorithm was able to detect a change in data that had been detrended using an age-dependent spline better than when applied to the raw time series or any of the other detrended time series.

Code Box 6.24: Repeated simulations and change-point detection; 20 values for `jump`

```
jumps <- seq(0, 1, by=0.05)
manyjumps <- matrix(NA, ncol=length(jumps), nrow=7)
set.seed(123)
for (i in seq_along(jumps)){
  jump <- jumps[i]
  chpts.simu <- replicate(100, {
    lightextinction <- rangit(sapply((time-1970)/100, \(x) -1.2*x^5 + .5*x^4
        -3*x^3 + 2.75*x^2 -1*x + .21) + rnorm(length(time), 0, .1) )
    lightextinction <- lightextinction + rep(c(-jump, jump), each=26) + jump
    ts.raw <- lightextinction
    ts.det <- detrend.series(ts.raw)[,-4]
    chpts <- apply(cbind(ts.raw, ts.det), 2, cpt.mean)
    sapply(chpts, function(x) x@cpts[x@cpts != 52])
  })
  manyjumps[,i] <- apply(chpts.simu, 1, \(x) length(unlist(x)))
  cat(i, " ")
}

dats <- data.frame(jumps, t(manyjumps))
colnames(dats) <- c("size", rownames(chpts.simu))
```

2022 (Figure 6.14). The data show clear seasonal cycles, a weak increasing trend, and a pronounced drop in 2020–2021 associated with the COVID-19 pandemic (for another COVID-19 example, albeit a politically sensitive one, see Wood 2022).

A quick way to explore these data is with the `stl` function in R (Code Box 6.25). This function decomposes a time series into components that represent its trend and seasonal pattern, along with the residuals. The `stl` function uses local regression (loess), so there is substantial flexibility in identifying linear or nonlinear trends.

Figure 6.15 shows the result of decomposing the air-passenger time series assuming a linear trend. The grey range bars on the right side of each plot illustrate the explanatory power of each component if it were plotted on the same scale (y-axis) as the data. Here, the data range from 0 to 60 (millions of passengers), whereas the extent of the trend line ranges from 52 to 57. Thus, we would conclude that the linear trend has relatively weak explanatory power (the range bar is very large relative to the range bar for the raw data; if the trend line were plotted on the same scale as the data, it would be nearly flat). In contrast, the pronounced seasonal pattern (after detrending) is much more informative than the trend (data range of the seasonal cycles is $[-8, 6]$, nearly three times that of the trend; the range bar is intermediate in size between the data's and the trend's). The residuals (what are left after detrending and removing the seasonal pattern) contribute about the same information as the data (the ranges and range bars are about the same size). The residuals also illustrate that the noise in the data is much higher in the COVID-19 years, but the increasing variance in the residuals well before the pandemic raises a red flag about a non-linear trend.

In contrast, Figure 6.16 shows the result of decomposing the air-passenger time series with a loess trend. Here, the trend is not monotonic but is more reflective of the data, which during $\approx$ 2005–2015 was relatively flat. The loess fit captures the dramatic fall and subsequent rise in passengers associated with COVID-19 lock-downs and releases; the range bar suggests that this overall trend indeed is capturing as much information as the data. The

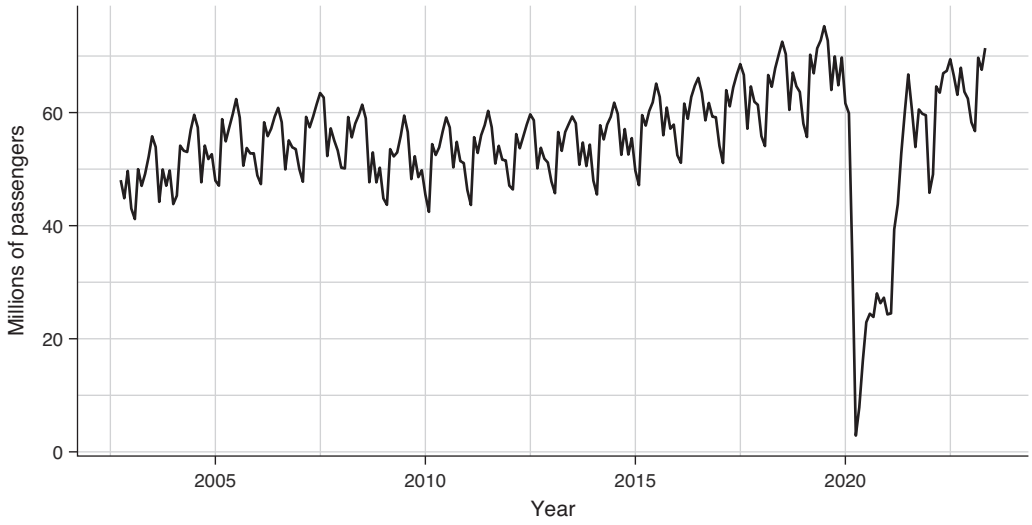


Figure 6.14 Total number of passengers on airlines flying monthly within the USA. Data from the US Department of Transportation's Bureau of Transportation Statistics (https://www.transtats.bts.gov/Data_Elements.aspx?Data=1).

Code Box 6.25: Decomposition of the air passenger data shown in Figure 6.14

```
# Read in the data (local copy)
 air <- read.csv("../Data/AirPassengers/BTSairpassengers.csv", header=TRUE, skip
     =1)

# Create the time series from the data frame
# Domestic US passengers only (data in column 3), scale to millions
 air.ts <- ts(air[,3]/1000000, start=c(2002,10), frequency=12)

# Decompose the data with linear trend (default) and plot it
 plot(stl(air.ts, s.window="per", t.window=1000))

# Decompose with robust smooth for trend and plot it
 plot(stl(air.ts, s.window="per", robust=TRUE))
```

residuals have about the same information, too, and now are much more in line with the actual data than they were with the linear decomposition. Finally, there is little additional information in the seasonal cycles.

Although we could argue that the detrending in Figure 6.16 is helpful, it doesn't really illustrate a trend in a particular direction. Rather, it shows the ups and downs of air travel that actually occurred. This information is lacking in the residuals of Figure 6.15, which, if examined without information about the trend, might suggest that the pandemic caused greater variance, but not a precipitous decline in the number of air passengers in 2020 and 2021.

We conclude that the rule-of-thumb to always detrend time-series data before modelling and analysing them is an oversimplification (see also Peters et al. 2015; McPartland et al. 2020). Yes, there are good reasons to detrend some time series before analysing and modelling them, but in general, we should take a more nuanced approach and examine

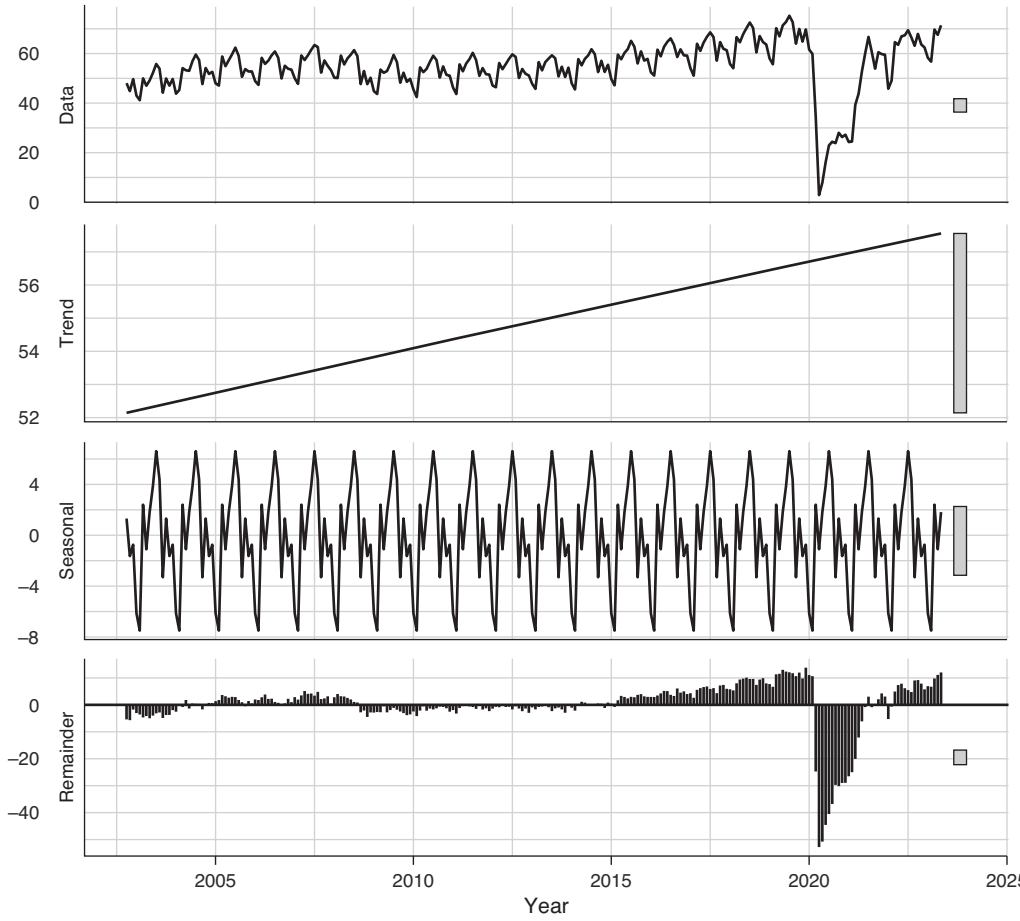


Figure 6.15 Decomposition of the air passenger data (in millions of passengers per month) with a linear trend.

the data thoroughly before deciding whether to detrend them or not. Simply plotting the results of decomposing a time series into its various components can point us in the best direction.[28]

6.6 Machine learning and Big Data do not obviate rules-of-thumb

Machine learning and deep learning have been taken up rapidly by scientists over the last two decades. This rapid uptake has occurred for at least three reasons. First, the availability of massive datasets ("Big Data") that are not easily analysed or modelled with classical statistical methods has led to widespread adoption of machine-learning models that have been promoted as the next scientific revolution (Anderson 2008; but see, e.g., Succi & Coveney 2019 and Hansen & Quinon 2023 for salient critiques). Second, the software has improved rapidly. Default settings of many machine-learning approaches are reasonable and provide a number of turn-key (albeit opaque-box) solutions for individuals interested in using them. Finally, our interest in prediction and forecasting has grown rapidly and is expected more and more by policy analysts and decision makers. Machine-learning models

[28] We note that the R function `stl` provides more flexibility than the `decompose` function.

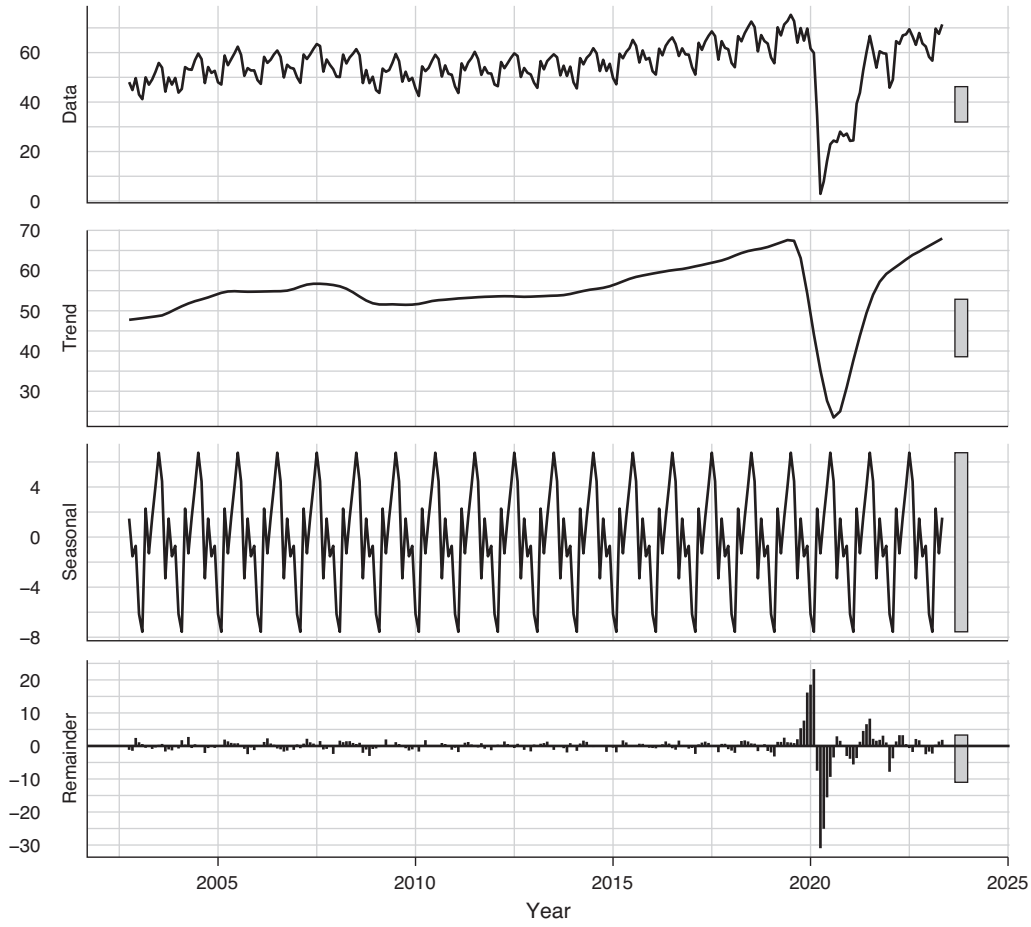


Figure 6.16 Decomposition of the air passenger data (in millions of passengers per month) with a robust loess trend.

easily provide these forecasts and add to them an au courant and seemingly unbiased patina.

Many popular and technical publications suggest that machine-learning methods are intrinsically superior to classical, model-based parametric statistical approaches. We illustrated in §6.2 that this contemporary folkloric assertion is untrue in the simple case of choosing between correlated predictors in an ordinary linear regression or a random forest. We share with recent commentators (e.g., Succi & Coveney 2019; Hansen & Quinon 2023) the opinion that Big Data and artificial intelligence (with its machine- and deep-learning offspring) do not spell the end of theory. But we do not yet want to toss out the AI babies with the parametric bathwater. In the remainder of this chapter, we discuss a few more complex cases than a comparison of regression and random forests for which further work and associated simulation would prove valuable in testing the relative merits of classical statistical methods and machine-learning models.

6.6.1 Machine learning, distributional assumptions, and non-independence

The large number of publications featuring statements such as "machine learning does not make assumptions" or "machine learning is robust [to this or that assumption]" might

imply that machine learning and deep learning are part of an anything goes, lawless area of statistics. They are not. Just because we do not need parametric assumptions to use machine-learning models does not mean they don't have any such assumptions.

All machine-learning approaches optimise a default or user-specified loss function. Loss functions are not free of assumptions. Indeed, and without any exception of which we are aware, these loss functions assume independence of observations. As a result, there are virtually no hierarchical machine-learning loss functions. For example, the popular random forest algorithm averages over hundreds of classification and regression trees (CARTs). Each of those trees splits the data by predictors using some measure of split quality, i.e., the loss function of the tree. For regression trees, the loss function is typically the root-mean-square error (RMSE), which essentially is equivalent to assuming a Gaussian log-likelihood. For classification trees, various other loss functions are used. The common default is the Gini-impurity (= the Gini-Simpson index; Farris 2010), which multiplies misclassification rates. Others use cross-entropy as a loss function, which for binary data is just a fancy way of using the negative Bernoulli log-likelihood as a minimisation target.

In fact, many loss functions in machine learning are rehashed (negative) log-likelihoods or slightly solidified (less sensitive to extremes) versions thereof.[29] That should not come as a surprise, since probability distributions serve exactly that aim in statistics: trying to find a simple description of the variation in observed data. Minimising the surprise in data is thus a common goal of both classical statistics and machine learning.

Both probability distributions and machine-learning loss functions evaluate each observation and then combine these assessments into a single value. In classical statistics, this value is the probability (density) of observing Y_i under an assumed distribution for the expected value; for example, $P(Y_i|\mu = \hat{Y}_i, \sigma) = \mathcal{N}(Y_i|\mu = a, \sigma = b)$. In machine learning, the single value may be the distance between observation and prediction measured as Euclidian distance; for example, the RMSE implies $\mathrm{d}(Y_i|\hat{Y}_i) = \sqrt{(Y_i - \hat{Y}_i)^2}$.

Whether we combine many probability densities (in classical statistics) or distances (in machine learning) into a single value, we assume independence: $P((Y_i, Y_j)|\mu = (\hat{Y}_i, \hat{Y}_j), \sigma) = P(Y_i|\mu = \hat{Y}_i, \sigma) \cdot P(Y_j|\mu = \hat{Y}_j, \sigma)$.[30] We do essentially the same thing when we combine distances into a single measure in machine-learning models. For example, RMSE $= \sqrt{\frac{1}{N}(\mathrm{d}(Y_i|\hat{Y}_i) + \mathrm{d}(Y_j|\hat{Y}_j) + \ldots + \mathrm{d}(Y_N|\hat{Y}_N))}$.[31] If the observations were not independent, then they would not contribute the same amount of information to the total RMSE score, and this approach could be biased when the sample size is correlated with a predictor variable (Boyd et al. 2023 call this "big data" becoming "small data" under non-representative sampling).

In sum, machine learning uses loss functions that are similar to (negative) log-likelihoods, although these loss functions may be somewhat more robust to deviations from or violations of specific distributional assumptions (Murphy 2012). Additionally, loss functions by default assume independence of observations, unless the loss functions are specifically tailored to alternative assumptions. Thus, claims that machine learning frees us from the shackles of distributional assumptions and non-independence are worse than folklore—they are downright wrong. If you remain unconvinced, code up and run some

[29] A loss function is minimised, whereas a log-likelihood is maximised. Hence, the negative log-likelihood is the loss function for GLMs.

[30] For the log-likelihood, the log of these probability densities is summed, implying that the probability densities can be multiplied, which is only correct when the observations are (conditionally) independent.

[31] This RMSE summation is equivalent to the sum in the log-likelihood and similarly implies independence of observations.

simulations comparing the susceptibility of machine-learning to non-independence (e.g., temporal autocorrelation) and notice that many people have tried to put non-independence into machine learning for this reason (e.g., Goulon-Sigwalt-Abram et al. 2005; Seok et al. 2011; Zhang et al. 2011; Luts et al. 2012; Sela & Simonoff 2012; Chen et al. 2014; Cheng et al. 2014; Hajjem et al. 2014; Kiaee et al. 2016; Fokkema et al. 2018; Alaa & van der Schaar 2020; Joseph 2020).

6.6.2 Machine-learning, non-linear functions, and statistical interactions

Machine learning is largely unsuitable for hypothesis testing, as it does not churn out *p*-values (Hastie et al. 2009; Murphy 2012; Pichler & Hartig 2023). If we are not interested in *p*-values or correctly estimated confidence intervals for a prediction, using machine learning is great and typically outperforms traditional statistical approaches (e.g. Elith et al. 2006).

However, a reason for its better performance presented in many method comparisons with the GLM—because the GLM is inflexible—is not why machine learning typically outperforms traditional approaches. Rather, and this seems to be a large caveat, GLMs require careful "hand-crafting" to approximate flexibility at a level that machine-learning models deliver out of their opaque box. The two key advantages, in particular of CARTs, are that machine learning automatically models non-linear effects and statistical interactions among predictors. Both can be easily coded into GLMs, but they typically are not.[32]

Several machine-learning approaches also can handle the $P > N$ problem, i.e., when we have more predictors than observations. Machine-learning models accomplish this feat by randomly selecting only a subset of predictors and then repeating (internally re-running) the model with new subsets of predictors. This also could be done with GLMs (see Code Box 6.2), but, as with building flexibility into a GLM, this requires statistical hand-crafting. The other advantage of machine learning is that it uses—or at least tries to use—cross-validation to reduce model complexity so as to hit the sweet spot in the bias-variance trade-off. This is what model selection or regularisation procedures try to do in GLM. Once again, the off-the-shelf solution is much more convenient to use than manually rigging a classical GLM.

In a nutshell, machine learning often is far quicker and more conveniently applied than tailoring traditional regression-style models to a specific dataset. In a sense, that makes machine learning more foolproof, a characteristic that often is, we think, mistaken to confer on machine learning an intrinsic superiority to traditional statistics.[33] Simulations of very specific problems, e.g., thresholds or cyclical functions, can help identify the most suitable approach.

6.6.3 Machine-learning, extrapolation, and prediction error

Our final reflection on machine-learning models concerns their ability to extrapolate beyond the range of the data and the estimates of uncertainty that (should) accompany it. A common misconception is that machine learning can do this better than a traditional statistical model such as a GLM. The following points of debate each could be addressed with different kinds of simulation.

[32] We often encounter statements such as "interactions were excluded to aid interpretation," as if a wrongly specified model is easier to interpret than a correct, albeit complicated, one.

[33] Interestingly the other larger movement in statistics in recent decades is into Bayesian statistics. Bayesian models are pure statistical handcraft and require a lot of thought and skill to fit. One could argue that the statistics and modelling community is splitting up into "carefree" and "geeky" groups, but on the other hand, there are increasing numbers of implementations of Bayesian deep-learning approaches (e.g., Arbel et al. 2023; Bharadiya 2023; Wikle & Zammit-Mangion 2023).

First, non-linear models, which include all machine-learning models, "learn" about the behaviour of the fitted function $f(x)$ at the extremes (at high and low values of x) mostly from observed values near those extremes. A linear regression model forces a line through the cloud of all the data points, but the local character of a GAM, for example, will extrapolate the trend in the data near the limits of x:[34] if extreme observations are increasing, the extrapolation will increase, too.[35] In contrast, any CART-based approach (including random forest and extreme gradient boosting) will remain flat at the extremes. Although it also fits local subsets of observations, unless a trend is imposed by some other method, CART estimates cannot exceed the observed range of values; out-of-range predictions cannot exceed observed values, either.[36] Neither of these extrapolations are automatically wrong, but we must be aware of the structural constraints each type of model imposes on extrapolation and out-of-range estimates.

Second, most machine-learning approaches are non-probabilistic—they use loss functions other than negative log-likelihoods. In contrast to likelihood-based, probabilistic models (such as a GLMs), many machine-learning algorithms do not have an analytical (closed-form) description of the estimates they produce (Hastie et al. 2009; Murphy et al. 2004; Kuhn & Johnson 2013). As a result, no closed-form estimates for the standard error of predictions exist for machine-learning models, although in a few cases there are approximations (Wager et al. 2014; Gal & Ghahramani 2016). Thus, in an unacceptable number of publications, effect plots are depicted without error bars, giving the reader the misleading impression of high precision (Ellison 2001). Bootstrapping (see §2.3.3 in Chapter 2) is a common, computer-intensive way to generate confidence intervals for computer-age effect plots (Hastie et al. 2020), but it is not foolproof (see §8.5.5 and Figure 8.14 in Chapter 8).

Finally, the datasets fed into machine- and deep-learning models are often "Big," but the internal non-independence of the data results in optimistically (and inaccurately) high assessments of model fit when using cross-validation to evaluate model quality and prediction errors (Bergmeir & Benítez 2012; Le Rest et al. 2014; Roberts et al. 2017; Milà et al. 2022). The reason for this is that temporal or spatial autocorrelation in the response data (not the residuals!) invalidates random splits into training and test subsets of the dataset and results in overfitting the model (Ying 2019; Roelofs et al. 2019; Peng & Nagata 2020). Such overfitting can be avoided if the cross-validation folds are temporally or spatially blocked (e.g., Ploton et al. 2020; Kattenborn et al. 2022). Judging by the number of citations these aforementioned papers have received, recognition of this problem when applying machine-learning models to Big Data is slowly seeping into common practice. In the main, however, traditional random cross-validation continues to dominate machine-learning models used for spatial and temporal analyses, underestimating the true prediction error, perhaps by a large margin (e.g. Bahn & McGill 2013; Bergmeir et al. 2018).

6.7 Summary

Simulations confirm that there is some good statistical folklore in the world. However, there are also some not so useful rules-of-thumb, and even some near-superstitious beliefs in the

[34] The number of observations a GAM uses to fit each segment can be specified in the code, but most users stick with built-in defaults without considering their consequences for model fit and extrapolation.

[35] For example, the hind- and forecasting of Figure 8.3 in Chapter 8 would result in flat lines and stupendous upswings, respectively.

[36] In contrast to the GAM example (note 35), both the hind- and forecasting of Figure 8.3 using CART would result in flat lines.

magic of machine learning (not to speak of "artificial intelligence"; see, e.g., Searle 1982). Key take-homes from this chapter include:

- Model selection rarely improves interpretation.
- The logical problem of identifying the correct predictor from a pair (or more) of highly correlated ones is insoluble.
- Categorising predictor variables is poor practice and can lead to incorrect inferences.
- Carefully considered and interpreted, Monte Carlo simulation can deal with heteroscedasticity.
- Time-series data should be detrended before analysing them only if estimating the trend is not of interest. If time-series data include both trends and cycles, different methods of detrending can give very different results.
- Both classical parametric models and contemporary machine-learning and deep-learning models use loss functions in similar ways. Thus, machine- and deep-learning models are not intrinsically superior to parametric models, but can be quicker and easier to use than handcrafting classical GLMs for complex datasets. Simulations of specific problems can help us identify the best approach for any given dataset.

Theoretically motivated and derived rules-of-thumb as in van Belle (2008) certainly are more helpful and they return smart-yet-intuitive insights. Similarly, common errors also reveal where our minds wander off the path, particularly when we lack either the time to check what we are doing or formal statistical education (Good & Hardin 2012). Questioning common practice is a key element of scientific progress and has led to improved theory and knowledge (e.g., Juggins 2013), so we should keep doing it.

References

Ajmera, Y., Singhal, S., Dwivedi, S. N., & Dey, A. B. (2021). The changing perspective of clinical trial designs. *Perspectives in Clinincal Research*, 12(2), 66–71. doi: 10.4103/picr.PICR_138_20.

Alaa, A. M. & van der Schaar, M. (2020). Frequentist uncertainty in recurrent neural networks via blockwise influence functions. *arXiv*, 2006.13707, 1–16. arXiv: 2006.13707. doi: 10.48550/arXiv.2006.13707.

Altman, D. G. & Royston, P. (2006). The cost of dichotomising continuous variables. *British Medical Journal*, 332(7549), 1080. doi: 10.1136/bmj.332.7549.1080.

Anderson, C. (2008). The end of theory: The data deluge makes the scientific method obsolete. *WIRED*, 16(7). https://www.wired.com/2008/06/pb-theory/.

Andrew Gelman (2017). Frank Harrell's Statistics Blog! https://statmodeling.stat.columbia.edu/2017/01/14/frank-harrell-statistics-blog/. Accessed 16 January 2024.

Antonovics, J. & Fowler, N. L. (1985). Analysis of frequency and density effects on growth in mixtures of *Salvia splendens* and *Linum grandiflorum* using hexagonal fan designs. *Journal of Ecology*, 73(1), 219–234. doi: 10.2307/2259779.

Arbel, J., Pitas, K., Vladimirova, M., & Fortuin, V. (2023). A primer on Bayesian neural networks: Review and debates. *arXiv*, 2309.16314, 1–64. doi: 10.48550/arXiv.2309.16314.

Bahn, V. & McGill, B. J. (2013). Testing the predictive performance of distribution models. *Oikos*, 122(3), 321–331. doi: 10.1111/j.1600-0706.2012.00299.x.

Bartoń, K. (2023). MuMIn: Multi-Model Inference, version 1.47.5. https://cran.r-project.org/package=MuMIn.

Bennedsen, M., Hillebrand, E., & Koopman, S. J. (2023). On the evidence of a trend in the CO_2 airborne fraction. *Nature*, 616(7956), E1–E3. doi: 10.1038/s41586-023-05871-6.

Bergmeir, C. & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, 191, 192–213. doi: 10.1016/j.ins.2011.12.028.

Bergmeir, C., Hyndman, R. J., & Koo, B. (2018). A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics & Data Analysis*, 120, 70–83. doi: 10.1016/j.csda.2017.11.003.

Bestelmeyer, B. T., Ellison, A. M., Fraser, W. R., et al. (2011). Detecting and managing abrupt transitions in ecological systems. *Ecosphere*, 2(12), art129. doi: 10.1890/ES11-00216.1.

Bharadiya, J. P. (2023). A review of Bayesian machine learning principles, methods, and applications. *International Journal of Innovative Science and Research Technology*, 8(5), 2033–2038. doi: 10.5281/zenodo.8020825.

Bondell, H. D. & Reich, B. J. (2008). Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with OSCAR. *Biometrics*, 64(1), 115–121. doi: 10.1111/j.1541-0420.2007.00843.x.

Bowalekar, S. (2011). Adaptive designs in clinical trials. *Perspectives in Clinincal Research*, 2(1), 23–27. doi: 10.4103/2229-3485.76286.

Boyd, R. J., Powney, G. D., & Pescott, O. L. (2023). We need to talk about nonprobability samples. *Trends in Ecology & Evolution*, 38(6), 521–531. doi: 10.1016/j.tree.2023.01.001.

Bunn, A. G. (2008). A dendrochronology program library in R (dplR). *Dendrochronologia*, 26(2), 115–124. doi: 10.1016/j.dendro.2008.01.002.

Burnham, K. P. & Anderson, D. R. (2001). Kullback-Leibler information as a basis for strong inference in ecological studies. *Wildlife Research*, 28(2), 111–119. doi: 10.1071/WR99107.

Burnham, K. P. & Anderson, D. R. (2002). *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. New York, NY, USA: Springer.

Carpenter, S. R. (2003). *Regime Shifts in Lake Ecosystems: Pattern and Variation*. Oldendorf/Luhe, Germany: Ecology Institute.

Chambers, J. M. (1992). Linear models. In J. M. Chambers & T. J. Hastie (Eds.), *Statistical Models in S*. Milton Park, UK: Routledge.

Chen, A. H., Ge, W., Metcalf, W., et al. (2019). An assessment of true and false positive detection rates of stepwise epistatic model selection as a function of sample size and number of markers. *Heredity*, 122, 660–671. doi: 10.1038/s41437-018-0162-2.

Chen, S., Grant, E., Wu, T. T., & Bowman, F. D. (2014). Some recent statistical learning methods for longitudinal high-dimensional data. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(1), 10–18. doi: 10.1002/wics.1282.

Cheng, Q., Tezcan, J., & Cheng, J. (2014). Confidence and prediction intervals for semiparametric mixed-effect least squares support vector machine. *Pattern Recognition Letters*, 40, 88–95. doi: 10.1016/j.patrec.2013.12.010.

Cialdi, M. & Secchi, P. A. (1865). Sur la transparence de la mer. *Comptes Rendu de l'Acadamie des Sciences*, 61, 100–104.

Contamin, R. & Ellison, A. M. (2009). Indicators of regime shifts in ecological systems: What do we need to know and when do we need to know it? *Ecological Applications*, 19(3), 799–816. doi: 10.1890/08-0109.1.

Cottingham, K. L., Lennon, J. T., & Brown, B. L. (2005). Knowing when to draw the line: Designing more informative ecological experiments. *Frontiers in Ecology and the Environment*, 3(3), 145–152. doi: 10.1890/1540-9295(2005)003[0145:KWTDTL]2.0.CO;2.

Cressie, N. & Wikle, C. K. (2011). *Statistics for Spatio-Temporal Data*. Hoboken, NJ, USA: John Wiley & Sons.

Dakos, V., Carpenter, S. R., Brock, W. A., et al. (2012). Methods for detecting early warnings of critical transitions in time series illustrated using simulated ecological data. *PLoS ONE*, 7(7), e41010. doi: 10.1371/journal.pone.0041010.

Dawkins, R. (2011). The tyranny of the discontinuous mind. *New Statesman*, 100(5084/5085), 54–57.

Delap, L. (2006). 'Thus does man prove his fitness to be the master of things': Shipwrecks, chivalry and masculinities in nineteenth- and twentieth-century Britain. *Cultural and Social History*, 3(1), 45–74. doi: 10.1191/1478003805cs044oa.

Dormann, C. F., Elith, J., Bacher, S., et al. (2013). Collinearity: A review of methods to deal with it and a simulation study evaluating their performance. *Ecography*, 36(1), 27–46. doi: 10.1111/j.1600-0587.2012.07348.x.

Durham, J. (1658). *A Commentarie upon the Book of Revelation*. Edinburgh, Scotland: Printed by the heir of Andrew Anderson.

Durham, J. (1659). *The Dying man's Testament to the Church of Scotland; or, A Treatise Concerning Scandal*. London, England: The Company of Stationers.

Durham, J. (1685). *Heaven upon Earth, in the Serene Tranquility of a Good Conscience*. Edinburgh, Scotland: Printed by the heir of Andrew Anderson.

Elith, J., Graham, C. H., Anderson, R. P., et al. (2006). Novel methods improve prediction of species' distributions from occurrence data. *Ecography*, 29(2), 129–151. doi: 10.1111/j.2006.0906-7590.04596.x.

Ellison, A. M. (2001). Exploratory data analysis and graphic display. In S. M. Scheiner & J. Gurevitch (Eds.), *Design and Analysis of Ecological Experiments*. Oxford, UK: Oxford University Press, 2nd edition.

Ellison, A. M., Lavine, M., Kerson, P. B., et al. (2014). Building a foundation: Land-use history and dendrochronology reveal temporal dynamics of a *Tsuga canadensis* (Pinaceae) forest. *Rhodora*, 116(968), 377–427. doi: 10.3119/14-04.

Emiliano, P. C., Vivanco, M. J. F., & de Menezes, F. S. (2014). Information criteria: How do they behave in different models? *Computational Statistics & Data Analysis*, 69, 141–153. doi: 10.1016/j.csda.2013.07.032.

Faraway, J. (2022). faraway: Functions and Datasets for Books by Julian Faraway, version 1.0.8. https://cran.r-project.org/package=faraway.

Farris, F. A. (2010). The Gini index and measures of inequality. *The American Mathematical Monthly*, 117(10), 851–864. doi: 10.4169/000298910X523344.

Fisher, R. A. (1925). *Statistical Methods for Research Workers*. Edinburgh, Scotland, UK: Oliver and Boyd.

Fokkema, M., Smits, N., Zeileis, A., et al. (2018). Detecting treatment-subgroup interactions in clustered data with generalized linear mixed-effects model trees. *Behavior Research Methods*, 50(5), 2016–2034. doi: 10.3758/s13428-017-0971-x.

Fox, J. & Monette, G. (1992). Generalized collinearity diagnostics. *Journal of the American Statistical Association*, 87(417), 178–183. doi: 10.1080/01621459.1992.10475190.

Frankot, M., Mueller, P. M., Young, M. E., & Haar, C. V. (2023). Statistical power and false positive rates for interdependent outcomes are strongly influenced by test type: Implications for behavioral neuroscience. *Neuropsychopharmacology*, 48, 1612–1622. doi: 10.1038/s41386-023-01592-6.

Gal, Y. & Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML '16: Proceedings of the 33rd International Conference on Machine Learning–Volume 48* (pp. 1050–1059). New York, NY, USA: JMLR.org.

Gelman, A. & Carlin, J. (2014). Beyond power calculations: Assessing type S (sign) and type M (magnitude) errors. *Perspectives on Psychological Science*, 9(6), 641–651. doi: 10.1177/1745691614551642.

Gelman, A., Vehtari, A., Simpson, D., et al. (2020). Bayesian workflow. *arXiv*, 2011.01808, 1–77. doi: arXiv.2011.01808.

Global Carbon Budget Office (2023). Global Carbon Budget. https://globalcarbonbudget.org/. Accessed 5 February 2024.

Good, P. I. & Hardin, J. W. (2012). *Common Errors in Statistics (and How to Avoid Them)*, 4th edition. Hoboken, NJ, USA: Wiley.

Gotelli, N. J. & Ellison, A. M. (2012). *A Primer of Ecological Statistics*, 2nd edition. New York, NY, USA: Oxford University Press.

Goulon-Sigwalt-Abram, A., Duprat, A., & Dreyfus, G. (2005). From Hopfield nets to recursive networks to graph machines: Numerical machine learning for structured data. *Theoretical Computer Science*, 344(2), 298–334. doi: 10.1016/j.tcs.2005.08.026.

Hajjem, A., Bellavance, F., & Larocque, D. (2014). Mixed-effects random forest for clustered data. *Journal of Statistical Computation and Simulation*, 84(6), 1313–1328. doi: 10.1080/00949655.2012.741599.

Hansen, J. U. & Quinon, P. (2023). The importance of expert knowledge in big data and machine learning. *Synthese*, 201, 35. doi: 10.1007/s11229-023-04041-5.

Harrell, F. (2022). Problems Caused by Categorizing Continuous Variables. https://discourse.datamethods.org/t/categorizing-continuous-variables/3402. Accessed 16 January 2024.

Hastie, T., Tibshirani, R., & Wainwright, M. (2020). *Statistical Learning with Sparsity*. New York: Routledge.

Hastie, T., Tibshirani, R. J., & Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edition. Berlin: Springer.

Heinze, G., Wallisch, C., & Dunkler, D. (2018). Variable selection—A review and recommendations for the practicing statistician. *Biometrical Journal*, 60(3), 431–449. DOI: 10.1002/bimj.201700067.

Holfelder, K. D. (2004). Durham, James (1622–1658), Church of Scotland minister. DOI: 10.1093/ref:odnb/8314. Accessed 4 May 2023.

Hosmer, D. W., Jovanovic, B., & Lemeshow, S. (1989). Best subsets logistic regression. *Biometrics*, 45(4), 1265–1270. doi: /10.2307/2531779.

Houghton, R. A. & Nassikas, A. A. (2017). Global and regional fluxes of carbon from land use and land cover change 1850–2015. Global Biogeochemical Cycles, 31(3), 456–472. doi: 10.1002/2016GB005546.

Houle, C. & Bodea, C. (2017). Ethnic inequality and coups in sub-Saharan Africa. *Journal of Peace Research*, 54(3), 382–396. doi: 10.1177/0022343316685140.

Huang, N. E., Shen, Z., Long, S. R., et al. (1998). The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1971), 903–995. doi: 10.1098/rspa.1998.0193.

IISS (2021). Sub-Saharan Africa. *Armed Conflict Survey, 2022*, 8(1), 200–321. doi: 10.1080/23740973.2022.2135793.

Inouye, B. D. (2001). Response surface experimental designs for investigating interspecific competition. *Ecology*, 82(10), 2696–2706. doi: 10.1890/0012-9658(2001)082[2696:RSEDFI]2.0.CO;2.

Ives, A. (2022). Random errors are neither: On the interpretation of correlated data. *Methods in Ecology and Evolution*, 13(10), 2092–2105. doi: 10.1111/2041-210X.13971.

Joseph, M. B. (2020). Neural hierarchical models of ecological populations. *Ecology Letters*, 23(4), 734–747. doi: 10.1111/ele.13462.

Juggins, S. (2013). Quantitative reconstructions in palaeolimnology: New paradigm or sick science? *Quaternary Science Reviews*, 64, 20–32. doi: 10.1016/j.quascirev.2012.12.014.

Kattenborn, T., Schiefer, F., Frey, J., et al. (2022). Spatially autocorrelated training and validation samples inflate performance assessment of convolutional neural networks. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 5, 100018. doi: 10.1016/j.ophoto.2022.100018.

Kelly, H. A. (1994). "Rule of thumb" and the folklaw of the husband's stick. *Journal of Legal Education*, 44(3), 341–365. http://www.jstor.org/stable/42893341.

Kendall, M. G. (1975). *Rank Correlation Methods*, 4th edition. London, UK: Charles Griffin.

Kiaee, F., Sheikhzadeh, H., & Eftekhari Mahabadi, S. (2016). Sparse Bayesian mixed-effects extreme learning machine, an approach for unobserved clustered heterogeneity. *Neurocomputing*, 175(A), 411–420. doi: 10.1016/j.neucom.2015.10.073.

Killick, R., Haynes, K., & Eckley, I. A. (2022). changepoint: An R package for changepoint analysis, version 2.2.4. https://CRAN.R-project.org/package=changepoint. Accessed 6 September 2023.

Kohout, F. J. & Norwood, G. J. (1981). Interpretation of research data: Analysis of variance. *American Journal of Health-System Pharmacy*, 38(1), 96–104. doi: 10.1093/ajhp/38.1.96.

Kotz, M., Levermann, A., & Wenz, L. (2022). The effect of rainfall changes on economic production. *Nature*, 601(7892), 223–227. doi: 10.1038/s41586-021-04283-8.

Kreyling, J., Schweiger, A. H., Bahn, M., et al. (2018). To replicate, or not to replicate—That is the question: How to tackle nonlinear responses in ecological experiments. *Ecology Letters*, 21(11), 1629–1638. doi: 10.1111/ele.13134.

Kuchibhotla, A. K., Kolassa, J. E., & Kuffner, T. A. (2022). Post-selection inference. *Annual Review of Statistics and Its Application*, 9(1), 505–527. doi: 10.1146/annurev-statistics-100421-044639.

Kuhn, M. & Johnson, K. (2013). *Applied Predictive Modeling*. Berlin, Germany: Springer.

Le Rest, K., Pinaud, D., Monestiez, P., et al. (2014). Spatial leave-one-out cross-validation for variable selection in the presence of spatial autocorrelation. *Global Ecology and Biogeography*, 23(7), 811–820. doi: 10.1111/geb.12161.

Leeb, H. & Pötscher, B. M. (2005). Model selection and inference: Facts and fiction. *Econometric Theory*, 21(1), 21–59. doi: 10.1017/S0266466605050036.

Luts, J., Molenberghs, G., Verbeke, G., et al. (2012). A mixed effects least squares support vector machine model for classification of longitudinal data. *Computational Statistics & Data Analysis*, 56(3), 611–628. doi: 10.1016/j.csda.2011.09.008.

Magnuson, J. J., Carpenter, S. R., & Stanley, E. H. (2022). North Temperate Lakes LTER: Secchi Disk Depth; Other Auxiliary Base Crew Sample Data 1981. https://doi.org/10.6073/pasta/4c5b055143e8b7a5de695f4514e18142. Accessed 6 September 2023.

Mann, H. B. (1945). Non-parametric tests against trend. *Econometrica*, 13(3), 245–259. doi: 10.2307/1907187.

McElreath, R. (2020). *Statistical Rethinking: A Bayesian Course with Examples in R and STAN*, 2nd edition. Boca Raton, FL, USA: Chapman & Hall/CRC.

McPartland, M. Y., St. George, S., Pederson, G. T., & Anchukaitis, K. J. (2020). Does signal-free detrending increase chronology coherence in large tree-ring networks? *Dendrochronologia*, 63, 125755. doi: 10.1016/j.dendro.2020.125755.

Milà, C., Mateu, J., Pebesma, E., & Meyer, H. (2022). Nearest neighbour distance matching Leave-One-Out Cross-Validation for map validation. *Methods in Ecology and Evolution*, 13(6), 1304–1316. doi: 10.1111/2041-210X.13851.

Morrissey, M. B. & Ruxton, G. D. (2018). Multiple regression is not multiple regressions: The meaning of multiple regression and the non-problem of collinearity. *Philosophy, Theory, and Practice in Biology*, 10, 003. doi: 10.3998/ptpbio.16039257.0010.003.

Murphy, J. M., Sexton, D. M. H., Barnett, D. N., et al. (2004). Quantification of modelling uncertainties in a large ensemble of climate change simulations. *Nature*, 430(7001), 768–772. doi: 10.1038/nature02771.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press.

Nakagawa, S. & Freckleton, R. P. (2011). Model averaging, missing data and multiple imputation: A case study for behavioural ecology. *Behavioral Ecology and Sociobiology*, 65(1), 91–101. doi: 10.1007/s00265-010-1044-7.

Nelder, J. (1962). New kinds of systematic designs for spacing experiments. *Biometrics*, 18(3), 283–307. doi: 10.2307/2527473.

Nielsen, S. F. (2003). Proper and improper multiple imputation. *International Statistical Review*, 71(3), 593–607. doi: 10.2307/1403831.

OED Online (2023). Rule of thumb. https://www.oed.com/view/Entry/168726. Accessed 4 May 2023.

Oppel, S., Powell, A. N., & Dickson, D. L. (2009). Using an algorithmic model to reveal individually variable movement decisions in a wintering sea duck. *Journal of Animal Ecology*, 78(3), 524–531. doi: 10.1111/j.1365-2656.2008.01513.x.

Peng, Y. & Nagata, M. H. (2020). An empirical overview of nonlinearity and overfitting in machine learning using COVID-19 data. *Chaos, Solitons & Fractals*, 139, 110055. doi: 10.1016/j.chaos.2020.110055.

Peters, R. L., Groenendijk, P., Vlam, M., & Zuidema, P. A. (2015). Detecting long-term growth trends using tree rings: A critical evaluation of methods. *Global Change Biology*, 21(5), 2040–2054. doi: 10.1111/gcb.12826.

Pichler, M. & Hartig, F. (2023). Machine learning and deep learning—A review for ecologists. *Methods in Ecology and Evolution*, 14(4), 994–1016. doi: 10.1111/2041-210X.14061.

Ploton, P., Mortier, F., Réjou-Méchain, M., et al. (2020). Spatial validation reveals poor predictive performance of large-scale ecological mapping models. *Nature Communications*, 11(11), 4540. doi: 10.1038/s41467-020-18321-y.

Quinn, T. P., Le, V., & Cardilini, A. P. A. (2021). Test set verification is an essential step in model building. *Methods in Ecology and Evolution*, 12(1), 127–129. doi: 10.1111/2041-210X.13495.

Roberts, D. R., Bahn, V., Ciuti, S., et al. (2017). Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40(8), 913–929. doi: 10.1111/ecog.02881.

Roelofs, R., Shankar, V., Recht, B., et al. (2019). A meta-analysis of overfitting in machine learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, volume 32 (pp. 1–11). Red Hook, NY, USA: Curran Associates.

Roessler, P. (2011). The enemy within: Personal rule, coups, and civil war in Africa. *World Politics*, 63(2), 300–346. doi: 10.1017/S0043887111000049.

Royston, P., Altman, D. G., & Sauerbrei, W. (2006). Dichotomizing continuous predictors in multiple regression: A bad idea. *Statistics in Medicine*, 25(1), 127–141. doi: 10.1002/sim.2331.

Rozins, C., Hood, M. E., Cho, J. H., & Antonovics, J. (2020). Exploring density- and frequency-dependent interactions experimentally: An R program for generating hexagonal fan designs. *Methods in Ecology and Evolution*, 11(5), 678–683. doi: 10.1111/2041-210X.13382.

Sawilowsky, S. (2003). You think you've got trivials? *Journal of Modern Applied Statistical Methods*, 2(1), 218–225. doi: 10.22237/jmasm/1051748460.

Schmidt, R. (1988). Dose-finding studies in clinical drug development. *European Journal of Clinical Pharmacolgy*, 34(1), 15–19. doi: 10.1007/BF01061410.

Schofield, M. R., Barker, R. J., Gelman, A., et al. (2016). A model-based approach to climate reconstruction using tree-ring data. *Journal of the American Statistical Association*, 111(513), 93–106. doi: 10.1080/01621459.2015.1110524.

Searle, J. (1982). The myth of the computer. *The New York Review of Books*, 29 (April 29), 3–7. doi: https://www.nybooks.com/articles/1982/04/29/the-myth-of-the-computer/.

Sela, R. J. & Simonoff, J. S. (2012). RE-EM trees: A data mining approach for longitudinal and clustered data. *Machine Learning*, 86(2), 169–207. doi: 10.1007/s10994-011-5258-3.

Seok, K. H., Shim, J., Cho, D., et al. (2011). Semiparametric mixed-effect least squares support vector machine for analyzing pharmacokinetic and pharmacodynamic data. *Neurocomputing*, 74(17), 3412–3419. doi: 10.1016/j.neucom.2011.05.012.

Shepard, L. A. (2019). Classroom assessment to support teaching and learning. *The Annals of the American Academy of Political and Social Science*, 683(1), 183–200. doi: 10.1177/0002716219843818.

Shumway, R. H. & Stoffer, D. S. (2006). *Time Series Analysis and Its Applications: With R Examples*. New York, NY, USA: Springer.

Stewart, P. S., Stephens, P. A., Hill, R. A., et al. (2023). Model selection in occupancy models: Inference versus prediction. *Ecology*, 104(3), e3942. doi: 10.1002/ecy.3942.

Succi, S. & Coveney, P. V. (2019). Big data: The end of the scientific method? *Philosophical Transactions of the Royal Society, Series A*, 377, 20180145. doi: 10.1098/rsta.2018.0145.

Ting, N., Ed. (2006). *Dose Finding in Drug Development*. New York, NY, USA: Springer.

Tredennick, A. T., Hooker, G., Ellner, S. P., & Adler, P. B. (2021). A practical guide to selecting models for exploration, inference, and prediction in ecology. *Ecology*, 102(6), e03336. doi: 10.1002/ecy.3336.

Underwood, A. J. (1997). *Experiments in Ecology: Their Logical Design and Interpretation Using Analysis of Variance*. Cambridge, UK: Cambridge University Press.

U.S. DOE (2020). Free-Air CO2 Enrichment Experiments: FACE Results, Lessons, and Legacy, DOE/SC-0202. DOI: 10.2172/1615612. Accessed 5 October 2023.

van Belle, G. (2008). *Statistical Rules of Thumb*, 2nd edition. Hoboken, N.J: Wiley.

van Marle, M. J. E., van Wees, D., Houghton, R. A., et al. (2022a). Code for "new land-use-change emissions indicate a declining CO_2 airborne fraction." doi: 10.5281/zenodo.5617953.

van Marle, M. J. E., van Wees, D., Houghton, R. A., et al. (2022b). RETRACTED ARTICLE: New land-use-change emissions indicate a declining CO_2 airborne fraction. *Nature*, 603(7901), 450–454. DOI: 10.1038/s41586-021-04376-4.

van Marle, M. J. E., van Wees, D., Houghton, R. A., et al. (2023). Retraction note: New land-use-change emissions indicate a declining CO_2 airborne fraction. *Nature*, 622(7981), 202. doi: 10.1038/s41586-023-06605-4.

Wagenmakers, E. J. & Farrell, S. (2004). AIC model selection using Akaike weights. *Psychonomic Bulletin & Review*, 11(1), 192–196. doi: 10.3758/BF03206482.

Wager, S., Hastie, T., & Efron, B. (2014). Confidence intervals for random forests: The jackknife and the infinitesimal jackknife. *Journal of Machine Learning Research*, 15(1), 1625–1651. doi: 10.5555/2627435.2638587.

Wasserstein, R. L. & Lazar, N. A. (2016). The ASA statement on *p*-values: Context, process, and purpose. *The American Statistician*, 70(2), 129–133. doi: 10.1080/00031305.2016.1154108.

Wheeler, B. E. & Peet, R. K. (2016). A multi-scale analysis of plant diversity along soil nutrient gradients. In E. O. Box (Ed.), *Vegetation Structure and Function at Multiple Spatial, Temporal and Conceptual Scales* (pp. 425–444). Cham, Switzerland: Springer.

Wikle, C. K. & Zammit-Mangion, A. (2023). Statistical deep learning for spatial and spatiotemporal data. *Annual Review of Statistics and Its Application*, 10(1), 247–270. doi: 10.1146/annurev-statistics-033021-112628.

Wood, S. N. (2022). Inferring UK COVID-19 fatal infection trajectories from daily mortality data: Were infections already in decline before the UK lockdowns? *Biometrics*, 78(3), 1127–140. doi: 10.1111/biom.13462.

Ying, X. (2019). An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168, 022022. doi: 10.1088/1742-6596/1168/2/022022.

Zhang, D., Liu, Y., Si, L., et al. (2011). Multiple instance learning on structured data. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 24* (pp. 145–153). Red Hook, NY, USA: Curran Associates.

7

Workflows and pipelines can introduce and propagate artefacts

Statistical analyses, particularly for non-experimental data, are becoming more and more sophisticated. This is a result of technological advances that yield large datasets that require substantial data cleaning and because interesting questions can be addressed by linking different databases. Common workflows for analysing Big Data comprise many steps, beginning with quality-related screening, correction, and imputation of data; continuing with feature engineering (i.e., derivation of additional predictors from the data at hand); and then moving on to hyper-parameter optimisation, model selection, and model fitting (e.g., Keller et al. 2008; Rieber et al. 2009; Zhang & Storey 2016; Golding et al. 2018; Dago et al. 2019; SoRelle et al. 2020; Wang et al. 2020; Gupte et al. 2022; Schilder et al. 2022; Adde et al. 2023). Stringing together these (and other) steps in an analytical workflow may be just fine, or artefacts may be introduced at one or more steps that could lead to wildly distorted model fits, biased estimates, and corrupted *p*-values. How can we distinguish reliable workflows from unreliable ones?

We develop workflows with good intentions. Whenever such workflows are proposed in a manuscript, however, it is considered good practice to verify them with simulations. Verification here means that the individual components of the workflow behave correctly so that the overall workflow does what it is supposed to do (Schmolke et al. 2010). However, we can properly verify and evaluate the consistency of our workflows only with simulated data. For example, when Zayed et al. (2022) analysed transcriptome data from 120 ocean water samples, they used a number of data filtering and controlling steps for their workflow. To ensure that these did not alter the original data in unexpected ways, Zayed et al. (2022) followed the methods developed by Roux et al. (2017) and simulated data akin to their samples to verify their workflow. We should be sure to confront the workflow with a wide range of simulated datasets and determine whether its output matches the parameters of our data simulations. Finally, uncertainties can accumulate and propagate through the workflow. Simulations should be used to test whether the final results are unbiased and have the nominal coverage, i.e., a *p*-value of 0.05 should actually detect a spurious effect in only 5% of the cases.

In this chapter we first present an analysis of the initial steps that are common to many workflows. We start with methods to deal with missing values—imputation of missing predictors, coping with censored observations, pre-selection of variables, and dimensional reduction—and assess whether (and by how much) they introduce bias. We then move on to examine error propagation in workflows to investigate bias in estimates, *p*-values, prediction, and uncertainty. As in the previous chapters, all these methods and analyses are motivated through case studies.

What you will learn from this chapter

- Methods for dealing with missing data that occur randomly or with some relationship to the predictors or response variables themselves.
- Methods for estimating or inferring values for truncated or censored observations.
- Whether and how to pre-select a small set of predictors to use in a larger analytical workflow.
- The pro et contra of regression on residuals and of multiple regression when predictors are correlated.
- How propagation of errors in statistical workflows can affect estimates of uncertainty.

7.1 What can we do about missing data?

Datasets of any size may contain missing data. Because many different statistical methods, including generalised linear models, principal component analysis, other multivariate data reduction techniques, and a range of machine-learning and deep-learning models all require complete datasets,[1] we usually have to do something about missing values. The easiest option is to delete all the rows in a spreadsheet for which any column has a missing value; this option is called list-wise deletion. However, if missing values are scattered randomly throughout our dataset, list-wise deletion can dramatically reduce the size of the resulting dataset, even if only a few values are missing from any single column. For example, imagine we have 20 variables (= columns) and 100 cases (= rows). If each variable has only two cases missing at random, then list-wise deletion will result, on average, in a proportional loss of $1 - (1 - 0.02)^{20} = 0.33$ of our rows—from 100 cases in the original dataset to only 67 cases in the "clean" dataset.

This simple example shows that the most important consequence arising from missing data is the loss of information in the dataset. This loss of information (i.e., a smaller sample size) leads directly to a substantial loss of power for our analyses (see Chapter 4). A second, and more insidious, problem arises if the missing data are not missing completely at random. This problem changes further if the probability of missing values increases (or decreases) with the value of the predictor or response variables. Furthermore, we may be missing values of either the predictor variables or the response variables. Thus, list-wise deletion of a dataset with missing values rarely is a good approach to dealing with missing data. We need to be more deliberate.

However, paraphrasing Deeks et al. (2023), single imputation is commonly used and is practical in most circumstances. However, it fails to acknowledge uncertainty in the imputed values and results, typically in confidence intervals that are too narrow. In contrast, multiple imputation and using statistical models that allow for missing data would require involvement of a knowledgeable statistician. In other words, dealing with missing data is not for the faint-hearted.

7.2 Types of missing data

We start by distinguishing three broad classes of missing data that share some features but differ in others that have large impacts on our analyses (Enders 2022):

[1] Essentially, any method that relies on matrix operations cannot deal with missing values that lead to empty entries in a matrix.

Missing completely at random (MCAR): The simplest case is that the probability that any value in our dataset is missing follows a uniform distribution on all dimensions of the dataset. By analogy, if we consider our spreadsheet of data to be a two-dimensional spatial array with X rows of observations and Y columns of variables, the probability that any cell (X_i, Y_i) is missing is uniform on X and uniform on Y.

Missing at random (MAR): Unlike MCAR data, MAR data are not missing completely at random. Rather, the probability of a missing value is related in some way to a predictor variable, which also may be unobserved. For example, when using questionnaires to assess wealth, people from some parts of town tend to respond less frequently. Although MAR observations result in a sample that is less representative of the population as a whole, they do not necessarily bias the analysis. In this example, there are ways to correct for unequal response rates.

Missing not at random (MNAR): The probability of a missing value in an MNAR dataset is related to a response variable. Continuing with our example of the questionnaire assessing wealth, the probability of not reporting holiday expenditures may increase with how much was actually spent. The resulting dataset not only is less representative of the population (as with MAR data), but also is biased because only high holiday expenditures are not reported. Even if we correct for non-responses, we will still incorrectly parameterise the distribution of actual holiday expenditures.

Despite the differences in their generating processes, two key features unite these types of missing data. First, we typically cannot statistically test for the underlying cause(s) of missing values. Rather, we have to deduce or infer their values using critical reasoning. Second, missing data in themselves can, but need not, bias our analysis. In general, the consequences of MCAR and MAR data are unproblematic, but MNAR data can cause substantial problems for parameter estimation and statistical inference; using causal discovery to impute missing values is the topic of cutting-edge research (e.g., Liu & Constantinou 2023).

7.3 Imputation of missing predictors

We start our discussion of dealing with missing values with approaches for imputing missing values of predictor variables. "Imputation" is a fancy way to say that we are filling in the blanks in our dataset, replacing missing values with actual values. There are dozens of approaches to impute values for missing values, and these approaches range from the downright silly to the very clever. For MCAR and MAR data, imputation often has little effect on parameter estimates but does fill out the dataset and increases the statistical power of our analyses. For MNAR data, however, imputation requires an explicit representation of the process that causes missingness. Such a model should be based on our understanding of the system itself and likely will be different for each analysis. The main challenge for analysts is not to do the imputation, but to justify it to themselves, their colleagues, and in many cases, their clients. Using simulations can help demonstrate not only how we imputed missing data but also why the imputation we did was reasonable and was done correctly.[2]

7.3.1 A case study: Predictors of type 2 diabetes

Diabetes often is considered a disease of affluence (Shi 2016). But from the 1960s to the 1980s, the startling increase in type 2 diabetes was difficult to account for because of a

[2] It would seem intuitive that imputing a small amount of missing data would be less influential or problematic than imputing a larger amount. However, Madley-Dowd et al. (2019) used simulations to demonstrate that the proportion of missing data is not a reliable guide for how effective imputation is, and that in the case of MAR, even imputing a substantial proportion yields unbiased estimates.

lack of data on its prevalence and on correlates of this disease. In the 1960s, as part of broader public health efforts focused on displaced Native Americans, a voluntary longitudinal study of the "Gila River Indian Community" began (the "Gila River Indian Community" is properly called the Akimel O'odham but also is called the "Pima" by anthropologists and non-Native Americans outside the community; Pritzker 1998; Radin 2017). The resulting dataset has been very important for diabetes research; it also is a standard dataset used in machine-learning competitions (Smith et al. 1988).[3]

THE DATA

The Pima Indian Diabetic Disease dataset includes observational data from 768 women. The binary response variable is whether or not the individual shows clinical signs of diabetes. The eight predictor variables include: two standard indicators for diabetes (blood-sugar concentration [μg/mL] and serum insulin levels [μU/mL] measured two hours after the individual has consumed a large amount of glucose in liquid);[4] age; body-mass index (kg/m^2); number of previous pregnancies; diastolic blood pressure (mm Hg); triceps skin-fold thickness (mm); and a diabetes pedigree function.[5] In total, the dataset has 768 rows and 9 columns; if there were no missing entries, this dataset would have 6,912 individual values. However, there are 647 missing entries in this dataset,[6] and list-wise deletion would result in a loss of 375 rows—information about more than half of the cases. The information gleaned from the remaining 393 women would still be valuable, but it would be unfortunate to ignore all the other individuals only because their insulin level is unknown.

TWO ISSUES WITH IMPUTATION

Can imputation rescue these individuals? Filling the gaps can be done in many ways, but we are here concerned mainly with two issues: (1) the inappropriate use of the response variable (`test`, the clinical manifestation of other symptoms of diabetes) to impute predictors; and (2) whether to use the correlation structure among the predictor variables to estimate missing data.

---

[3] The Pima Indian Diabetic Disease database has been one of many used for benchmarking machine-learning approaches, and was hosted by the University of Califoria at Irvine's Machine Learning repository (https://archive.ics.uci.edu/. After Radin (2017) described the history of both the Akimel O'odham and this dataset, UCI removed the data from their repository, but it is still widely available (e.g., it is part of the Kaggle database used for improving machine-learning methods and models [Kaggle 2017] and it is included within R's `faraway` library). Radin (2017) identified the colonial history of the dataset and we acknowledge the contention that such an origin may be attached to the data themselves. Radin (2017), however, does not identify any actual criticism or privacy concerns raised by the Akimel O'odham regarding the use of this anonymized dataset. Although twenty-first-century Institutional Review Boards and other ethics committees might impose a different strategy to achieve voluntary consent to collect such data, these data were recorded while consensual medical attention was provided to the Akimel O'odham community. The dataset has been important for our understanding of the onset of diabetes and it does have substantial scientific value. We use it here respectfully as an exemplar realistic statistical dataset that is reasonably small and includes a binary response with several correlated predictors, zero-inflation, and missing data. Surprisingly, there are few comparable datasets in the public domain.

[4] A blood-sugar concentration $\geq 200\,\mu g$/mL or a serum insulin level $\geq 80\,\mu$U/mL indicates the occurrence of diabetes (CDC 2023). IU stands for "international units"; 1 IU of insulin represents the biological activity of 0.0347 mg of human insulin.

[5] The diabetes pedigree function provides a measure of the expected genetic influence of affected and unaffected relatives on the eventual diabetes risk of an individual (Smith et al. 1988).

[6] The original dataset has entries of "0" that ought to be "NA"s. These false zeros are in the variables (names as in the `pima` dataset in the `faraway` library) `glucose` (N *zeros* $= 5$), `diastolic` ($N = 35$), `triceps` ($N = 227$), `insulin`($N = 374$), and `bmi` ($N = 11$). Individuals with zero-values for any of these variables would be dead. In contrast, zeros in `pregnant` are considered to be true zeros.

It is inappropriate to use the response variable to impute missing values of the predictors simply because it is circular to use Y to estimate missing X, which are then used to model Y (Nakagawa 2015; Enders 2022).[7] So, when we impute missing predictor values, we do it only within the other predictors (the results would be better if we did use Y, but the p-values would be distorted).

The second issue is less straightforward. It should be clear that we should not fill in data gaps with random values, but rather use values that are more likely to be correct. We could regress the predictor with missing values against the remaining predictors and use the fitted model to estimate each missing value. Such models include ordination or random-forest-based "guess-from-the-neighbours" approaches (Enders 2022), and a coupled (chained) regression using all predictors together (e.g., predictive mean matching; Su et al. 2011; van Buuren & Groothuis-Oudshoorn 2011). A simpler approach would be to replace missing values of a specific predictor by the mean or median of the observed values of that same variable. This latter, simple approach appears on first glance to be a bad idea because it ignores the information from the correlation structure among the predictors. After an initial, qualitative comparison of the results using three different approaches (unweighted means, random forests, and predictive mean matching), we turn to simulation to more rigorously compare how they do with MCAR and MAR datasets.

A QUALITATIVE COMPARISON

We first compared the fits of logistic regression models using the raw, unimputed data (considering all seemingly false zeros as true zeros) and the data with imputed values for zeros in `diastolic`, `triceps`, `insulin`, and `bmi`. Our logistic regression model used six (of the eight) predictors in the dataset: `pregnant`, `diastolic`, `triceps`, `bmi`, `age`, `insulin` (Code Box 7.1).

Code Box 7.1: Effects of imputing missing values in the Pima dataset

```
# Load required libraries
 library(faraway) # Includes Pima dataset
 library(mice) # For imputation of missing values

 summary(pima)
```

```
|      pregnant         glucose        diastolic         triceps
|  Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
|  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
|  Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
|  Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
|  3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
|  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
|
|     insulin            bmi           diabetes           age
|  Min.   :  0.0   Min.   : 0.00   Min.   :0.0780   Min.   :21.00
|  1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437   1st Qu.:24.00
|  Median : 30.5   Median :32.00   Median :0.3725   Median :29.00
```

[7] An interesting simulation exercise that we leave to you is to demonstrate that such circularity indeed affects model estimates.

```
|  Mean   : 79.8   Mean   :31.99   Mean   :0.4719   Mean   :33.24
|  3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262   3rd Qu.:41.00
|  Max.   :846.0   Max.   :67.10   Max.   :2.4200   Max.   :81.00
|
|       test
|  Min.   :0.000
|  1st Qu.:0.000
|  Median :0.000
|  Mean   :0.349
|  3rd Qu.:1.000
|  Max.   :1.000

# Turn zeros into "NA"s in columns 3 - 6 (for illustrative purposes)
 pimaNA <- pima
 for (i in 3:6) pimaNA[which(pimaNA[, i] == 0), i] <- NA
 summary(pimaNA) # With NAs; note differences, especially in triceps and insulin

|     pregnant         glucose        diastolic         triceps
|  Min.   : 0.000   Min.   :  0.0   Min.   : 24.00   Min.   : 7.00
|  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 64.00   1st Qu.:22.00
|  Median : 3.000   Median :117.0   Median : 72.00   Median :29.00
|  Mean   : 3.845   Mean   :120.9   Mean   : 72.41   Mean   :29.15
|  3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:36.00
|  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
|                                   NA's   :35       NA's   :227
|
|     insulin            bmi           diabetes           age
|  Min.   : 14.00   Min.   :18.20   Min.   :0.0780   Min.   :21.00
|  1st Qu.: 76.25   1st Qu.:27.50   1st Qu.:0.2437   1st Qu.:24.00
|  Median :125.00   Median :32.30   Median :0.3725   Median :29.00
|  Mean   :155.55   Mean   :32.46   Mean   :0.4719   Mean   :33.24
|  3rd Qu.:190.00   3rd Qu.:36.60   3rd Qu.:0.6262   3rd Qu.:41.00
|  Max.   :846.00   Max.   :67.10   Max.   :2.4200   Max.   :81.00
|  NA's   :374      NA's   :11
|
|       test
|  Min.   :0.000
|  1st Qu.:0.000
|  Median :0.000
|  Mean   :0.349
|  3rd Qu.:1.000

 md.pattern(pimaNA, plot=FALSE) # Pattern of missing values (plot not shown)

|     pregnant glucose diabetes age test bmi diastolic triceps insulin
| 393        1       1        1   1    1   1         1       1       1   0
| 144        1       1        1   1    1   1         1       1       0   1
| 192        1       1        1   1    1   1         1       0       0   2
| 2          1       1        1   1    1   1         0       1       0   2
```

```
| 26         1       1        1   1    1   1         0       0       0   3
| 1          1       1        1   1    1   0         1       1       1   1
| 1          1       1        1   1    1   0         1       1       0   2
| 2          1       1        1   1    1   0         1       0       0   3
| 7          1       1        1   1    1   0         0       0       0   4
|            0       0        0   0    0  11        35     227     374 647

# GLM on raw data with complete cases only ("listwise deletion" of NAs)
 summary(fm.raw <- glm(test ~ pregnant + diastolic + triceps + bmi + age +
     insulin, data=pimaNA, family=binomial))

| Call:
| glm(formula = test ~ pregnant + diastolic + triceps + bmi + age +
|     insulin, family = binomial, data = pima)
|
| Deviance Residuals:
|     Min       1Q   Median       3Q      Max
| -2.0655  -0.7735  -0.4889   0.9077   2.3474
|
| Coefficients:
|              Estimate Std. Error z value Pr(>|z|)
| (Intercept) -6.183266   0.918230  -6.734 1.65e-11 ***
| pregnant     0.059344   0.049098   1.209 0.226776
| diastolic    0.003717   0.010635   0.349 0.726734
| triceps      0.016433   0.015471   1.062 0.288150
| bmi          0.062838   0.024346   2.581 0.009851 **
| age          0.053732   0.016738   3.210 0.001327 **
| insulin      0.004105   0.001114   3.686 0.000228 ***
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| (Dispersion parameter for binomial family taken to be 1)
|
|     Null deviance: 498.90  on 392  degrees of freedom
| Residual deviance: 405.13  on 386  degrees of freedom
|   (375 observations deleted due to missingness)
| AIC: 419.13


# Imputation with only predictors ("test" removed before imputation)
# Create data frames of pimaNA with imputed values computed three different ways
     (The mice package has 28 different built-in imputation methods)
 pima.impMean <- cbind.data.frame(complete(mice(pimaNA[,-9], method="mean", m=5,
     print=F)), "test"=pimaNA$test ) # Imputation with unconditional means
 pima.impPMM <- cbind.data.frame(complete(mice(pimaNA[,-9], method="pmm", m=5,
     print=F)), "test"=pimaNA$test ) # Imputation with predictive mean matching
 pima.impRF <- cbind.data.frame(complete(mice(pimaNA[,-9], method="rf", m=5,
     print=F)), "test"=pimaNA$test ) # Imputation with random forests
```

```
# GLMs on the datasets with different imputations
# Dataset with imputation using unconditional means
 summary(fm.mean <- glm(test ~ pregnant + diastolic + triceps + bmi + age +
     insulin, data=pima.impMean, family=binomial))

| Call:
| glm(formula = test ~ pregnant + diastolic + triceps + bmi + age +
|     insulin, family = binomial, data = pima.impMean)
|
| Deviance Residuals:
|     Min       1Q   Median       3Q      Max
| -2.3549  -0.8507  -0.5355   1.0648   2.3229
|
| Coefficients:
|               Estimate Std. Error z value Pr(>|z|)
| (Intercept) -6.1289071  0.6651798  -9.214  < 2e-16 ***
| pregnant     0.0931678  0.0284438   3.275 0.001055 **
| diastolic    0.0002825  0.0078156   0.036 0.971166
| triceps      0.0073458  0.0114795   0.640 0.522232
| bmi          0.0981497  0.0158358   6.198 5.72e-10 ***
| age          0.0289068  0.0085435   3.383 0.000716 ***
| insulin      0.0040754  0.0010884   3.744 0.000181 ***
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| (Dispersion parameter for binomial family taken to be 1)
|
|     Null deviance: 993.48  on 767  degrees of freedom
| Residual deviance: 843.66  on 761  degrees of freedom
| AIC: 857.66

# Dataset with imputation by predictive mean matching
 summary(fm.pmm <- glm(test ~ pregnant + diastolic + triceps + bmi + age +
     insulin, data=pima.impPMM, family=binomial))

| Call:
| glm(formula = test ~ pregnant + diastolic + triceps + bmi + age +
|     insulin, family = binomial, data = pima.impPMM)

| Deviance Residuals:
|     Min       1Q   Median       3Q      Max
| -2.5236  -0.8104  -0.5042   0.9794   2.3003
|
| Coefficients:
|               Estimate Std. Error z value Pr(>|z|)
| (Intercept) -5.8672813  0.6543019  -8.967  < 2e-16 ***
| pregnant     0.0969302  0.0293707   3.300 0.000966 ***
| diastolic    0.0025061  0.0078435   0.320 0.749337
| triceps      0.0055993  0.0098773   0.567 0.570788
```

```
| bmi          0.0848055  0.0174418   4.862 1.16e-06 ***
| age          0.0252784  0.0089080   2.838 0.004544 **
| insulin      0.0052682  0.0008344   6.314 2.72e-10 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| (Dispersion parameter for binomial family taken to be 1)
|
|     Null deviance: 993.48  on 767  degrees of freedom
| Residual deviance: 811.26  on 761  degrees of freedom
| AIC: 825.26

# Dataset with imputation by random forest
 summary(fm.rf <- glm(test ~ pregnant + diastolic + triceps + bmi + age +
     insulin, data=pima.impRF, family=binomial))

| Call:
| glm(formula = test ~ pregnant + diastolic + triceps + bmi + age +
|     insulin, family = binomial, data = pima.impRF)
|
| Deviance Residuals:
|     Min       1Q   Median       3Q      Max
| -2.1632  -0.8508  -0.5375   1.0134   2.2172
|
| Coefficients:
|               Estimate Std. Error z value Pr(>|z|)
| (Intercept) -5.7420443  0.6438076  -8.919  < 2e-16 ***
| pregnant     0.0960254  0.0285894   3.359 0.000783 ***
| diastolic    0.0021124  0.0077094   0.274 0.784081
| triceps     -0.0006314  0.0103077  -0.061 0.951158
| bmi          0.0979260  0.0166289   5.889 3.89e-09 ***
| age          0.0248121  0.0087345   2.841 0.004501 **
| insulin      0.0031161  0.0007366   4.230 2.33e-05 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| (Dispersion parameter for binomial family taken to be 1)
|
|     Null deviance: 993.48  on 767  degrees of freedom
| Residual deviance: 842.32  on 761  degrees of freedom
| AIC: 856.32
```

The results of the four logistic regressions qualitatively agree (Figure 7.1). However, `pregnant` (the number of prior pregnancies), which was not a significant term in the model fitted to the unimputed data, was a significant term in the three models fitted to imputed data. This result suggests the smaller sample size in the unimputed data led to an inflated type-II error rate. Thus, there is a potential inferential value for imputation. What it does not tell us, however, is whether imputation yields unbiased estimates. For that, we need to do simulations.

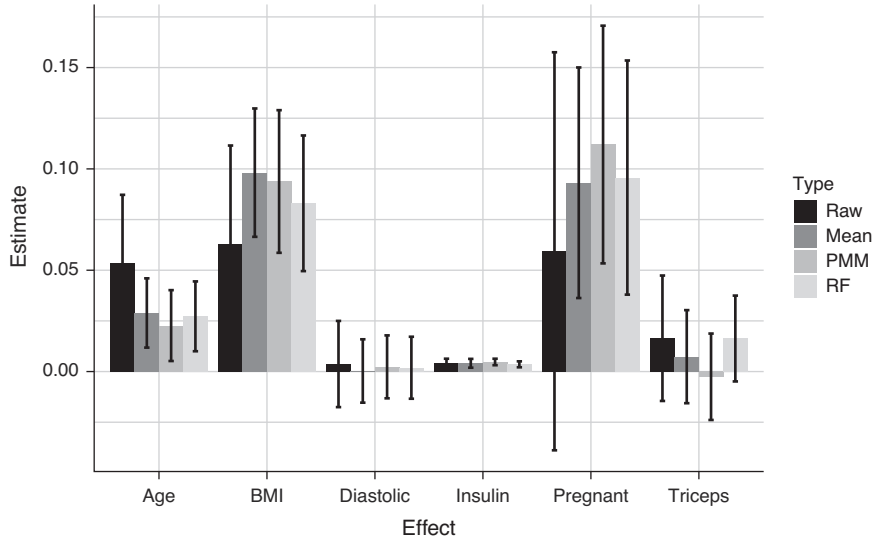


Figure 7.1 Model parameter estimates for raw data (black bars) and data imputed in three different ways (unweighted means ["mean"; dark grey]; random forests ["RF"; light grey]; and predictive mean matching ["PMM"; intermediate grey]). Note in particular the substantial change in the effect of pregnancy (`pregnant`), but there are other, smaller quantitative changes. Error bars are 95% confidence intervals on the parameter estimates.

7.3.2 A missing completely at random (MCAR) simulation

In the MCAR setting, we only have to sprinkle NAs into the predictors and compare the estimates based on the "holey" and the imputed data (Code Box 7.2). Note that for this and the two subsequent simulations, we use re-use the dataset `pima.impPMM` for which we imputed values for missing data using predictive mean matching (Code Box 7.1).

Code Box 7.2: A single simulation of imputing and modelling of MCAR data

```
# We re-use the imputed dataset pima.impPMM from Code Box 7.1
 sim <- pima.impPMM

# Simulation
 set.seed(123)
 sim$test <- with(pima.impPMM,  rbinom(nrow(pima), size=1, prob=plogis(-6 +
      0.06*pregnant + 0.004*diastolic + 0.015*triceps + 0.06*bmi + 0.05*age +
      0.004*insulin))) # values from raw data analysis
# Remove 30% of the predictor values at random
# Removal from matrices is easier than from data frames
 holes <- matrix(1, ncol=8, nrow=738)
 holes[sample(round(738*8), size=0.3*8*738)] <- NA
# Identify the position of the holes (NAs), using row/column indexing:
 NAindex <- which(is.na(holes), arr.ind=T)
# Transfer the holes:
 sim[NAindex] <- NA
 summary(sim)
```

```
|     pregnant        glucose        diastolic         triceps
|  Min.   : 0.00   Min.   :  0.0   Min.   : 30.00   Min.   : 7.00
|  1st Qu.: 1.00   1st Qu.:100.0   1st Qu.: 64.00   1st Qu.:21.25
|  Median : 3.00   Median :117.0   Median : 72.00   Median :30.00
|  Mean   : 3.89   Mean   :121.5   Mean   : 72.02   Mean   :29.15
|  3rd Qu.: 6.00   3rd Qu.:142.0   3rd Qu.: 80.00   3rd Qu.:36.00
|  Max.   :17.00   Max.   :198.0   Max.   :122.00   Max.   :99.00
|  NA's   :204     NA's   :230     NA's   :230      NA's   :222
|
| insulin           bmi           diabetes           age
|  Min.   : 14.0   Min.   :18.20   Min.   :0.0840   Min.   :21.00
|  1st Qu.: 74.0   1st Qu.:27.40   1st Qu.:0.2475   1st Qu.:24.00
|  Median :120.0   Median :32.00   Median :0.3750   Median :29.00
|  Mean   :150.8   Mean   :32.51   Mean   :0.4698   Mean   :33.66
|  3rd Qu.:184.0   3rd Qu.:36.80   3rd Qu.:0.6095   3rd Qu.:41.00
|  Max.   :846.0   Max.   :67.10   Max.   :2.4200   Max.   :81.00
|  NA's   :219     NA's   :240     NA's   :217      NA's   :209
|
| test
|  Min.   :0.0000
|  1st Qu.:0.0000
|  Median :0.0000
|  Mean   :0.3242
|  3rd Qu.:1.0000
|  Max.   :1.0000
```

```
# Impute values into the holey dataset
 sim.impMean <- cbind.data.frame(complete(mice(sim[,-9], method="mean", m=5,
     print=F)), "test"=sim$test )
 sim.impPMM <- cbind.data.frame(complete(mice(sim[,-9], method="pmm", m=5,
     print=F)), "test"=sim$test )

# Re-run the GLMs and extract coefficients into results matrix
 fm.raw <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
     data=sim, family=binomial)
 fm.mean <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
     data=sim.impMean, family=binomial)
 fm.pmm <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
     data=sim.impPMM, family=binomial)
 cbind(coef(fm.raw), coef(fm.mean), coef(fm.pmm))
```

```
|                    [,1]         [,2]         [,3]
| (Intercept) -7.460719449 -7.611603775 -5.920992364
| pregnant     0.053844980  0.125471849  0.100189283
| diastolic    0.002016267  0.012701019  0.005560633
| triceps     -0.006427809  0.021844160  0.023720471
| bmi          0.096131058  0.076660849  0.052560436
| age          0.068002027  0.040689440  0.036188855
| insulin      0.006801407  0.005424256  0.003989206
```

Next, we run this code several hundred times and summarise the data in a box plot (Code Box 7.3; Figure 7.2). The results demonstrate that imputation reduces the variance of the parameter estimates because it increases the total number of samples used in the models. However, for MCAR data such as these, imputation has no discernible effect on bias.

7.3.3 A missing at random (MAR) simulation

Data missing at random (MAR) present a slightly more worrying situation than data missing completely at random. In MAR data, there is a correlation between missingness and another variable and thus we may expect there to be a bias in the raw data. We also hope that imputation will alleviate the bias. However, Enders (2022) suggests that MAR is not problematic because the model can condition out the effect of missingness.

We directly jump to the many-simulations step (Code Box 7.4) because the initial few lines are only slightly different from those used for the MCAR case (Code Box 7.3). However, instead of sprinkling NAs over all the predictors in the dataset, we impose them on only two, `triceps` and `insulin`, and make missingness a function of pregnancies. Again, we use reuse the dataset `pima.impPMM` for which we imputed values for missing data using predictive mean matching (Code Box 7.1).

MAR data appear to result in biased estimates (Figure 7.3, black boxes). Imputation corrected the estimates for `pregnant` but not `bmi`, and it introduced some bias into the `estimate` for `triceps`—one of the targets for missing data. Enders' (2022) suggestion that MAR is corrected by conditioning on the causal variable (`pregnant`) seems optimistic. However, imputation using unconditional means (dark grey in Figure 7.3) is more biased, at least for some parameters, than imputation using predictive mean matching (light grey in Figure 7.3). In other words, using information about correlations among predictors does improve imputation. The effect is visible for `bmi`, `age`, `triceps`, and `insulin`, but negligible for `pregnant` and `diastolic` (Code Box 7.4 and Figure 7.3).

7.3.4 Conclusions

To summarise, for MCAR and MAR data, imputation improves parameter estimates and decreases variances by retaining a larger sample size. Imputation using unconditional means may increase bias because it does not use additional information available in the data. However, imputation using predictive mean matching takes advantage of this additional information in the form of the correlations among predictors. In our example, using this information reduces the bias in the parameter estimates. However, this would not necessarily be the case for missing not at random (MNAR) data. We do not discuss MNAR data here because we would need to model the process that itself leads to missingness. There are no generic routines for imputing MNAR data because the process model has to be determined for each individual problem. Rather, a process model uniquely tailored to a specific problem or case study would be needed to explore the effects of imputation on parameter estimation and bias when the data are MNAR.

7.4 Estimating values for censored observations

The data we record for response variables are not always as accurate or as precise as we would like, or they may be missing in a way that is dependent on their value. For example, it is common that there is a fixed, lower detection limit for an organic, inorganic, or heavy metal compound in solution (for example, 0.5 parts per million [ppm]). Continuing with this example, if the true value in the solution is too small to be recorded by our sensor, we will record values of $X_i = 0$ when in fact the values should be 0.5 ppm $\geq X \geq 0$. The resulting data are referred to as left-censored or left-truncated because we know that these values are

Code Box 7.3: Multiple simulations of imputing and modelling MCAR data

```
set.seed(123)
res <- replicate(500, {
  sim <- pima.impPMM
  sim$test <- with(pima.impPMM, rbinom(nrow(pima), size=1, prob=plogis(-6 +
      0.06*pregnant + 0.004*diastolic + 0.015*triceps + 0.06*bmi + 0.05*age +
      0.004*insulin)))
  holes <- matrix(1, ncol=8, nrow=738)
  holes[sample(round(738*8), size=0.3*8*738)] <- NA
  NAindex <- which(is.na(holes), arr.ind=T)
  sim[NAindex] <- NA
  sim.impMean <- cbind.data.frame(complete(mice(sim[,-9], method="mean", m=5,
      print=F)), "test"=sim$test )
  sim.impPMM <- cbind.data.frame(complete(mice(sim[,-9], method="pmm", m=5,
      print=F)), "test"=sim$test )

  fm.raw <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
      data=sim, family=binomial)
  fm.mean <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
      data=sim.impMean, family=binomial)
  fm.pmm <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
      data=sim.impPMM, family=binomial)

  cbind("none"=coef(fm.raw), "mean"=coef(fm.mean), "PMM"=coef(fm.pmm))
})

signif(apply(res, c(1,2), mean), 2)
```

```
|                none    mean     PMM
| (Intercept) -6.4000 -6.8000 -5.2000
| pregnant     0.0630  0.0820  0.0570
| diastolic    0.0044  0.0120  0.0079
| triceps      0.0150  0.0240  0.0150
| bmi          0.0610  0.0560  0.0450
| age          0.0550  0.0490  0.0370
| insulin      0.0045  0.0045  0.0033
```

```
signif(apply(res, c(1,2), sd), 3)
```

```
|                none     mean      PMM
| (Intercept) 1.91000 0.811000 0.629000
| pregnant    0.07780 0.031500 0.029000
| diastolic   0.02350 0.009250 0.007780
| triceps     0.03160 0.010600 0.011000
| bmi         0.04920 0.016800 0.016800
| age         0.02730 0.009820 0.008910
| insulin     0.00249 0.000973 0.000795
```

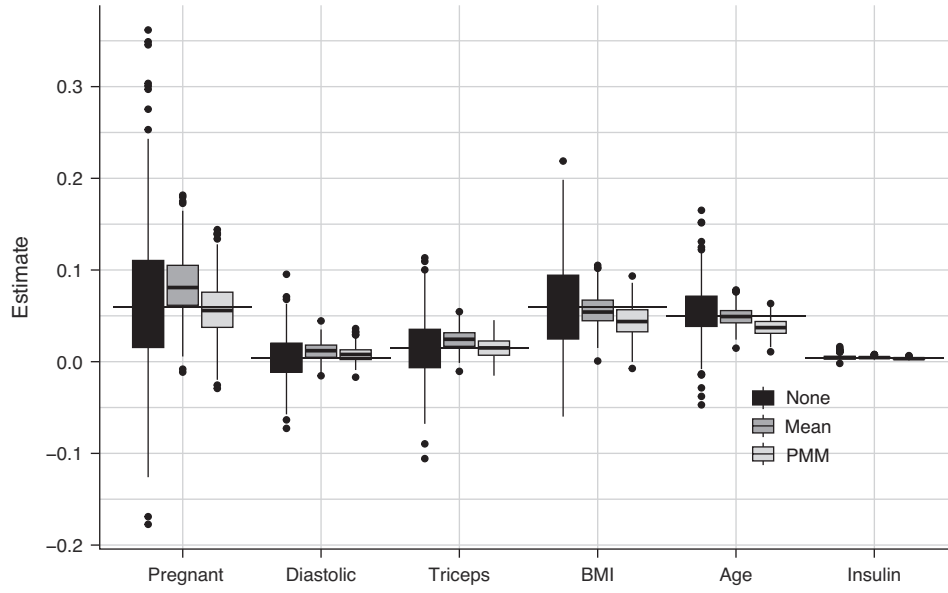


Figure 7.2 Estimation bias for raw ("none"; black boxes) and imputed data for data missing completely at random (MCAR). Horizontal line segments are true values. We expect (and observe) no bias for the data without imputation, as the NAs are imposed without pattern and hence without bias. Imputation substantially reduces estimation variance simply by maintaining a high sample size. The results from data imputation using unconditional means (dark grey boxes) or predictive mean matching (PMM; light grey boxes) are very similar, as we can expect with MCAR; PMM has consistently lower variance (see output in Code Box 7.3).

below the detection limit, but not by how much (Figure 7.4; Hansen et al. 2015; Canales et al. 2018; Cantoni et al. 2020).[8]

Similarly, there are right-censored data. The most common situation in which we encounter right-censored data is in survival analysis; at the end of a study we know the number of subjects for which an event (i.e., death) did not occur, but not when it might. Consider a five-year-long study of a medical treatment on life expectancy. If 10% of subjects have died by the end of the study, we know their life expectancy exactly. Because the other 90% of the participants are still alive, we know only their age at the end of the study. From this we can infer only the lower limit of their life expectancy, not its exact value (e.g., Gijbels 2010; Bogaerts et al. 2017). Right-censored data also are common in hydrology. For example, if the discharge rate of a river exceeds the capacity of the flow gauge, we might know only that the peak discharge rate was $\geq 400\ \mathrm{m^3 s^{-1}}$.

Censored data are different from the kinds of missing data we discussed in §7.3 in three ways. First, it is the response variable that is censored, not the predictor(s). Second, we know that the recorded value of a censored observation is, in fact, not correct. For example, if our detection limit for cadmium concentration in solution is 0.025 pg/L and the recorded value is 0.02 pg/l, we know the recorded value is incorrect. However, we do not know its exact value,

[8] Gijbels (2010) uses the term "truncated data" to refer to what we call left- and right-censored data in Figure 7.4. They use the term "censored data" to apply only to missing values of the survivorship to time T of N independent individuals in longitudinal studies: $T \overset{\text{i.i.d.}}{\sim} \{T_1, T_2, \ldots, T_N\}$. Because we can use the same methods to estimate both truncated data and left- or right-censored data, we use the term "censored data" to refer to both types of missing observations. For other types of censoring in survivorship data, see Gijbels (2010).

Code Box 7.4: Multiple simulations of imputing and modelling missing at random (MAR) data

```
set.seed(123)
resMAR <- replicate(500, {
  simMAR <- pima.impPMM

 # Define the probability model for the MAR data
  simMAR$prob <- with(pima.impPMM,  plogis(-6 + 0.06*pregnant + 0.004*diastolic
      + 0.015*triceps + 0.06*bmi + 0.05*age + 0.004*insulin))
  simMAR$test <- rbinom(nrow(pima), size=1, prob=simMAR$prob)

 # First, remove 30% of two variables with probability proportional to
     pregnant:
  holes1 <- rbinom(nrow(simMAR), prob=pregnant.ranged, size=1)
  holes2 <- rbinom(nrow(simMAR), prob=pregnant.ranged, size=1)
  simMAR$triceps[which(holes1==1)] <- NA
  simMAR$insulin[which(holes2==1)] <- NA

 # Imputation
  simMAR.impMean <- cbind.data.frame(complete(mice(simMAR[,-9], method="mean",
      m=5, print=F)), "test"=simMAR$test )
  simMAR.impPMM <- cbind.data.frame(complete(mice(simMAR[,-9], method="pmm",
      m=5, print=F)), "test"=simMAR$test )

 # Logistic regression models
  fm.raw <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
      data=simMAR, family=binomial)
  fm.mean <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
      data=simMAR.impMean, family=binomial)
  fm.pmm <- glm(test ~ pregnant + diastolic + triceps + bmi + age + insulin,
      data=simMAR.impPMM, family=binomial)

  cbind("none"=coef(fm.raw), "mean"=coef(fm.mean), "PMM"=coef(fm.pmm))
})
signif(apply(resMAR, c(1,2), mean), 2)
```

	none	mean	PMM
(Intercept)	-6.1000	-6.3000	-5.9000
pregnant	0.0610	0.0580	0.0550
diastolic	0.0033	0.0036	0.0034
triceps	0.0150	0.0120	0.0130
bmi	0.0620	0.0700	0.0610
age	0.0520	0.0540	0.0510
insulin	0.0040	0.0040	0.0039

```
signif(apply(resMAR, c(1,2), sd), 3)
```

| | none | mean | PMM |

```
| (Intercept) 0.840000   0.676000  0.670000
| pregnant    0.0492000  0.030500  0.031400
| diastolic   0.009960   0.007770  0.007840
| triceps     0.014600   0.012100  0.011700
| bmi         0.022900   0.017300  0.018800
| age         0.011900   0.008950  0.009100
| insulin     0.000923   0.000841  0.000768
```

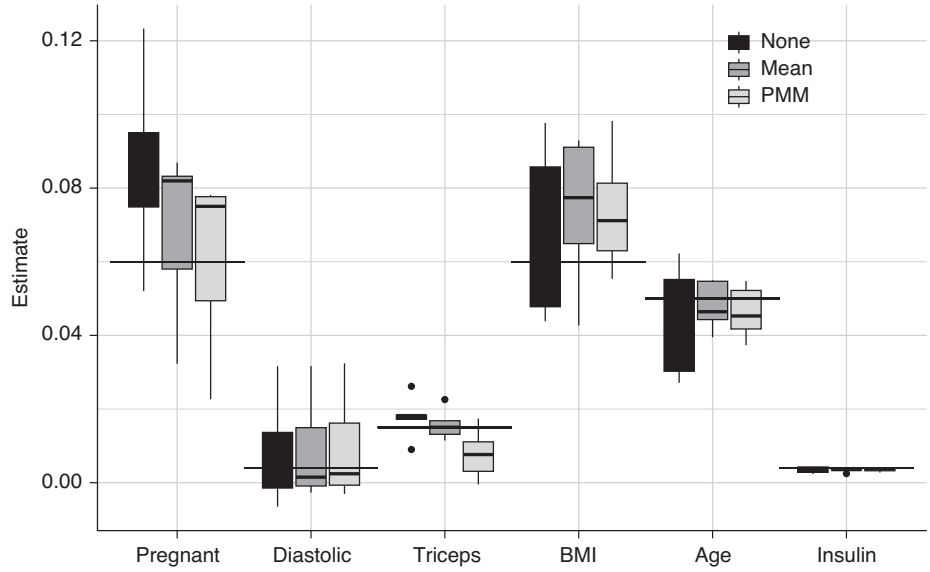


Figure 7.3 Estimation bias for raw and imputed data for data missing at random (MAR). Horizontal segments are true values and box shading as in Figure 7.2.

which is somewhere in the interval [0–0.025). Third, analysing datasets with censored observations yields biased parameter estimates. For example, if income is reported in categories, and the highest category is, say, "US$ > 100, 000/year," then we can only estimate the lower bound for that category. Consequently, we will underestimate the mean and quantiles of the income distribution (e.g., Helsel 2005; Bogaerts et al. 2017).

There are few general methods for estimating or inferring values for censored observations; the available statistical tools are specific to particular kinds of datasets and analytical methods. For example, we have very good ways to handle censored survivourship data, but it is difficult to use these methods in a workflow that also includes, for example, temporal autocorrelation, missing values for predictors, and machine learning. We can do so only if we can estimate correctly the censored values in the first step of the workflow and then use the corrected dataset in the remaining steps of the workflow.[9]

Simulations have been used regularly to explore how to best replace left-censored data; with values equal to zero, the detection limit, half that value, or other values imputed after assigning them as missing values (Helsel 2012). It is safe to say that none of these approaches

[9] Indeed, this is how a Bayesian workflow (Gelman et al. 2020) would deal with censored data. But Bayesian statistics are not known for their friendly embrace of machine- or deep-learning methods, nor for swift execution.

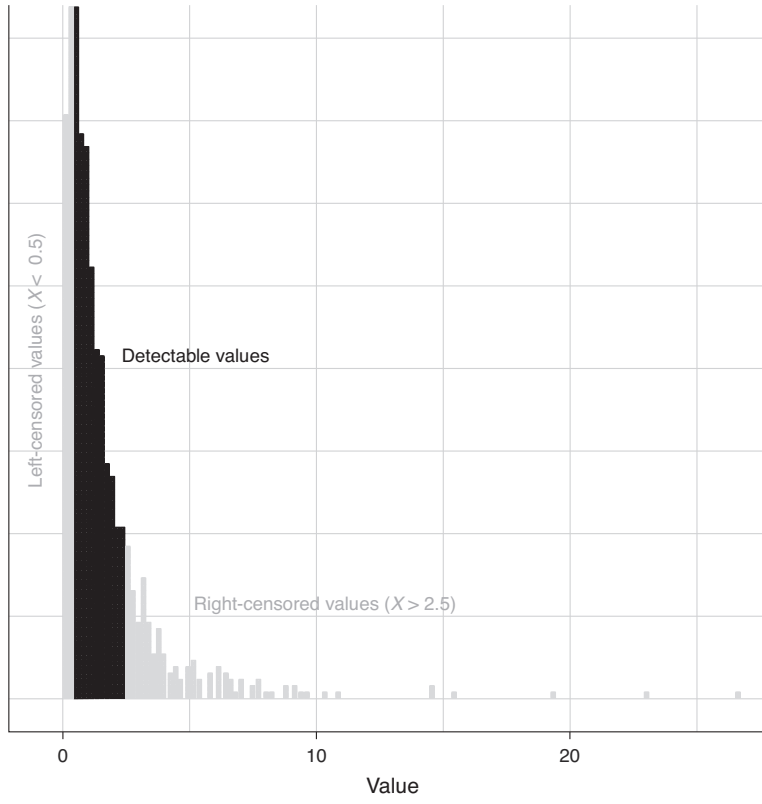


Figure 7.4 Left- and right-censored (a.k.a. truncated) example data. Values below the detection limit (here < 0.5) would be recorded as "0", whereas those above the capacity of the instrument (here > 2.5) would be recorded either as equal to "2.5" or "> 2.5" (grey points). Values that are detectable by our instrument would be recorded precisely (black points).

has met with universal approval, and hence the current statistical standard is censored regression, i.e., purpose-build methods for specific types of censored data (Helsel 2005; Bogaerts et al. 2017).

7.4.1 A case study: Atrazine in rivers and streams

Atrazine is an herbicide banned in the European Union since 2004, but it is the second-most commonly used agrochemical (after glyphosate) in the USA. The data we want to model are concentrations of atrazine in Midwestern USA rivers and streams (Helsel 2005); the detection limit of atrazine at the time these data were collected was $\approx 0.05 \mu$g/L.[10] We use these data to illustrate how such left-censored measurements affect models that consider predictors of atrazine concentrations that are related to geography (e.g., catchment area, streamflow rate), the environment (e.g., temperature or precipitation during the growing season), and management (e.g., days since planting, how atrazine is applied to crops, proportion of arable land in the river's catchment).

[10] Because of the serious health effects of atrazine at very low concentrations, development of methods to improve its detection is an active area of research. As of late 2022, atrazine can be detected in water samples in concentrations as low as 1 ng/L (50 times lower than could be detected in the 1990s when the data we use in this case study were collected), but reliably quantified only at concentrations ≥ 20 ng/L (Steffens et al. 2022).

A FIRST SET OF REGRESSION MODELS

We start with a simple question (and model): Is atrazine concentration in rivers and streams related to the catchment area—the area of land drained by the sampled river or stream? We know there is a limit to the concentration that we can detect, and the minimum value for atrazine concentration entered in the dataset (0.05 µg/L) is the detection limit (Figure 7.5).[11]

[11] This is not completely accurate. We provide here a lengthy description of the history of this dataset to justify what we actually did (in three lines of code in Code Box 7.5) to prepare the data for subsequent analysis. This exegesis provides an instructive illustration of the importance of data management, metadata, and provenance (e.g. Lerner et al. 2022) and their relationship to one another for ensuring the reliability of statistical workflows and analytical pipelines.

The `Recon` dataset we use in this case study of atrazine in rivers and streams is included in the `R` library `NADA` (Lee 2020; the acronym refers to the title of Helsel's (2005) book—*Nondetects and Data Analysis: Statistics for Censored Environmental Data*—and was created to accompany, and includes the methods and associated datasets used in, the first and the second (retitled) editions: Helsel 2005, 2012). The description of the `Recon` dataset in `NADA` identifies the detection limit (0.05 µg/L) from Chapter 12 of Helsel (2005); the dataset is introduced in both editions of the book in Exercise 12.1, and it is used again in both editions in Exercise 12.3. However, the dataset description in the Appendix: Datasets of Helsel (2005) calls 0.05 µg/L the "detection limit," whereas this is changed in the Appendix: Datasets of Helsel (2012) to the "reporting limit."

What's the difference between the reporting limit, the limit of detection, and yet another limit, the limit of quantification (Currie 1999)? The limit of detection, which is the term used most commonly when discussing censored environmental data, is "the minimum measured concentration of a substance that can be reported with 99% confidence that the measured concentration is distinguishable from method blank results" (40 CFR §136, Appendix B). The limit of quantification, also called the minimum level of quantitation or simply the "minimum level," is defined as "the lowest level at which the entire analytical system must give a recognisable signal and acceptable calibration point for the analyte. … It is calculated by multiplying the MDL [minimal detection limit] by 3.18 and rounding the result to the number nearest to (1, 2, or 5) $\times 10^n$, where n is an integer [appropriate to the MDL]" (EPA 2002, p. 32). Although the multiplier 3.18 is used routinely to determine the limit of quantification from the detection limit, the US Environmental Protection Agency (EPA) defined this multiplier only for quantifying concentrations of mercury (Hg) in water (EPA 2002). More generally, the EPA defines the "practical quantification limit" as "the lowest concentration of an analyte that can be reliably measured within specified limits of precision and accuracy during routine laboratory operation conditions" (EPA 2010, p. 1). However, this limit of quantification has been considered "problematic," because it had been determined for different analytes using three different and incompatible methods. Therefore, EPA (2010) proposed an alternative to the limit of quantification, the lowest concentration minimum reporting level (LCMRL) (a.k.a. the reporting limit). The EPA went on to define this LCMRL to be the lowest spiking concentration such that the probability of spike recovery in the 50% to 150% range is at least 99% (EPA 2010). Procedures, algorithms, and `R` code on GitHub are maintained by the EPA so that analysts can estimate reporting levels required for regulatory decision making.

To what does the minimum value for atrazine concentration in `Recon` refer? Is it the detection limit? The limit of quantification? Or the lowest concentration minimum reporting level? The description in `NADA` (Lee 2020) and the source it cites are in conflict, and it turns out that the source of the original data provided by Helsel (2005, 2012), and Lee (2020) does not help resolve the question. The putative source of the dataset itself—Mueller et al. (1997)—is given in the `NADA` documentation (Lee 2020), but Mueller et al. (1997) neither analysed atrazine concentrations (they analysed nitrate fertiliser concentrations) nor did they publish the dataset. Rather, Mueller et al. (1997) placed the origin of the dataset in earlier work by Donald Goolsby and his colleagues at the US Geological Survey (USGS) (Goolsby et al. 1991; Thurman et al. 1991, 1992), who had collected both the herbicide and the nitrate data but reported in those papers only the analysis of the herbicide data. Although we have not been able to locate the original data table in any of Goolsby's publications or in the online archives of the USGS (and it is unlikely that an electronic version of the data would have been archived anywhere in the early 1990s), Thurman et al. (1992) provided a detailed description of the analytical methods they used and their precision.

Thurman et al. (1992) used two methods to quantify atrazine concentration: immunoassays (ELISA) and gas chromatography/mass spectroscopy (GC/MS). At the time, GC/MS was the standard method, and Thurman et al. (1992) reported that their Hewlett-Packard Model 5890 GC with a 5970A mass selective detector had a detection limit of 0.05 µg/mL of atrazine. In contrast, their ELISA immunoassay run using microtiter plates had a limit of quantification of 0.05 µg/mL; the (presumably lower) detection limit was not specified. For values > 0.2 µg/mL, the ELISA and GC/MS results were strongly correlated ($r^2 = 0.86$). Because Thurman et al.'s (1992) attention was focused on whether measured concentrations of atrazine exceeded the regulatory maximum contaminant level of 3 µg/mL, and their tests gave neither false negatives nor false positives at concentrations > 0.2 µg/mL, there was

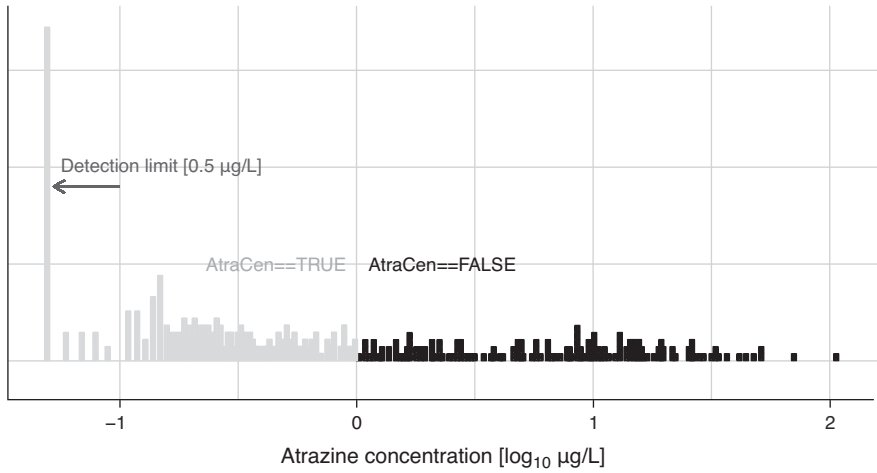


Figure 7.5 Distribution of values of atrazine concentration in the `Recon` data file. The grey points are those flagged as censored (`AtraCen`==TRUE; $N = 266$) in the data file, whereas the black points are those flagged as actual values ($N = 157$). The "detection limit" (see footnote 11) in the documentation is given as 0.05 µg/L, and indeed there are 47 values in the file with that value and none with smaller values.

We could simply ignore those values at the detection limit (model 1 in Code Box 7.5; Figure 7.6, black line) or use the values reported at the detection limit as if they are the actual values (model 2 in Code Box 7.5; Figure 7.6, light grey line), but both of those options clearly are inappropriate. Censored regression (Helsel 2012) is the standard way to handle such problems, but its use is limited to specific data structures (e.g., common distributions), types of models (e.g., no random effects), and model comparisons (e.g., standard methods of model selection cannot be used). Censored regression is a generalisation of tobit regression (Tobin 1958), which modifies the likelihood function so that the sampling probabilities

little cause for them to distinguish among different censoring limits in their paper. Indeed, in their key figure, they simply referred to 0.05 µg/mL of atrazine as "minimum value or detection limit."

All of this would be of only passing historical interest except that the `Recon` dataset contains a variable, `AtraCen`, that flags whether a value should be considered censored (i.e., at or below the relevant limit) or not. Curiously, `AtraCen` flags any value of atrazine concentration ≤ 1.0 µg/mL as being censored. This value clearly corresponds to none of the plausible limits (detection, quantification, reporting) nor to the maximum contaminant level. If we had thoughtlessly used this censor flag in our analyses, 63% of the observations of atrazine concentration would have been censored, even though they had a range of actual values much larger than either the detection limit or the limit of quantification of 0.05 µg/L (Figure 7.5).

Why are values < 1.0 µg/L flagged as censored in `Recon`? It turns out that Exercise 12.3 in Helsel (2005, 2012) asks the reader to "compute a logistic regression equation for predicting the probability of observing an atrazine concentration above 1 mg/L" (the typographic error here of mg/L instead of µg/L occurs twice in Exercise 12.3 in Helsel 2012). The variable `GE_1` [in the `recon.xls` file referred to in Helsel (2012)] has a value equal to 1 for all atrazine concentrations greater than 1 mg/L, and 0 otherwise. The value 1 normally is used to flag a censored value in a dataset, and this flag was first changed to a logical (TRUE/FALSE) variable, renamed as `AtraCen` in `Recon`, and then used incorrectly as the censor-indicator flag. Further confusion is engendered by the `NADA2` package, in which the included dataset `ReconLogistic`, corresponding directly to Exercise 12.3 in Helsel (2012), includes only predictor variables and the variable `GT_1` (renamed from `GE_1` in the exercise) as a binary variable for atrazine concentration > 1 (but not ≥ 1) µg/L. The actual concentration values are not provided in `ReconLogistic`, and `Recon` is not included in `NADA2`, either.

As we were interested in using the `Recon` dataset to explore censored regression of environmental data with a detection limit, we re-coded the censoring flag `AtraCen` in `Recon` so that it was true only if the value of `AtraConc` ≤ 0.05 µg/L.

Code Box 7.5: Modelling the atrazine data using different methods of regression

```
# Load required libraries
 library(NADA)      # Includes the atrazine dataset (Recon)
 library(NADA2)     # For censored regression function (censorreg)

# Load the dataset
 data(Recon)
 Recon$sqrtArea <- sqrt(Recon$Area) #Square-root transform the area
 attach(Recon)

# Deal with miscoding of the censoring flag (see footnote 11 in this Chapter)
Recon$AtraPHLevel <- Recon$AtraCen #Put existing AtraCen into AtraPHLevel
Recon$AtraCen <- FALSE
Recon$AtraCen[Recon$AtraConc<=.05] <- TRUE #Change AtraCen to TRUE if \leq 0.05

# Model 1. Analysis omitting the censored data:
fmNoCens <- lm(log10(AtraConc) ~ sqrtArea, data=Recon[Recon$AtraCen==FALSE,])
summary(fmNoCens)

|  Coefficients:
|              Estimate Std. Error t value Pr(>|t|)
| (Intercept)  0.080880   0.073374   1.102    0.271
| sqrt(Area)  -0.001056   0.001096  -0.964    0.336
|
| Residual standard error: 0.7852 on 374 degrees of freedom
| Multiple R-squared:  0.002476,  Adjusted R-squared:  -0.0001908
| F-statistic: 0.9285 on 1 and 374 DF,  p-value: 0.3359

# Model 2. Analysis with censored dataset to reported (lowest measured) value:
 fmLowlim <- lm(log10(AtraConc) ~ sqrtArea, data=Recon)
 summary(fmLowlim)

| Coefficients:
|               Estimate Std. Error t value Pr(>|t|)
| (Intercept) -0.1123846  0.0735967  -1.527    0.128
| sqrt(Area)  -0.0002306  0.0011067  -0.208    0.835
|
| Residual standard error: 0.8501 on 421 degrees of freedom
| Multiple R-squared:  0.0001031, Adjusted R-squared:  -0.002272
| F-statistic: 0.04341 on 1 and 421 DF,  p-value: 0.8351

# Model 3. Analysis assuming left-censored regression with a detection limit of
    0.05 and censored data flagged with variable AtraCen
 fmcen <- cencorreg(Recon$AtraConc, Recon$AtraCen, Recon$sqrtArea, LOG=T)
 summary(fmcen)

|                       Value Std. Error     z      p
| (Intercept)        -4.06e-01   1.89e-01 -2.14  0.032
| sqrt(Recon$Area)    1.57e-05   2.84e-03  0.01  0.996
```

```
| Log(scale)          7.68e-01   3.75e-02 20.46 <2e-16
|
| Scale= 2.16
|
| Gaussian distribution # Because data were log10-transformed
| Loglik(model)= -876.4   Loglik(intercept only)= -876.4
|         Chisq= 0 on 1 degrees of freedom, p= 1
| Number of Newton-Raphson Iterations: 3
| n= 423

detach(Recon)
```

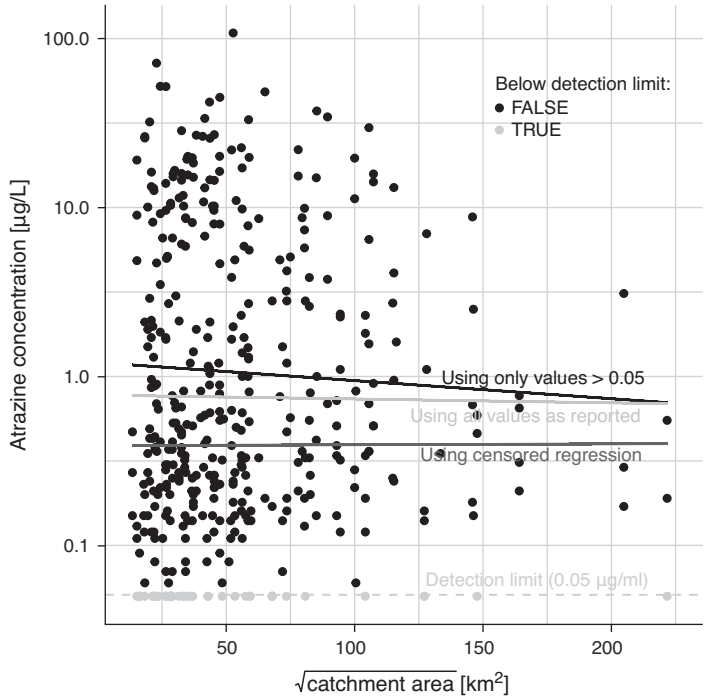


Figure 7.6 Atrazine concentrations in Midwestern USA rivers and streams as a function of catchment area. Values below the detection limit (0.05 µg/l) are shaded in grey for illustration. Regression lines indicate different ways of handling the problem: black line—regression using only values above the detection limit (47 observations ≤ 0.05 µg/L removed; remaining $N = 326$); light grey line—regression using all values as reported (retained all 47 values reported as equal to the detection limit of 0.05 µg/L at that value, $N = 423$); dark grey line—censored regression line. The code to fit these regression models is in Code Box 7.5.

depend on whether the observation is either greater than ($>$) or less than or equal to ($\leq$) the detection limit. We use Julian and Helsel's (2023) implementation of censored regression in R's `censorreg` function to model the atrazine data with a detection limit of 0.05 µg/L (model 3 in Code Box 7.5; Figure 7.6, dark grey line).

7.4.2 Estimating values below the detection limit

A value below the detection limit cannot be taken at face value. Although the tobit regression approach (Code Box 7.5 and Figure 7.6; Tobin 1958; Helsel 2012; Julian & Helsel 2023) is a

reasonable first option, there are others. In particular, if we had a justifiable and reproducible procedure to impute values below the detection limit, we could dispense with censored regression and analyse the imputed dataset using a much broader range of methods. However, imputation of data outside the range of observed values is a very different problem from imputation of values within the range of observed values. Simulation can be useful for addressing this problem (e.g., Hansen et al. 2015).

Here we compare four methods for estimating unobserved values below a detection limit and how these methods affect parameter estimates in a linear model. The four methods are: (1) omit from the analysis any value that is less than or equal to the detection limit; (2) replace each of them by half the detection limit; (3) replace each of them by the detection limit itself; or (4) use imputation to estimate the values. We use the predictors in the atrazine dataset (`Recon`) to simulate atrazine concentrations, censor the data with a fixed detection limit (0.05 µg/L), impute the censored values with these four different methods, and finally analyse the simulated data using the linear model we used to simulate the true (simulated and uncensored) data (Code Box 7.6). We note that like the original atrazine dataset, our simulated dataset had $\approx 10\%$ of the values below the detection limit (47 in the original `Recon` dataset, 50 in our simulated dataset).

In this single simulation, none of the methods came close to the truth for more than one or two parameter estimates. To decide whether this is a consistent pattern, we run many simulations (Code Box 7.7). For comparison, we also analyse the (simulated) raw data—those without a detection limit—to show that the analysis itself is not biased.

The result from multiple simulations is similar to that of the single simulation. No method could be said definitively to be the best. The results of the simulations using the different imputation methods for the effect of the square root of the catchment area (as used in Code Box 7.5 and Figure 7.6) are summarised in Figure 7.7. We conclude that replacing censored data by any reasonable value is unlikely to undo the censoring, even if it is estimated using a reliable method for data imputation (see §7.3).

7.4.3 Conclusions

Censoring itself is already a statistical challenge, requiring dedicated modelling approaches. Typically, it is challenging to combine censored data with more complex model structures (such as hierarchical or machine-learning models, but see Wang et al. 2019; Nagpal et al. 2021), unless one employs a fully Bayesian approach (Brilleman et al. 2020; McElreath 2020). But even using purpose-built regression models for censored data requires evaluating different approaches, as no single regression approach fits all censoring (Leung et al. 1997; Sun 2006; Fox 2015; Jones 2018). If we are going to use censored data in a workflow, we need to proceed very carefully and be completely transparent about our methods of imputation.

7.5 Pre-selecting predictors

Identifying predictors suggests clear causal mechanisms; these can be especially important in clinical trials, where we are looking for effective therapies, and in environmental conservation, where we would like to have predictable responses resulting from management strategies. It is thus common in these (and other) research settings to use interpretable GLMs in analysis, even when a machine-learning approach may be superior in prediction; the coefficients of a GLM are readily interpretable as causes, whereas we cannot readily find causes in the unlit corners of the opaque box that is any machine-learning model.

Code Box 7.6: A single simulated comparison among linear models with different methods of imputation

```
# Load required libraries
 library(NADA) # For Recon dataset
 library(mice) # For imputation

# Simulation
 set.seed(123)

# Define the "true" relationship between atrazine concentration and the
    predictor variables in Recon, with normal error
 attach(Recon)
 y <- 10^(-2 + .15*Temp + 0.015*PctCorn -0.004*Dyplant + 0.01*Pctl + 0.0003*
     Precip + 0.0015*sqrt(Area) -0.007*Applic + rnorm(423, 0, 0.5))

# Impose censoring, and impute data defined as below the detection limit
 detectlim <- 0.05   # This can be modified to explore different detection limits
 Recon$AtCo_NA <- ifelse(y < detectlim, NA, y) # Omits values below the detection
      limit
 Recon$AtCo_half <- ifelse(y < detectlim, 0.5*detectlim, y) # Replaces values
     below the detection limit with 50% of the detection limit
 Recon$AtCo_uplim <- ifelse(y < detectlim, detectlim, y) # Replaces values below
     the detection limit with the detection limit
 Recon$AtCo_impute <- complete(mice(data=Recon[, c(1:8, 12)], m=5, method="pmm",
     printFlag=F))$AtCo_NA # Imputes values below the detection limit using
     predictive mean matching

# Analyse and summarise the four imputed datasets with a simple additive linear
    model:
 coefs <- rbind(
  "omit"=coef(lm(log10(AtCo_NA) ~ Temp+PctCorn+Dyplant+Pctl+Precip+sqrt(Area)+
      Applic, data=Recon)),
  "half"=coef(lm(log10(AtCo_half) ~ Temp+PctCorn+Dyplant+Pctl+Precip+sqrt(Area)+
      Applic, data=Recon)),
  "upper"=coef(lm(log10(AtCo_uplim) ~ Temp+PctCorn+Dyplant+Pctl+Precip+
      sqrt(Area)+Applic, data=Recon)),
  "impute"=coef(lm(log10(AtCo_impute) ~ Temp+PctCorn+Dyplant+Pctl+Precip+
      sqrt(Area)+Applic, data=Recon)),
  "true"=c(-2, .15, .015, -.004, .01, .0003, .0015, -.007)
 )
coefs

|        (Intercept)   Temp PctCorn Dyplant   Pctl Precip sqrt(Area)  Applic
| omit       -1.9882 0.1499  0.0168 -0.0032 0.0099 0.0010     0.0012 -0.0118
| half       -2.2337 0.1529  0.0174 -0.0037 0.0109 0.0027     0.0011 -0.0120
| upper      -2.0310 0.1389  0.0157 -0.0034 0.0104 0.0026     0.0010 -0.0114
| impute     -1.7530 0.1146  0.0127 -0.0029 0.0100 0.0028     0.0012 -0.0106
| true       -2.0000 0.1500  0.0150 -0.0040 0.0100 0.0003     0.0015 -0.0070

detach(recon)
```

Code Box 7.7: A full simulation comparing among linear models with different methods of imputation

```
attach(recon)  # Dataset from NADA
set.seed(1234)
detectlim <- 0.05
coefsMany <- replicate(100, {
  y <- 10^(-2 + .15*Temp + 0.015*PctCorn -0.004*Dyplant + 0.01*Pctl + 0.0003*
      Precip + 0.0015*sqrt(Area) -0.007*Applic + rnorm(423, 0, 0.5))
  Recon$AtCo_raw <- y # use uncensored data for comparison
  y <- ifelse(y<detectlim, 0, y)
  Recon$AtCo_NA <- ifelse(y < detectlim, NA, y)
  Recon$AtCo_half <-  ifelse(y < detectlim, detectlim*.05, y)
  Recon$AtCo_uplim <- ifelse(y < detectlim, detectlim, y)
  Recon$AtCo_impute <- complete(mice(data=Recon[, c(1:8, 11)], m=5,
      method="pmm", printFlag=F))$AtCo_NA

  coefs <- rbind(
   "omit"=coef(lm(log10(AtCo_NA) ~ Temp+PctCorn+Dyplant+Pctl+Precip+sqrt(Area)+
       Applic, data=Recon)),
   "half"=coef(lm(log10(AtCo_half) ~ Temp+PctCorn+Dyplant+Pctl+Precip+
       sqrt(Area)+Applic, data=Recon)),
   "upper"=coef(lm(log10(AtCo_uplim) ~ Temp+PctCorn+Dyplant+Pctl+Precip+
       sqrt(Area)+Applic, data=Recon)),
   "impute"=coef(lm(log10(AtCo_impute) ~ Temp+PctCorn+Dyplant+Pctl+Precip+
       sqrt(Area)+Applic, data=Recon)),
   "raw"=coef(lm(log10(AtCo_raw) ~ Temp+PctCorn+Dyplant+Pctl+Precip+sqrt(Area)+
       Applic, data=Recon))
   )
})
round(apply(coefsMany, c(1,2), mean), 4)

|        (Intercept)   Temp PctCorn Dyplant   Pctl Precip sqrt(Area)  Applic
| omit       -1.6221 0.1228  0.0134 -0.0034 0.0092  2e-04     0.0011 -0.0067
| half       -1.9097 0.1410  0.0143 -0.0039 0.0100  6e-04     0.0013 -0.0070
| upper      -1.7291 0.1264  0.0130 -0.0037 0.0097  7e-04     0.0012 -0.0067
| impute     -1.4670 0.1073  0.0116 -0.0032 0.0092  5e-04     0.0010 -0.0063
| raw        -1.9944 0.1507  0.0155 -0.0040 0.0100  2e-04     0.0014 -0.0074

detach(recon)
```

Even with GLMs, however, when we work with a large number of diagnostic tools or management options, the number of plausible predictors, p, can become large relative to the number of subjects or observations, N, for even the largest of our datasets. For example, assuming only first-order interactions among predictors and possible quadratic effects leading to nonlinear responses,[12] the number of parameters in a GLM increases as

[12] That is, p predictors $+p$ quadratic terms $+\binom{p}{2}$ first-order interactions

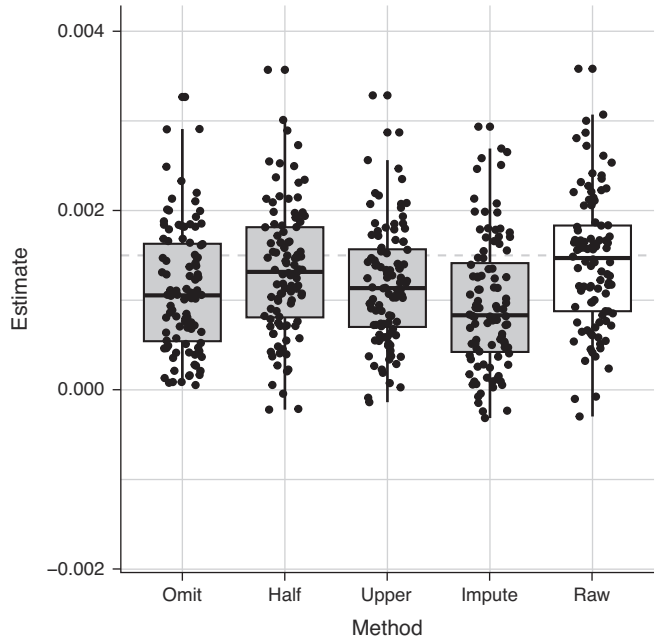


Figure 7.7 Estimates of effect of catchment area (square-root transformed) on atrazine concentrations for which four different methods were used to impute values to measurements below the detection limit: **Omit** values below the detection limit; replace each such value with **Half** the detection limit; replace each such value with the detection limit itself (**Upper**); and **Impute** each such value using predictive mean matching (see §7.3). The true value is indicated by the dashed line. Simulated values below the detection limit were assigned a value of NA. To demonstrate that without detection limit the analysis yields unbiased estimates, the estimates from fitting the model to the simulated **Raw** data are shown overlaying the white box.

$\binom{p}{2} + 2p$ with the number of predictors; even $p = 30$ predictors would require fitting nearly 600 parameters. That is a reasonable thing to do as long as $N \gg p$, but if we then want to use model selection (§6.1) to whittle the parameter set of this monstrous model down to an interpretable and, we hope, useful set of predictors, we may be in for a long wait.[13]

To get around this problem in an analytical workflow, researchers often try to identify a smaller set of relevant predictors that can be handled either with fewer data (especially if $N < p$ in the original dataset) or in less CPU time. Murray & Conner (2009) used simulation to evaluate simple approaches to determine a ranking of predictors, such as using variations among the correlations between predictors and outcomes. Clearly, this approach is unlikely to work if effects are non-linear or interactions between predictors are significant.

7.5.1 A case study: Diagnostic markers of breast cancer

To illustrate this point, we use a dataset on breast cancer diagnostics (Wolberg et al. 1995). For each of 569 images of biopsy samples, the shapes of the nuclei in the cells were quantified in three dimensions and with 10 shape measures, yielding 30 predictors per sample (methods detailed in Street et al. 1993). Instead of attempting to fit a model to these data, we use a simulation to explore whether variable-screening introduces bias. That is, our final result will be the difference from known true parameter values θ^* relative to the actual values:

[13] The help page for the `dredge` package states that the number of model evaluations is of order 2^p, so processing time scales exponentially with the number of parameters.

Code Box 7.8: Simulating a complicated regression

```
# Read in the data (local copy)
 BCWD <- read.csv(file="BCWD.csv", stringsAsFactors = T, row.names = 1)
 BCWD$Diagnosis <- ifelse(BCWD$Diagnosis == "B", 0, 1) # Recode diagnosis as
     binary variable 0 (benign) and 1 (malign)
 BCWDs <- as.data.frame(scale(BCWD)) # Standardize predictors
 BCWDs[,1] <- BCWD[,1] # Keep Diagnosis as 0s and 1s

# Simulation
 set.seed(123)
 picks <- sample(30, size=6) # select random predictors to use
 colnames(BCWD)[picks]

| [1] "area2"           "concave_points2" "perimeter2"      "texture1"
|     "symmetry1"       "concavity2"

# An imaginary and simulated regression structure
 fDGM <- glm(Diagnosis ~ (area2 + concave_points2 + perimeter2 + texture1 +
     symmetry1)^3 + I(concavity2^2), family=binomial, data=BCWDs)

 beta <- coef(fDGM)
 X <- model.matrix(fDGM)
```

relative bias $= \frac{\hat{\theta}-\theta^*}{\max(|\hat{\theta}|,|\theta^*|)}$. A value of 0 indicates no bias, a value of ± 1 indicates that at least one incorrect predictor has been selected, and all other values indicate a bias as multiples of the true values.

7.5.2 Simulating a complex relationship between diagnosis and predictors

We start by randomly defining a subset of six predictors that we want to use, and some regression structure. Then, we fit the model to the data and extract its coefficients, β, and the model matrix, X (Code Box 7.8). With this definition of the true underlying model we can start the simulations.

SIMULATE PRE-SELECTION OF PREDICTORS USING CORRELATIONS AND A GLM

First, we simulate a dataset as a realisation of a Bernoulli trial with coefficients and predictors as defined above. Then we screen for the k predictors that are most strongly correlated with the response. With these pre-selected predictors, we fit a logistic regression, which also includes quadratic terms and first-order interactions among predictors. Finally, we use stepwise backward model selection to reduce the full model to a final model (Code Box 7.9).

Comparing this model with the known true model (with predictors `area2`, `concave_points2`, `perimeter2`, `texture1`, `symmetry1`, and `concavity2` from Code Box 7.8), we can immediately see that none of the variables in the pre-selected model are correct (poor specificity), and none of the parameters in the true model were detected (poor sensitivity).

Code Box 7.9: Simulating pre-selection using correlations

```
set.seed(123)

# The simulation uses X and beta from Code Box 7.8
sim <- cbind.data.frame(Diagnosis=rbinom(569, prob=plogis(X %*% beta), size=1),
    BCWDs[,-1])

# Two-step GLM: first, pick the k best predictors (here, we set k = 5); then
    step through the simulation
k <- 5

# Screen the predictors for their correlation with the response
sort(abs(cor(sim)[,1]))

|         symmetry2        smoothness2           texture2 fractal_dimension2
|        0.01378699         0.02532776         0.05095281         0.06966615
|
|fractal_dimension1 fractal_dimension3         concavity2       compactness2
|        0.07364657         0.17500188         0.21915355         0.24798278
|
|       smoothness3          symmetry3        smoothness1          symmetry1
|        0.25954050         0.26170925         0.27542691         0.31613842
|
|          texture3           texture1    concave_points2       compactness3
|        0.38143464         0.39335486         0.40479538         0.41410395
|
|      compactness1         concavity3         perimeter2              area2
|        0.49084606         0.50331420         0.53993970         0.54009014
|
|           radius2         concavity1    concave_points3              area1
|        0.56974977         0.59487496         0.65049661         0.67295691
|
|             area3    concave_points1            radius1         perimeter1
|        0.68145546         0.68733427         0.69124178         0.69823333
|
|        perimeter3            radius3          Diagnosis
|        0.70779994         0.71349513         1.00000000

varsC <- names(sort(abs(cor(sim)[,1]), decr=T))[2:(k+1)] # Sort the predictors
    in descending order
RHS1C <- paste0(varsC, collapse=" + ") # First-order interactions
RHS2C <- paste0(varsC, collapse="^2) + I(") # Quadratic terms
fC <- formula(paste0("Diagnosis ~ (", RHS1C, ")^2 + I(", RHS2C, "^2)"))
    # Formula for GLM
fglmC <- glm(fC, family=binomial, data=sim)
fstepC <- step(fglmC, k=log(569), trace=F) # use BIC
summary(fstepC)
```

```
| Call:
| glm(formula = Diagnosis ~ radius3 + perimeter3 + concave_points1,
|     family = binomial, data = sim)
|
| Coefficients:
|                 Estimate Std. Error z value Pr(>|z|)
| (Intercept)      -0.4226     0.1428  -2.959  0.00309 **
| radius3           6.9119     1.5304   4.516 6.29e-06 ***
| perimeter3       -5.0577     1.5873  -3.186  0.00144 **
| concave_points1   1.6524     0.3078   5.369 7.94e-08 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| (Dispersion parameter for binomial family taken to be 1)
|
|     Null deviance: 755.49  on 568  degrees of freedom
| Residual deviance: 335.70  on 565  degrees of freedom
| AIC: 343.7
```

SIMULATE PRE-SELECTION OF PREDICTORS USING MACHINE LEARNING

We did the same simulation using a machine-learning algorithm (random forest) to see if it can pre-select at least some of the correct predictors (Code Box 7.10).

Even using a random-forest model to pre-select the predictors, we only got two predictors—one correct one and one incorrect one—but none of the other predictors.

DETERMINING BIAS

To compare among our models and see whether they are biased, we need to save the coefficient estimates of the true model and the two models based on pre-selected variables resulting from pre-selection. Because the coefficients differ for each run, this takes a bit more data wrangling (Code Box 7.11). What this code does is to compute a relative bias: 0 is ideal (unbiased). A value $\approx |1|$ indicates that the estimates are virtually all wrong, either estimated as 0 when they should not be, or estimated as different from 0 when they should be 0.

The results from the single simulations in Code Boxes 7.9 and 7.10 are strongly biased.

REPEATED SIMULATION OF PRE-SELECTION

We use repeated simulations to see if this conclusion is robust. In our multiple simulations, we also compute the relative bias at each step.

After a hundred repetitions, virtually all pre-selected parameters estimated were wrong. That does not mean that the model is a poor predictor, only that it completely fails at inferring the correct model parameters (Figure 7.8).

CONCLUSIONS

There are many good reasons why pre-selection failed in our simulation but may actually succeed in another example. Correlation among the predictors in the breast-cancer dataset ranged from −0.3 to 0.99 (between radius and perimeter in each dimension), so either of the correlated predictors could have been selected without loss of fit. However, other real-world datasets may well exhibit similar collinearity, and random-forest models will spread the estimated importance over correlated predictors (§6.2 in Chapter 6).

Code Box 7.10: Simulating pre-selection using random forest and a GLM

```
# Load required libraries
 library(ranger) # For random forest models

# Run the random forest on the simulated dataset ("sim") from Code Box 7.9
 fr <- ranger(Diagnosis ~., data=sim, importance="impurity")
 sort(importance(fr), decr=T)[1:k]

|     area3    radius3      area2 perimeter3      area1
| 15.181869  14.483436  13.944780   9.651902   7.980150

 varsR <- names(sort(importance(fr), decr=T)[1:k])
 RHS1R <- paste0(varsR, collapse=" + ") # First-order interactions
 RHS2R <- paste0(varsR, collapse="^2) + I(") # Quadratic terms
 fR <- formula(paste0("Diagnosis ~ ", RHS1R, "+ I(", RHS2R, "^2)")) # Formula for
      GLM
 fglmR <- glm(fR, family=binomial, data=sim)
 fstepR <- step(fglmR, k=log(569), trace=F) # use BIC
 summary(fstepR)

| Call:
| glm(formula = Diagnosis ~ area3 + area2, family = binomial, data = sim)

| Coefficients:
|             Estimate Std. Error z value Pr(>|z|)
| (Intercept)   0.4197     0.2067   2.030   0.0423 *
| area3         2.4186     0.4125   5.864 4.52e-09 ***
| area2         3.1764     0.6572   4.833 1.34e-06 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| (Dispersion parameter for binomial family taken to be 1)
|
|     Null deviance: 755.49  on 568  degrees of freedom
| Residual deviance: 333.77  on 566  degrees of freedom
| AIC: 339.77
```

Because we allowed for statistical interactions to be relevant in our model (Code Box 7.8), correlations with a predictor may severely impair measures of variable importance (Murray & Conner 2009). If the true effect was only a linear additive effect, then correlation could work better in pre-selecting predictors. Although many researchers fit simple additive models to their data, it is unlikely that many real-world causal relationships are simply additive.

Finally, we imposed a rather strict selection by setting $k = 5$. With a larger selected set of predictors, different models would have been selected, possibly decreasing bias.[14] Still, we were surprised that seemingly none out of our 200 pre-selection simulations (100 for

[14] We encourage you to try this by modifying the *k* values in Code Boxes 7.8–7.11.

Code Box 7.11: Collate coefficient estimates from Code Boxes 7.8–7.10

```
# Set up a matrix with true (from Code Box 7.8) and estimated (from Code Box
    7.9) coefficients:
paraEsts <- t(beta)
paraEsts <- rbind(paraEsts, rep(0, ncol(paraEsts)))
for (i in 1:length(coef(fstepC))){
  putthere <- which(colnames(paraEsts) == names(coef(fstepC))[i])
  if (length(putthere) == 0) { # If parameter name does not yet exist
    paraEsts <- cbind(paraEsts, c(0, coef(fstepC)[i]))
    colnames(paraEsts)[ncol(paraEsts)] <- names(coef(fstepC))[i]
  } else { # if parameter name does exist
    paraEsts[2, putthere] <- coef(fstepC)[i]
  }
}

# Add randomForest-preselection coefficients (from Code Box 7.10) to the
    paraEsts matrix:
paraEsts <- rbind(paraEsts, rep(0, ncol(paraEsts)))
for (i in 1:length(coef(fstepR))){
  putthere <- which(colnames(paraEsts) == names(coef(fstepR))[i])
  if (length(putthere) == 0) { # If parameter name does not exist yet
    paraEsts <- cbind(paraEsts, c(0, 0, coef(fstepR)[i]))
    colnames(paraEsts)[ncol(paraEsts)] <- names(coef(fstepR))[i]
  } else { # if parameter name does exist
    paraEsts[3, putthere] <- coef(fstepR)[i]
  }
}
rownames(paraEsts) <- c("true", "cor", "rf")

# Compute average relative bias
mean(abs((paraEsts[2,] - paraEsts[1,]) / pmax(abs(paraEsts[2,]),
    abs(paraEsts[1,]))), na.rm=T) # True vs. correlation-derived parameter
    estimates

| [1] 1.010055

mean(abs((paraEsts[3,] - paraEsts[1,]) / pmax(abs(paraEsts[3,]),
    abs(paraEsts[1,]))), na.rm=T) #True vs. random forest-derived parameter
    estimates

| [1] 0.9825349
```

each pre-selection method) selected any of the correct predictors. This result is in line with what we found when exploring model selection (§6.1 in Chapter 6), which also generally introduced bias into model structure and estimates. Continuing analysis in a workflow with such biased models, as simulated here, will only exacerbate the problem. Our conclusion, therefore, is to avoid at all costs pre-selection of predictors in statistical workflows.

Code Box 7.12: Multiple simulations of pre-selection of predictors and their relative biases

```
# Load required libraries
 library(ranger) # For random forest model

# Simulations
 set.seed(123)
 relBiasSim <- replicate(100, { # 100 Simulations
     sim <- cbind.data.frame(Diagnosis=rbinom(569, prob=plogis(X %*% beta),
         size=1), BCWDs[,-1])

    # Choose predictors based on correlations (as in Code Box 7.9)
     k <- 5
     varsC <- names(sort(abs(cor(sim)[,1]), decr=T))[2:(k+1)]
     RHS1C <- paste0(varsC, collapse=" + ") # First-order interactions
     RHS2C <- paste0(varsC, collapse="^2) + I(") # Quadratic terms
     fC <- formula(paste0("Diagnosis ~ (", RHS1C, ")^2 + I(", RHS2C, "^2)"))
         # Formula for GLM
     fglmC <- glm(fC, family=binomial, data=sim)
     fstepC <- step(fglmC, k=log(569), trace=F)

    # Set up a matrix with true and estimated coefficients
     paraEsts <- t(beta)
     paraEsts <- rbind(paraEsts, rep(0, ncol(paraEsts)))
     for (i in 1:length(coef(fstepC))){
         putthere <- which(colnames(paraEsts) == names(coef(fstepC))[i])
         if (length(putthere) == 0) { # If parameter name does not yet exist
             paraEsts <- cbind(paraEsts, c(0, coef(fstepC)[i]))
             colnames(paraEsts)[ncol(paraEsts)] <- names(coef(fstepC))[i]
         } else { # If parameter name does exist
             paraEsts[2, putthere] <- coef(fstepC)[i]
         }
     }

    # Choose predictors using random forest model (as in Code Box 7.10)
     fr <- ranger(Diagnosis ~., data=sim, importance="impurity")
     varsR <- names(sort(importance(fr), decr=T)[1:k])
     RHS1R <- paste0(varsR, collapse=" + ") # First-order interactions
     RHS2R <- paste0(varsR, collapse="^2) + I(") # Quadratic terms
     fR <- formula(paste0("Diagnosis ~ ", RHS1R, "+ I(", RHS2R, "^2)")) # Formula
         for GLM
     fglmR <- glm(fR, family=binomial, data=sim)
     fstepR <- step(fglmR, k=log(569), trace=F)

    # Add the coefficients to the matrix
     paraEsts <- rbind(paraEsts, rep(0, ncol(paraEsts)))
     for (i in 1:length(coef(fstepR))){
         putthere <- which(colnames(paraEsts) == names(coef(fstepR))[i])
```

```
            if (length(putthere) == 0) { # if parameter name does not exist yet
                paraEsts <- cbind(paraEsts, c(0, 0, coef(fstepR)[i]))
                colnames(paraEsts)[ncol(paraEsts)] <- names(coef(fstepR))[i]
            } else { # if parameter name does exist
                paraEsts[3, putthere] <- coef(fstepR)[i]
            }
        }

      # Calculate bias
       c(mean(abs((paraEsts[2,] - paraEsts[1,]) / pmax(abs(paraEsts[2,]),
            abs(paraEsts[1,]))), na.rm=T),
       mean(abs((paraEsts[3,] - paraEsts[1,]) / pmax(abs(paraEsts[3,]),
            abs(paraEsts[1,]))), na.rm=T))
  })
  rowMeans(relBiasSim)

| [1] 1.0017143 0.9869893
```

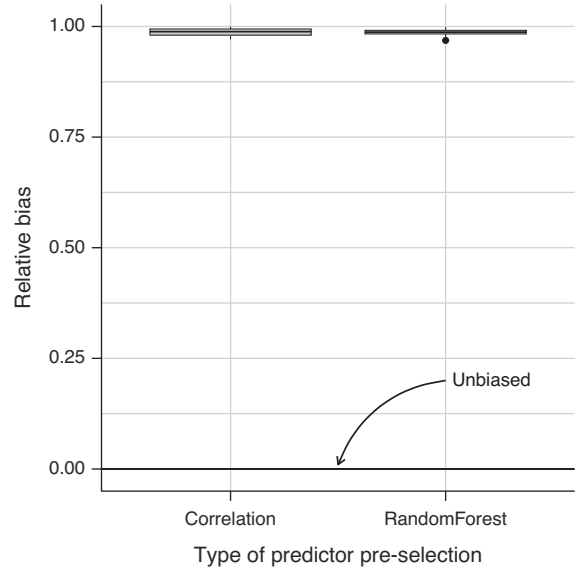


Figure 7.8 Bias in parameter estimates after variable pre-selection. *Left*, using correlation of predictor with response; *right*, using importance in the random-forest model as the criterion.

7.6 Regression on residuals

Workflows consist of steps building on the results of a previous step. If the previous step is a statistical analysis, and the next step uses the residuals from the first step, then we know that estimates in the second step will be biased (Darlington & Smulders 2001).[15] Imagine a longer workflow, in which at some point the analyst wants to remove a known bias in the

[15] Interestingly, the authors wrote (p. 599): "The flaws of residual analysis are widely known among statistical specialists, but none of those we asked were able to point to a paper describing these flaws to a general scientific

data collection by regressing it against the observations. Then the residuals of this "correction" cannot be used anymore as an unbiased sample simply because more often than not the correction term is correlated with the predictors in the next step. As this correlation increases, any effect of the actual predictor already has been corrected for in the previous step.

Freckleton (2002) wrote a paper that attempted to bring this practice to a stop (at least among ecologists). His paper was motivated by Darlington & Smulders's (2001) one, and he presented the results of a simulation illustrating the bias incurred by regression of residuals, and comparing it to multiple regression (i.e., combining both steps into one). Since Freckleton's (2002) paper was published before code-sharing became common practice, we rework his code here (Code Box 7.13) and repeat the simulation.

REGRESSION ON RESIDUALS VERSUS MULTIPLE REGRESSION

We simulate the simple situation of two predictors with a predefined (and manipulable) correlation between them; the response variable is a linear combination of the predictors: $Y = \beta_0 + \beta X_1 + \beta X_2 + \varepsilon$.[16] To be able to generate any amount of correlation between X_1 and X_2, they are generated from a multivariate normal distribution, with variance = 1 and covariance ranging from 0.1 to 0.99.[17] We then analyse the data once as a two-step procedure, the regression on residuals that we want to demonstrate is wrong, and then using multiple regression, which should yield the correct parameter estimates (Code Box 7.13).

The correct estimates of β_1 and β_2 in the true model, 0.2, as defined in line 11, are estimated incorrectly (as 0.11) by the regression on residuals (model `fm2`), but estimated correctly (as ≈ 0.2) in the multiple regression (model `fm3`).

THE MAGNITUDE OF THE COVARIANCE MATTERS

Next, we repeat the simulation with 10 different covariance values (Code Box 7.14). Unsurprisingly, both methods become increasingly unreliable as the covariance increases.

FULL SIMULATION

Finally, we repeat the simulation in Code Box 7.14 many times to see if the results are consistent (Code Box 7.15). We first use a small sample size ($N = 50$) and small effect ($\beta_1 = \beta_2 = 0.2$), and follow that with a larger sample size ($N = 200$) and larger effect ($\beta_1 = \beta_2 = 0.7$). Note that we multiply the covariance matrix by 0.05 to recreate the large spread reported by Freckleton (2002).

The results of our simulations (Code Box 7.15, Figure 7.9) nicely reproduce the results of Freckleton (2002). As the correlation between X_1 and X_2 increases from 0.2 to 0.7, the bias in the estimate of β_2 also increases. In contrast, multiple regression estimates become increasingly uncertain, but remain unbiased (see also Morrissey & Ruxton 2018 for this latter point).

CONCLUSIONS

Regression on residuals not only yields biased estimates for correlated predictors, but also is too optimistic (smaller confidence interval) about the correctness of this biased estimate.

---

audience." This quote suggests that even such simple facts are not necessarily textbook knowledge (this is the reverse of folklore; see Chapter 6). Or maybe statisticians consider them too obviously wrong. Yet Darlington & Smulders (2001) cited several studies from that year alone that committed this error.

[16] Note that Freckleton (2002) chose to give both predictors the same effect on Y, but the effect differed in the two later simulations: $\beta_1 = \beta_2 = 0.2$ and $\beta_1 = \beta_2 = 0.7$, respectively.

[17] As we set the variance equal to 1, the covariance equals the correlation. This remains the case when we later rescale the entire variance-covariance matrix, multiplying it by 0.05.

Code Box 7.13: A single simulation of regression on residuals versus multiple regression

```
# Load required libraries
 library(mvtnorm) # For multivariate normal distribution functions

# Simulate the predictors and their variance-covariance structure
 set.seed(123)
 N <- 50
 X <- rmvnorm(N, mean=c(0,0), sigma=matrix(c(1, 0.7, 0.7, 1), 2, 2))
 cor(X)

|           [,1]      [,2]
| [1,] 1.0000000 0.7036116
| [2,] 0.7036116 1.0000000

# Define the two variables and fix the parameters (beta1 = beta2 = 0.2)
 x1 <- X[,1]
 x2 <- X[,2]
 y = 1 + .2*x1 + .2*x2 + runif(N) # The "true" (simulated) model

# Sequential regression: First get residuals, then regress on residuals
 fm1 <- lm(y ~ x1) # Regression only on the first predictor, x1
 fm2 <- lm(residuals(fm1) ~ x2) # Regression using as predictors the residuals of
       the first regression and x2
 summary(fm2)

| Call:
| lm(formula = residuals(fm1) ~ x2)
|
| Residuals:
|     Min      1Q  Median      3Q     Max
| -0.5941 -0.1418  0.0112  0.1975  0.6960
|
| Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept) -0.01203    0.04072  -0.295  0.76892
| x2           0.11257    0.04167   2.701  0.00951 **
|---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| Residual standard error: 0.2862 on 48 degrees of freedom
| Multiple R-squared:  0.132, Adjusted R-squared:  0.1139
| F-statistic: 7.298 on 1 and 48 DF,  p-value: 0.009514

# Multiple regression as the alternative
 fm3 <- lm(y ~ x1 + x2)
 summary(fm3)

| Call:
```

```
| lm(formula = y ~ x1 + x2)
|
| Residuals:
|      Min       1Q   Median       3Q      Max
| -0.50376 -0.20287  0.01656  0.21488  0.46037
|
|Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept)  1.50070    0.03818  39.308  < 2e-16 ***
| x1           0.27371    0.06366   4.299 8.57e-05 ***
| x2           0.22294    0.05467   4.078 0.000174 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.2668 on 47 degrees of freedom
| Multiple R-squared:  0.7158,  Adjusted R-squared:  0.7038
| F-statistic:  59.2 on 2 and 47 DF,  p-value: 1.44e-13
```

For extremely correlated predictors, this approach very clearly suggests that no effect of X_2 exists, whereas multiple regression (correctly) literally goes off the chart in its uncertainty estimation (Figure 7.9, right panel).

The result of our simple simulation raises a red flag to watch out for in statistical workflows: beware of sequential statistical modelling steps. Detrending also is a preparatory step in an analytical pipeline: does it cause the same problems as demonstrated here? Not necessarily. Because detrending a time series does not use information about the predictors, it need not suffer from the same type of bias (see §6.5 in Chapter 6).

7.7 Error propagation

The last topic we need to cover before building a complete workflow is how errors or uncertainty propagate through analysis. When we report results from a set of observations or experiments, we should not be satisfied with only a numerical estimate for a parameter or test statistic of interest; we also should require a quantitative measure of its uncertainty. The two main reasons for wanting to know the uncertainty in our estimates are first, for statistical inference (we need to know how likely a value could have arisen by chance; the larger the uncertainty, the higher the probability it is not statistically significant); and second, for decision making (as we cannot put too much weight on a statement that is hugely uncertain). In thinking about error propagation, our goal is to have a reasonable idea about how the variability of our estimates can be affected by uncertainty.

Determining how uncertainty propagates is mathematically challenging. Paraphrasing an unnamed physicist, we usually spend 10% of our time deriving a formula and 90% of our time estimating its error. This is true, by and large, for analytical approximations of uncertainty.[18] Another approach, computational rather than analytical, is bootstrapping; if we want to quantify the effect of noise in the data on our analysis, we can repeat the entire analysis many times on data sampled with replacement to yield bootstrapped estimates

[18] The mathematics of error estimation and propagation go back to investigations by Carl Friedrich Gauss (Gauss 1816) that were then elaborated by Helmert (1875, 1876); hence it often is referred to as "Gaussian error propagation" or "Gauss-Helmert theory" (for an in-depth review, see Sheynin 1995).

Code Box 7.14: Testing the effect of different covariance values on regression on residuals versus multiple regression

```
# Load required libraries
 library(mvtnorm) # For multivariate normal distribution functions

 cova <- c(.1, .2, .3, .4, .5, .6, .7, .8, .9, .99) # Ten different covariances
 coefs <- matrix(NA, ncol=2, nrow=10) # Create a matrix in which to store the
     results
 colnames(coefs) <- c("resReg", "multReg")

# Simulation
 set.seed(123)
 for (i in 1:10){
   X <- rmvnorm(N, mean=c(0,0), sigma=matrix(c(1, cova[i], cova[i], 1), 2, 2))
   x1 <- X[,1]
   x2 <- X[,2]
   y = 1 + .2*x1 + .2*x2 + runif(N) # The "true" model

  # Regression on residuals
   fm1 <- lm(y ~ x1)
   coefs[i, 1] <- coef(lm(residuals(fm1) ~ x2))[2]

  # Multiple regression
   coefs[i, 2] <- coef(lm(y ~ x1 + x2))[3]
 }
 coefs # Comparison of the results (estimated value of beta2)

|             resReg     multReg
|  [1,]  0.243281115  0.24542367
|  [2,]  0.191398656  0.20408179
|  [3,]  0.164125449  0.19432754
|  [4,]  0.202662721  0.21365443
|  [5,]  0.184705485  0.23156336
|  [6,]  0.074420332  0.13563373
|  [7,]  0.050552826  0.12317649
|  [8,]  0.046034939  0.18415448
|  [9,]  0.008570341  0.05239294
| [10,] -0.004422754 -0.22590620
```

(Efron 1982). Last, because Bayesian inference uses distributions for any quantity—data or parameters—it naturally propagates any uncertainty through the analysis and the uncertainty is reflected in the posterior distributions (Lista 2023).

We briefly review these approaches here, but we only use bootstrapping in our worked example. The complexity of the analytical solution for our example is not worth the effort, and a Bayesian implementation is incompatible with the machine-learning approach we use.

Code Box 7.15: Simulations of regressions on residuals versus multiple regression for different sample sizes and effect sizes

```
# Load required libraries
 library(mvtnorm) # For multivariate normal distribution functions

# Small sample size (N=50) and small effect size (beta1 = beta2 = 0.2)
 N <- 50
 set.seed(123)
 coef.arr <- replicate(1000, { # 1000 replicate simulations
   coefs <- matrix(NA, ncol=2, nrow=10)
   colnames(coefs) <- c("resReg", "multReg")
   for (i in 1:10){
     X <- rmvnorm(N, mean=c(0,0), sigma=matrix(0.05*c(1, cova[i], cova[i], 1), 2,
          2)) # Multiply the covariance c(.) by 0.05 to get Freckleton's spread
     x1 <- X[,1]
     x2 <- X[,2]
     y = 1 + .2*x1 + .2*x2 + runif(N)
     fm1 <- lm(y ~ x1)
     coefs[i, 1] <- coef(lm(residuals(fm1) ~ x2))[2]
     coefs[i, 2] <- coef(lm(y ~ x1 + x2))[3]
   }
 })

# Large sample size (N=200) and large effect size (beta1 = beta2 = 0.7)
 N <- 200
 set.seed(123)
 coef.arr <- replicate(1000, { # 1000 replicate simulations
   coefs <- matrix(NA, ncol=2, nrow=10)
   colnames(coefs) <- c("resReg", "multReg")
   for (i in 1:10){
     X <- rmvnorm(N, mean=c(0,0), sigma=matrix(0.05*c(1, cova[i], cova[i], 1), 2,
          2)) # Multiply the covariance c(.) by 0.05 to get Freckleton's spread
     x1 <- X[,1]
     x2 <- X[,2]
     y = 1 + .7*x1 + .7*x2 + runif(N)
     fm1 <- lm(y ~ x1)
     coefs[i, 1] <- coef(lm(residuals(fm1) ~ x2))[2]
     coefs[i, 2] <- coef(lm(y ~ x1 + x2))[3]
   }
 })
```

7.7.1 Propagation of uncertainty and its analytical approximation

We typically describe a variable of interest by its expectation (i.e., its mean) and variance. An (observed) random variable x thus has mean $E(x) = \bar{x}$ and variance $= \text{var}(x)$. The variance of x feeds through to any computation that uses $\bar{x}$. That is, if we multiply $\bar{x}$ by a factor k, the variance of $k\bar{x}$ has to be recomputed: $\text{var}(k\bar{x}) = k^2\text{var}(x)$.

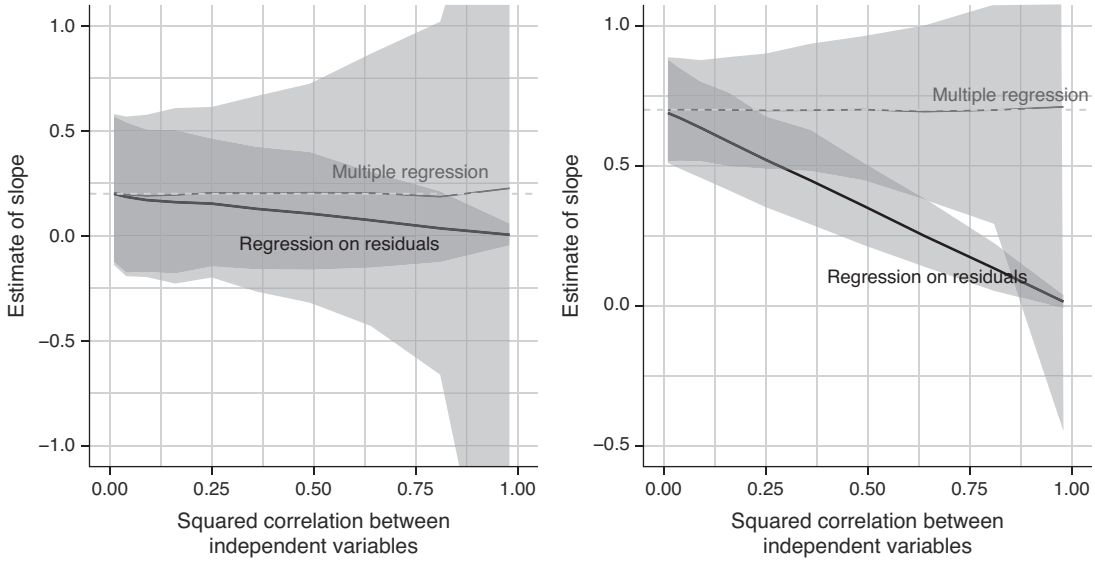


Figure 7.9 Estimated values of the coefficient β_2 in regression on residuals (black line) and multiple regression (grey line) for different sample sizes and effect sizes. *Left*: $N = 50,\ \beta = 0.2$; *right*: $N = 200,\ \beta = 0.7$. Shaded areas span the 0.025–0.975 quantiles. The true value is indicated by the dashed horizontal line. Note that this figure is equivalent to Freckleton's (2002) Figures 1a and 1d overlaid on his Figures 2a and 2d.

For more complicated functions f, the analytical equation for $\text{var}(f(x))$ becomes more tedious (Wikipedia Contributors 2023). For example, for $f = \frac{\bar{x}_A}{\bar{x}_B}$, $\text{var}(f) = f^2 \cdot \left(\frac{\sigma^2_{x_A}}{\bar{x}_A} + \frac{\sigma^2_{x_B}}{\bar{x}_B} - 2\frac{\sigma_{x_A x_B}}{\bar{x}_A \bar{x}_B} \right)$, where $\sigma_{x_A x_B}$ is the covariance of x_A and x_B. Such an equation takes getting used to, even if it makes intuitive sense: the variances are scaled by their means, and the stronger x_A and x_B are correlated, the more their errors cancel (because f is a ratio of x_A and x_B). The same equation for $f = \bar{x}_A \bar{x}_B$ would replace $-2\frac{\sigma_{x_A x_B}}{\bar{x}_A \bar{x}_B}$ with $+2\frac{\sigma_{x_A x_B}}{\bar{x}_A \bar{x}_B}$.

For more than two variables, the situation gets even more challenging, since each one of them has a variance. The entries are correlated, thus exhibiting covariance:

$$\Sigma_x = E\{(x - \bar{x}) \otimes (x - \bar{x}))\}, \tag{7.1}$$

where $\otimes$ symbolises the outer product: each value of the length-N vector on the left $(x - \bar{x})$ is multiplied with every value of the vector on the right $(x - \bar{x})$, yielding the $N \times N$ variance-covariance matrix Σ_x.

Consider a function f that is defined as a linear function of the elements of x: $f(x) = a + bx_1 + cx_2 + \ldots + nx_n = Bx$. This is another way of writing the predicted value of our model, as $f(x)$ is the result of fitting a (in this case linear) model to observations y. Then the variance-covariance matrix of $f(x)$ is computed as:

$$\Sigma_f = B\Sigma_x B^\top. \tag{7.2}$$

In other words, the error on a prediction from model $f(x)$ depends on the covariance matrix Σ_f, which in turn depends on the way the uncertainties in the different x combine or propagate.

The situation is still more complicated if f is a non-linear function, as would be the case for many GLMs or machine-learning models. For only a few cases can an analytical expression be derived for the variance of x. In most cases, however, we need to use a Taylor expansion, evaluated for x. An approximation, after dropping everything after the first term, yields a very similar expression, but now using the Jacobian matrix J instead of the variance-covariance matrix of x:

$$\Sigma_f = J\Sigma_x J^\top. \tag{7.3}$$

For most non-linear functions f, the Jacobian matrix is, by definition, determined by taking partial derivatives of $f(\cdot)$, which are then evaluated in x. This is also referred to in the statistical literature as the "Delta method." There also are two simpler ways to estimate Σ, provided we assume that the main source of uncertainty is the data and that all steps in a workflow depend on decisions we take based on these data.

7.7.2 Monte Carlo simulation

One common way to represent the effect of uncertainty on an estimated value is to do a forwards simulation with randomly drawn values.[19] More precisely, for our function $f(x)$ with parameters B, we generate a realisation B' by drawing, for each value B_i, from the distribution that describes its uncertainty. Then, we evaluate $f_{B'}(x)$ and repeat this evaluation many times over, yielding a simulated distribution of outcomes $f'(x)$, from which we compute the variance, confidence intervals, or other quantities used to represent uncertainty.

A general way to describe uncertainty is with the variance-covariance matrix Σ (Equation 7.1 or, more generally, Equation 7.3). This means that rather than drawing from the distributions of each B_i separately, we must draw from the multivariate distribution of B. Otherwise, we likely will overestimate the uncertainty in $f(x)$.[20]

We can compare analytical and Monte Carlo methods for Gaussian error propagation using simulation (Code Box 7.16).

The simulation shows that the predicted values are much closer to the mean (the distribution is sharper) when considering the variance-covariance matrix (`pred3` in Code Box 7.16), particularly when the correlation between slope and intercept is as large ($r = -0.87$) as in this (representative) case (Figure 7.10).

In summary, Monte Carlo simulation is a standard way to propagate from uncertainty in parameters to predictions from a model, but unless the parameters were derived by fitting a model, and hence their variance-covariance matrix is known, simply drawing Monte Carlo values from each parameter independently will overestimate uncertainty.

7.7.3 Bayesian and Markov chain Monte Carlo (MCMC) error propagation

Despite the "Monte Carlo" in Markov chain Monte Carlo (MCMC), the latter deviates substantially from the Monte Carlo approach to error propagation. The most relevant difference is that MCMC samples from the (posterior) distribution of a function $f(x)$, given input x, parameters B and prior distributions for each B_i (or their variance-covariance matrix).[21]

[19] The "randomly drawn" bit is the origin of the term Monte Carlo simulation, named by Stanislaw Ulam for the centre of gambling in Europe. See footnote 17 in Chapter 2 for more detail.

[20] There are very few off-the-shelf multivariate random number generators. But we can transform draws from the multivariate normal distribution to follow almost any marginal distribution using the cumulative distribution function (CDF) of the normal distribution (which transforms to uniformity) and then the probability distribution function (PDF) of whichever distribution we seek to represent. An alternative, and general, approach is through the use of copulas (Kojadinovic & Yan 2010; Hofert et al. 2023; Yan 2023).

[21] See also §3.3.1 in Chapter 3.

Code Box 7.16: Monte Carlo simulation of regression uncertainty

```
#Load required libraries
library(mvtnorm)

# Simulation
 set.seed(123)
 x <- runif(100, 0, 1)
 y <- 2 + 2*x + rnorm(100, sd=1)
 fm <- lm(y ~ x)

# Analytical error propagation:
 preds1 <- predict(fm, newdata=list(x=0.9), se.fit=T)

# Monte Carlo sampling (without the variance-covariance matrix)
 preds2 <- rnorm(1000, mean=coef(fm)[1], sd=summary(fm)$coefficient[1,2]) +
      0.9* rnorm(1000, mean=coef(fm)[2], sd=summary(fm)$coefficient[2,2])

 summary(preds2)

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
|  2.499   3.461   3.733   3.719   3.975   4.968

# Monte Carlo sampling (from the variance-covariance matrix)
 Sigma <- vcov(fm)
 cov2cor(Sigma)

|              (Intercept)          x
| (Intercept)   1.0000000 -0.8692359
| x            -0.8692359  1.0000000

 coefs <- rmvnorm(1000, mean=coef(fm), sigma=Sigma)
 preds3 <- coefs[,1] + 0.9*coefs[,2]

summary(preds3)

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
|  3.240   3.595   3.705   3.709   3.819   4.200
```

As long as we can formulate our analytical workflow entirely in terms of probabilistic operations (e.g., GLMs, stochastic search variable selection, etc.), we can use MCMC to estimate all the parameters in the model, use the model for predictions, and automatically get a quantifiable measure of uncertainty. Although this is seemingly straightforward, Bayesian methods (and MCMC) are not used or taught as frequently as we might expect because their computational burdens can be very high. Tailoring a workflow and the associated analysis beyond standard models and procedures requires programming well beyond what we have encountered in any simulations thus far presented. Gilks et al. (1996) and Chen et al. (2012) provide good introductions to Bayesian MCMC methods.

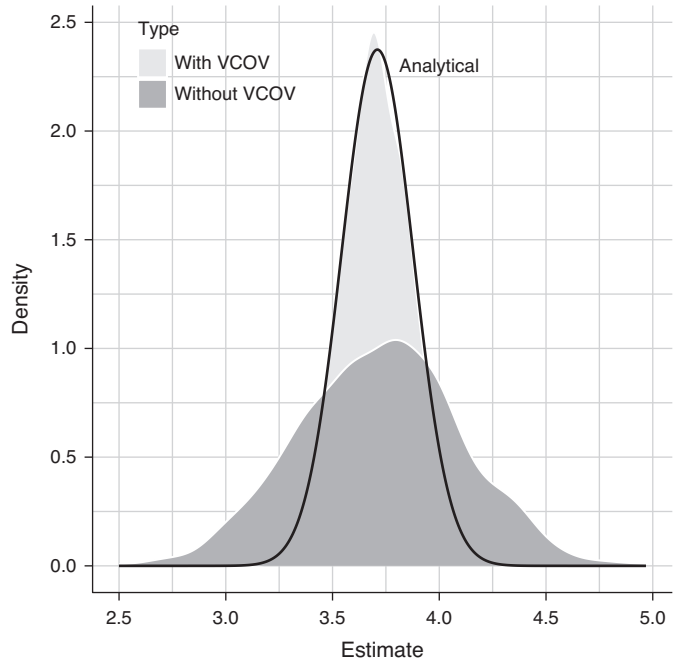


Figure 7.10 A little simulation that shows that univariate Monte Carlo draws from the estimates are not correctly representing the uncertainty of a prediction (light grey), compared to a multivariate draw (dark grey) or indeed the analytical propagation of uncertainty (solid black line).

7.7.4 Bootstrapping

Probably the most commonly-used method to propagate uncertainty through a statistical workflow is bootstrapping—repeatedly sampling the data with replacement (§2.3.3 in Chapter 2; Efron 1982). This is certainly appropriate if the main source of error is in the data themselves. That means that the various estimates are derived from the data and all decisions that occur between individual steps in a workflow also depend only on the data.

For example, if consecutive steps in an analysis depend on the R^2 of the fitted model, then the conditions for bootstrapping would be met. This also holds true for steps that are related to the values of the predictors and response, e.g., a standardisation of predictors. Any step that is data-independent (e.g., subtracting 5 from every value in our dataset) would have no effect on error propagation.[22] Thus, for all practical purposes, if a statistical workflow depends only on the values of the input (Xs and Ys), bootstrapping these inputs is a suitable way to propagate data-driven uncertainty through the workflow.

7.8 Workflow: Stringing multiple statistical steps into an analytical pipeline

In the preceding sections in this chapter, we have discussed four critical steps that are encountered typically in a statistical workflow—imputing missing values for predictors or response variables, handling censored data, pre-selection of predictors, and error

[22] To see this, recall that the Jacobian matrix is comprised of first-order partial derivatives. Since the slope of a function does not change by shifting it up or down, a constant added to y has no effect on the derivatives and hence on the analytical error propagation.

propagation—but there are many more (data transformation, handling multicollinearity, addressing autocorrelation in the response, ad hoc formulating of predictors through feature engineering, deriving and filtering indices, using proxies, etc.). Stringing more than one of these, and certainly combining many of them, into a workflow that at some or all steps requires decisions that depend on what the data or intermediate results suggest, could be a recipe for disaster.[23]

Even if disaster is averted, however, every decision we take in a statistical workflow can add to the uncertainty in our results. If, given the result of a step in a workflow, we have to decide (or the workflow forks) to follow one of a set of possible next steps, and if each alternative step heads into a different part of the parameter space, the combination of the different steps may increase the uncertainty of the final result. That is, uncertainty results not only from measurement error but also from variability in any part of the analytical workflow.

There are three things that keep the uncertainty in check. First, the uncertainty introduced at each decision may be independent of the uncertainty introduced at another step. If this is the case, the total variance is only additive (§7.7). Second, many statistical procedures have been devised to be unbiased, so chaining such procedures together should not result in biased estimates. For example, even the rather involved automatic analysis of Dormann et al. (2008) was unbiased, despite looking suspiciously like fishing for significance. Third, and most importantly in the context of this book, we can use simulation to test our workflow and determine if the estimates are biased, the variance expands unexpectedly, or our tests have inflated type-I or type-II errors. It usually is much easier to set up and run an appropriate simulation than it is to define the workflow we want to use.

We illustrate two key points with a relatively simple workflow and simulation. First, we use bootstrapping to illustrate how a model estimated at the end of a workflow can under-represent the true uncertainty in parameter estimates and prediction. Second, we use a simulated dataset (for which we know the true parameter values) to show that a workflow can create biased estimates. With a bit of thought, you should be able to transfer this approach to more complicated, real-world cases.

7.8.1 A case study: Estimating river discharge across the USA

The river discharge of 671 large hydrological basins in the contiguous USA is continuously monitored. Addor et al. (2017) published a homogenised dataset of the mean annual discharges and watershed characteristics (soil, topology, vegetation, climate, geology).[24] Our analysis of mean discharge as a function of the catchment characteristics is not particularly exciting (at least to hydrologists), but we think that it illustrates the typical considerations that enter into a statistical pipeline. Note that we *deliberately* make questionable analytical decisions for illustrative purposes and to highlight key issues we need to consider.

The response variable is mean discharge averaged over the years 1989–2009. For the analysis we $\log_{10}$-transformed the mean discharge of each basin (Newman et al. 2015). The residuals of this model were distributed approximately normally, justifying this decision.

[23] Many such disasters depend not only on data-driven decisions but also on systematic errors based on an individual's experience and expectations (e.g., Frankel 1976; Simmons et al. 2011; Baker 2016; Silberzahn et al. 2018; Bishop 2020; Gelman et al. 2020; Breznau et al. 2022; Gould et al. 2023) or on ill-formed questions (Auspurg & Brüderl 2021). These are much more difficult to deal with, much less to simulate (and play into the design of "federated" analyses; see Chapter 10).

[24] The complete dataset is available at https://gdex.ucar.edu/dataset/camels/file.html. We used the small files, whose names start with `camels`.

Code Box 7.17: Setting up the data for the workflow

```
# Read in the dataset (local copy)
 camelsI <- read.csv("camelsImputed.csv", stringsAsFactors = T, row.names=1)

# Remove the five categorical predictors
 camelsI <- camelsI[, -c(21, 31, 34, 52, 54)]

# Remove names, and q5 and q95 because they are tautological predictors for the
    response variable, q_mean
 camelsI <- camelsI[, -c(1, 7, 8)]

# Delete 15% of the predictor values at random (MCAR)
camelsHoles <- camelsI
set.seed(123)
for (i in c(1:50)){
  shot <- sample(671, floor(0.15*671))
  camelsHoles[shot, i] <- NA
}
```

To add realism, we introduced missing data into the dataset; we deleted at random 15% of the values for each predictor. Admittedly this is a high value, but this MCAR scenario allows for imputation without much further consideration of the reasons of missingness (§7.1, above). We also used only the continuous predictors in the dataset.[25] In sum, we have a dataset with of 671 observations (basins), each with a single response variable (mean daily discharge in mm/d), and a set of continuous predictor variables each of which has 15% of its values missing (Code Box 7.17). This is the dataset we send into our workflow.

THE WORKFLOW

Our illustrative workflow consists of seven steps:

1. Impute the missing (MCAR) data using random forest imputation.[26]
2. Quantify the importance of each variable using measures of impurity derived from a random-forest model.
3. Reduce the dimensionality and retain only those predictors that together contribute 90% of variable importance.[27]
4. Identify collinear predictors and, for any pair of them whose correlation $|r| > 0.7$, remove from the dataset the predictor with the lower importance.[28]

[25] This is not a hydrologically-motivated choice, but rather is a practical one. Including categorical predictors makes the workflow more complex and didactically more problematic. A real-world analysis would, of course, include categorical predictors.

[26] Because of near-aliasing, we used a random-forest model for imputation, rather than the default predictive mean matching.

[27] Any dimensional reduction results in a loss of information. It certainly will add uncertainty to our results but does not inevitably introduce bias.

[28] We recognise that this is a very problematic choice. That predictors are correlated is a fact we cannot remove (recall §6.2 in Chapter 6). With this choice, which we make for pedagogical reasons, we ensure we will introduce bias into our estimation and interpretation.

5. Fit a basic ordinary least-squares (OLS) linear model with $\log_{10}$(mean discharge) as the response and all the remaining variables as predictors, with both linear and quadratic terms.[29]
6. Simplify the model using backward-stepwise model selection and the minimum AIC as our target.[30]
7. Predict the mean discharge of the first basin. Save and output the prediction, its standard error, all model estimates, and their standard errors.

We create an `R` function that we call `workflow` that includes all seven of these steps (Code Box 7.18). We then execute the function using the dataset `camelsHoles` that we created in Code Box 7.17.[31]

ILLUSTRATING WORKFLOW-INDUCED UNCERTAINTY

Now that we have our workflow up and running, we can use it to illustrate the first of the two key points we introduced in §7.8.1: identifying the uncertainty that is introduced by the various data-dependent decisions in the steps of the workflow. To do this, we bootstrap the entire workflow (Code Box 7.19). Other than setting the random seed to ensure repeatable results, the bootstrapping requires only a single line of `R` code applied to the `workflow` function we created in Code Box 7.18. Each run of the workflow takes approximately six seconds, so using 500 bootstraps takes about five minutes.

Since every run results in a different set of predictors, we need to wrangle the results into a format suitable for visualisation. For now, we save space and only look at the main effects (Code Box 7.20).

Summarising the results, we first observe that the most important predictors that were included in the single workflow run (Code Box 7.18) also were included in most bootstrapping models (grey dots in Figure 7.11). But a much wider range of potentially useful predictors were identified across all the bootstrapped runs than were identified in the single run.

To quantify the effect of bootstrapping the workflow on the uncertainty of parameter estimates and prediction, we need to put the estimates on a common scale. To do this, we use the dimensionless coefficient of variation: $\text{CV} = \frac{\text{SE}}{|\bar{x}|}$ (Code Box 7.21).

Bootstrapping the workflow reveals a two- to eight-fold increase in parameter uncertainty (Figure 7.12), and a three-fold increase in the uncertainty of our predictions from a single run of the workflow.

We conclude that our workflow substantially inflates parameter and prediction uncertainty, both of which are inadequately reflected in the final model. Figure 7.11 also suggests a high probability of overlooking relevant predictors and haphazard inclusion of less important ones. However, this bootstrap analysis cannot help us determine whether the parameter estimates are actually biased (the second key point raised in §7.8). For that we need simulated data with known parameter values.

---

[29] We recognize that this is also a somewhat problematic choice. In an ideal world we also may want to fit interactions between predictors but we soon run out of degrees of freedom. With this choice, we have constrained the flexibility and realism of our model. However, in our experience, fitting such additive models is done routinely.

[30] This is a problematic choice for parameter estimation, but not for prediction. This choice changes the uncertainty but the resulting final model will not know its origin and hence is likely to underestimate the uncertainty. Analysts in all fields still incorrectly practice variable selection-then-inference; see Sclove et al. (1972); Freedman et al. (1988); Breiman (1992); Viallefont et al. (2001); Greenland (2008); Hill et al. (2011) for discussion.

[31] Of course, we also could use this `workflow` function on different datasets.

Code Box 7.18: Create and execute the `workflow` function

```
# Load required libraries
 library(mice) # For imputation routines
 library(ranger) # For random forest models

# Create the workflow as a function
 workflow <- function(data){
  # Step 1: Imputation
   camelsPlugged <- complete(mice(data, method="rf", print=F, m=1)) # !! ''data''
        (generic) is passed to the function instead of the specific ''
       camelsHoles''. Note there is no default for ''data''.
  # Step 2: Quantification of importance using random forest's impurity
   fr <- ranger(log10(q_mean) ~ ., data=camelsPlugged, importance="impurity")
   imps <- sort(importance(fr), decreasing=T) # Sorted importance values
  # Step 3: Dimensional reduction: Keep only those predictors that together
      contribute 90% of variable importance
   contribTo100 <- cumsum(imps) / sum(imps)
   cutoff <- which.min(abs(contribTo100 - 0.9)) # Closest to .9
   subCamelsPlugged <- camelsPlugged[, names(contribTo100)[1:cutoff] ]
  # Step 4: Identify collinearity predictors and remove any of a pair whose r >
      |0.7|
   cors <- cor(subCamelsPlugged, method="sp")
   highcor <- which(abs(cors) > 0.7 & cors != 1, arr.ind=T)
   dealWith <- which(highcor[,1] < highcor[,2])
   removeHighCollinear <- if (length(dealWith) > 0) TRUE else FALSE
    while (removeHighCollinear){
     pickThese <- highcor[dealWith[1],]
     remove <- which.min(imps[colnames(cors)[pickThese]])
     subCamelsPlugged <- subCamelsPlugged[,-which(colnames(subCamelsPlugged) ==
         names(remove))]
     cors <- cor(subCamelsPlugged, method="sp")
     highcor <- which(abs(cors) > 0.7 & cors != 1, arr.ind=T)
     dealWith <- which(highcor[,1] < highcor[,2])
     removeHighCollinear <- if (length(dealWith) > 0)  TRUE else FALSE
    }
  # Step 5: Fit an OLS model
   RHS <- paste(colnames(subCamelsPlugged), collapse=" + ")
   RHS2 <- paste0("I(", paste(colnames(subCamelsPlugged), collapse="^2) + I("),
       "^2)")
   f <- as.formula(paste("log10(q_mean) ~", RHS, "+", RHS2))
   fm <- lm(f, data=cbind.data.frame("q_mean"=camelsPlugged$q_mean,
       subCamelsPlugged))
  # Step 6: Simplify the model using stepwise model selection and AIC
   fmstep <- step(fm, trace=F)
  # Step 7: Save estimates and predictions
   as.matrix(rbind(summary(fmstep)$coefficients[,1:2],
                   "gauge1"=unlist(predict(fmstep, newdata=camelsI[1,],
                       se.fit=T))[1:2]))
}
```

```
# Full run of the workflow function using camelsHoles from Code Box 7.17 as data
 fullrun <- workflow(data=camelsHoles)
 fullrun

|                          Estimate   Std. Error
| (Intercept)        -1.896960e+00 1.818948e-01
| runoff_ratio        2.143451e+00 1.161712e-01
| p_mean              2.665705e-01 2.296815e-02
| frac_forest         3.542514e-01 7.420312e-02
| p_seasonality      -1.397722e-01 2.015741e-02
| low_prec_dur       -3.180653e-02 6.560033e-03
| slope_fdc           1.448435e-01 5.224617e-02
| slope_mean         -9.157225e-04 4.487626e-04
| hfd_mean            5.942241e-04 2.574483e-04
| stream_elas         6.915396e-02 2.289648e-02
| area_gages2        -2.102407e-05 6.365008e-06
| root_depth_50       5.234006e+00 1.628602e+00
| I(runoff_ratio^2)  -1.203459e+00 9.661979e-02
| I(p_mean^2)        -1.756873e-02 2.418287e-03
| I(frac_forest^2)   -2.978154e-01 7.001603e-02
| I(low_prec_dur^2)   3.850653e-04 1.971200e-04
| I(slope_fdc^2)     -5.786769e-02 2.041899e-02
| I(slope_mean^2)     5.445615e-06 2.518318e-06
| I(stream_elas^2)   -1.098226e-02 4.744542e-03
| I(zero_q_freq^2)   -4.858955e-01 1.079449e-01
| I(area_gages2^2)    5.485360e-10 3.534245e-10
| I(root_depth_50^2) -1.352989e+01 4.306492e+00
| gauge1              2.621590e-01 1.773648e-02
```

Code Box 7.19: Bootstrappping the workflow

```
set.seed(123)
ResultsBootstrapWorkflow <- replicate(500, workflow(data=camelsHoles[sample(671,
     size=671, replace=T),]))
```

ARE THE PARAMETER ESTIMATES BIASED?

When simulating data, the first question we always ask is: how do we decide the values of the parameters to use? That is, why should a simulation have the specific form we set up? As we discussed way back in Chapter 1, any individual simulation typically is tied to a specific dataset. Thus, we only have to devise a simulation of similar complexity as we might believe to be true. Our simulation typically does not have to be the correct model; it only needs to be in the right ballpark. We might have a clear expectation of which predictors affect river discharge from previous studies or we might run some exploratory analysis to derive some model structure that more-or-less fits the data. Either way, voilà, we have the coefficients for our simulation.

For example, we can think of a few reasons why river discharge could differ among catchments. Knowing that discharge (m^3/t) is computed from velocity $\times$ area

Code Box 7.20: Wrangle the bootstrapped results

```
res.arr <- array(0, dim=c(length(ResultsBootstrapWorkflow), 50, 2),
                 dimnames=list(1:length(ResultsBootstrapWorkflow),
                      c(colnames(camelsI)[-1], "gauge1"), c("estimate", "se")))
                # Create the array to hold all the replicates, 49 estimates plus
                      prediction, mean, and standard errors
for (i in 1:dim(res.arr)[1]){ # Loop over replicates
  for (j in 1:50){ # Loop through effects
    findthis <- dimnames(res.arr)[[2]][j]
    puthere <- which(rownames(ResultsBootstrapWorkflow[[i]]) %in% findthis)
    if (length(puthere) > 0){
      res.arr[i, j, ] <- ResultsBootstrapWorkflow[[i]][findthis, ]
    }
  }
}
```

($m \cdot t^{-1} \cdot m^2 = m^3/t$), we immediately can see that the amount of rainfall (m) and the catchment area (m^2) will matter.[32] We also think that two other predictors could have large effects: the amount of forest (which increases transpiration loss but also facilitates infiltration) and aridity (which increases evaporational losses). Thus, we use these four variables to fit a model to our data and then use the resulting estimates as the true parameter values for our simulation (Code Box 7.22). Note that this is a much simpler model than we recovered from our workflow (Figure 7.11). However, we are not interested in recovering the model that best fits the data. Rather, we are interested in whether the workflow can identify correctly the variables that we used as the predictors in our simulation and also give unbiased estimates of the parameters.

The last line of the output in Code Box 7.22 shows the mean parameter estimates from the 100 simulations, which we should compare with the true values used in the simulation (0.135, 0.366, −0.275, 0.00004, and 0.09, respectively). In the simulations, the effects of precipitation (`p_mean`), `aridity`, and `area` at least are within a factor of 2 of the true values, but the effect of forested area of the catchment (`frac_forest`) is off by a factor of 8. The overall prediction (`gauge1`) is off only by a factor of 1.5, suggesting that the effect on prediction is less severe than on parameter estimation.

We can express this as relative bias by dividing the difference between the true and estimated values by the true value; all estimates are substantially biased (Figure 7.13). Computing standardised regression coefficients (as $\frac{s_x}{s_y}\hat{\beta}$) ranks the predictors as `aridity` (−0.12), `p_mean` (0.10), `area` (−0.021) and `frac_forest` (0.0082). Although `p_mean` and area have the least bias, they are still biased. This suggests that correlations within a dataset have an impact on which predictor is selected and not even the strongest predictors come through the workflow without bias.

[32] Although our response variable, river discharge, is measured using a standard stream gauge (see USGS 2018) and already has been corrected for catchment area in the `camel` dataset, there may well be an additional effect of area. For example, larger catchments have longer average retention times, allowing precipitation to evaporate or infiltrate into deeper ground water.

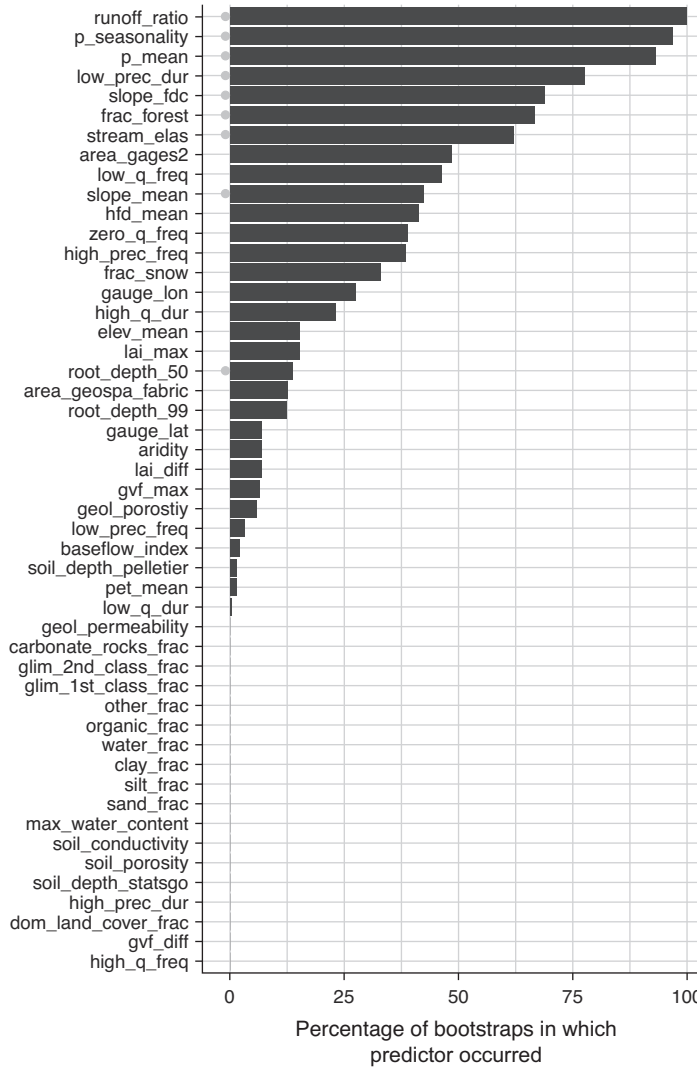


Figure 7.11 Proportion of models that include a given predictor for the discharge analysis workflow. Grey dots indicate variables included in the full data workflow run (Code Box 7.18). (The incorrect spelling of "geo_porostiy" implies that these are the real data, as provided at the URL given in note 25.)

CONCLUSIONS

In this case study, analysing the effect of catchment characteristics on river discharge, we deliberately used a workflow that we knew would be likely to have problems with estimation and bias. We did this to illustrate that specific data-driven decisions could lead to problems, which is not to say that all workflows will have similar problems. Rather, we use this example to demonstrate that common decisions can lead to problems in a workflow and how to use bootstrapping and simulation to find them (and, by implication, return to the workflow and correct them).

This exercise revealed that the selection of predictors—first using dimensional reduction, then (de)selecting some of them because of collinearity, and finally determining the final set of predictors using model selection—led to a wide range of model structures when bootstrapping the data (Figure 7.11). These (automated) decisions, along with imputation

Code Box 7.21: Comparing parameter and prediction uncertainty

```
# Compute coefficients of variation
 CVone <- fullrun[,2] / abs(fullrun[,1]) * 100
 CVbs <- apply(res.arr[-51, , 1], 2, sd) / abs(colMeans(res.arr[-51, , 1])) * 100
 cbind(CVone[c(2:11, 20)], na.omit(CVbs[bsOrder]))
```

```
|                    [,1]      [,2]
| runoff_ratio   4.656990  12.89267
| p_mean         9.808280  32.72270
| frac_forest   25.164589  78.85933
| low_prec_dur  18.068959  79.00987
| p_seasonality 19.255093  38.16169
| slope_fdc     28.526573 104.46083
| stream_elas   25.928433 109.28080
| area_gages2   17.607957 116.79866
| frac_snow     71.189493 674.27568
| root_depth_50 62.776889 267.58386
| gauge1         6.888866  17.46253
```

of missing values, added substantial variability to the estimates that would go unnoticed if we had looked only at the output of the resulting model (the left side, `full`, in Figure 7.12). Harrell (2015) used the term "prescribed model" to highlight the point that this final model cannot have any information in it about how its predictors were selected; it is as if it was prescribed and its standard errors are optimistic.

To add insult to injury, the parameter estimates themselves are biased. In particular, the various model-selection steps yielded different models with every bootstrapping analysis; true predictors are missing from many models, which further negatively biases their estimates (Figure 7.13).

We can confidently state that this analytical workflow gives poor results for both parameter inference and prediction (Figure 7.13). However, our simulations do show that even substantial statistical distortions do not necessarily render predictions unreliable. A prediction can be right for the wrong reason (Persson 2020; Sun et al. 2023). It is more common that model selection acts as an optimiser in the trade-off between variance and bias, reducing variance while biasing estimates (Hastie et al. 2009).

7.9 Summary

Any workflow can introduce artefacts but some steps are particularly problematic and raise red flags, thus meriting additional careful attention. These include:

- Any step that uses the response Y before the final analysis. Such "data snooping" (White 2000; Leippold & Lohre 2012) is problematic for two reasons. First, steps that result in re-use of data de facto suggest a much higher sample size than is in the dataset. Second, by using the same data for pre-selection of predictors and their estimation, we are assuming implicitly that they are independent. This second problem also includes any loop[33] that uses such results to refine the analysis or change steps in the workflow. It

[33] Or coding equivalents to for-loops, such as `while`, `rep`, or `sim` statements.

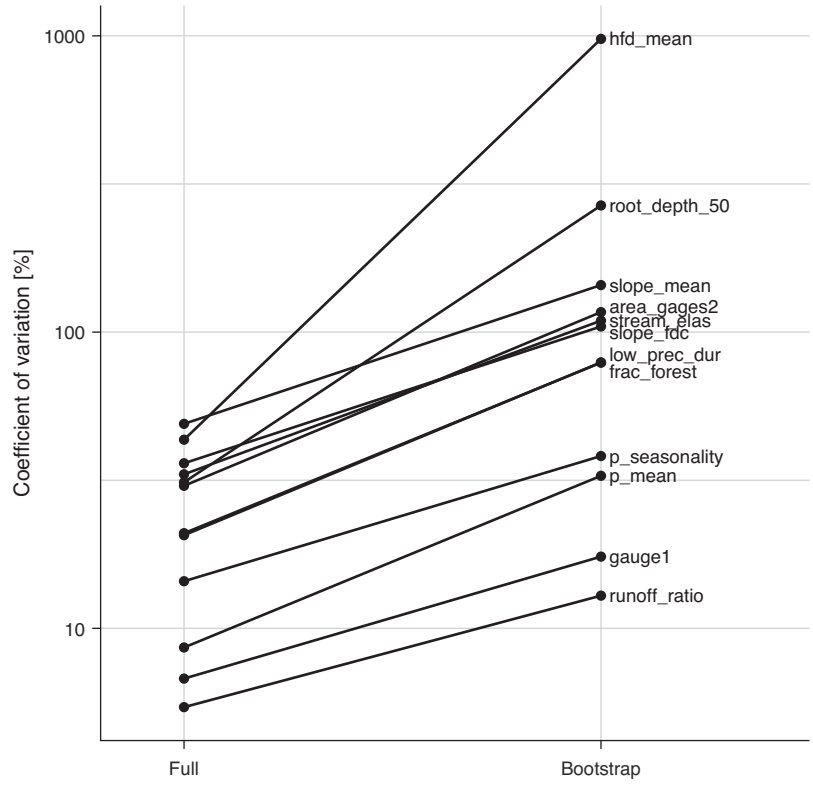


Figure 7.12 Change in parameter uncertainty from single full data analysis workflow to its bootstrapping equivalent (500 bootstraps). "gauge1" refers to the prediction made by the model to the first catchment in the dataset. Note the logarithmic scaling of the y-axis.

is also important that any feature engineering step must be done without knowledge of Y.

- Using as response an index that mixes X and Y. This is, in a way, an extension of the previous point, but more subtle. A response ratio of X/Y as a function of Z/Y will introduce a spurious correlation (Williams et al. 2022).
- Outlier removal. Tempting as it may be, "outliers" that are real measurements cannot simply be ignored.[34] Removing them a priori inevitably will result in a biased model; the substantial uncertainty that comes with using real outliers does represent the true variability.
- Steps using model selection *and* subsequent inferential statements.
- Any branching in the workflow related to Y, even without formal analysis of Y, that is related to, for example, sample size, type of distribution, correlation in predictors, range of predictors, and the like.

We note that independently of all these problems, biased data typically are extremely difficult to analyse correctly. Faced with survivourship bias or other forms of censoring, we

[34] Of course, some outliers may not be valid observations for methodological reasons, such as sensor failures, or because of errors in data entry or transmission. Unless it can be demonstrated that an outlier (or any other "suspicious" datum) really is invalid, it must be retained in the analysis. Screening out data more than $N\sigma$ (where N is often set ≥ 3) away from our expectation or model prediction should never be done.

Code Box 7.22: Using simulation to quantify bias in the workflow

```
# Fit a simple model to the data and use the parameter estimates for the
    simulation
summary(lm(log10(q_mean) ~ p_mean + frac_forest  + aridity + area_gages2, data=
    camelsI))

| <—-snip--->
| Coefficients:
|               Estimate Std. Error t value Pr(>|t|)
| (Intercept) -3.962e-01  5.665e-02  -6.994 6.51e-12 ***
| p_mean       1.348e-01  1.019e-02  13.225  < 2e-16 ***
| frac_forest  3.661e-01  3.113e-02  11.759  < 2e-16 ***
| aridity     -2.755e-01  2.325e-02 -11.852  < 2e-16 ***
| area_gages2 -4.004e-05  5.435e-06  -7.367 5.18e-13 ***
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| Residual standard error: 0.2338 on 666 degrees of freedom
| Multiple R-squared:  0.791, Adjusted R-squared:  0.7898

# Simulations
set.seed(123)
camelsSims <- replicate(100, { # 100 simulations using the original dataset from
     Code Box 7.17
 y <- with(camelsI, 10^(-0.4 + 0.135*p_mean + 0.366*frac_forest - 0.275*aridity
     -0.00004*area_gages2 + rnorm(671, 0, 0.2)))
 datos$q_mean <- y
 fullrun.sim <- workflow(data=datos)
 fullrun.sim
})

# Re-use the code from Code Box 7.20 to collate the results
sim.res.arr <- array(0, dim=c(length(camelsSims), 50, 2),
                dimnames=list(1:length(camelsSims), c(colnames(camelsI)[-1],
                    "gauge1"), c("estimate", "se")))
for (i in 1:dim(sim.res.arr)[1]){
 for (j in 1:50){
   findthis <- dimnames(sim.res.arr)[[2]][j]
   puthere <- which(rownames(camelsSims[[i]]) %in% findthis)
   if (length(puthere) > 0){
     sim.res.arr[i, j, ] <- camelsSims[[i]][findthis, ]
   }
 }
}
colMeans(sim.res.arr[, c(19, 11, 23, 43, 50),1])

|       p_mean  frac_forest      aridity  area_gages2       gauge1
|  0.143317471  0.043802601 -0.381779103 -0.000023914  0.066100597
```

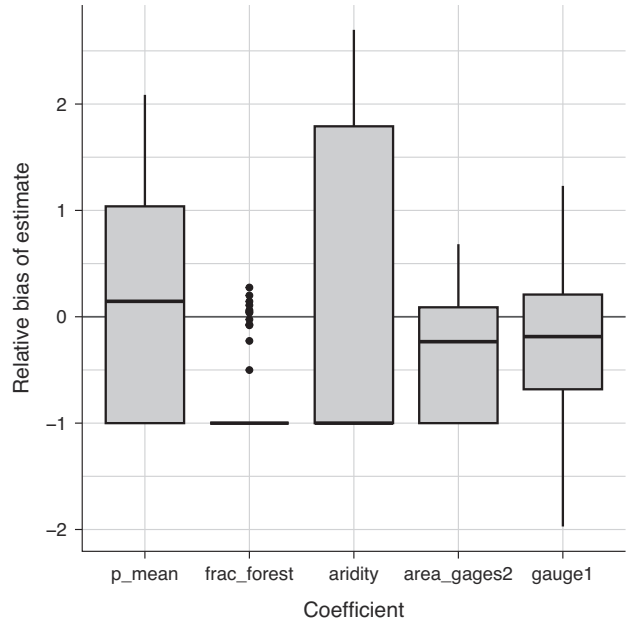


Figure 7.13 Relative bias of the four predictor variables in the simulation (`p_mean`: mean annual precipitation).

must have a comprehensive knowledge of the biases introduced when the data were collected to be able to account for them. For example, when plants flower is difficult to estimate from herbarium data without bias because botanists often wait to collect specimens until they find flowers and seeds, they only collect specific families or genera, collect near to universities, or collect only during particular times of the week (Daru et al. 2018).[35] Biases can also occur because we use only data that are easily available, such as decisions made by billionaires that are documented in their (auto)biographies. Where might we get similar data from less wealthy individuals?

However, automated workflows are extremely useful and our forced example does not aim at discouraging their use. Rather, we hope that we have illustrated how to evaluate whether a workflow introduces artefacts and biases, and to think about correcting the workflow before proceeding with the full analysis.

References

Adde, A., Rey, P.-L., Brun, P., et al. (2023). N-SDM: A high-performance computing pipeline for Nested Species Distribution Modelling. *Ecography*, 2023(6), e06540. doi: 10.1111/ecog.06540.

Addor, N., Newman, A. J., Mizukami, N., & Clark, M. P. (2017). The CAMELS data set: Catchment attributes and meteorology for large-sample studies. *Hydrology and Earth System Sciences*, 21(10), 5293–5313. doi: 10.5194/hess-21-5293-2017.

Auspurg, K. & Brüderl, J. (2021). Has the credibility of the social sciences been credibly destroyed? Reanalyzing the "Many Analysts, One Data Set" project. *Socius: Sociological Research for a Dynamic World*, 7, 1–14. doi: 10.1177/23780231211024421.

[35] Similar biases also occur in zoological collections and in data collected by citizen-scientists, community-scientists, and crowd-sourcers (Meineke & Daru 2021; Daru & Rodriguez 2023), and when pooling data across different sampling designs (where one can use thinning to the lowest level of communality as comparison: Sabatini et al. 2022).

Baker, M. (2016). Is there a reproducibility crisis? *Nature*, 533(7604), 452–454. doi: 10.1038/533452a.

Bishop, D. (2020). How scientists can stop fooling themselves over statistics. *Nature*, 584(7819), 9. doi: 10.1038/d41586-020-02275-8.

Bogaerts, K., Komarek, A., & Lesaffre, E. (2017). *Survival Analysis with Interval-Censored Data: A Practical Approach with Examples in R, SAS, and BUGS*. New York, NY, USA: Chapman & Hall/CRC.

Breiman, L. (1992). The little bootstrap and other methods for dimensionality selection in regression: X-fixed prediction error. *Journal of the American Statistical Association*, 87(419), 738–754. doi: 10.1080/01621459.1992.10475276

Breznau, N., Rinke, E. M., Wuttke, A., et al. (2022). Observing many researchers using the same data and hypothesis reveals a hidden universe of uncertainty. *Proceedings of the National Academy of Sciences, USA*, 119(44), e2203150119. doi: 10.1073/pnas.2203150119.

Brilleman, S. L., Elci, E. M., Novik, J. B., & Wolfe, R. (2020). Bayesian survival analysis using the rstanarm R package. *arXiv*, 002.09633v1, 1–50. doi: 10.48550/arXiv.2002.09633.

Canales, R. A., Wilson, A. M., Pearce-Walker, J. I., et al. (2018). Methods for handling left-censored data in quantitative microbial risk assessment. *Applied and Environmental Microbiology*, 84(20), e01203–e01218. doi: 10.1128/AEM.01203-18.

Cantoni, B., Delli Compagni, R., Turolla, A., et al. (2020). A statistical assessment of micropollutants occurrence, time trend, fate and human health risk using left-censored water quality data. *Chemosphere*, 257, 127095. doi: 10.1016/j.chemosphere.2020.127095.

CDC: Centers for Disease Control and Prevention (2023). Diabetes Tests. https://www.cdc.gov/diabetes/basics/getting-tested.html. Accessed 16 January 2024.

Chen, M.-H., Shao, Q.-M., & Ibrahim, J. G. (2012). *Monte Carlo Methods in Bayesian Computation*. New York, NY, USA: Springer.

Currie, L. A. (1999). Detection and quantification limits: Origins and historical overview. *Analytica Chimica Acta*, 391(2), 127–134. doi: 10.1016/S0003-2670(99)00105-1.

Dago, N. D., Fofana, I. J., Diarrassouba, N., et al. (2019). A quick computational statistical pipeline developed in R programing environment for agronomic metric data analysis. *American Journal of Bioinformatics Research*, 9(1), 22–44. doi: 10.5923/j.bioinformatics.20190901.03.

Darlington, R. B. & Smulders, T. V. (2001). Problems with residual analysis. *Animal Behaviour*, 62(3), 599–602. doi: 10.1006/anbe.2001.1806.

Daru, B. H., Park, D. S., Primack, R. B., et al. (2018). Widespread sampling biases in herbaria revealed from large-scale digitization. *New Phytologist*, 217(2), 939–955. doi: 10.1111/nph.14855.

Daru, B. H. & Rodriguez, J. (2023). Mass production of unvouchered records fails to represent global biodiversity patterns. *Nature Ecology & Evolution*, 7(6), 816–831. doi: 10.1038/s41559-023-02047-3.

Deeks, J. J., Higgins, J. P. T., & Altman, D. G. (2023). Chapter 10: Analysing data and undertaking meta-analyses. In J. P. T. Higgins, J. Thomas, J. Chandler, M. Cumpston, T. Li, M. G. Page, & V. A. Welch (Eds.), *Cochrane Handbook for Systematic Reviews of Interventions*. Version 6.4, 2nd edition (updated August 2023). Chichester, UK: John Wiley & Sons.

Dormann, C. F., Purschke, O., García Marquéz, J. R., et al. (2008). Components of uncertainty in species distribution analysis: A case study of the great grey shrike. *Ecology*, 89(12), 3371–3386.

Efron, B. (1982). *The Jackknife, the Bootstrap, and Other Resampling Plans*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Enders, C. K. (2022). *Applied Missing Data Analysis*, 2nd edition. New York, NY, USA: Guildford Press.

EPA (United States Environmental Protection Agency) (2002). *Method 1631, Revision E: Mercury in water by oxidation, purge and trap, and cold vapor atomic fluorescence spectrometry*. Technical Report EPA-821-R-02-019, United States Environmental Protection Agency, Office of Water.

EPA (United States Environmental Protection Agency) (2010). *Technical basis for the Lowest Concentration Minimum Reporting Level (LCMRL) calculator*. Technical Report EPA 815-R-11-001, United States Environmental Protection Agency, Office of Water.

Fox, G. A. (2015). What you don't know can hurt you: Censored and truncated data in ecological research. In G. A. Fox, S. Negrete-Yankelevich, & V. J. Sosa (Eds.), *Ecological Statistics: Contemporary Theory and Application* (pp. 106–130). Oxford, UK: Oxford University Press.

Frankel, L. R. (1976). Statistics and people–the statistician's responsibilities. *Journal of the American Statistical Association*, 71(353), 8–16. doi: 10.1080/01621459.1976.10481471.

Freckleton, R. P. (2002). On the misuse of residuals in ecology: Regression of residuals vs. multiple regression. *Journal of Animal Ecology*, 71(3), 542–545. doi: 10.1046/j.1365-2656.2002.00618.x.

Freedman, D. A., Navidi, W., & Peters, S. C. (1988). On the impact of variable selection in fitting regression equations. In T. Dijkstra (Ed.), *On Model Uncertainty and Its Statistical Implications* (pp. 1–16). Berlin, Gernany: Springer.

Gauss, C. F. (1816). Bestimmung der Genauigkeit der Beobachtungen. *Zeitschrift für Astronomie und verwandte Wissenschaften*, 1, 185–197.

Gelman, A., Vehtari, A., Simpson, D., et al. (2020). Bayesian workflow. *arXiv*, 2011.01808, 1–77. doi: arXiv.2011.01808.

Gijbels, I. (2010). Censored data. *WIREs Computational Statistics*, 2(2), 178–188. doi: 10.1002/wics.80.

Gilks, W., Richardson, S., & Spiegelhalter, D., Eds. (1996). *Markov Chain Monte Carlo in Practice*. Boca Raton, FL, USA: Chapman & Hall/CRC.

Golding, N., August, T. A., Lucas, T. C. D., et al. (2018). The zoon R package for reproducible and shareable species distribution modelling. *Methods in Ecology and Evolution*, 9(2), 260–268. doi: 10.1111/2041-210X.12858.

Goolsby, D. A., Thurman, E. M., & Kolpin, D. W. (1991). Geographic and temporal distribution of herbicides in surface waters of the Upper-Midwestern United States, 1989–90. In G. E. Mallard & D. A. Aronson (Eds.), *U.S. Geological Survey Toxic Substances Hydrology Program—Proceedings of the Technical Meeting, Monterey, CA. 11–15 March 1991* (pp. 183–188). Reston, VA, USA: U.S. Geological Survey, Water-Resources Investigations Reports 91-4034.

Gould, E., Fraser, H. S., Parker, T. H., et al. (2023). Same data, different analysts: Variation in effect sizes due to analytical decisions in ecology and evolutionary biology. *EcoEvoRxiv*, 2023, X2GG62. doi: 10.32942/X2GG62.

Greenland, S. (2008). Invited commentary: Variable selection versus shrinkage in the control of multiple confounders. *American Journal of Epidemiology*, 167(5), 523–529. doi: 10.1093/aje/kwm355.

Gupte, P. R., Beardsworth, C. E., Spiegel, O., et al. (2022). A guide to pre-processing high-throughput animal tracking data. *Journal of Animal Ecology*, 91(2), 287–307. doi: 10.1111/1365-2656.13610.

Hansen, C. T., Ritz, C., Gerhard, D., et al. (2015). Re-evaluation of groundwater monitoring data for glyphosate and bentazone by taking detection limits into account. *Science of the Total Environment*, 536, 68–71. doi: 10.1016/j.scitotenv.2015.07.047.

Harrell, F. E. (2015). *Regression Modeling Strategies—With Applications to Linear Models, Logistic and Ordinal Regression, and Survival Analysis*, 2nd edition. New York, NY, USA: Springer.

Hastie, T., Tibshirani, R. J., & Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edition. Berlin: Springer.

Helmert, F. R. (1875). Über die Berechnung des wahrscheinlichen Fehlers aus einer endlichen Anzahl wahrer Beobachtungsfehler. *Zeitschrift für Mathematik und Physik*, 20, 300–303.

Helmert, F. R. (1876). Über die Wahrscheinlichkeit der Potenzsummen der Beobachtungsfehler und über einige damit im Zusammenhange stehende Fragen. *Zeitschrift für Mathematik und Physik*, 21, 192–218.

Helsel, D. R. (2005). *Nondetects and Data Analysis: Statistics for Censored Environmental Data*. New York, NY, USA: Wiley.

Helsel, D. R. (2012). *Statistics For Censored Environmental Data Using Minitab and R*, 2nd edition. Hoboken, NJ, USA: John Wiley & Sons.

Hill, M. P., Hoffmann, A. A., Macfadyen, S., et al. (2011). Understanding niche shifts: Using current and historical data to model the invasive redlegged earth mite, *Halotydeus destructor*. *Diversity and Distributions*, 18(2), 191–203. doi: 10.1111/j.1472-4642.2011.00844.x.

Hofert, M., Kojadinovic, I., Maechler, M., & Yan, J. (2023). copula: Multivariate dependence with copulas. R package version 1.1-2. https://CRAN.R-project.org/package=copula.

Jones, M. P. (2018). Linear regression with left-censored covariates and outcome using a pseudolikelihood approach. *Environmetrics*, 29(8), e2536. doi: 10.1002/env.2536.

Julian, P. & Helsel, D. (2023). NADA2: Data Analysis for Censored Environmental Data, version 1.1.5. https://cran.r-project.org/package=NADA2.

Kaggle (2017). Pima Indians Diabetes Database. https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database. Accessed 16 January 2024.

Keller, A., Backes, C., Al-Awadhi, M., et al. (2008). GeneTrailExpress: A web-based pipeline for the statistical evaluation of microarray experiments. *BMC Bioinformatics*, 9(1), 552. doi: 10.1186/1471-2105-9-552.

Kojadinovic, I. & Yan, J. (2010). Modeling multivariate distributions with continuous margins using the copula R package. *Journal of Statistical Software*, 34(9), 1–20. doi: 10.18637/jss.v034.i09.

Lee, L. (2020). NADA: Nondetects and data analysis for environmental data, version 1.6-1.1. https://cran.r-project.org/package=NADA.

Leippold, M. & Lohre, H. (2012). Data snooping and the global accrual anomaly. *Applied Financial Economics*, 22(7), 509–535. doi: 10.1080/09603107.2011.631892.

Lerner, B., Boose, E., Brand, O., et al. (2022). Making provenance work for you. *R Journal*, 14(4), 141–159. doi: 10.32614/RJ-2023-003.

Leung, K.-M., Elashoff, R. M., & Afifi, A. A. (1997). Censoring issues in survival analysis. *Annual Review of Public Health*, 18(1), 83–104. doi: 10.1146/annurev.publhealth.18.1.83.

Lista, L. (2023). Bayesian probability and inference. In *Statistical Methods for Data Analysis: With Applications in Particle Physics* (pp. 95–125). Cham, Switzerland: Springer.

Liu, Y. & Constantinou, A. (2023). Improving the imputation of missing data with Markov blanket discovery. In *The Eleventh International Conference on Learning Representations*.

Madley-Dowd, P., Hughes, R., Tilling, K., & Heron, J. (2019). The proportion of missing data should not be used to guide decisions on multiple imputation. *Journal of Clinical Epidemiology*, 110, 63–73. doi: 10.1016/j.jclinepi.2019.02.016.

McElreath, R. (2020). *Statistical Rethinking: A Bayesian Course with Examples in R and STAN*, 2nd edition. Boca Raton, FL, USA: Chapman & Hall/CRC.

Meineke, E. K. & Daru, B. H. (2021). Bias assessments to expand research harnessing biological collections. *Trends in Ecology & Evolution*, 36(12), 1071–1082. doi: 10.1016/j.tree.2021.08.003.

Morrissey, M. B. & Ruxton, G. D. (2018). Multiple regression is not multiple regressions: The meaning of multiple regression and the non-problem of collinearity. *Philosophy, Theory, and Practice in Biology*, 10, 003. doi: 10.3998/ptpbio.16039257.0010.003.

Mueller, D. K., Ruddy, B. C., & Battaglin, W. A. (1997). Logistic model of nitrate in streams of the Upper-Midwestern United States. *Journal of Environmental Quality*, 26(5), 1223–1230. doi: 10.2134/jeq1997.00472425002600050005x.

Murray, K. & Conner, M. M. (2009). Methods to quantify variable importance: Implications for the analysis of noisy ecological data. *Ecology*, 90(2), 348–355. doi: 10.1890/07-1929.1.

Nagpal, C., Li, X., & Dubrawski, A. (2021). Deep survival machines: Fully parametric survival regression and representation learning for censored data with competing risks. *IEEE Journal of Biomedical and Health Informatics*, 25(8), 3163–3175. doi: 10.1109/JBHI.2021.3052441.

Nakagawa, S. (2015). Missing data: Mechanisms, methods, and messages. In G. A. Fox, S. Negrete-Yankelevich, & V. J. Sosa (Eds.), *Ecological Statistics: Contemporary Theory and Application* (pp. 81–105). Oxford, UK: Oxford University Press.

Newman, A. J., Clark, M. P., Sampson, K., et al. (2015). Development of a large-sample watershed-scale hydrometeorological data set for the contiguous USA: Data set characteristics and assessment of regional variability in hydrologic model performance. *Hydrology and Earth System Sciences*, 19(1), 209–223. doi: 10.5194/hess-19-209-2015.

Persson, A. (2020). Right for the wrong reason? A new look at the 6 June 1944 D-day forecast by a neutral Swede. *Bulletin of the American Meteorological Society*, 101(7), E993–E1006. doi: 10.1175/BAMS-D-18-0311.1.

Pritzker, B. (1998). *A Native American Encyclopedia*. Oxford, UK: Oxford University Press.

Radin, J. (2017). "Digital natives": How medical and indigenous histories matter for Big Data. *Osiris*, 32(1), 43–64. doi: 10.1086/693853.

Rieber, N., Knapp, B., Eils, R., & Kaderali, L. (2009). RNAither, an automated pipeline for the statistical analysis of high-throughput RNAi screens. *Bioinformatics*, 25(5), 678–679. doi: 10.1093/bioinformatics/btp014.

Roux, S., Emerson, J. B., Eloe-Fadrosh, E. A., & Sullivan, M. B. (2017). Benchmarking viromics: An *in silico* evaluation of metagenome-enabled estimates of viral community composition and diversity. *PeerJ*, 5, e3817. doi: 10.7717/peerj.3817.

Sabatini, F. M., Jiménez-Alfaro, B., Jandt, U., et al. (2022). Global patterns of vascular plant alpha diversity. *Nature Communications*, 13(11), 4683. doi: 10.1038/s41467-022-32063-z.

Schilder, B. M., Humphrey, J., & Raj, T. (2022). echolocatoR: an automated end-to-end statistical and functional genomic fine-mapping pipeline. *Bioinformatics*, 38(2), 536–539. doi: 10.1093/bioinformatics/btab658.

Schmolke, A., Thorbek, P., DeAngelis, D. L., & Grimm, V. (2010). Ecological models supporting environmental decision making: A strategy for the future. *Trends in Ecology & Evolution*, 25(8), 479–486. doi: 10.1016/j.tree.2010.05.001.

Sclove, S. L., Morris, C., & Radhakrishnan, R. (1972). Non-optimality of preliminary-test estimators for the mean of a multivariate normal distribution. *The Annals of Mathematical Statistics*, 43(5), 1481–1490. doi: 10.1214/aoms/1177692380.

Sheynin, O. (1995). Helmert's work in the theory of errors. *Archive for History of Exact Sciences*, 49(1), 73–104. doi: 10.1007/BF00374700.

Shi, B.-Y. (2016). The importance and strategy of diabetes prevention. *Chronic Diseases and Translational Medicine*, 2(4), 204–207. doi: 10.1016/j.cdtm.2016.11.013.

Silberzahn, R., Uhlmann, E. L., Martin, D. P., et al. (2018). Many analysts, one data set: Making transparent how variations in analytic choices affect results. *Advances in Methods and Practices in Psychological Science*, 1(3), 337–356. doi: 10.1177/2515245917747646.

Simmons, J. P., Nelson, L. D., & Simonsohn, U. (2011). False-positive psychology: Undisclosed flexibility in data collection and analysis allows presenting anything as significant. *Psychological Science*, 22(11), 1359–1366. doi: 10.1177/0956797611417632.

Smith, J., Everhart, J., Dickson, W., et al. (1988). Using the ADAP learning algorithm to forcast the onset of diabetes mellitus. *Proceedings of the Annual Symposium on Computer Applications in Medical Care*, 10, 261–265.

SoRelle, J. A., Wachsmann, M., & Cantarel, B. L. (2020). Assembling and validating bioinformatic pipelines for next-generation sequencing clinical assays. *Archives of Pathology & Laboratory Medicine*, 144(9), 1118–1130. doi: 10.5858/arpa.2019-0476-RA.

Steffens, C., Ballen, S. C., Scapin, E., et al. (2022). Advances of nanobiosensors and its application in atrazine detection in water: A review. *Sensors and Actuators Reports*, 4, 100096. doi: 10.1016/j.snr.2022.100096.

Street, W. N., Wolberg, W. H., & Mangasarian, O. L. (1993). Nuclear feature extraction for breast tumor diagnosis. In *Proceedings of SPIE—The International Society for Optical Engineering* (pp. 861–870). San Jose, CA: Society of Photo-optical Instrumentation Engineers.

Su, Y.-S., Gelman, A., Hill, J., & Yajima, M. (2011). Multiple imputation with diagnostics (mi) in R: Opening windows into the black box. *Journal of Statistical Software*, 45(2), 1–31. doi: 10.18637/jss.v045.i02.

Sun, J. (2006). *The Statistical Analysis of Interval-Censored Failure Time Data*. New York, NY, USA: Springer.

Sun, S., Koch, L. M., & Baumgartner, C. F. (2023). Right for the wrong reason: Can interpretable ML techniques detect spurious correlations? In H. Greenspan, A. Madabhushi, P. Mousavi, S. Salcudean, J. Duncan, T. Syeda-Mahmood, & R. Taylor (Eds.), *Medical Image Computing and Computer Assisted Intervention—MICCAI 2023* (pp. 425–434). Cham, Switzerland: Springer.

Thurman, E. M., Goolsby, D. A., Meyer, M. T., & Kolpin, D. W. (1991). Herbicides in surface waters of the midwestern United States: The effects of spring flush. *Environmental Science and Technology*, 25(10), 1794–1796. doi: 10.1021/es00022a018.

Thurman, E. M., Goolsby, D. A., Meyer, M. T., et al. (1992). A reconnaissance study of herbicides and their metabolites in surface waters of the midwestern United States using immunoassay and gas chromatography/mass spectrometry. *Environmental Science and Technology*, 26(12), 2440–2447. doi: 10.1021/es00036a016.

Tobin, J. (1958). Estimation of relationships for limited dependent variables. *Econometrica*, 26(1), 24–36. doi: 10.2307/1907382.

USGS (2018). How Streamflow is Measured. https://www.usgs.gov/special-topics/water-science-school/science/how-streamflow-measured. Accessed 16 January 2024.

van Buuren, S. & Groothuis-Oudshoorn, K. (2011). mice: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software*, 45(3), 1–67. doi: 10.18637/jss.v045.i03.

Viallefont, V., Raftery, A. E., & Richardson, S. (2001). Variable selection and Bayesian model averaging in case-control studies. *Statistics in Medicine*, 20(21), 3215–3230. doi: 10.1002/sim.976.

Wang, B., Sudijono, T., Kirveslahti, H., et al. (2020). A statistical pipeline for identifying physical features that differentiate classes of 3D shapes. *Annals of Applied Statistics*, 15(2), 638–661. doi: 10.1214/20-AOAS1430.

Wang, P., Li, Y., & Reddy, C. K. (2019). Machine learning for survival analysis: A survey. *ACM Computing Surveys*, 51(6), 110. doi: 10.1145/3214306.

White, H. (2000). A reality check for data snooping. *Econometrica*, 68(5), 1097–1126. doi: 10.1111/1468-0262.00152.

Wikipedia Contributors (2023). Propagation of uncertainty—Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Propagation_of_uncertainty&oldid=1174273326. Accessed 25 December 2023.

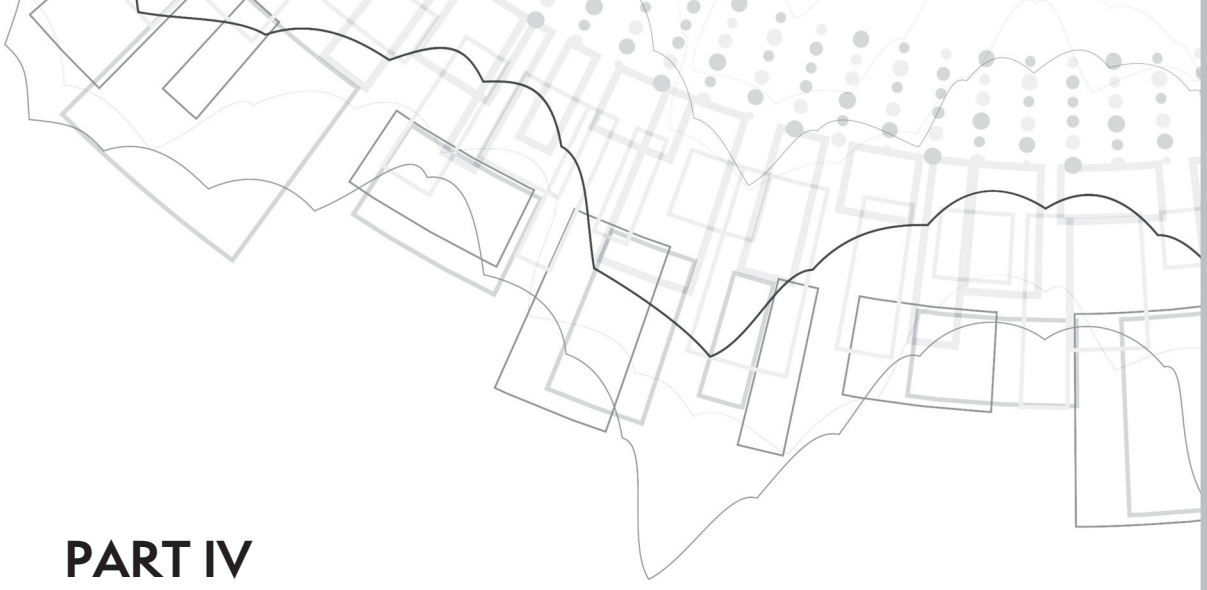
Williams, M. R., Lamont, B. B., & He, T. (2022). Dealing with 'the spectre of "spurious" correlations': Hazards in comparing ratios and other derived variables with a randomization test to determine if a biological interpretation is justified. *Oikos*, 2022(3), e08575. doi: 10.1111/oik.08575.

Wolberg, W., Mangasarian, O., Street, N., & Street, W. (1995). Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository, 10.24432/C5DW2B.

Yan, J. (2023). Multivariate modeling with copulas and engineering applications. In H. Pham (Ed.), *Springer Handbook of Engineering Statistics* (pp. 931–945). London, UK: Springer.

Zayed, A. A., Wainaina, J. M., Dominguez-Huerta, G., et al. (2022). Cryptic and abundant marine viruses at the evolutionary origins of Earth's RNA virome. *Science*, 376(6589), 156–162. doi: 10.1126/science.abm5847.

Zhang, J. & Storey, K. B. (2016). RBioplot: An easy-to-use R pipeline for automated statistical analysis and data visualization in molecular biology and biochemistry. *PeerJ*, 4, e2436. doi: 10.7717/peerj.2436.



PART IV

Post exemplum: Diagnostic simulations

A great deal of thought went into the planning, design, and execution of research, all of which ideally should have included prospective simulations and pre-registration (part II). As a result, the analyses and model fitting may have been much simpler but still should have included some simulations to explore the consequences of violating assumptions, using popular methods, or revising them (part III). However, we rarely complete a project with only the analysis and models. From the analyses, we draw inferences about the reliability and reproducibility of our theories. If our model is a structural misfit, we cannot rely on its parameters. Hence, using model diagnostics to check the validity of our analyses and models is an important part of the our results.

The two chapters in this part address how to evaluate the analyses we have done and the models we have fit. In short, we ask, did the analyses do what we thought they did, did they do it accurately, and do the models actually fit the data well? We illustrate how to use simulations after fitting a model, to investigate whether the parameter estimates seem reasonable, and whether the residuals are well-behaved or not (Chapter 8). These checks help us determine what the model actually means for prediction and to think about what to do with results that are not statistically significant. One cause of unreliable and irreproducible statistical findings is that many non-statisticians do not know what to do when their analysis yields a non-significant result. In Chapter 9, we discuss what we can or should do with non-significant findings and the problems associated with retrospective power analysis. Most of these problems are based on well-known and repeatedly derived analytical formulas, but simulation still proves to be useful for specific types of retrospective analyses.

8

Evaluating models: How well do they really fit?

When we fit a statistical model to our data, we do much more than test an hypothesis. In fact, we are implicitly quantifying—that is, predicting—what we expect to see if additional data were collected. Despite our best efforts, however, the predictions from our models (the point estimates, to be precise) often are wrong, for at least two reasons. First, our model may have been inappropriate—wrong predictors, failed assumptions, etc. Second, even if the model is appropriate, the predictions may be very uncertain. Quantifying this uncertainty is important for evaluating a prediction, and actually is as important as estimating the parameters themselves. That is because making a wrong prediction when we are sure our model is correct is much worse than being wrong when we were doubtful about our model. In this chapter, we present several examples of how to use simulations to compute and visualise the uncertainty in our predictions.

Before we can visualise and quantify the uncertainty in our model, however, we first may want to quantify our expectation before we fit our model to the data. Unless our study truly is sui generis or we have never found a paper on the subject, we should have some idea of the range of measurements we expect to make or the response we expect to see as a result of our experimental treatments. Recall from §3.3.1 in Chapter 3 that when we use Bayesian inference, our expectation before seeing the data or fitting the model is called the prior distribution (or "prior" for short). Even if we are not using Bayesian methods, however, thinking about our prior is very useful to answer the question: What do we expect to learn from our analysis that we did not know before we did the study?

If we specify a prior before we start our study, we can simulate from it what we would expect to learn after we complete our study. You may think that the uninformative (flat or vague) priors used by default in many Bayesian software packages may not be very useful in this regard. Think again. An uninformative prior forces us to face the consequences of our decision to use it: could our expected results really take on *any* value? Probably not. We do, of course, get a much better sense of the (implicit) constraints in our study and our models from informative priors, and this is especially true for hierarchical models.

Simulation from an informative prior also will tell us whether the data we are about to collect (i.e., the likelihood) is likely to change our model substantially; that is, whether we actually expect to learn something new from our study.[1] It could be argued that the ultimate goal of scientific inquiry is to know how the world works, so at some point any new

[1] More formally, if we really are going to learn something new from our study, we expect the likelihood to dominate the prior, in which case the posterior distribution of the model either will have less uncertainty than the prior or have a very different expected value. See §3.3.1 in Chapter 3.

data should align nicely with our priors. However, even in that world, the uncertainty in our knowledge could still lead to surprising results.[2]

We draw all of these themes together throughout this chapter. In the main, we are interested in visualising the consequences of uncertainty in parameter estimation, especially for effect plots and for single-point predictions. Using (multivariate) Monte Carlo draws to plot the range of possible relationships between predictors and responses also can give us a much better sense of the variability in our predictions than confidence intervals, even though they have essentially the same information content (see also Figure 2.4 in Chapter 2).

Indeed, for a prediction, we can use this same approach to quantify not only the confidence interval (for the expectation) but also the prediction interval (for a new observation). Such plots may be a familiar part of the toolkit of classical parametric and hierarchical models, but they are underused with machine-learning models. Thus, we include in §8.5.5 a simulation that allow us to peek into the otherwise opaque box of machine- or deep-learning models, and use the simulation to begin to identify which predictors actually are the most influential for a given prediction (i.e., so-called "explainable AI"; Phillips et al. 2021). Whether we are presenting classical parametric or machine-learning models, however, our simulations and associated plots illustrate in a clear and simple way how variable we expect a new measurement to be, which is usually much more variable than we thought when we defined our prior. All our examples also show how simulation leads to a deeper understanding of what our fitted models actually predict.

What you will learn from this chapter

- How to use prior information effectively to determine how much you can expect to learn from a study.
- Alternatives to simple bivariate plots that can improve the understanding and communication of multivariate models.
- The importance of diagnostic plots.
- The difference between confidence intervals and prediction intervals.

8.1 Learning from the prior

In a Bayesian analysis, we have to specify prior probability distributions (and the correlations among them) for all model parameters. As we introduced in §3.3.1 in Chapter 3, these priors represent our knowledge of the system before we do our study and are based on data and results that are different from the data we will collect and analyse. Typically, we gain this prior knowledge from a pilot study or a review and synthesis of existing literature on our research topic. From either or both of these efforts, we extract reasonable estimates, variances, and ranges for key parameters. Sometimes physical or chemical constraints also dictate or constrain the shape or range of the prior. Finally, priors can be elicited from experts (e.g., Azzolina et al. 2021; Falconer et al. 2022).

Regardless of how we got our priors, however, once we have them, we can simulate data from them. Such simulations represent what the data would look like if our priors were accurate. We also can analyse such simulated data to show us what the analysis is likely to reveal. Importantly, these data and the analysis define our expectations before we collect, look at, and analyse new data. Simulating from the prior is a way to show us the implications

[2] For example, we know that rolling a fair die should, on average, turn up a six in one-sixth of the rolls. However, the actual result of rolling the die is still unpredictable and hence "surprising," even though we would not be surprised when we occasionally really do roll a 6. Shannon's (1948) information-theoretic approaches are centred on quantifying and comparing how surprising data really are (Freiberger 2015).

of our analysis; for example: How steep could a slope become? How much scatter can we expect? Is a positive relationship between x and y inevitable (Doak et al. 1998; Frank 2014). Further, such simulations can help us think through what the most important figure or table will be once we complete our study (recall Figure 3.1 in Chapter 3). This, in turn, focuses our attention on the most appropriate analyses to do once we have completed our study, and also may encourage us to preregister our study design (§3.5 in Chapter 3). We first illustrate these ideas with a simple, didactic example, and then we return to them with a more complex example at the end of the chapter (§8.6).

8.1.1 A didactic example: Alice's adventures in restaurants

Alice has taken notes on her recent experiences in five different restaurants. She skims the data and senses some correlation between the formality of the setting (attentiveness of the maître d'hôtel, cleanliness of the table cloths, how many types of wine glasses, forks, knives, and spoons are pre-set on the table, etc.) and how much she enjoyed her meal. She has only been to a handful of different restaurants so far, and when her old friend Arlo comes to town for a visit, she wants to impress him by taking him to two new (for her) and (she hopes) good restaurants. Assuming she continues to make comparable observations of setting and enjoyment when she goes to these two new restaurants, how much can she actually expect to learn about the relationship between setting and enjoyment from these additional two data points?

We can use a simple simulation to answer this question. The key point here is that Alice has an informative prior based on her five previous restaurant visits (this is similar to having an informative prior based on a small pilot study). Hence, Alice's expectation is not evenly distributed (i.e., it is not uniform). If the two new restaurants fall on or are very close to her prior expectation, she will not actually learn much about the estimate (i.e., the slope of the relationship between setting and enjoyment), but she still will reduce the uncertainty around her estimate. On the other hand, if her experiences at the new restaurants depart substantially from her prior expectation, she might want to revise her hypothesis in light of the new data.

Thus, we have two sequential goals with our simulation. First, we want to quantify Alice's expectation based on her prior. Second, we want to compare the prior and posterior probability distributions to evaluate how much Alice learned from dining at two new restaurants. Our simulation (Code Box 8.1) includes four steps:

1. Define the prior by fitting a regression to the data from the assessments of the five restaurants that Alice has already visited;
2. Use this prior in a Bayesian analysis that includes the new observations from the two restaurants that Alice goes to with Arlo;
3. Plot the regression using the prior data only and with the full dataset that includes observations from the two new restaurants. This plot illustrates how much the new data change Alice's perception of the relationship between setting and enjoyment. Parameter estimates quantify any change in her perception of the relationship;
4. Plot the prior and posterior for each of the two new restaurants. This plot allows us to see how much the two new observations changed the expectation for an individual restaurant.

Note that we do not need to repeat this simulation 1,000 (or any other number of) times because we are investigating the contribution of two new data points whose values we already know, rather than those we expect to collect. We would need replicate simulations only if we were were still in the design (prospective) stage of our study or if we were forecasting from existing data (see §8.6, below).

Code Box 8.1: Alice's Bayesian restaurant experience

```
# Load required libraries
 library(mvtnorm) # For multivariate normal random numbers
 library(brms) # For fitting Bayesian generalised (non-)linear multivariate
      multilevel models using Stan

# Make up data
 restaurant <- data.frame("Name"=c("Blue Lagoon", "Joe's Finger Food",
 "Beanache", "Real Saigon", "Tete a Tete"),
  "Formality" = c(5, 3, 7, 7, 9),
  "Enjoyment" = c(3, 6, 4, 6, 8))

# Model the data using ordinary linear regression and summarise the model
 fm <- lm(Enjoyment ~ Formality, data=restaurant)
 summary(fm)

| <--snip--->
| Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept)   3.1346     2.9122   1.076    0.361
| Formality     0.3654     0.4462   0.819    0.473
|
| Residual standard error: 2.035 on 3 degrees of freedom
| Multiple R-squared:  0.1827,  Adjusted R-squared:  -0.08974
| F-statistic: 0.6706 on 1 and 3 DF,  p-value: 0.4728

# Simulation
 set.seed(1234)
 priorSimus <- rmvnorm(1000, mean=coef(fm), sigma=vcov(fm)) # Prior based on
      estimate and covariance matrix from model "fm"

# Add data from the two new restaurants
 newRestaurant <- data.frame("Name"=c("Paradise", "Backstreet"),
  "Formality" = c(2, 5),
  "Enjoyment" = c(2, 7))

# Draw from priors:
 priorExpectation <- sapply(1:1000, function(x) priorSimus[x,1] + priorSimus[x,2]
       * newRestaurant$Formality)

# Compute posterior:
 bprior <- prior(multi_normal(M, V), class = "b") # See ?stanvar for syntax
 # extract priors as estimates from previous LM:
 stanvars <- stanvar(unname(coef(fm)), "M", scode = "  vector[2] M;") +
   stanvar(unname(vcov(fm)), "V", scode = "  matrix[2, 2] V;")
 fit <- brm(Enjoyment ~ 0 + Intercept + Formality, data = newRestaurant, prior =
      bprior, stanvars = stanvars)
 fit
```

```
 Family: gaussian
  Links: mu = identity; sigma = identity
Formula: Enjoyment ~ 0 + Intercept + Formality
   Data: newRestaurant (Number of observations: 2)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Population-Level Effects:
          Estimate Est.Error l-95%CI u-95%CI Rhat Bulk_ESS Tail_ESS
Intercept     2.64      2.31   -1.62    7.36 1.00     1259     1309
Formality     0.47      0.38   -0.29    1.20 1.00     1274     1404

Family Specific Parameters:
     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     3.19      2.03     1.01     8.44 1.00     1865     1825
<--snip--->
```

Comparing the two regressions—one using only the prior data (grey line and points in Figure 8.1) and the other using the prior data together with the new data from the two new restaurants (black line and the two additional black points in Figure 8.1)—we can see clearly that the data from the two new restaurants have reduced the uncertainty around the regression line but have had very little effect on the slope itself. Finally, even though the actual observations at each restaurant deviated quite a bit from the precise expectation, there was no effect on Alice's overall expectation for her next visit to another fancy restaurant (Figure 8.2).

8.2 What does a model tell us, and what does it not tell us?

Once we have collected our data, we routinely use a model to test an hypothesis. To do this, we first have to define the model structure. We may have done this based on our prior, but we could also define it based on other knowledge about how the system we are studying works. Either way, when we assume that the model is a good representation of the specific theory we are testing, we impose or force the model parameters onto the data. If the model structure is reasonable and includes the correct predictors (or at least very close correlates of them), then the fitted model parameters are, indeed, informative. For example, if we observe a steep, positive relationship (i.e., the slope) between predictor and response, we can interpret the result at face value as a strong, positive effect.

In other cases, however, a fitted model is, in and of itself, not as informative as it may seem at first glance. When the model structure is only poorly related to the data, perhaps because we cannot measure the real causal predictors and have to work with noisy or incomplete surrogates, maximisation of the log-likelihood will still ensure a fit that is as good as possible, given the model structure. The results may include biased estimates and poor predictions, both of which are not obvious in the estimates returned by the model. We regularly use two strategies to assess the quality of a model: bootstrapping, to get a fair estimate of uncertainty, particularly for small data and for workflows (see §7.8 in Chapter 7); and validation of the model applied to independent data.

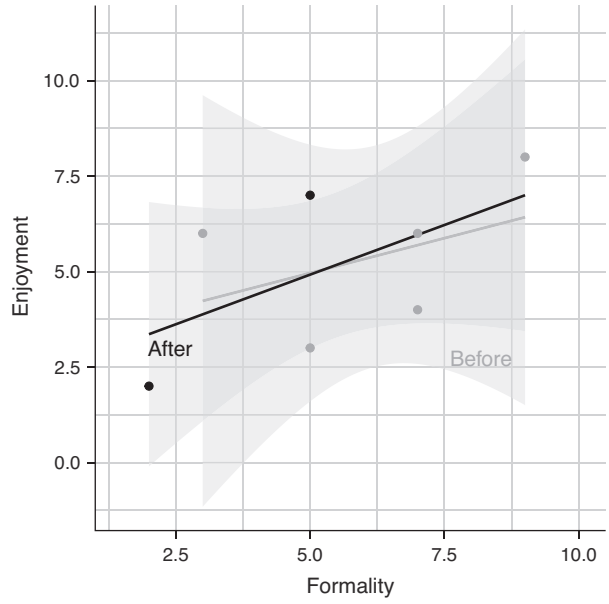


Figure 8.1 Alice's prior belief (in the Bayesian sense) about the relationship between formality and enjoyment in a restaurant (grey line) based on dining at five different restaurants (grey points), and the updated belief (and its uncertainty; black line) after dining at two additional restaurants (black points).

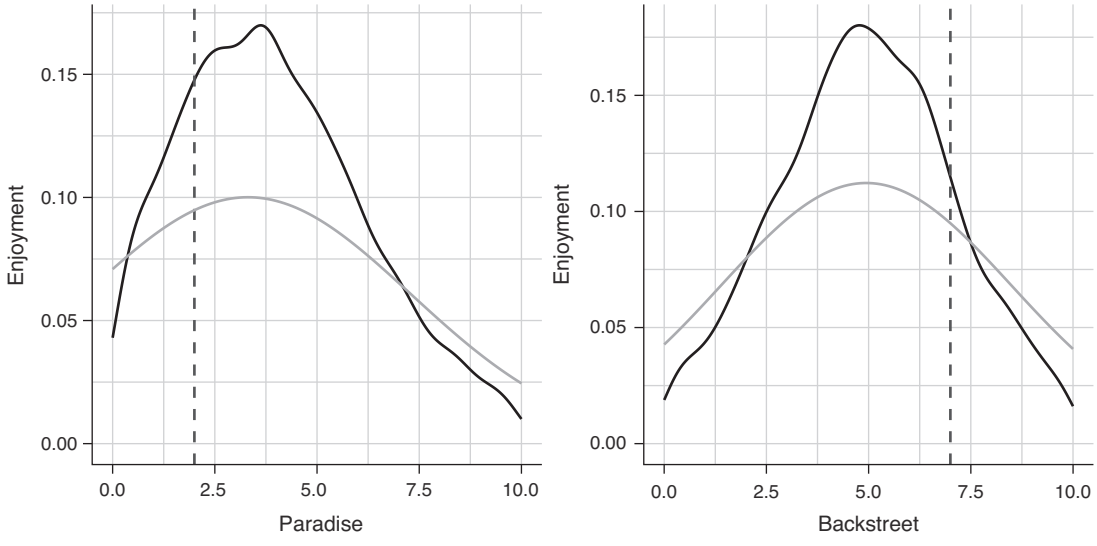


Figure 8.2 Alice's expectation of enjoyment at the restaurants "Paradise" and "Backstreet" before visiting them (grey probability distributions). Afterwards (black probability distributions), her expectation for a restaurant with her index of formality equal to 2 or 7 (dashed vertical lines) would be similar, but now she would be more certain about it.

AN ILLUSTRATIVE EXAMPLE: HINDCASTING AND FORECASTING TEMPERATURE DATA

To illustrate how such validation works, we briefly (and superficially) analyse temperature records for the northern hemisphere, based on data collected between 1856 and 2000 at a number of weather stations.[3]

SPLIT THE DATASET INTO TRAINING AND TESTING SETS

We fit our model on the data collected from 1881 to 1974 (the "training" data), and then validate it on the data collected earlier (1855–1880) and later (1975–2000). We use a general additive model (GAM) and fit a spline to the training data. We then use the parameter estimates to hindcast the temperatures before, and forecast the temperatures after, the years included in the training data (Code Box 8.2).

HOW WELL DOES THE MODEL FIT?

The model output (at the end of Code Box 8.2) indicates a decent fit; 69% of the variance is explained by a moderately complicated spline with six degrees of freedom. Model diagnostics are fine (we will look at and discuss them later; see §8.4 and Figure 8.7, below), so this looks like a good model. The fit plotted on the test data (between the vertical lines in Figure 8.3) appears reasonable. Note, however, the uncertainty communicated in the output. The residual variance (`Scale est.` in the last line of Code Box 8.2) is ≈ 0.012, so the standard error is $\approx \sqrt{0.012}$°C ≈ 0.11°C. Since the range of temperature anomalies in the fitted data is only ≈ 0.4°C, the residual variance suggests a substantial amount of uncertainty.

HINDCASTING AND FORECASTING USING BOOTSTRAPPING

Bootstrapping the GAM (Code Box 8.3) and plotting the hind- and forecasts (Figure 8.3) clearly illustrate the uncertainty implicit in the model that is not apparent in the printed output (Code Box 8.2). We can easily see that the performance of the hindcast may be completely independent of the performance of the forecast (Figure 8.3; the hindcasts are the fitted lines to the left of the left-hand vertical line, whereas the forecasts are the fitted lines to the right of the right-hand vertical line). A possible reason for the mismatch between the good fits of the hindcasts and the poor fits of the forecasts is that global warming took a two-decade break in the 1960s and 1970s (because of aerosol formation caused by heavy sulfur emissions at that time; IPCC 2001) before becoming once again much more prominent in the 1980s. Our model forecasts are, however, substantially affected by the most recent trend in our training data, which in this case incorrectly predicts a continuous decline in northern hemisphere temperature after 1975 (Figure 8.3, right of the vertical line at 1975).

CONCLUSIONS

We conclude from this simple illustration that the basic output of a model can be useful for inference, but not necessarily much more. We can use additional ways to explore model fits to determine just how much more we can learn from fitting a particular model. These include: different ways to visualise a fitted model with many predictors and their interactions; meaningfully checking model assumptions using model diagnostics; and computing and plotting both prediction and uncertainty in ways that are relevant to the specific question or hypothesis. Each of these are explored in turn in the subsequent sections of this chapter.

[3] The data are actually temperature anomalies: deviations from the overall mean temperature (1856–1980). See Jones & Mann (2004) for details.

Code Box 8.2: Fit a trend to the northern hemisphere temperature anomalies using a GAM

```
# Load required libraries
 library(faraway) # Includes northern hemisphere temperature data
 library(mgcv) # For GAM

# Load dataset
 data(globwarm) # Northern hemisphere temperature data
  globwarm$year <- as.numeric(rownames(globwarm)) # Defines new variable "year"
  train <- globwarm[globwarm$year > 1880 & globwarm$year < 1975, ] # Training
      data
  before <- globwarm[globwarm$year <= 1880 & globwarm$year > 1855, ] # Data
       for hindcasting
  after <- globwarm[globwarm$year >= 1975, ] # Data for forecasting

# Fit the GAM
 fm <- gam(nhtemp ~ s(year), data=train)
 summary(fm)

| <--snip--->
| Approximate significance of smooth terms:
|           edf Ref.df     F p-value
| s(year) 5.966  7.126 26.25  <2e-16 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| R-sq.(adj) =  0.667   Deviance explained = 68.8%
| GCV = 0.012055  Scale est. = 0.011161  n = 94
```

8.3 Visualising more complex effects: Conditional, marginal, and partial plots

Visualisation of model fits along with the data is trickier for more complex models, especially multiple linear or nonlinear regression models with many predictors and high-order interactions, or machine-learning models. The difficulties arise at least in part because a potentially large proportion of the variability in the data can be explained by predictors that are not shown on any given plot.

AN ILLUSTRATIVE EXAMPLE: PREFERENCES FOR WINE AS A FUNCTION OF ITS CHEMICAL PROPERTIES

An individual's preference for wine can be modeled as a function of its chemical properties, such as pH, alcohol and sugar concentrations, and amounts of trace minerals (Cortez et al. 2009a). We used a standard wine-chemistry dataset often utilised in machine-learning competitions (Cortez et al. 2009b) as a basis for a straightforward simulation relating wine preference to multiple variables that characterise its chemistry (Code Box 8.4). The fitted model (which, in our simulation, includes two spurious predictors) very nicely estimates the true parameters.

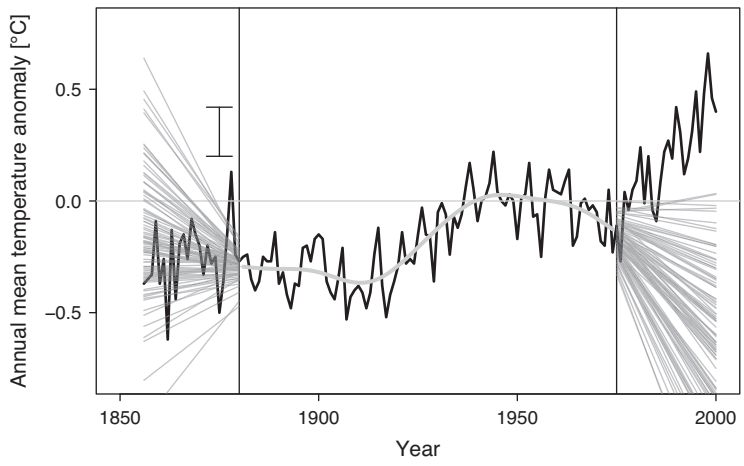


Figure 8.3 Temperature of the northern hemisphere expressed as the annual anomaly relative to the full period 1856–1980 (black line), and model fit of the GAM to the training period (grey line between 1880 and 1975). Grey lines to the left and right are hindcasts to the past and forecasts to the future of the training data. Hindcasts and forecasts are from 100 bootstrapped replicates of the GAM and represent the uncertainty inherent in the fitted model. Note that hindcasting is uncertain (error bar indicates RMSE for hindcasts) but largely unbiased, whereas forecasting is completely outside (below) the range of the observed data.

Code Box 8.3: Hindcasting and forecasting of global temperature using bootstrapping and the GAM

```
# Continues with dataset and libraries from Code Box 8.1
 B <- 100 # number of bootstraps
 set.seed(123)
 modellist <- replicate(B, simplify=F,
    {
        fmbs <- gam(nhtemp ~ s(year), data=train[sample(nrow(train),
            replace=T), ])
    })
 predsHind <- sapply(modellist,  predict, newdata=before, simplify=F) # Hindcasts
 predsFore <- sapply(modellist,  predict, newdata=after, simplify=F)  # Forecasts

 RMSEs <- sapply(predsHind, function(x)  sqrt(mean((before$nhtemp - x)^2)) )
     # Errors for hindcasts
 mean(RMSEs)

| [1] 0.2198188
```

SIMPLE PLOTS OBSCURE THE MODEL FIT

Plotting wine preference as a function of a single (non-linear) predictor `fixed.acidity` (i.e., the concentration of tartaric acid in g/L) yields a full impression of the variability of the data (Figure 8.4). The data plotted in Figure 8.4 do not suggest that there should be a downward trend in the fitted model for values of total acidity > 13. Further, this plot makes it appear as if the model fits the data poorly, when in fact it fits it very well ($R^2 = 0.98$; see the penultimate line of output in Code Box 8.4).

Code Box 8.4: Simulating wine preference as a function of its chemical properties

```
# Load the data (local copy). We used the red-wine dataset that we downloaded
    from
# https://archive.ics.uci.edu/dataset/186/wine+quality
# https://doi.org/10.24432/C56S3T
 wine <- read.csv("../Data/wine+quality/winequality-red.csv", sep=";")

# Simulation
 set.seed(123)
 wine$preference <- with(wine, -10 + 3*fixed.acidity - 0.15*fixed.acidity^2 +
     0.4*fixed.acidity*residual.sugar  + 1*residual.sugar - 0.01*total.sulfur.
     dioxide + 1*alcohol + rnorm(nrow(wine), 0, 1)) # Simulated "true" model

 summary(fm <- lm(preference ~ fixed.acidity + I(fixed.acidity^2) + fixed.acidity
     :residual.sugar + residual.sugar + total.sulfur.dioxide + alcohol +
     chlorides + sulphates, data=wine))

| <--snip--->
| Coefficients:
|                               Estimate Std. Error t value Pr(>|t|)
| (Intercept)                 -10.191777   0.608656 -16.745   <2e-16 ***
| fixed.acidity                 3.018790   0.105892  28.508   <2e-16 ***
| I(fixed.acidity^2)           -0.152210   0.005826 -26.127   <2e-16 ***
| residual.sugar                0.913638   0.080795  11.308   <2e-16 ***
| total.sulfur.dioxide         -0.009991   0.000800 -12.490   <2e-16 ***
| alcohol                       1.021549   0.025540  39.998   <2e-16 ***
| chlorides                     0.208622   0.591290   0.353    0.724
| sulphates                     0.130206   0.163192   0.798    0.425
| fixed.acidity:residual.sugar  0.404925   0.009289  43.591   <2e-16 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 0.9851 on 1590 degrees of freedom
| Multiple R-squared:  0.9804,  Adjusted R-squared:  0.9803
| F-statistic:  9945 on 8 and 1590 DF,  p-value: < 2.2e-16
```

MULTIVARIATE PLOTS, FOUR WAYS

There are four main ways of plotting the effect of a single response variable Y as a function of a single, focal predictor (X_f) that is one of many predictors used in a multivariate model:

1. A *conditional* plot, in which the predicted values $\hat{Y}$ are a function of a changing X_f, while each of the other predictors (called the "complement," $\{X_c\}$) is set to some fixed value, typically its median. The line representing this conditional fit to the simulated wine data is shown in grey in Figure 8.4.
2. A *marginal* plot, in which each observed combination of $\{X_c\}$ (i.e., every row) is used once to predict the effect of X_f; the predicted value of $\hat{Y}$ is the average of these combinations

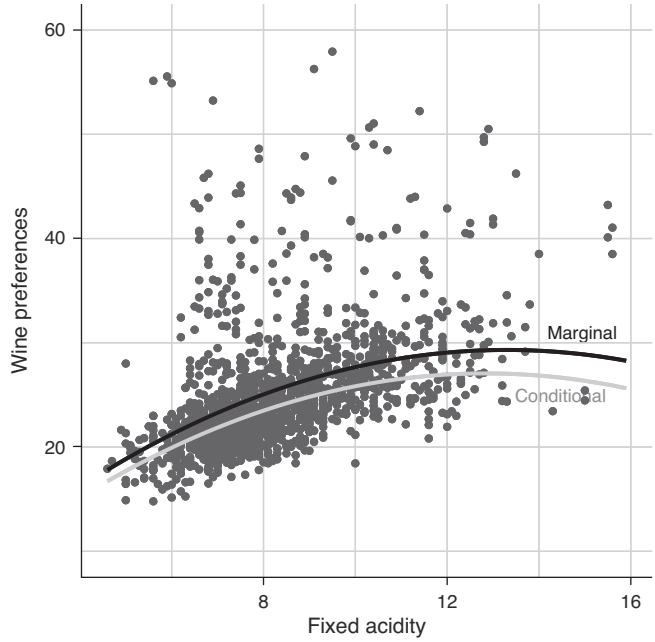


Figure 8.4 Preference for wine as a function of its fixed acidity. Lines are conditional (grey) and marginal (black) effects of fixed acidity. This plot nicely illustrates the full variability in the response, but it is not apparent from the plotted data why the fitted lines should curve downwards when $X > 13$.

for each X_f. The line representing this marginal fit to the simulated wine data is shown in black in Figure 8.4.

3. A *partial dependence* plot, in which the focal variable X_f is set to a constant value (e.g., its mean) when predicting with the training data to compute $\hat{Y}$. This represents the effect of the complement $\{X_c\}$ on the response; subtracting it from the observations yields a new variable Y', which is the variance in the data after accounting for X_f. Plotting Y' against X_f yields a very different impression of the data and the model fit (Figure 8.5, left panel).
4. A *partial residual* plot, where each of the complement predictors $\{X_c\}$ are set to a constant value (e.g., their means); the model then predicts $\hat{Y}$ as a function of the observed values of X_f. Finally, the residuals are added to this prediction and the resulting values are plotted (Figure 8.5, right panel). Partial dependence and partial residual plots typically are highly correlated with (and look similar to) one another, as both subtract the effect of the other predictors from the observations.

The algorithmic differences in computing predicted values $\hat{Y}$ in these four different ways can be seen in the code in Code Box 8.5.

Why do the conditional and marginal predictions suggest a decline in preference for wine when `fixed.acidity` > 13 even though the data themselves do not (Figure 8.4)? When we look at the two plots of the partial predictions, we realise that the actual effect of `fixed.acidity` is explained by the other predictors ($\{X_c\}$) that are not shown in Figure 8.4. The partial dependence and partial residual plots (Figure 8.5) include the effects of the other predictors $\{X_c\}$ on the predictions. Although this may feel like cheating, the resulting plots actually are a much better representation of what happens in the model, as they account for the effects of non-depicted predictors. In our wine-preference example, when we account for

Code Box 8.5: Four different predictions of preference for wine from its multiple chemical properties

```
# These models all take as input the data and model from Code Box 8.3

# Conditional predictions:
 newFA <- data.frame("fixed.acidity" = seq(min(wine$fixed.acidity),
      max(wine$fixed.acidity), len=100), t(apply(wine[,-1], 2, median)))
 newFA$predsFA <- predict(fm, newdata = newFA)

# Marginalpredictions:
 seqFA <- seq(min(wine$fixed.acidity), max(wine$fixed.acidity), len=100)
 newFAmarg <- data.frame("fixed.acidity" = rep(seqFA, times = nrow(wine)),
      wine[rep(1:nrow(wine), each = 100), 2:15] )
 preds <- predict(fm, newdata = newFAmarg)
 predsMeans <- tapply(preds, newFAmarg$fixed.acidity, mean)

# Partial dependence predictions:
 wineFAmean <- wine
 wineFAmean$fixed.acidity <- mean(wine$fixed.acidity)
 wine$FApredsPartial <- predict(fm, newdata = wineFAmean)
 wine$FAResidualsPartial <- wine$preference - wine$FApredsPartial
 newFApdp <- data.frame("fixed.acidity" = seq(min(wine$fixed.acidity),
      max(wine$fixed.acidity), len = 100), t(colMeans(wine[,-1])))
 newFApdp$predsFA <- predict(fm, newdata = newFApdp) - mean(wine$preference)

# Partial residual predictions:
 wineFAprp <- data.frame("fixed.acidity" = wine$fixed.acidity,
      t(colMeans(wine[,-1])))
 wineFAprp$partialRes <- predict(fm, newdata = wineFAprp) + residuals(fm)
 newFAprp <- data.frame("fixed.acidity" = seq(min(wine$fixed.acidity),
      max(wine$fixed.acidity), len = 100), t(colMeans(wine[,-1])))
 newFAprp$predictedPreference <- predict(fm, newdata = newFAprp)
```

the additional predictors, the quadratic effect seen in our initial plot (Figure 8.4) becomes more visible and more credible.

CONCLUSIONS

There is no right or wrong way to plot and visualise the results of a multivariate model. Each of the four different approaches listed above serves a different aim. Conditional and marginal plots illustrate clearly how relevant a specific predictor is for the entire analysis, whereas the partial plots give a much better impression of the data that inform the model. Conditional plots are much more common in publications, whereas the partial dependence plot is the default plot for many machine-learning plotting functions.[4]

[4] For example, a partial dependence plot is returned when plotting the `gam` output from the `mgcv` library, and when using the `pdp` library for plotting.

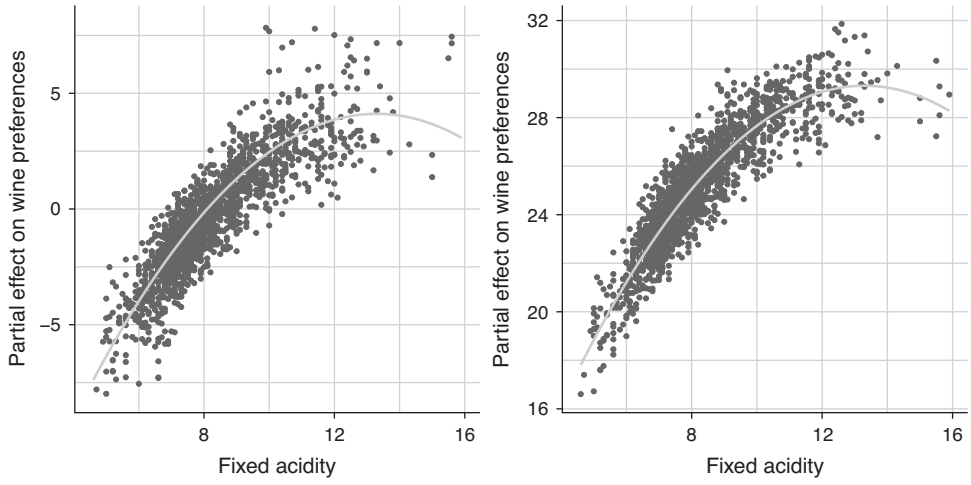


Figure 8.5 Partial predictions of an individuals preference for wine preference as a function of its fixed acidity. *Left*: Partial dependence plot of the effect of fixed acidity on preference for wine. The predictions here account for all the other predictors in the model. Note that the response variable in this plot is centred on zero. *Right*: Partial residual plot of the same effect fixed acidity on preference for wine. The predictions here are centred on $\bar{Y}$. Note that to emphasize the similarity of the plots, some values were clipped from the partial dependence plot so that the *y*-axes of both plots have roughly the same spread.

8.4 Model diagnostics

After we fit a model to data and visualise how well it describes the pattern(s) in the data, we routinely run a set of diagnostic checks to ensure that we did not miss any flagrant violations of assumptions or that a small number of points had an overwhelming influence on the model fit. For linear regression, the suite of model diagnostics is highly standardized (e.g. Crawley 2007; Good & Hardin 2012; Gotelli & Ellison 2012; Faraway & Augustin 2018; Dormann 2020b): residuals are plotted against predicted values to test for non-linearity in the data (the expectation is a 1:1 relationship); a quantile-quantile ("Q-Q") plot provides a quick visual assessment of whether or not the data meet normality assumptions; standardised residuals are plotted against predicted values to test for homoscedasticity (the expectation is no pattern); and residuals are plotted against leverage—and their relationship to Cook's distance is used to identify influential data points (outliers have large leverage and are outside the contour lines of Cook's distance) (Cox & Snell 1968; Pierce & Schafer 1986).[5] In Figure 8.6, we illustrate all four of these for the simple linear regression model we used to model wine preferences as a function of its chemical properties (Code Box 8.4). Visual inspection of these diagnostic plots suggests that the overall linear model is not unreasonable.

For non-Gaussian data and other types of models, however, these diagnostics are not always reliable. For example, residuals from time-series data are likely to be autocorrelated (see §2.4 in Chapter 2), whereas count and binary data are not Gaussian and should exhibit patterns in the residuals.

[5] Indeed, these diagnostics are so standard that simply plotting a linear model object in `R` using `plot(lm(x ~ y,...))` returns all four of these plots.

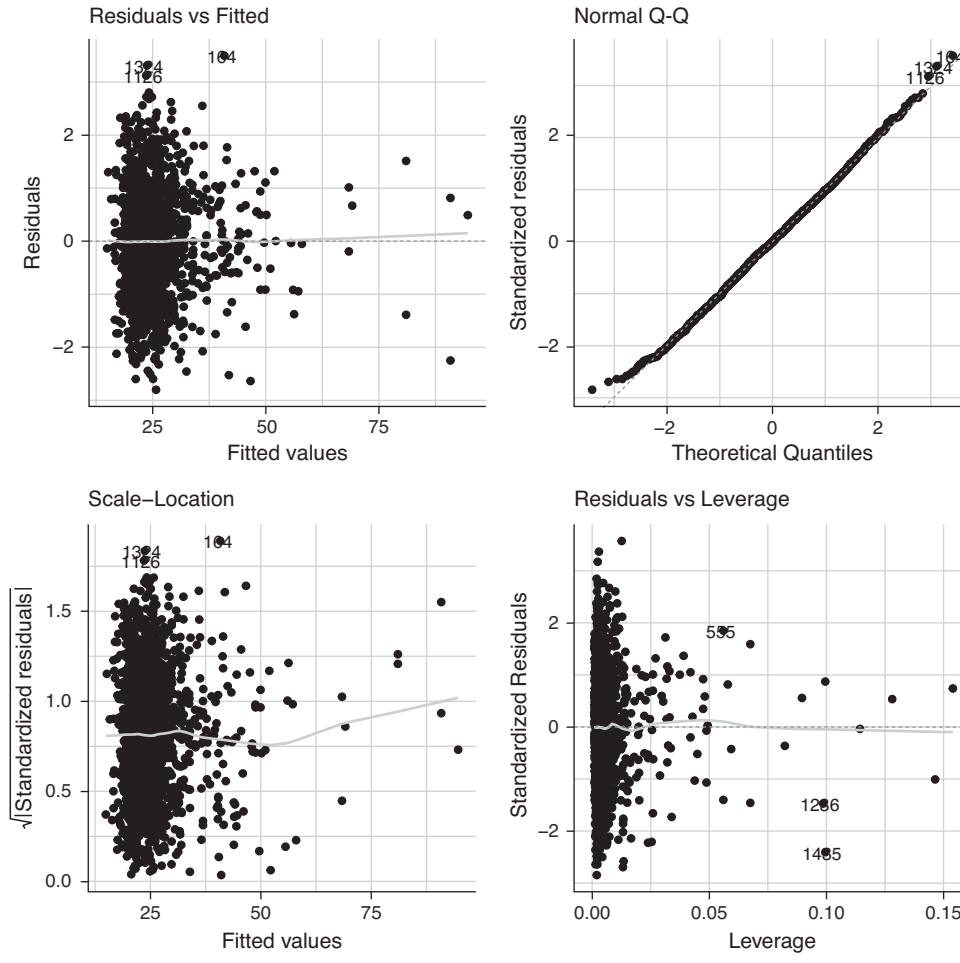


Figure 8.6 Standard diagnostic plots of the linear model for the wine data (Code Box 8.4). Clockwise from top left: Residuals versus fitted values; a quantile-quantile (Q-Q) plot; residuals versus leverage with contours of Cook's distance; and standardised residuals versus fitted values. These plots are automatically generated applying the `ggfortify::autoplot()` function to a linear model object generated by the `lm()` function in R.

8.4.1 Generalised diagnostics

Any quantitative model can make an in-sample prediction—predicting responses from the same set and combinations of predictor variables we used for fitting the model.[6] This is, at its most basic, the entire purpose of a statistical model: generating fitted values (i.e., predictions) from hypothesised causal variables.

One way to generalise model diagnostics flows from the idea that if a model is appropriate for describing the data, then data simulated using the fitted model should look like the real data. Following from this idea, an appropriate set of model diagnostics would compare the observed data to the simulated data. In a sense, the latter are used in the same way that

[6] No matter whether our model is univariate, multivariate, linear or nonlinear regression, a random forest, an agent-based simulation, or a calibrated set of stochastic differential equations, it has parameters and equations. From these parameters and equations, an "output" can be derived from the "input." If the input is the observed (or measured) range of predictor variables, the output is called the "fitted values" or, more generally the "prediction."

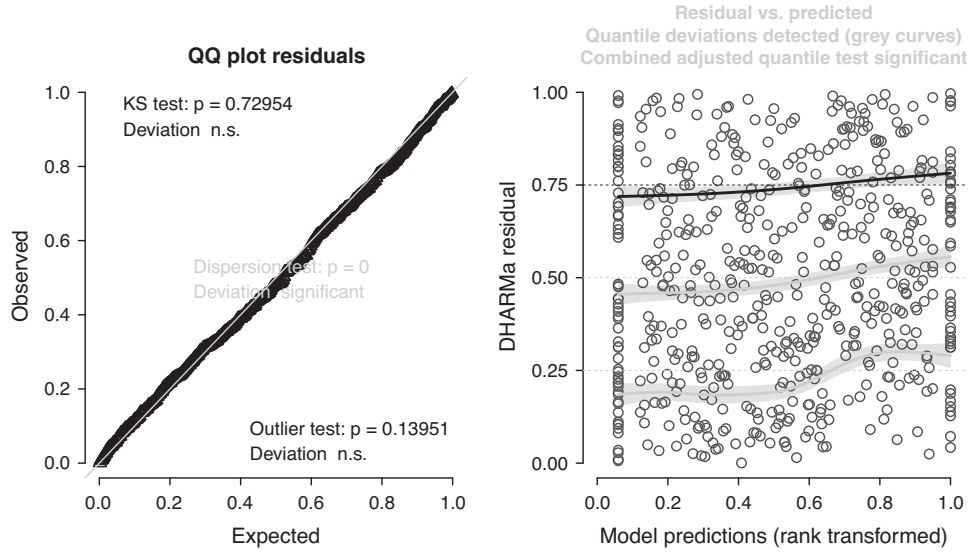


Figure 8.7 Default residual plots of the temperature analysis (§8.2) from the `DHARMa` package in `R`. *Left*: A standard quantile-quantile (Q-Q plot) with results of different tests for normality, outliers, and dispersion of residuals. *Right*: Regressions of the 25th, 50th, and 75th quantiles of the residuals. These plots are automatically generated by applying the `plot()` function to a `DHARMa` object. Grey text indicates deviation from expectation.

fitted values are used in the standard diagnostics of the linear model, whereas the difference (simple, squared, standardised, etc.) between the observed and fitted (i.e., simulated) values are the residuals. This idea is not new. It dates back at least to Pierce & Schafer (1986), was then improved on successively by Dawid (1984), Dunn & Smyth (1996), and Gelman & Hill (2007), and is now implemented nicely in the `DHARMa` library in `R` (Hartig 2022).[7] For example, plotting simulated residuals for the initial model that we fit to the northern-hemisphere temperature data from 1881 through 1974 (§8.2: Code Box 8.2 and Figure 8.3) suggests no deviations from the distributional assumptions for the residuals (Figure 8.7). Such simulation-based model diagnostics can be done similarly for any approach that is able to make predictions from a fitted model, including machine-learning and Bayesian (hierarchical) models.

USE QUANTILE RESIDUALS

Standard model diagnostics make good use of residuals, and for ordinary linear regression models, the residuals are expected to follow a Gaussian distribution. More generally, however, we use quantile residuals (which Hartig 2022 calls scaled residuals) in lieu of ordinary residuals for model diagnostics (Figure 8.7, right panel).

The general approach to using quantile residuals for model diagnostics comprises three steps, which are done sequentially after we have fit our model to the data:

1. Simulate datasets from the model.
2. Compute, for each data point, on which quantile it falls in the simulations.
3. Use the quantile residuals instead of ordinary residuals in model diagnostics.

[7] Indeed, thanks to that package most of the detailed code we use in this chapter is unnecessary, except that writing it all out helps us to better understand the statistics and methods, which is our main goal with this book!

Code Box 8.6: Simulating a simple quadratic relationship that we then analyse as a linear one

```
set.seed(123)

# Create the "observations" and "predictors"
 x <- runif(100, 0, 1)
 y <- -x^2 + x + rnorm(100, 0 , .05)

 # Model them with an ordinary linear regression
 fm <- lm(y ~ x)
 fm.fitted <- predict(fm)
 fm.residuals <- y - fm.fitted

# Simulate from this model:
 sim1 <- predict(fm) + rnorm(100, mean=0, sd=sd(residuals(fm)))

# Repeat it 200 times:
 sims <- replicate(200, {predict(fm) + rnorm(100, mean=0, sd=sd(residuals(fm)))})
```

Code Box 8.7: Computing quantiles from simulations and observations

```
# We continue to use the "sims" object from Code Box 8.5
 ecdf(sims[1,])(y[1])

# Repeat this for all 100 data points:
 qs <- 0
 for (i in 1:100) qs[i] <- ecdf(sims[i,])(y[i])
```

Rather than going right to DHARMa and blindly trusting this approach, we first establish the equivalence of model residuals and quantile residuals in the context of a simple regression. Then we demonstrate how this simulation approach can be extended to binary data (logistic regression) and machine-learning models.

SIMULATION WITH A LINEAR MODEL

Following Anscombe (1973), we simulate a quadratic relationship and then incorrectly analyse it as linear. Such a deliberately mis-fit model should be revealed by the model diagnostics. Then, we use the fitted values plus residual noise to simulate a new dataset from this model (Code Box 8.6).

COMPUTING QUANTILES

For each of the 100 data points in our original data (the observations) we now have 200 simulated data points for comparison. We can plot these 200 values either as a density plot or, better, as an empirical cumulative density (ECD) plot to which we add the location of the observation (Code Box 8.7; Figure 8.8).[8] These quantile residuals are highly correlated with

[8] Note that the `ecdf()()` function in R yields a function as its output. Thus, we can use that directly to compute the quantile of a value in this ECDF. That makes the code very concise, but also looks a bit weird, because of the two sets of parentheses in the `ecdf(...)(...)` function that appear in the last line of Code Box 8.7.

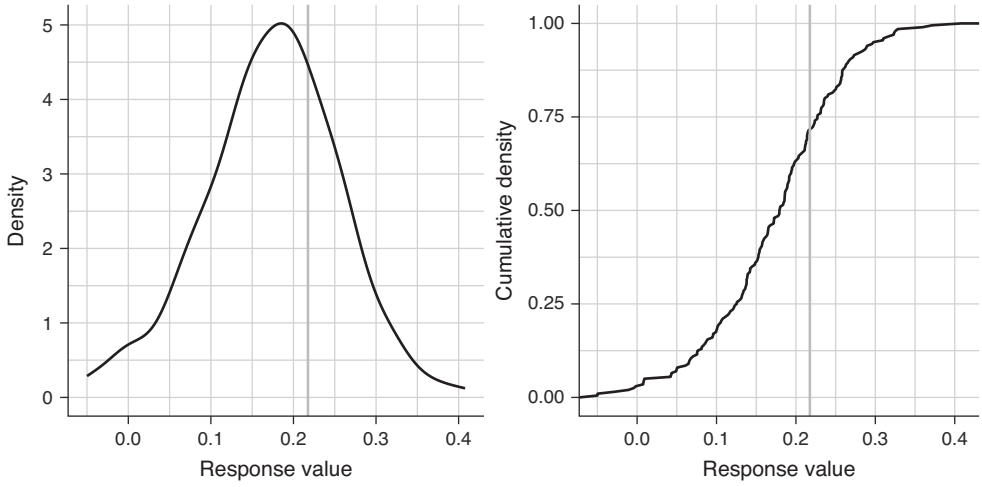


Figure 8.8 Example of how the quantile of a data point is computed. *Left*: Density plot of 200 simulations for data point 1, which is indicated by the vertical grey line. *Right*: Empirical cumulative density (ECD) of the 200 simulations for data point 1; the response value is 0.218, and its quantile is thus 0.715.

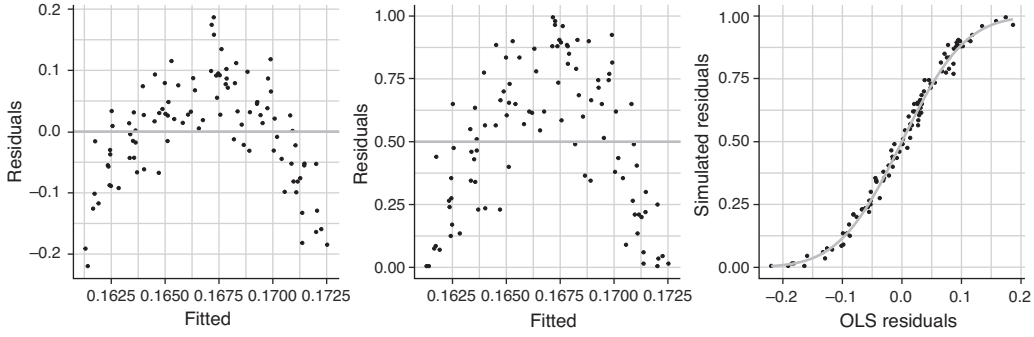


Figure 8.9 Residual vs. fitted plot of the mis-specified linear model from Code Box 8.6. *Left*: Residuals taken from the linear model. *Center*: Quantile residuals of 200 simulations of each data point from the fitted linear model (Code Box 8.7). The Spearman correlation coefficient between these two sets of residuals is quite high: $\rho = 0.993$. *Right*: Relationship of ordinary least-squares (OLS) residuals (x-axis) vs. quantile residuals, for all 100 data points in the dataset. The grey line is the normal cumulative density function (CDF) with $\sigma =$ the standard deviation of the OLS residuals. This illustrates that the quantile residuals represent a transformation to uniformity that emerges automatically from the empirical cumulative density (ECD) function. The variation around the curve is due to the stochastic nature of the simulations and disappears as the number of simulations goes to infinity.

the ordinary residuals of the OLS regression (Figure 8.9; Spearman's $\rho = 0.993$), making the point that indeed they are valid stand-ins for residuals as we know them.

USING QUANTILE RESIDUALS IN MODEL DIAGNOSTICS

Now that we are convinced that quantile residuals are equivalent to the standard residuals we are used to working with in a linear model, we move on to using the quantile residuals in the typical model diagnostics. First, we can do a quantile-quantile plot. Since the ECD-transformation automatically yields a uniform distribution (Figure 8.9, right panel), we plot the observed quantile against that expected in a uniform distribution.[9] The expectation for N values of a uniform distribution is that they lie on the quantiles

[9] This transformation is also known as the "probability integral transform" (Fisher 1925).

Code Box 8.8: Quantile-quantile plot of quantile residuals

```
# We continue to use the "qs" object from Code Box 8.6

dats <- data.frame("expected"=(1:100)/(100+1), "observed"=sort(qs))
ggplot(dats, aes(observed, expected)) +
 stat_function(fun=punif, geom="line", args=list(0, 1), col="grey") +
 geom_point() +
 theme_minimal()
```

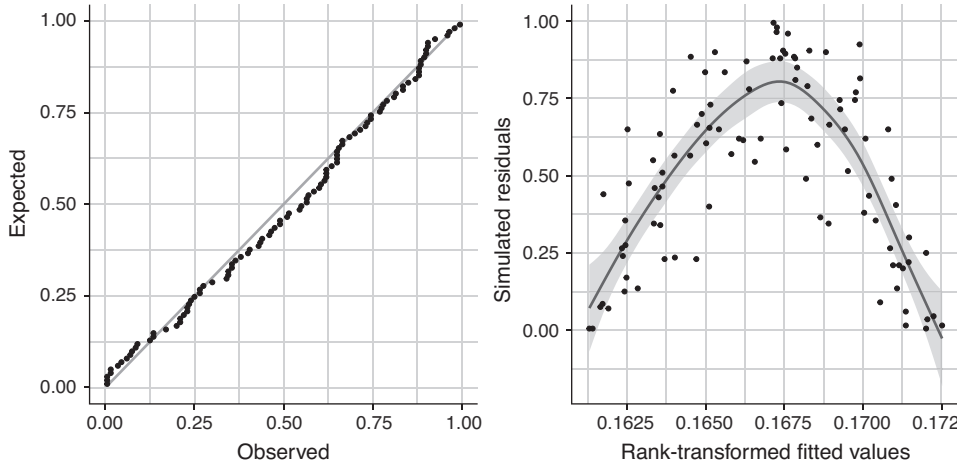


Figure 8.10 Two typical diagnostic plots: the quantile-quantile (Q-Q) plot (*left*; points should be close to the line) and the residual-fitted plot (*right*; points should exhibit no pattern). Because the data easily pass Berkson's "interocular traumatic test" (see footnote 15 in Chapter 9), the spline curve with 95% confidence interval (shaded area) added to the right panel serves only as a visual guide to the obvious deviation from "no pattern." Because there should be no relationship between fitted values and their residuals, this line ideally should be horizontal with $y = 0.5$.

$(1:N)/(N+1)$. We therefore plot our sorted quantile residuals against this expectation (Code Box 8.8; Figure 8.10, left panel).

We also can compute diagnostic statistics on these quantile residuals (Code Box 8.9), such as with the Kolmogorov-Smirnov test for deviation from the expected uniform (see also left panel in Figure 8.7). Further, we can compare the observed variance of the model residuals with those in the simulations as a test of over- or under-dispersion (a value close to 1 indicates no dispersion issues). Additionally, we may want to quantify whether our dataset contains outliers. In terms of the simulated residuals, an outlier is simply a data point for which all simulations are larger or smaller than the observed value, and hence for which the quantile would be 0 or 1. Since the number of outliers follows a binomial distribution with probability $\approx 1/(N+1)$ (Hartig 2022), we can even assign a p-value to the observed number of outliers. In our simple example (Code Box 8.6), the quantile residuals are uniformly distributed, neither under- nor over-dispersed, and lack outliers (see output in Code Box 8.9).

Finally, and probably most importantly, we can plot the quantile residuals as a function of the fitted values (Code Box 8.10). To guide the eye, we can even add a regression spline

Code Box 8.9: Model diagnostics using the quantile residuals

```
# We continue to use the object "qs" from Code Box 8.6
 ks.test(qs, "punif", 0, 1) # Kolmogorov-Smirnov test on the residuals

|   Asymptotic one-sample Kolmogorov-Smirnov test
|
| data:  qs
| D = 0.065, p-value = 0.792
| alternative hypothesis: two-sided
|

 summary(apply(sims, 2, function(x) var(fm.residuals) / var(x))) # Examine for
      dispersion

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
| 0.7185  0.9210  1.0104  1.0171  1.1016  1.4787

 range(qs) # Any outliers?

| [1] 0.005 0.995

 sum(qs==0) + sum(qs==1)

| [1] 0
```

Code Box 8.10: Quantile residual plot

```
ggplot(data.frame("fitted"=fm.fitted, "quantresid"=qs), aes(fitted, quantresid))
      +
  xlab("fitted values") + ylab("simulated residuals") +
  geom_smooth(method="gam", color="grey40") +
  geom_point() +
  theme_minimal()
```

through the plot (although that is almost too obvious in this example; Figure 8.10, right panel).

8.4.2 Using quantile residuals beyond the linear model

Quantile residuals come into their own when used in situations when the Q-Q plot and other diagnostic plots fail: if errors are not normally distributed or if we use "distribution-free" machine-learning approaches.[10]

[10] We use scare quotes around "distribution-free" because this frequent claim is correct only in an irrelevant way: machine-learning models often do not assume or require a specified distribution. However, any model fitted to data—including machine-learning models—has to have a score function (a.k.a. a loss function), so

AN ILLUSTRATIVE EXAMPLE: REVISITING THE BREAST CANCER DATASET

As an example, we revisit the breast cancer study from §7.5 in Chapter 7. We try to predict a malignant cancer from characteristics of tissue images. We use a random-forest model to classify the images into the two diagnoses (benign, malignant). The predictions from this analysis are probabilities of a datum being labelled malignant.

CHANGE THE SIMULATION TO DEAL WITH BINARY DATA

We need to change two elements of our simulation for binary data like these. First, we need to realise that observations and simulated data can only take on two values: 0 or 1. How do we compute the quantiles of a binary dataset? A solution proposed by Dunn & Smyth (1996) is to add uniform noise ($\mathcal{U}(-0.5, 0.5)$) to all observations and simulated values when we have integer or binary responses (Code Box 8.11). Second, we can rank-transform the fitted values before plotting them. This is because machine learning does not necessarily use the log-likelihood as a score function; its predictions are uncalibrated pseudo-probabilities (Platt 2000; Niculescu-Mizil & Caruana 2005; Dormann 2020a), which might give a misleading impression when plotted in diagnostic plots.

The model diagnostics suggest that this model is fine, although not perfect. The quantile mean is not exactly equal to 0.5, but it is quite close (0.52); the deviation of quantile residuals from uniformity is just barely not statistically significant ($p = 0.050$); and there are three cases (out of 569) where the quantile is 1.[11] Most importantly, the Q-Q and residual-vs.-fit plots look convincingly innocuous (Figure 8.11).

CONCLUSIONS

If the diagnostics showed a clear pattern different from what we expect, we would reconsider the model, possibly allowing for more flexibility and non-linearity,[12] or the inclusion of predictors that code grouping structures that we might have omitted in our initial model formulation. It is important to always remember that model diagnostics only show us if there are obvious violations of basic explicit and implicit assumptions. These same diagnostics do not tell us if the model is actually a good one.

8.5 Predicting with confidence is not the same as confidence in prediction

The first example in this chapter illustrated in-sample prediction and out-of-sample hindcasting and forecasting (§8.2). The bootstrapped lines in Figure 8.3 showed the challenges in hindcasting and forecasting that emerge from the uncertainty in the data. In contrast, Figure 2.4 in Chapter 2 showed how bootstrapped replicates relate to a 95% confidence interval: the region of the observed data that includes the true expectation in 95% of the cases. As we have noted repeatedly throughout this book (see, e.g., §2.3.1 and Figure 2.3 in Chapter 2)—and it bears repeating again—this region *is not the region that contains an observation* in 95% of the cases. That latter region is called the prediction interval, and it is wider than the corresponding confidence interval because we add residual error to the confidence interval to obtain the prediction interval.

---

machine-learning models implicitly include distributional assumptions. These are either least-squares (implying a normal distribution) or something akin to Gini impurity, which also implies assumptions about expected frequencies for any category (see §6.6.1 in Chapter 6).

[11] Following Hartig (2022) we can compute the probability of three outliers in 569 data points as a binomial random variate. In R: `dbinom(3, p=1/(569+1), size=569)` = 0.061. Thus, the three outliers are what we can expect ($p > 0.05$).

[12] Non-linearity is built into random-forest models, of course.

Code Box 8.11: Random-forest modelling of binary data and the associated diagnostics

```
# Load required libraries
 library(ranger)

# Read in the data (local copy; same as the dataset used in Code Box 7.8 in
    Chapter 7)
 BCWD <- read.csv(file="BCWD.csv", stringsAsFactors = T, row.names = 1)
 BCWD$Diagnosis <- ifelse(BCWD$Diagnosis == "B", 0, 1) # recode as 0 (benign) and
      1 (malignant)

# Randmom-forest model
 fr <- ranger(factor(Diagnosis) ~., data=BCWD, probability=T)
 fr.fitted <- predict(fr, data=BCWD)$prediction[,2] # prob of y=1
 n <- nrow(BCWD)

# Simulations
 set.seed(123)
 sim1 <- rbinom(nrow(BCWD), p=fr.fitted, size=1)
 sims <- replicate(200, rbinom(n, p=fr.fitted, size=1))

# Add uniform noise to simulations and observations when computing quantile
    residuals:
 qr <- 0
 for (i in 1:n) qr[i] <- ecdf(sims[i,] + runif(200, -0.5, 0.5))(BCWD$Diagnosis[i]
      + runif(1, -0.5, 0.5))
 summary(qr)

|    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
|   0.010   0.300   0.520   0.519   0.750   1.000

ks.test(qr, "punif", 0, 1) # Test distribution

| Asymptotic one-sample Kolmogorov-Smirnov test
|
| data:  qr
| D = 0.056924, p-value = 0.05007
| alternative hypothesis: two-sided

range(qr) # Any outliers?

| [1] 0.01 1.00

sum(qr==0) + sum(qr==1)

| [1] 3
```

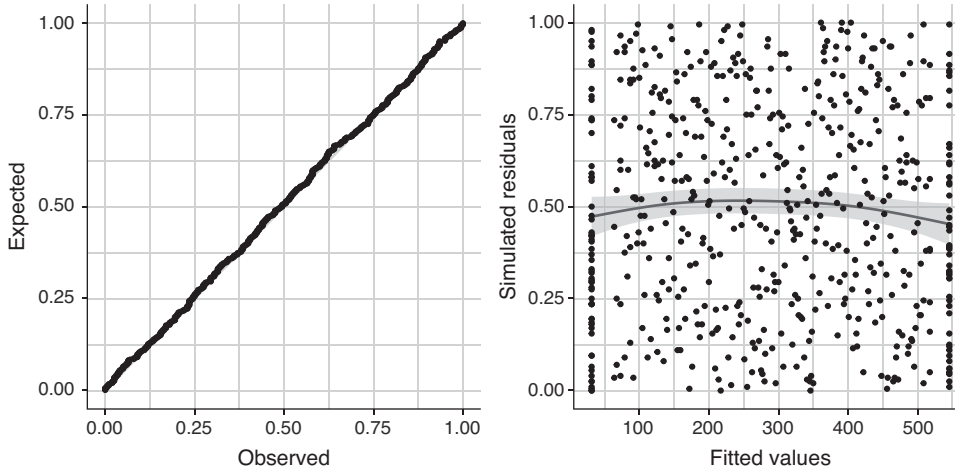


Figure 8.11 Diagnostic plots using quantile residuals for a random-forest model fit to binary data (Code Box 8.11). Note rank-transformation of the *x*-axis in the right-hand side plot.

Whether we are more interested in a confidence interval or a prediction interval depends on the question we are trying to answer. For example, if we are trying to forecast the path and strength of a hurricane, we are much less interested in being right on average, and much more interested in being right about this specific storm.[13] In contrast, if we are trading stocks and making several thousand transactions every day, we are likely to be more interested in the actual confidence interval. No single transaction makes that much difference, but our client (or our boss) expects us to make money on average.

8.5.1 Reprise: The confidence interval

The confidence interval is a statement about a statistic. Often it is the interval around an expectation that contains, in the desired proportion of cases, the true value. When we state tomorrow's expected lunchtime temperature as 15 ± 2°C (95% confidence interval), then we mean that in 95 out of 100 days like this, we find the true lunchtime temperature will be in the interval [13, 17]°C. Here are a number things this statement does *not* mean:

- We are 95% certain that tomorrow at lunchtime the air temperature will be between 13 and 17°C. This is incorrect because it is unclear what "we are *x*% certain" actually means. The confidence interval is a statement about an expectation, not about us.
- In 95% of cases tomorrow's lunchtime temperature will fall into the range between 13 and 17°C. This is incorrect, because the confidence interval is an assertion about the expectation, not the realisation.[14]
- The expectation based on such data has an uncertainty of 2°C in 95% of cases. This is incorrect because every new dataset and hence every new prediction from it will have a different confidence interval. Some of these will be larger and others will be smaller than the specific confidence interval we estimated from our particular dataset.

[13] This is quite different from the daily weather forecast, where an assertion that the probability of rain today is 70% means that on 70% of other days like today, it rains. But will it rain today? Perhaps, or perhaps not.

[14] Another way to phrase this is with the familiar adage "climate is what you expect, weather is what you get." Climate is the expectation, weather is the realisation. The confusion between the two is highlighted every time a politician asserts that the Earth is not warming because it is snowing today.

The meaning of the confidence interval has received an enormous amount of coverage in scientific publications due to the difficulty of exactly stating what it means (see §2.3.1 in Chapter 2 and Ellison & Dennis 2010, for example). Although, for practical purposes, the three incorrect statements listed above point us in the right direction, they all miss the mark in helping us understand exactly what is the confidence interval of a prediction.

Formally, the confidence interval for a set of predictions $\hat{Y}$ from a multiple regression to a new dataset Z is calculated as:

$$\hat{Y} \pm t_{\alpha/2,N-(k+1)} \sqrt{\hat{\sigma}^2 Z^\intercal (X^\intercal X)^{-1} Z}, \tag{8.1}$$

where X are the predictor values used for fitting the model, σ^2 is the residual variance, and $^\intercal$ represents the transposition of a matrix. Intuitively, the expression under the square-root operator represents the distance between the centre of the training data X and the predicted data Z: the larger this distance, the larger is the factor multiplied by the residual variance. The expression to the left of the square-root operator (i.e., the value from the t-distribution with $N-(k+1)$ degrees of freedom) yields the factor discussed in §2.3.1 in Chapter 2; for "large" datasets this is ≈ 2, yielding the familiar rule-of-thumb that the "95% confidence interval = estimate $\pm$ 2· its standard error."

Simulations require more than a rule-of-thumb approximation, however. When we simulate data, we must specify and code exactly what we intend. We then proceed, using repeated simulations, to determine (validate) if, indeed, 95% of cases, and not 80% or 99%, behave as expected (i.e., the confidence interval has nominal coverage: it does what it says on the can). Figure 2.3 in §2.3.1 of Chapter 2 showed the results of an initial, and simple, simulation that clearly illustrates the meaning and nominal coverage of the 95% confidence interval of the mean.

8.5.2 The prediction interval

The prediction interval, also confusingly sometimes referred to as the "confidence interval of a new observation," is different from a confidence interval both in meaning and in computation. Whereas the confidence interval is a statement about the expectation (i.e., the average value we would get in an infinite number of trials, observations, or experiments), the prediction interval is a statement about a single, new observation. As a result, the prediction interval always is wider than the confidence interval because we do not average a large number of realisations. Accordingly, the equation to compute a prediction interval is subtly different from the one we used to compute the confidence interval (Equation 8.1):

$$\hat{Y} \pm t_{\alpha/2,N-(k+1)} \sqrt{\hat{\sigma}^2 (1 + Z^\intercal (X^\intercal X)^{-1} Z)}, \tag{8.2}$$

Note that the "$1 + \ldots$" in the expression within the square root means that the residual variance is added to whatever we compute as uncertainty in Z. All other terms in Equation 8.2 are identical to those in Equation 8.1. Intuitively, the prediction interval adds the residual noise to the confidence interval.

8.5.3 An illustrative example: Predicting the volume of a tree from its diameter

Using linear regression to create a calibration line and then predicting new observations from the calibration results illustrates the difference between confidence intervals and

Code Box 8.12: Regression of tree volume on its diameter

```
# The "trees" dataset comes bundled in R, so we don't need to load any special
    libraries.
# Note that in the "trees" dataset, the variable name for the vector of
    diameters is "Girth"

# Load the data
 data(trees)

# Fit and summarise the model
 fm <- lm(Volume ~ Girth, data=trees)
 summary(fm)

| <—-snip--->
| Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept) -36.9435     3.3651  -10.98 7.62e-12 ***
| Girth         5.0659     0.2474   20.48  < 2e-16 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 4.252 on 29 degrees of freedom
| Multiple R-squared:  0.9353,  Adjusted R-squared:  0.9331
| F-statistic: 419.4 on 1 and 29 DF,  p-value: < 2.2e-16
```

prediction intervals. Our example comes from the common goal of estimating the volume of timber (usable wood) in a tree from a measurement of its diameter.

ESTIMATE THE CONFIDENCE INTERVAL

Given a set of measurements, we use linear regression to get the calibration (Code Box 8.12).

From this fitted model, we can extract all relevant information to simulate the estimated amount of timber for a tree of, for example, 20 inches diameter (see §6.4 in Chapter 6 for details of Monte Carlo sampling from a multivariate distribution), and compare it with direct computation of the confidence interval using Equation 8.1 (Code Box 8.13).[15]

There is very good agreement between the approaches, demonstrating that we correctly understood how the confidence interval is computed.

ESTIMATE THE PREDICTION INTERVAL

For the tree diameter-volume calibration line we can also compute a 95% prediction interval for volumes of trees with diameters equal to 20 inches (Code Box 8.14).

Once again, the Monte Carlo simulation agrees nicely with the analytical solutions. Note that the width of the prediction interval (Code Box 8.14) is more than twice that of the confidence interval (Code Box 8.13).

[15] The classic "trees" dataset, reprinted in Atkinson (1985) from Ryan et al. (1978), is sufficiently old-school that the values are in inches (diamter), feet (height), and cubic feet (volume). The data were recorded from 31 black cherry trees (*Prunus serotina*) harvested from the Allegheny National Forest in Pennsylvania, USA.

Code Box 8.13: Confidence interval derived from Monte Carlo estimates

```
# We continue to use the model object "fm" from Code Box 8.11
# Load required library
 library(mvtnorm) # For generating multivariate random normal data

 Sig <- vcov(fm) # Variance-covariance matrix of the fitted model "fm" from Code
      Box 8.11
 b <- rmvnorm(1000, mean=coef(fm), sigma=Sig)

# Monte Carlo estimates (n = 1000) of the volume of a tree whose diameter is 20
     inches
 predsMC <- c(1, 20) %*% t(b)
 mean(predsMC)

| [1] 64.45536

 quantile(predsMC, c(0.025, 0.975))

|     2.5%    97.5%
| 60.99790 68.24657

# 95% confidence interval from build-in function
 predsLM <- predict(fm, newdata=list("Girth"=20), interval="confidence")
 rownames(predsLM) <- "LM"
 rbind(predsLM, "MC"=c(mean(predsMC), quantile(predsMC, c(0.025,  0.975))))

# Analytical, using eqn. 8.1:
 sigma <- summary(fm)$sigma
 X <- cbind(1, trees$Girth) # include intercept!
 Z <- c(1, 20)
 mean.analytical <- c(1, 20) %*% coef(fm)
 SECI.analytical <- qt(0.025, df=fm$df.residual) * as.numeric(sqrt(sigma^2 * t(Z)
      %*% solve(t(X) %*% X) %*% Z))

rbind("LM"=as.vector(predsLM),
      "AN"=mean.analytical + c(0, 1, -1) * SECI.analytical,
      "MC"=c(mean(predsMC), quantile(predsMC, c(0.025, 0.975)))
      )

|         fit      lwr      upr
| LM 64.37367 60.61759 68.12974     # predict.lm
| AN 64.37367 60.61759 68.12974     # eqn 8.1
| MC 64.45536 60.99790 68.24657     # Monte Carlo
```

Code Box 8.14: Prediction interval for a tree whose diameter equals 20 inches

```
# We continue to use the model object "fm" from Code Box 8.11 and the Monte
    Carlo predictions from Code Box 8.12
predsLM.PI <- predict(fm, newdata=list("Girth"=20), interval="prediction")
rownames(predsLM) <- "LM"
predsMC.PI <- rnorm(1000, mean=predsMC, sd(residuals(fm)))
rbind(predsLM.PI, "MC"=c(mean(predsMC.PI), quantile(predsMC.PI, c(0.025,
    0.975))))

# analytical: PI from eqn. 8.2 #
SEPI.analytical <- qt(0.025, df=fm$df.residual) * as.numeric(sqrt(sigma^2 * (1 +
    t(Z) %*% solve(t(X) %*% X) %*% Z)))

rbind("LM"=as.vector(predsLM.PI),
    "AN"= c(mean.analytical + c(0, -1, 1) * SEPI.analytical),
    "MC"=c(mean(predsMC.PI), quantile(predsMC.PI, c(0.025, 0.975)))
)

|         fit      lwr      upr
| LM 64.37367 54.90089 73.84645     # predict.lm
| AN 64.37367 54.90089 73.84645     # eqn. 8.2
| MC 64.27568 55.83571 73.73132     # Monte Carlo
```

8.5.4 Confidence and prediction intervals for non-Gaussian data and models

For non-Gaussian GLMs, the situation is very different. The additive error is valid only for Gaussian data; for all other distributions we have to propagate both the uncertainty in the estimate(s)[16] and the uncertainty in the distribution. We do this by drawing realisations using the Monte Carlo predictions as distributional parameters. Note that the simulations carried out at the link-scale need to be back-transformed to the response scale using the appropriate inverse link-function. That is, after the last step presented above fits, lower and upper confidence bounds need to be transformed; the resulting confidence intervals will be asymmetric.

CONFIDENCE AND PREDICTION INTERVALS FOR COUNT DATA

To illustrate the estimation of confidence intervals and prediction intervals for count (Poisson) data, we revisit the dataset on military coups in sub-Saharan Africa described in §6.1 of Chapter 6 (Code Box 8.15).

As we would expect, the confidence interval [1.96, 3.66] is substantially narrower than the prediction interval [0, 7]. It is instructive to realize why the prediction interval is so much larger.

To estimate the confidence and prediction intervals, we first generated 1,000 draws from the regression estimate distribution to include their uncertainty. Second, we drew samples

[16] Calculating the uncertainty of the estimate in R typically is provided by the `predict()` function when asking for the standard error of the prediction (with the option `se.fit=T`). However, for models that do not provide this standard error, such as most machine-learning models, we have to use bootstrapping to get such standard errors. See §8.5.5, below.

from the negative binomial distribution. We can determine the effect of the first step because we can skip it and use the values estimated by the GLM directly in the second step. The two distributions are very similar (Figure 8.12); the uncertainty of the parameter estimates did not substantially change the overall prediction distribution. In other words, drawing realisations from the negative binomial is the cause of the wide prediction interval; it simply can be a very spread-out distribution.

CONFIDENCE AND PREDICTION INTERVALS FOR BINARY DATA

We repeat this regression-simulation approach one last time, only now for binary data. A moment of reflection should reveal that the expected prediction interval must be $[0, 1]$ because any realisation can take on only values of 0 or 1. Unless the confidence interval has width equal to 0, there will be some probability of either 0 or 1 as outcome, the result of which will be a full-width prediction interval. In other words, our intuition suggests that the prediction interval really is uninformative for binary data. Simulation lets us check our intuition.

For this example, we revisit the diabetes dataset we introduce in §7.3 of Chapter 7, but now model these data using a GAM with a Bernoulli link function.[17] As we did for the count regression above, we compute both confidence and prediction intervals (Code Box 8.16).

The fit of this model is very poor; $< 8\%$ of the deviance is explained by it. Accordingly, the confidence interval of a prediction from this model to an insulin level of 500 μU/mL is very wide: $[0.48, 0.92]$. Using random draws from the (link-scale) predictions to compute realizations, we find that this leads to values of both 0 and 1; hence the prediction interval is effectively the full width of the two possible probabilities.

CONCLUSIONS

Because it adds unexplained noise to the predictions, a prediction interval is at least as wide as, and usually much wider than, the corresponding confidence interval. For normally distributed data, the situation is straightforward. For non-Gaussian data, the situation is somewhat murkier, as we have to consider the steps of link-response-scale transformation and drawing from the distribution. Depending on the underlying distribution, these steps may yield completely uninformative prediction distributions (as they did for the Bernoulli case).

8.5.5 Confidence intervals for machine-learning models

The previous subsections illustrated the meanings, computations, and simulations of confidence and prediction intervals for classical parametric models; can these approaches be extended to machine-learning or deep-learning methods and models? The challenge here is that the output from machine- and deep-learning models typically do not provide standard errors for the predictions.[18]

Looking for solutions to this problem, we have found only a few proposals. Qucit (2018) proposed using quantile regression to compute confidence intervals. When we read this, we thought that it would be an inappropriate approach—and to be fair, Qucit (2018) also discussed the challenges with quantile regression approaches to this problem—but to test it

[17] The Bernoulli is a special case of the binomial distribution with size = 1, and hence the default binomial in `glm(..., family=binomial)`.

[18] The popular and commonly-used random-forest model provides a salient example. Until recently, random forest would not return standard errors for its predictions. The approach now available in R's `ranger` library was devised much later than the original algorithm (Wager et al. 2014). However, we are unaware of any software implementation of artificial neural networks that provide a standard error for their estimates.

Code Box 8.15: Confidence and prediction intervals for count data

```
# Load required libraries
 library(faraway) # For the "africa" dataset
 library(MASS) # For the glm.nb function
 library(mvtnorm) # For generating multivariate random normal data

# Load data
 data(africa)

# Model the data with a negative binomial regression and summarise the model
 fmnb <- glm.nb(miltcoup ~ oligarchy, data=africa)
 summary(fmnb)

| <—-snip--->
| Coefficients:
|             Estimate Std. Error z value Pr(>|z|)
| (Intercept) -0.46233    0.22673  -2.039   0.0414 *
| oligarchy    0.12064    0.02179   5.536  3.1e-08 ***
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| (Dispersion parameter for Negative Binomial(7.9582) family taken to be 1)
|
|     Null deviance: 80.460  on 46  degrees of freedom
| Residual deviance: 49.273  on 45  degrees of freedom
| AIC: 136.03

# Confidence interval (exact, analytical)
 exp(c(pred.fmnb$fit-2*pred.fmnb$se.fit, pred.fmnb$fit+2*pred.fmnb$se.fit))

|        1        1
| 1.940272 3.697983

# Confidence interval (using Monte Carlo simulation)
 Sig.nb <- vcov(fmnb)
 set.seed(123)
 b.nb <- rmvnorm(1000, mean=coef(fmnb), sigma=Sig.nb)
 predsMC.nb <- c(1, 12) %*% t(b.nb) # link-scale MC predictions
 quantile(exp(predsMC.nb), c(0.025, 0.975)) # back-transform to the response
      scale

|     2.5%    97.5%
| 1.959987 3.655779

# Prediction interval (only Monte Carlo simulation)
 realizations.nb <- rnbinom(1000, mu=exp(predsMC.nb), size=fmnb$theta)
 quantile(realizations.nb, c(0.025, 0.975))
```

```
|  2.5% 97.5%
|    0     7

table(realizations.nb)/1000

| realizations.nb
|     0     1     2     3     4     5     6     7     8     9 ...
| 0.107 0.198 0.218 0.188 0.124 0.079 0.047 0.016 0.015 0.004 ...

# Compare these with their distribution without parameter uncertainty
 round(dnbinom(0:14, mu=exp(pred.fmnb$fit), size=fmnb$theta), 3)

| 0.099 0.199 0.225 0.188 0.130 0.078 0.042 0.021 0.010 0.004 ...
```

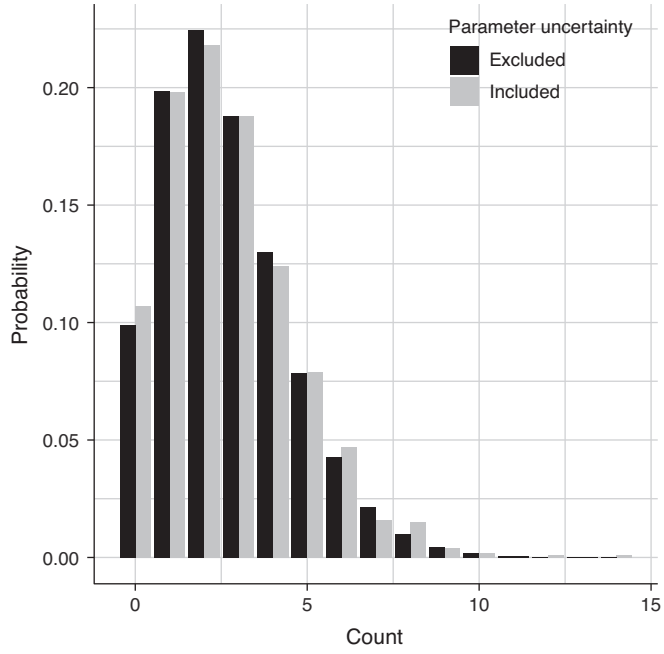


Figure 8.12 Distribution of predictions of the number of military coups for a hypothetical country which already witnessed 12 such coups in the past. Shown are the distributions of predictions from the negative binomial regression model, excluding (black) or including (grey) model parameter uncertainty. In contrast to the Gaussian example, the prediction distribution emerges almost exclusively from drawing from the negative binomial distribution. This adds a lot of uncertainty and dominates the much smaller amount of estimation uncertainty.

quantitatively and determine whether or not it would work, we use simulations. Although it turns out that there are different (and reasonable) approaches to constructing confidence (and prediction) intervals for machine- and deep-learning models, these come with considerable computational costs.

We note that machine learning often is used in a classification setting (i.e., models with discrete, categorical predictors), but it is much simpler to test the validity of quantile

Code Box 8.16: Confidence and prediction intervals for binary data

```
# Load required libraries
 library(faraway) # For the PIMA dataset
 library(mgcv) # for the gam function

# Load the data
 data(pima)

# Model the data with a binomail GAM and summarise the model
 fgam <- gam(test ~ s(insulin), data=pima, family=binomial)
 summary(fgam)

| <—-snip--->
| Approximate significance of smooth terms:
|              edf Ref.df Chi.sq p-value
| s(insulin) 8.173  8.797  54.99  <2e-16 ***
| ---
| Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
|
| R-sq.(adj) =  0.0778   Deviance explained = 7.53%
| UBRE = 0.22002  Scale est. = 1         n = 768

 pred.gam.link.scale <- predict(fgam, newdata=list(insulin=500), se.fit=T)
     # Predictions

| $fit
|       1
| 1.18176
|
| $se.fit
|         1
| 0.6403521

# Confidence interval
 plogis(c(pred.gam.link.scale$fit-2*pred.gam.link.scale$se.fit,
          pred.gam.link.scale$fit+2*pred.gam.link.scale$se.fit))

|         1         1
| 0.4752842 0.9214682

# Prediction interval
 set.seed(123)
 pred.gam.response.scale <- plogis(rnorm(1000, pred.gam.link.scale$fit,
                                   sd=pred.gam.link.scale$se.fit))
 table(rbinom(1000, prob=pred.gam.response.scale, size=1))

|   0   1
| 231 769
```

Code Box 8.17: Set up the simulated dataset for the random-forest model

```
# Load required libraries
 library(faraway) # For the wcgs dataset

# Load data and remove categorical predictors
 data(wcgs)
 wcgs <- as.data.frame(scale(na.omit(wcgs[, -c(7, 9:13)])))

# Simulated model
 MM <- model.matrix(~(age+height+weight+sdp+dbp+chol+cigs)^2, data=wcgs)
 b <- rep(c(1, -1), len=29)
 truth <- MM %*% b
 set.seed(1234)
 wcgs$y <- as.vector(truth + rnorm(nrow(MM), 0, 2)) # Add normal noise
```

regression-based confidence intervals in a regression setting (i.e., with continuous predictors). Further, regression settings are more common, if only our human minds were not so determined to make categories out of continuous variables (see §6.3 in Chapter 6). Thus, our simulated example uses random-forest regression models.

SIMULATE A DATASET

Our simulation is based on longitudinal data collected by the Western Collaborative Group Study (WCGS) from 3,524 male volunteers beginning in 1960 to study correlates of coronary heart disease (Rosenman et al. 1975). Our aim here is only to compare different approaches to compute confidence intervals for a random forest model, not to (re)interpret the original dataset and subsequent analyses.

We use only the (scaled) continuous predictors from the WCGS dataset, but to add some realism (and complexity), we also include in our model all pairwise interactions between these continuous predictors. Then, we define the model coefficients b as either 1 or -1 (and alternating between them), compute the expected $\hat{Y} = Xb$, and add normal noise to it (Code Box 8.17).

THREE WAYS TO COMPUTE A CONFIDENCE INTERVAL FOR A RANDOM-FOREST MODEL

Depending on which machine-learning model we use, there may or may not be analytical or approximate solutions for computing standard errors for its predictions.[19] For example, a GAM (which we used in §4.4.3 in Chapter 4) really is just a very flexible GLM, so we can compute its standard errors analytically. This is not the case with a random-forest model.

Including quantile regression (Qucit 2018), there are three ways to compute confidence intervals for the predictions of a random-forest model (Code Box 8.18; Breiman 1992; Wager et al. 2014; Qucit 2018):

1. Using internal bootstrapping results, Wager et al. (2014) devised an "infinitesimal jackknife" procedure to compute the standard error of the predictions.

[19] Note that when we work with machine-learning models, we are working entirely in a predictive setting. That is, machine-learning approaches typically do not have interpretable parameters and hence we are not interested in their standard errors or whether or not they are statistically significant. Rather, the goal with a machine-learning model is prediction from the fitted model.

Code Box 8.18: Three different methods to estimate a 95% confidence interval on the predictions of a random forest

```
# We continue to use the "wcgs" dataset from Code Box 8.17

# Load required library
 library(ranger) # For fast implemtation of random forest

# Fit the random forest model
 fr <- ranger(y ~ ., data=wcgs, keep.inbag=T, quantreg=T)

# Infinitesimal jackknife:
 predsse <- predict(fr, data=wcgs[1,], type="se", se.method="infjack")
 c(predsse$prediction-1.96*predsse$se, predsse$prediction+1.96*predsse$se)

| [1] -0.4672762  2.3672002 # Jackknifed confidence interval

# Bootstrapping:
 frbs <- replicate(500, {
  ranger(y ~ ., data=wcgs[sample(nrow(wcgs), replace=T),])
  }, simplify=F) # Note this takes 400 CPU seconds for 500 runs
 predsbs <- sapply(frbs, \(x) predict(x, data=wcgs[1,])$prediction)
 quantile(predsbs, c(0.025, 0.975))

|       2.5%      97.5%
| -0.3890788  3.1026183     # Bootstrapped confidence interval

# Quantile regression:
 predict(fr, data=wcgs[1,], type="quantiles",
         quantiles=c(0.025, 0.975))$prediction

|      quantile= 0.025 quantile= 0.975
| [1,]       -2.967768        7.028694  # Quantile confidence interval
```

2. Like any other model, the entire random forest itself can be bootstrapped. Note that this is computationally very intensive.
3. Random forests can be used to predict not only the mean, but also other quantiles of the data. Hence we can use the quantile regression approach to estimate the 2.5% and 97.5% quantiles, representing our 95% confidence interval (Qucit 2018).

The confidence intervals on the predictions that we computed using the three different methods are not the same (Code Box 8.18), and in general, the resulting densities may be far from smooth or even unimodal (Figure 8.13). The first two confidence intervals (infinite jackknife and bootstrap) are similarly narrow (range = 3–3.5 units), whereas the quantile regression is more than three times as wide (Code Box 8.18). Of course, this result is from only one run and the prediction is to a single data point. We now examine this result with a more substantial simulation.

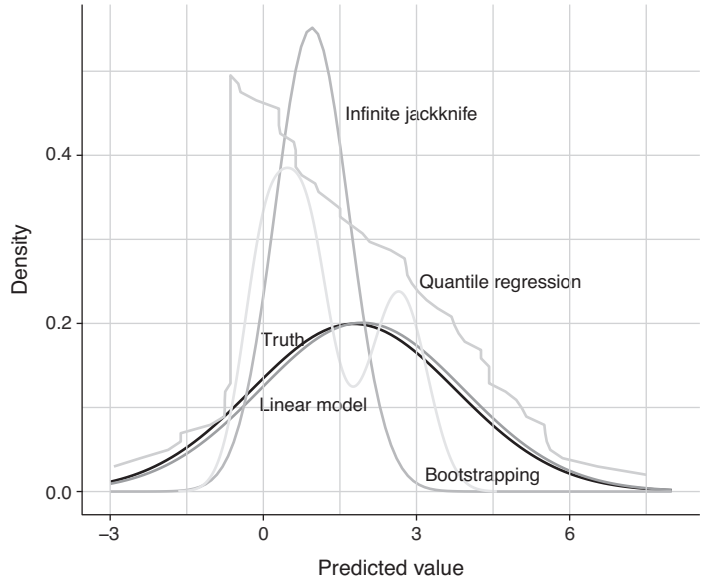


Figure 8.13 Probability densities of predicted values for the three random-forest approaches to construct confidence intervals. These densities are for the predictions of the first data point simulated in Code Box 8.18. `truth` refers to the distribution from which we expect to see an observation (i.e., within ±2 standard deviations of Xb). The linear model is included as reference, as in Code Box 8.19.

SIMULATIONS TO COMPARE AMONG COVERAGES

For our larger simulation, not only did we generate confidence intervals on the random-forest predictions in three different ways, but we also compared these results with the output from a classical linear model that has the true model structure (see also the linear model curve in Figure 8.13). For all methods and models, we varied the regression parameters and the noise with each simulated dataset; all were repeated 100 times. For each method and the two models (random-forest, linear model), we determined how many times each prediction was within its proposed 95% confidence interval; these counts yielded the average coverage per simulation. Finally, because the bootstrapping approaches are computationally intensive, we parallelized our code so that it can run simultaneously on multiple processors (Bengtsson 2021).[20]

Reassuringly, the linear model has the expected nominal coverage: the truth lies within the computed 95% confidence interval in 95% of the simulations. In contrast, the coverages for the three versions of the random-forest predictions were not as good (Figure 8.14). The confidence interval using the infinitesimal jackknife was close (92% nominal coverage), whereas the bootstrap (68% nominal coverage) and the quantile regression (99.7%) were much worse.

The boostrapped confidence intervals were much too narrow; 68% coverage is only $\approx \pm 1$ standard deviation, rather than the expected ≈ 2. Our computations used the quantiles from the bootstrapping estimates, which is standard practice. Alternatively, we could have computed their standard deviation and defined the 95%-confidence interval from that (as $[\hat{Y} \pm 1.96 s_{\hat{Y}}]$), which is the method used for the infinitesimal jackknife. Whether that would make for a better coverage is left as an exercise for the reader.

[20] Despite the parallelization across eight CPU cores, the code in Code Box 8.19 still took nearly three hours to run.

The quantile regression was even less useful. Its 95% confidence interval was so wide that it covered virtually all possible estimates. Although we might think this is a good outcome—because the model always will be right—we simply have no way of knowing how much too wide the confidence interval is. Further, the confidence interval does not deliver on its promise. If we use a particular method to compute a 95% confidence interval, we expect that it should deliver 95% coverage, not 100%.[21]

CONCLUSIONS

In brief, for models other than (generalised) linear ones, the propagation of errors from data through parameters to prediction presents serious challenges for estimating and communicating uncertainty. For (generalised) linear models, both confidence and prediction intervals are well-defined and can be determined analytically. For machine-learning models, however, there are no analytical formulas for estimating confidence intervals. Whenever new procedures are proposed for quantifying uncertainty, simulations will play a crucial role in demonstrating their correctness (Wager et al. 2014; Gal & Ghahramani 2016) and help us find the edible wheat within the otherwise inedible chaff. Unfortunately, our example (Code Box 8.19 and Figures 8.13 and 8.14) suggests that, more often than not, we will go hungry.

Simply asserting in an internet blog posting that quantile regression yields confidence quantiles around predictions from machine-learning models (Qucit 2018) is a far cry from actually delivering valid and repeatable solutions. We were skeptical about this statement before running the simulations because quantile regression suffers from an extreme attrition of data points as the quantiles become more extreme. Whereas all data points are informative for the mean, only 2.5% of them are informative for the upper boundary of a 95% confidence interval. That means that quantile regression often can be limited by sample size and estimation will be correspondingly unstable. Indeed, our simulations confirmed that conclusion (Code Box 8.19; Figure 8.14).

We are surprised—and concerned—that bootstrapping was a similarly poor solution to the problem of constructing confidence intervals on the random-forest predictions. After all, bootstrapping is a well-established way of estimating confidence intervals (Efron 1982; Efron & Tibshirani 1993). The key limitation of bootstrapping is that it samples only from the observed data, and these samples are taken randomly with replacement. As a result, the bootstrapped samples, on average, include only observations well within the interval of the original sample. This limitation is not a problem if we are trying to estimate the mean, but it becomes a problem when we try to estimate an extreme value of the full set of observations, such as the edge of the 95% confidence interval. On average, any bootstrapped sample will have a narrower range than the original dataset and hence yield 95% predictions that are closer to the mean than to the extremes. This effect is even more pronounced when using regression trees, which by definition cannot predict values larger (or smaller) than the data in the training set. As a consequence, bootstrapping inevitably leads to a reduced range of Y values in the training data, and thus to overly narrow predictions.

Very few machine-learning (and deep-learning) approaches have purpose-built methods to compute confidence intervals. Indeed, quantifying uncertainty of predictions is an active field of research in deep learning but the results will have little applied importance. For example, if an artificial neural network (or other similar approach) correctly predicts which item a customer will purchase next, the uncertainty of this classification is irrelevant.

[21] The reason for this should be clear. The more extreme the quantiles, the fewer data points there are that fall outside and can be used to estimate the confidence interval. The 50%-confidence interval should be much closer to the nominal coverage when using quantile regression with random forest. However, we should not allow the method to tell us what we want; we ask, it tells!

Code Box 8.19: Many simulations of prediction coverage.

```
# We continue to use the "wcgs" dataset from Code Box 8.17

# Load required libraries
 library(future.apply) # For parallelisation

# Set up parallelisation
 plan(multisession, workers = 8)

# Simulation
 reps <- 100
 set.seed(123)
 inCVs <- future_replicate(reps, {
  inCV <- c("LM"=1, "RFse"=NA, "RFbs"=NA, "RFqu"=NA) # Four approaches
  b <- runif(29, -2, 2) * rep(c(1, -1), len=29)
  truth <- MM %*% b
  wcgs$y <- as.vector(truth + rnorm(nrow(MM), 0, 2))
 # Reference Linear Model:
  fm <- lm(y ~ (age+height+weight+sdp+dbp+chol+cigs)^2, data=wcgs)
 # Random forest model, for jackknife and quantiles:
  fr <- ranger(y ~ ., data=wcgs, keep.inbag=T, quantreg=T)
 # Bootstrap the random forest:
  frbs <- replicate(200, {
    ranger(y ~ ., data=wcgs[sample(nrow(wcgs), replace=T),])
  }, simplify=FALSE)
 # Extract the confidence intervals
  # LM:
   predsLM <- predict(fm, newdata=wcgs, se.fit=T)
   CI_lm <- cbind(predsLM$fit -1.96*predsLM$se.fit,
                  predsLM$fit + 1.96*predsLM$se.fit)
   inCV["LM"] <- sum(truth > CI_lm[,1] & truth < CI_lm[,2])
  # Random forest SE:
   predsRFse <- predict(fr, data=wcgs, type="se")
   CI_RFse <- cbind(predsRFse$predictions - 1.96*predsRFse$se,
                    predsRFse$predictions + 1.96*predsRFse$se)
   inCV["RFse"] <- sum(truth > CI_RFse[,1] & truth < CI_RFse[,2])
  # Random forest boostrap:
   predsbs <- sapply(frbs, \(x) predict(x, data=wcgs)$prediction)
   CI_RFbs <- t(apply(predsbs, 1, quantile,  c(0.025, 0.975)))
   inCV["RFbs"] <- sum(truth > CI_RFbs[,1] & truth < CI_RFbs[,2])
  # Random forest quantile regression
   predsRFqu <- predict(fr, data=wcgs, type="quantiles",
                        quantiles=c(0.025, 0.975))
   CI_RFqu <- predsRFqu$predictions
   inCV["RFqu"] <- sum(truth > CI_RFqu[,1] & truth < CI_RFqu[,2])
  # Return the CIs
   inCV
 })
apply(inCVs/nrow(wcgs), 1, summary)
```

```
|                LM      RFse      RFbs      RFqu
| Min.    0.8325907 0.8876512 0.6546785 0.9926798
| 1st Qu. 0.9364258 0.9107257 0.6766391 0.9961808
| Median  0.9571929 0.9193189 0.6830045 0.9968173
| Mean    0.9508498 0.9209071 0.6851941 0.9968364
| 3rd Qu. 0.9737428 0.9303787 0.6948600 0.9974539
| Max.    0.9971356 0.9598982 0.7307447 0.9987269
```

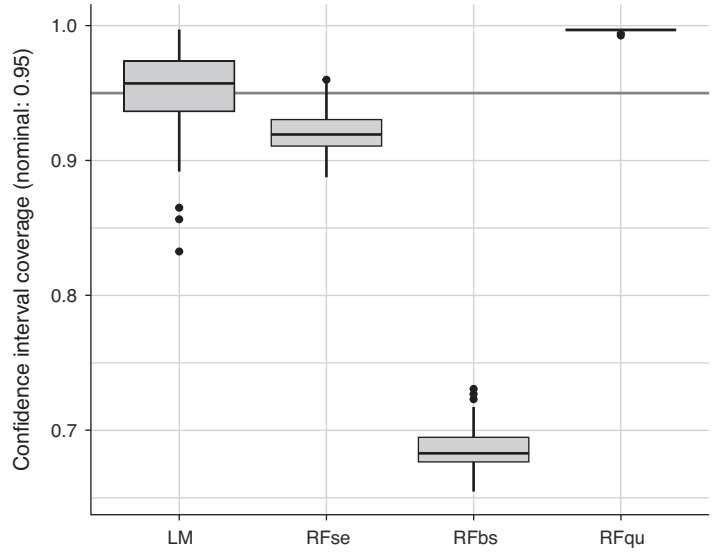


Figure 8.14 Coverages of the four different 95% confidence intervals across 100 simulations. From left to right, the box plots illustrate results from the classical linear model, and three methods of computing confidence intervals from random-forest models: the infinitesimal jackknife, bootstrap, and quantile regression.

For continuous data the situation is slightly different. In contrast, when Google's DeepMind announced that its GraphCast model[22] outperformed state-of-the-art process models in weather prediction (Lam et al. 2023), they had to show confidence intervals for their prediction error. However, Google did not provide details on the method they used for computing their confidence intervals and artificial neural networks are notorious for rapid and large propagation of errors (reviewed in Gawlikowski et al. 2023).

Finally, we have observed that in many fields of science, we are running out of appropriate words and terms to differentiate key concepts, and so we are prone to re-using existing terms without regard for the ambiguity and confusion this causes. For example, calling something the "prediction interval of a regression" is weird enough, but "prediction interval of a prediction," as opposed to the "confidence interval of a prediction," feels like taking the confusion a step too far. What these semantic monstrosities drive home, however, is the need to be clear about what we want from our analysis or prediction: do we make a statement about the mean or about a new data point? Are our "predictions," "forecasts," "hindcasts," or "projections" within or outside the range of our observed predictors? What we should report

[22] It is unclear why they did not call their model Google Cloud.

Code Box 8.20: Updating the forecasts of northern hemisphere temperature anomalies

```
# This code continues from Code Box 8.2, using the same libraries and the same
    dataset

# Redefine the "training" dataset (i.e., the prior)
train2 <- globwarm[globwarm$year > 1880 & globwarm$year < 1985, ] # Training
    data
after2 <- globwarm[globwarm$year >= 1985, ] # Data for forecasting

# Fit the GAM
fm2 <- gam(nhtemp ~ s(year), data=train2)

# Simulation (identical to that used in Code Box 8.3)
set.seed(123)
modellist2 <- replicate(B, simplify=F,
   {
       fmbs <- gam(nhtemp ~ s(year),
            data=train2[sample(nrow(train2), replace=T), ])
   })
predsHind2 <- sapply(modellist2,  predict, newdata=before, simplify=F) # Same
    date range for hindcasts
predsFore2 <- sapply(modellist2,  predict, newdata=after2, simplify=F)
    # Forecasts for new date range
```

on depends entirely on the question we are asking and the type of answer—estimate, uncertainty, confidence interval, prediction, 'cast, etc.—that will be the most useful for answering the questions in which we are most interested.

8.6 Iterative learning: New priors from old posteriors

In the first section of this chapter, we presented a didactic example of creating a simple prior for a linear regression and showed how it could be updated with only a pair of points. This example illustrated that with a reasonable and informative prior we can reduce the uncertainty in our new estimate (i.e., the posterior probability distribution) even if the estimate itself doesn't change very much. In the second section, however, we showed that our model is only as good as the data we use to fit it, and that we can still make very bad forecasts from overtly reasonable priors (Figure 8.3). In this section we show how we can update our model with new data and then re-run our model to get a new set of forecasts. Although in this example we know what we are actually looking for, this process is done every day by weather forecasters and economic analysts who are constantly updating their forecasts in near-real time based on new data that are continuously flowing onto their desks and into their computers.

TEMPERATURES OF THE NORTHERN HEMISPHERE, REVISITED

In §8.2, we analysed northern hemisphere temperature data from 1881 to 1974, and then hind- and forecast them with the fitted model. Our hindcasts were quite reasonable, but our forecasts completely missed the mark (Figure 8.3). We now return to the data as if a decade has passed and re-fit the model using the data from 1881 to 1985 (Code Box 8.20).

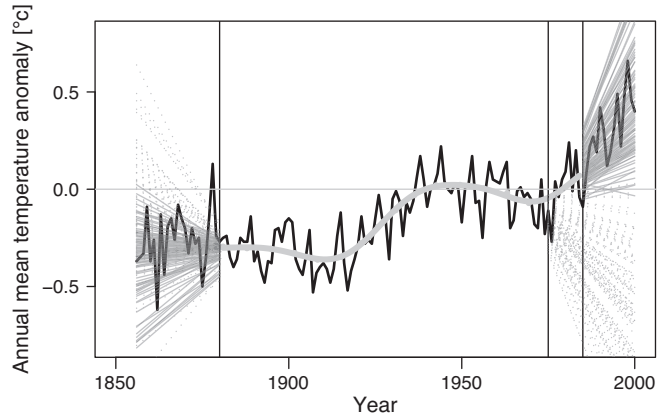


Figure 8.15 Temperature of the northern hemisphere expressed as the anomaly relative to the full period 1856–2000 (black line), and the model fit of the GAM to the training period (grey line between 1880 and 1985). Grey lines to the left and right are hindcasts to the past and forecasts to the future of the training data. Hindcasts and forecasts are from 100 bootstrapped replicates of the GAM and represent the uncertainty inherent in the fitted model. For both the hindcasts and the forecasts, we show the initial 'casts based on the model fit to the 1885–1975 data as dotted lines (identical to the 'casts shown in Figure 8.3) and the new 'casts as solid lines.

We then use these new parameter estimates as our updated prior and forecast again into the future (now, after 1985). We ask, how much does another ten years of data improve our expectations?

Although the hindcasts are qualitatively unaffected by our new model, the forecasts are dramatically different (Figure 8.15). With ten additional years of data, we learned nothing new about the past, but our prior data now include the decade (1975–1985) during which the emissions of SO_2 became more strongly regulated and their ameliorating effect on global warming began to disappear. This change is reflected in the data and including them in our model led us to forecast a substantial increase in atmospheric temperatures in the coming years. In short, learning from data changes our expectation (compare Figures 8.3 and 8.15).[23]

8.7 Outlook

Once a model has been fitted to data, the work still isn't done. Apart from model diagnostics that may force us into reconsidering our analysis, we should take a step back and evaluate what we had expected to see, and what we did find. In a way, that difference is the scientific progress resulting from our experiment, data collection, and analysis. We hope that the analysis did not shatter whatever ideas we had of how the world works. If it did, it may be more indicative of our ignorance before the analysis than about the quality of the data.

With machine- and deep-learning approaches, the model itself is opaque, in the sense that we can no longer comprehend the multi-dimensional representation of the data. We

[23] We note that Code Box 8.20 does not include a formal Bayesian analysis, but even without that, we can see clearly how changing the amount of prior information used in our model alters our expectations. Of course, the entire analysis could have been performed using Bayesian methods (including splines), resulting in proper posterior distributions for predictions in 1975, which could serve as priors for the analysis of only the subsequent ten years of new data, and resulting in new posteriors for 1986 and beyond. But our intention here is only to illustrate that we can learn from additional data and update our forecasts accordingly. This applies to both Bayesian and frequentist statistics.

may want to employ predictions to specific conditions, to specific points in time or space, to be better able to understand what that black-box model means. That is indeed the approach taken by explainable artificial intelligence (XAI; Adadi & Berrada 2018): not to take a full-picture view of everything the model represents, but only a "local" analysis. One of the most commonly employed XAI approaches indeed carries the local character of any effect evaluation in the name: LIME stands for "Locally Interpretable Model-agnostic Explanations" (Ribeiro et al. 2016). The field of XAI is still very much in flux, and there are as many voices raising concerns over its usefulness and reliability (e.g., Lipton 2018; Rudin 2019; Krishnan 2020) as there are methods proposed (e.g., Goldstein et al. 2015; Murdoch et al. 2019; Molnar 2023).

For the time being the jury is still out on how much we actually learn, scientifically, from more and more flexibility in the data-fitting elements of statistics. For one, simply describing data does not automatically result in good predictions. If the goal is to comprehend in a causal way how the world works, manipulative experimentation is still the gold standard and will increase our understanding much more rapidly than the statistical approaches used to explore and raise new ideas. Consolidating such ideas into testable theories is a much harder part of doing science.

8.8 Summary

- For a research project to be worth pursuing, there should be an expectation that there will be less uncertainty in the data than there is in the prior information about the system (i.e., the likelihood should dominate the prior). Both Bayesians and frequentists can use prior information to develop simulations about the system to be studied.
- For multivariate models, bivariate plots of the response to a single predictor can be confusing or misleading. Alternative plots that account better for multiple predictors and their effects on any focal predictor include conditional, marginal, partial dependence, and partial residual plots.
- Standard model diagnostic plots are used routinely to assess the fit and validity of simple linear models. Generalised versions of these plots derived from simulations can and should be used to assess the fit and validity of more complex models, including linear models with non-Gaussian link functions.
- A confidence interval asserts an expectation. An *n*% confidence interval is the interval around an expected value (a statistic) that contains the true value *n*% of the time. In contrast, a prediction interval asserts the uncertainty about a single, new observation. Prediction intervals generally are larger than corresponding confidence intervals. To date, algorithms to compute confidence intervals for machine-learning models often have provided unsatisfactory results.

References

Adadi, A. & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, 6, 52138–52160. doi: 10.1109/ACCESS.2018.2870052.

Anscombe, F. J. (1973). Graphs in statistical analysis. *The American Statistician*, 27(1), 17–21. doi: 10.1080/00031305.1973.10478966.

Atkinson, A. C. (1985). *Plots, Transformations, and Regression: An Introduction to Graphical Methods of Diagnostic Regression Analysis*. Oxford, UK: Clarendon Press.

Azzolina, D., Berchialla, P., Gregori, D., & Baldi, I. (2021). Prior elicitation for use in clinical trial design and analysis: A literature review. *International Journal of Environmental Research and Public Health*, 18(4), 1833. doi: 10.3390/ijerph18041833.

Bengtsson, H. (2021). A unifying framework for parallel and distributed processing in R using futures. *The R Journal*, 13(2), 208–227. doi: 10.32614/RJ-2021-048.

Breiman, L. (1992). The little bootstrap and other methods for dimensionality selection in regression: X-fixed prediction error. *Journal of the American Statistical Association*, 87(419), 738–754. doi: 10.1080/01621459.1992.10475276.

Cortez, P., Cerdeira, A., Almeida, F., et al. (2009a). Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4), 547–553. doi: 10.1016/j.dss.2009.05.016.

Cortez, P., Cerdeira, A., Almeida, F., et al. (2009b). Wine Quality. UCI Machine Learning Repository, 10.24432/C56S3T. Accessed 12 November 2023.

Cox, D. R. & Snell, E. J. (1968). A general definition of residuals. *Journal of the Royal Statistical Society. Series B (Methodological)*, 30(2), 248–275. doi: 10.1111/j.2517-6161.1968.tb00724.x.

Crawley, M. J. (2007). *The R Book*. Chichester, West Sussex, UK: John Wiley & Sons.

Dawid, A. P. (1984). Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society, Series A (General)*, 147(2), 278–292. doi: 10.2307/2981683.

Doak, D. F., Bigger, D., Harding, E. K., et al. (1998). The statistical inevitability of stability-diversity relationships in community ecology. *The American Naturalist*, 151(3), 264–276. doi: 10.1086/286117.

Dormann, C. F. (2020a). Calibration of probability predictions from machine-learning and statistical models. *Global Ecology and Biogeography*, 29(4), 760–765. doi: 10.1111/geb.13070.

Dormann, C. F. (2020b). *Environmental Data Analysis: An Introduction with Examples in R*. Heidelberg, Germany: Springer.

Dunn, P. K. & Smyth, G. K. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics*, 5(3), 236–236. doi: 10.2307/1390802.

Efron, B. (1982). *The Jackknife, the Bootstrap, and Other Resampling Plans*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Efron, B. & Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. London: Chapman & Hall.

Ellison, A. M. & Dennis, B. (2010). Paths to statistical fluency for ecologists. *Frontiers in Ecology and the Environment*, 8(7), 362–370. doi: 10.1890/080209.

Falconer, J. R., Frank, E., Polaschek, D. L. L., & Joshi, C. (2022). Methods for eliciting informative prior distributions: A critical review. *Decision Analysis*, 19(3), 189–204. doi: 10.1287/deca.2022.0451.

Faraway, J. J. & Augustin, N. H. (2018). When small data beats big data. *Statistics & Probability Letters*, 136, 142–145. doi: 10.1016/j.spl.2018.02.031.

Fisher, R. A. (1925). *Statistical Methods for Research Workers*. Edinburgh, Scotland, UK: Oliver and Boyd.

Frank, S. A. (2014). Generative models versus underlying symmetries to explain biological pattern. *Journal of Evolutionary Biology*, 27(6), 1172–1178. doi: 10.1111/Jeb.12388.

Freiberger, M. (2015). Information is surprise. https://plus.maths.org/content/information-surprise. Accessed 9 November 2023.

Gal, Y. & Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML '16: Proceedings of the 33rd International Conference on Machine Learning—Volume 48* (pp. 1050–1059). New York, NY, USA: JMLR.org.

Gawlikowski, J., Tassi, C. R. N., Ali, M., et al. (2023). A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56(1), 1513–1589. doi: 10.1007/s10462-023-10562-9.

Gelman, A. & Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge, UK: Cambridge University Press.

Goldstein, A., Kapelner, A., Bleich, J., & Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1), 44–65. doi: 10.1080/10618600.2014.907095.

Good, P. I. & Hardin, J. W. (2012). *Common Errors in Statistics (and How to Avoid Them)*. Hoboken, NJ, USA: Wiley, 4th edition.

Gotelli, N. J. & Ellison, A. M. (2012). *A Primer of Ecological Statistics*, 2nd edition. New York, NY, USA: Oxford University Press.

Hartig, F. (2022). DHARMa: Residual diagnostics for hierarchical (multi-level/mixed) regression models, version 0.4.5. https://cran.r-project.org/package=DHARMa.

IPCC (Intergovernmental Panel on Climate Change) (2001). *Climate Change 2001: The Scientific Basis. Contribution of Working Group I to the Third Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge, UK: Cambridge University Press.

Jones, P. D. & Mann, M. E. (2004). Climate over past millennia. *Reviews of Geophysics*, 42(2), RG2002. doi: 10.1029/2003RG000143.

Krishnan, M. (2020). Against interpretability: A critical examination of the interpretability problem in machine learning. *Philosophy & Technology*, 33(3), 487–502. doi: 10.1007/s13347-019-00372-9.

Lam, R., Sanchez-Gonzalez, A., Willson, M., et al. (2023). Learning skillful medium-range global weather forecasting. *Science*, 382(6677), 1416–1421. doi: 10.1126/science.adi2336.

Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3), 31–57. doi: 10.1145/3236386.3241340.

Molnar, C. (2023). Interpretable machine learning. https://christophm.github.io/interpretable-ml-book. Accessed 13 December 2024.

Murdoch, W. J., Singh, C., Kumbier, K., et al. (2019). Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences, USA*, 116(44), 22071–22080. doi: 10.1073/pnas.1900654116.

Niculescu-Mizil, A. & Caruana, R. (2005). Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)* (pp. 625–632). New York, NY, USA: Association for Computing Machinery.

Phillips, P. J., Hahn, C. A., Fontana, P. C., et al. (2021). *Four principles of explainable artificial intelligence, NISTIR 8312*. Technical Report NISTIR 8312, U. S. Department of Commerce, National Institute of Standards and Technology.

Pierce, D. A. & Schafer, D. W. (1986). Residuals in generalized linear models. *Journal of the American Statistical Association*, 81(396), 977–986. doi: 10.1080/01621459.1986.10478361.

Platt, J. C. (2000). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. J. Smola, P. Bartlett, B. Schölkopf, & D. Schurrmans (Eds.), *Advances in Large Margin Classifiers*, volume 10 (pp. 61–74). Cambridge, MA, USA: MIT Press.

Qucit (2018). A simple technique to estimate prediction intervals for any regression model. https://medium.com/@qucit/a-simple-technique-to-estimate-prediction-intervals-for-any-regression-model-2dd73f630bcb. Accessed 20 November 2023.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13–17, 2016* (pp. 1135–1144).

Rosenman, R. H., Brand, R. J., Jenkins, C. D., et al. (1975). Coronary heart disease in the Western Collaborative Group Study: Final follow-up experience of 8 1/2 years. *Journal of the American Medical Association*, 233(8), 872–877. doi: 10.1001/jama.1975.03260080034016.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(55), 206–215. doi: 10.1038/s42256-019-0048-x.

Ryan, Jr., T. A., Joiner, B. L., & Ryan, B. F. (1978). *Minitab Student Handbook*. Boston, MA, USA: PWS Publishers.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3), 379–423. doi: 10.1002/j.1538-7305.1948.tb01338.x.

Wager, S., Hastie, T., & Efron, B. (2014). Confidence intervals for random forests: The jackknife and the infinitesimal jackknife. *Journal of Machine Learning Research*, 15(1), 1625–1651. doi: 10.5555/2627435.2638587.

9

Post hoc alternatives to retrospective power analysis

In Chapter 3 we discussed the importance of thinking through our questions, framing our hypotheses, designing our studies with adequate replication to detect effects (if indeed they are there) of observational gradients or experimental treatments, and specifying appropriate analyses before proceeding with data collection. In Chapter 4, we illustrated how prospective power analyses can be used effectively to determine adequate sample sizes for our studies, conditional on a reasonable expectation of an effect size and standard values for a type-I error ($\alpha = 0.05$) and statistical power ($0.8 = 1 - \beta$, where $\beta =$ the probability of a type-II error). Despite all this advance planning and, we hope, having preregistered our study, we may still fail to find statistically significant results (i.e., we miss our crucial $p < 0.05$).

On the one hand, "negative results"—that is, failure to reject the null hypothesis—may be of great value. For example, the continued support for (i.e., failure to reject) the Standard Model of physics has, nonetheless, sparked many new discoveries at the sub-atomic scale (e.g., Butterworth 2016). At the same time, the failure to repeat exciting and apparently strong experimental results in fields as diverse as psychology, medicine, and ecology (e.g., Kimmel et al. 2023; Yang et al. 2023) has been held up as evidence of a reproducibility crisis in science (Munafò et al. 2017; Nosek et al. 2022). A good reason to publish negative results is to shine a light on over-hyped results or sloppy science.

On the other hand, what if the effect we were looking for really is there but we did not detect it (i.e., a false negative, a.k.a. type-II error)? There are many reasons for such failures. For example, the effect could have been smaller than we had expected; we did not sample enough of the range of variability in our study population; or our treatments were not extreme enough. Even if we had a good sampling design or a wide range of experimental treatments, the within-group variances may have been extremely heteroscedastic (see §5.4 in Chapter 5) or much greater than the between-group variance. We also may have failed to include potential confounding effects in either our experimental design or our statistical models. Sensors may have gone out of calibration or stopped working entirely. We may have made errors in transcribing data into spreadsheets from scrawled entries in lab or field notebooks and even the best data imputation methods turned out to be of little help (see §7.3 in Chapter 7).

Probably the most common reason that true effects are not detected is that the study is underpowered: the sample size simply is too small to detect the effect we expect to see.[1] We

[1] Of course, another reason is a result of using the standard $\alpha = 0.05$ critical value as our acceptable type-I error rate and critical value for statistical significance. When $\alpha = 0.05$, we expect 5% of repeated instances of this study to yield non-significant results even if they have a large enough sample size to have 80% power. Unfortunately,

illustrated in §3.4 in Chapter 3 that a surprisingly large number of samples can be required to detect even a relatively large difference between two groups or a significant correlation between two variables. Yet, even if the study design was justified based on a prospective power analysis, many researchers still go on to explain the lack of an observed effect (or its statistical significance) with a retrospective (a.k.a. post hoc) power analysis.[2] That is, they compute the power of their study conditional on the actual effect size they observed and the sample size they used (and, of course, with the traditional $\alpha = 0.05$). Apparent negative results are then wished away—and more research is called for—when the post hoc power turns out to be too low.

Please bear in mind the key take-home message of this chapter: along with virtually all statisticians **we do not recommend using retrospective power analysis**. We also note that Bayesian statisticians have no real need for either *p*-values or power computations. Bayesian posterior probability distributions are much more informative than even the most reasonable variant of post hoc frequentist analyses, and Bayesians can jump right to the last section of this chapter (§9.5). The other sections are more than justified by the frequency with which retrospective power analyses and its alternatives appear throughout the scientific literature.

What you will learn from this chapter

- The fundamental problems with retrospective power analysis.
- Reasonable alternatives to retrospective power analysis and when and how to use them.

9.1 Reprise: Prospective power analysis

Before defining and discussing *retro*spective power analysis, we briefly recap what *pro*spective power analysis is and how we should use it (for more details, reread Chapter 4). Prospective power analysis is based on the relationship between the acceptable rates of type-I (α) and type-II (β) errors, the estimated (or detectable) effect size (Δ), and the sample size (N). If we know or can estimate any three of these, the fourth can be computed analytically for most common study designs and determined using simulation for others (see Chapter 4 and Cohen 1988).

Since α and β are conventionally fixed at 0.05 and 0.2, respectively (see footnote 13 in Chapter 3), and the power of an experiment is defined as $1 - \beta$ (and hence conventionally fixed at 0.8), we normally use prospective power analysis to determine how many samples we need, given an estimated effect size, or identify how large an effect we can expect to detect, given a number of samples that have been predetermined by logistical or budgetary constraints. The former—determining the sample size needed to detect an effect of known size—is more common and is often used (or in some cases required) to justify

we have no way of knowing whether our study was in that 5%. This is an inevitable consequence of the fact that effect size and confidence intervals are random variables, and cannot be interpreted with respect to the truth. See Ellison & Dennis (2010) and §2.3.1 in Chapter 2 for more on how to interpret confidence intervals.

[2] Faul et al. (2007) distinguished between two kinds of post hoc power analyses: "retrospective power analysis" and "post hoc power analysis." The former, "retrospective power analysis", is identical to what most statisticians refer to as "post hoc power analysis"; Faul et al. (2007) concur with virtually all statisticians in identifying its problems. In contrast, what Faul et al. (2007) call "post hoc power analysis," like prospective power analysis (Chapter 4), requires an estimation of the effect size for the underlying population, not for the sample that was actually used in the analysis (see §9.3.3 for further discussion of this method). Later authors (e.g., Jobst et al. 2023), while citing Faul et al. (2007), ignore this distinction, used "post hoc power analysis" to mean the post hoc computation of power using the effect size observed in the sample (i.e., retrospective power analysis), and support the continued use and reporting of the long-discredited post hoc (i.e., retrospective) power.

a project budget.[3] Estimating predictive confidence intervals (Goodman & Berlin 1994, and see §9.3.1, below) can enhance our interpretation of prospective power, help identify appropriate sample sizes, and refine experimental designs.

9.2 What is retrospective power analysis?

9.2.1 Retrospective power analysis—The discredited standard approach

Retrospective power analysis normally is hauled out of our statistical toolbox only when our study does not turn out the way we expected and we fail to reject our null hypothesis. Because our sense of self-worth as scientists is inextricably (and inexplicably) wrapped up in "significant" *p*-values, we are driven to make sense of, and ultimately dismiss, our "negative" results. After explaining away and discounting all the possible technical reasons for our inability to detect the expected effect, we usually retreat to the idea that we would have found a significant effect if only our sample size had been larger or the experiment had run longer (and in either case, we would need additional funding for our research).

Retrospective power analysis aims to quantify how much larger the study would have had to have been for us to have detected the observed difference at a specified type-I error rate (usually $\alpha = 0.05$). Quite simply, we compute the retrospective power of our study in the same way that we (should have) determined its prospective power: we plug in the effect we observed, the sample size we used, and $\alpha = 0.05$, and then compute $1 - \beta$.[4] And voilà! We determine that our power was $\ll 0.8$ and conclude that it was unlikely that we would have found the effect we were looking for, given the size of our study.

Please don't do that. Bury your retrospective power calculator in a dark corner of your statistical toolbox where it can wither and die.

9.2.2 The problems with retrospective power analysis

Ever since Neyman & Pearson (1933) discriminated between type-I and type-II errors and tried to minimise them simultaneously, statisticians have been exploring the idea of statistical power. Jacob Cohen (1962) is usually cited as the first researcher to draw together the disparate threads and ideas about statistical power into a coherent package, although he credited Helen Walker and Joseph Lev's influential book *Statistical Inference* (1953) as being the lone prior exception to the rule of ignoring power in discussions of issues of statistical significance. In both his initial article (Cohen 1962) and his subsequent book (Cohen 1988, the first edition was published in 1969), Cohen framed power both as something that should be determined before undertaking a research project *and* as something that could be computed after the study to minimise the importance of failing to reject a null hypothesis. Indeed, he appeared to equate not only the calculations or prospective and retrospective power, but also their interpretation.[5] However, other statisticians rapidly determined that prospective and retrospective power are two completely different beasts.

[3] For example, §4.3 of Part G.500 of the US National Institutes of Health's (NIH) General Application Guide for NIH and Other PHS [Public Health Service] Agencies requires a detailed description of study design and prospective power for any research proposal involving clinical trials with human subjects (NIH 2022). Similarly, the ARRIVE Guidelines (Percie du Sert et al. 2020) include a requirement for justifying the number of animals used in research regulated by Institutional Animal Care and Use Committees (IACUC); see also Gaskill & Garner (2020).

[4] These computations can be done with any package (e.g., the `pwr` and `rdpower` packages in `R`) or standard equations (e.g., Cohen 1988) that also are used to compute prospective power. The calculations are identical and standard software has no way of knowing whether the inputs are a priori or post hoc.

[5] Cohen (1988, p. 4) wrote: "As another example, consider a completed experiment which led to nonrejection of the null hypothesis. An analysis which finds that the power was low should lead one to regard the negative results

BIASES IN ESTIMATION OF EFFECT SIZES

The first nail in the coffin of retrospective power analysis was hammered in by Lane & Dunlap (1978).[6] Using simulations (Code Box 9.1), they showed that if only significant results are reported in publications, then estimation of effect sizes and variances (of populations and that are identified by statistical models) from classic ANOVA designs will be biased, sometimes extremely so (Figure 9.1). In this case, they looked at the bias in estimates of ω^2, which is a standardised measure of effect size in ANOVA designs (Kroes & Finley 2023). Unlike Cohen's d, which is an absolute or standardised measure of difference between groups, ω^2 is a standardised measure of association between variables among the groups (Kroes & Finley 2023).[7]

Lane & Dunlap (1978) used the same simulations to show that this problem would only get worse if more conservative rejection regions were used for significance testing (e.g., $\alpha = 0.001$; Figure 9.1).[8] Hedges (1984) expanded on Lane & Dunlap's (1978) work by developing maximum likelihood estimators of effect sizes of one or more studies (the latter would be used in meta-analysis) when only significant results are published (i.e., when non-significant results are censored; see also §7.4 in Chapter 7).[9] These studies not only implied that retrospective power based on published effect sizes and p-values would be biased, but also showed that even prospective power calculations were likely to be overly optimistic without Hedges' correction (see also Correll et al. 2020).

PROSPECTIVE POWER IS A PROBABILITY BUT RETROSPECTIVE POWER IS NOT

Goodman & Berlin (1994) added the second nail in the coffin of retrospective power analysis when they presented one of the clearest critiques of retrospective power analysis. They wrote (Goodman & Berlin 1994, p. 201):

> Power is exclusively a pre-trial concept; it is the probability of a group of possible results (namely, all statistically significant outcomes) under a specified alternative hypothesis. A study produces only one result. . . . The perspective after the experiment differs from that before the experiment simply because the result is known. [Using the actual known results to estimate power] . . . would be like trying to convince someone that buying a lottery ticket was foolish (the before-experiment perspective) after they hit a lottery jackpot (the after-experiment perspective).

The key point that Goodman & Berlin (1994) made is that computation of power before the study is done returns the probability of finding a significant or a not significant difference (for which, of course, the critical value α also has to be defined). These computations are agnostic about particular p-values. But once we have reported a specific p-value (as opposed to stating that it is simply greater or less than α) and use it to make an assertion of

---

as ambiguous, since failure to reject the null hypothesis cannot have much substantive meaning when, even though the phenomenon exists (to some given degree), the *a priori* probability of rejecting the null hypothesis was low."

[6] This problem had been identified earlier by Sterling (1959) and Cohen (1962), but only conceptually.

[7] Cohen's d and ω^2 both use estimators of the variance (or its square root, the standard deviation) in their calculation. Cohen's $d = \frac{\overline{Y}_1 - \overline{Y}_2}{s_{\text{within}}}$ (Equation 3.8 in Chapter 3). The effect size of an ANOVA, $\omega^2 = \frac{\sigma_A^2}{\sigma_A^2 + \sigma_{\text{error}}^2}$, where σ_A^2 is the variance of the treatment effect and σ_{error}^2 is the total variance. For a one-way ANOVA, as in the simulations in Code Box 9.1, ω^2 can be estimated as $\hat{\omega}^2 = \frac{SS_{\text{effect}} - df_{\text{effect}} \times MS_{\text{error}}}{SS_{\text{total}} + MS_{\text{error}}}$, where SS_{effect} is the between-groups sums of squares, MS_{error} is the within-groups mean square, df_{effect} is the between-groups degrees of freedom, and SS_{total} is the total (between-groups + within-groups) sums of squares (Kroes & Finley 2023).

[8] At the same time, Lane & Dunlap (1978) presaged the pre-registration "revolution" (see §3.5 in Chapter 3) in arguing that publication decisions should be based solely on the soundness of the experimental design, not on the significance of the results.

[9] We introduced this estimator, Hedges' g, in §3.4 in Chapter 3. We discuss it more detail in §10.2 of Chapter 10.

Code Box 9.1: Simulation of estimated effect sizes if only significant results are reported

```
# Lane & Dunlap (1978) simulated studies of the comparison of 2 groups with
    known effect sizes (differences between groups) D = 4, 8, or 16,
    corresponding to 0.25, .5, 1 standard deviation units. The data for each
    group were drawn from normal distributions with mean = 0 and SD = 16. Group
     size varied from 5 to 100. Data were analysed using one-way ANOVA.

# Load required libraries
 library(effectsize) # For easily getting effect sizes from ANOVA
 library(parameters) # For easily formatting effect sizes

# Simulations (take a while!)
 set.seed(123)
 output <- data.frame("n"=numeric(), "effect"=numeric(), "pval"=numeric(),
     "omega2"=numeric())
 n <- c(5, 10, 25, 50, 75, 100) # Number of individuals per group
 eff <- c(4, 8, 16) # Actual differences between groups (Cohen's d)
 nreps <- 5000 # Number of simulations for each combination of number of
     individuals per group and actual difference between groups
 m <- 1 # Counter
 for (i in 1:length(n)) {
  for (j in 1:length(eff)) {
    for (k in 1:nreps) {
      group1 <- rnorm(n[i], 0, 16)
      group2 <- rnorm(n[i], eff[j], 16)
      temp <- stack(data.frame(group1, group2))
      names(temp) <- c("IQ", "group")
      a <- model_parameters(aov(IQ ~ group, data=temp),
        effectsize_type = "omega")
      output[m,1] <- n[i]
      output[m,2] <- eff[j]
      output[m,3] <- a$p[1] # p value
      output[m,4] <- a$Omega2[1] #ANOVA effect size
      m <- m+1
 }}} # see Figure 9.1
```

error rate on the observed value, we make the same logical fallacy as computing retrospective power:

> A *P* value is to a type-I error exactly as "1 − post hoc power" is to a type-II error; that is, where "5%" is almost always the type-I error rate set before the experiment; after the experiment, the *P* value is frequently described as the "observed" type-I rate or, in other words, the type-I error rate with respect to the observed difference. The notion of the *P* value as an "observed" type-I error suffers from the same logical problems as post hoc [i.e., retrospective] power. Once we know the observed result, the notion of an "error rate" is no longer meaningful in the same way that it was before the experiment. (Goodman & Berlin 1994, p. 202, using their typography of the *p*-value)

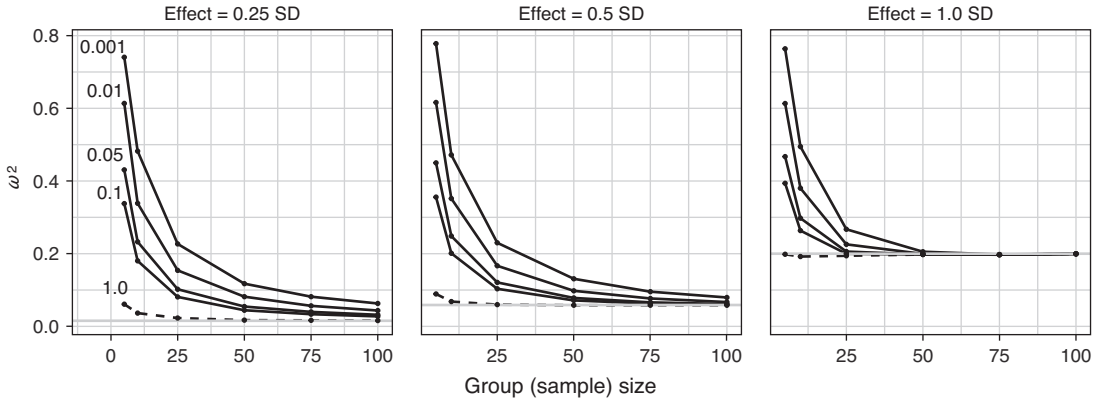


Figure 9.1 Average estimated standarized ANOVA effect sizes (ω^2; see footnote 7, above, and Kroes & Finley 2023) from published (i.e., "significant") studies for two-group comparisons tested with a one-way ANOVA and significant at α levels = 0.001, 0.01, 0.05, 0.1, and 1.0 (the last being the effect size of the entire set of simulated samples of group-size N and indicated with the dashed line) for three different values of Cohen's d (0.25, 0.5, and 1.0 standard deviation units). Values (points) shown are means from 5000 simulations for each group size × true effect size combination. For each value of d, the expected value of ω^2 is indicated by a horizontal grey line. As the effect size decreases, it takes an increasing number of observations to obtain unbiased results.

More succinctly, "because power is the probability of a significant result, the retrospective power is equal to 1 if the result is significant, and 0 if it is non-significant. It is certainly easier to compute that way!" (Lenth 2007, p. E26).

RETROSPECTIVE POWER OF A NON-SIGNIFICANT RESULT IS USUALLY < 0.5

Referring to the extended technical argument in Goodman (1992), Goodman & Berlin (1994) noted in a single sentence that if, instead of using Lenth's (2007) identity that retrospective power equals zero for a non-significant result, one actually calculated retrospective power using observed sample sizes, effect sizes, and non-significant p-values (interpreted illogically as the observed type-I error), the result would always be less than 0.5.[10] This result was illustrated by Hoening & Heisey (2001) (Figure 9.2) for a one-tailed Z-test with critical value $\alpha = 0.05$ (Code Box 9.2); variations on this figure have been published repeatedly in the last two decades (e.g., Gelman 2019). Nuzzo (2018) zoomed in on it and showed that to obtain a retrospective power equal to the conventional aspirational goal of 80%, the p-value would need to be 0.005 (Figure 9.2, inset). All these studies (and many others) have shown that there is a 1:1 relationship between retrospective power and a p-value, so there is no added value to computing, let alone reporting, the retrospective power of a single study.[11]

DOES ANYONE STILL REPORT AND USE RETROSPECTIVE POWER?

For nearly 50 years, statisticians repeatedly have pointed out the problems with retrospective power and published their critiques not only in statistical journals but also in journals published in a broad range of fields that routinely use statistics. A quarter-century after the first problems with retrospective power were identified, Hoening & Heisey (2001) listed 19 highly-ranked journals in fields ranging from physical anthropology and psychology through ecology and on to human and veterinary medicine that from 1983 to 1997

[10] There are, however, some instances where retrospective power computed for a non-significant result can exceed 0.5, sometimes by a large margin (Hoening & Heisey 2001).

[11] Note that this 1:1 relationship does not mean, as has been occasionally suggested, that a small p-value implies a high power (see Gelman & Carlin 2014).

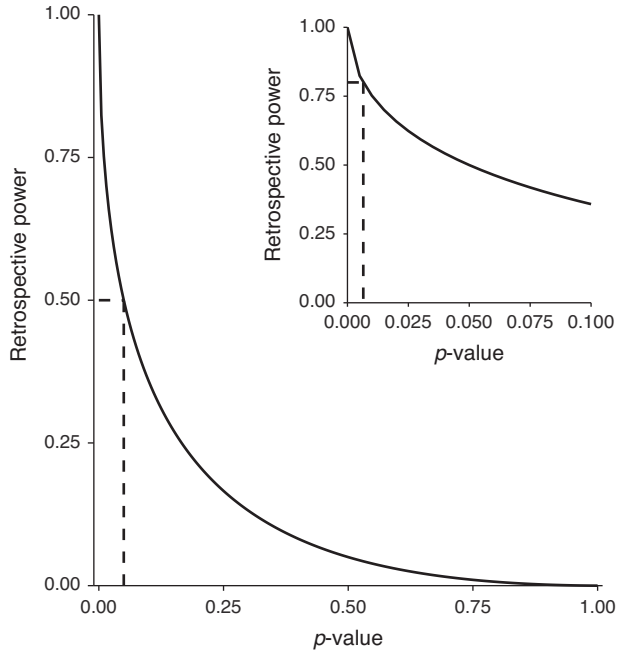


Figure 9.2 Retrospective power as a function of the observed p-value for a one-tailed Z-test in which $\alpha = 0.05$. When the hypothesis test is marginally significant (i.e., $p = 0.05$), the retrospective power equals 50%. Inset shows that to achieve a retrospective power of 80%, the p-value needs to equal 0.005.

Code Box 9.2: Computation of retrospective power from p values for the one-tailed Z test with $\alpha = 0.05$

```
# Analytical computations to show the relationship between p values and
    retrospective power

# Define a vector of Z scores on the interval [0,1] (percentiles of the standard
     normal distribution)
 zscores <- qnorm(seq(0,1,.005))

# p values for the Z test are 1 - CDF(Z scores) for the standard normal
    distribution
 pvals <- 1-pnorm(zscores)

# Retrospective power = 1 - CDF(Z_alpha - Z_pval)
 power <- 1-(pnorm(qnorm(0.95)-zscores))

# Compute retrospective power for pval = 0.05
 power <- 1-(pnorm(qnorm(0.95)-qnorm(0.95)))
 power

| 0.5
```

either published articles with retrospective power analysis or required it as a condition of publication if results of hypothesis tests were found to be not statistically significant.

This trend has not abated in the subsequent 25 years. A Web of Science search using the terms {"retrospective power"} OR {"post hoc power"} returned > 100 articles in the interval 2020–2023.[12] About half of these papers reported the results of retrospective power analysis, whereas the other half were critiques of its continued use. Most of the papers were in medical journals,[13] but the use of retrospective power remains widespread in a diversity of scientific disciplines (recent examples include: Aviles-Rosa et al. 2020 [animal behaviour]; Park & Snyder 2020 [atmospheric organic chemistry]; Ribeiro et al. 2020 and Yasdiman et al. 2022 [human behaviour]; Bartlett et al. 2022 [robotics]; Lutermann et al. 2022 [ecology]; Jobst et al. 2023 [psychology and structural equation models]; and Stommes et al. 2023 [political science].)

9.3 Post hoc alternatives to retrospective power analysis

Retrospective power analysis should not be done for three reasons. First, retrospective estimates of effect sizes are likely to be biased, so using them in power calculations without correction (e.g., with Hedges' g) will yield biased results. Second, prospective power identifies the *probability* that our study will not result in a false negative, whereas retrospective power is not a probability; it can only be equal to 1 if the result is statistically significant, otherwise it equals 0. This result holds regardless of the fact that we can use standard software to compute retrospective power in the same way that it computes prospective power. Finally, because the value that we can compute for retrospective power has the same information content as a p-value (Figure 9.2), it is sufficient to report only the p-value.[14]

So, are there any alternative post hoc analyses that provide useful information? Many have been proposed, and some of them have the potential to be useful. We discuss four alternatives to retrospective power analysis that have attained a reasonable amount of traction and use in the scientific literature. Despite available turn-key software solutions to implement these alternatives, all of them require care and detailed thought to use correctly and effectively.

THE FIRST QUESTION: DO YOU REALLY NEED POST HOC ANALYSIS?

If the results are so clear that the meaning of the data hits us right between the eyes, it has passed what Edwards et al. (1963) called the "interocular traumatic test."[15] We should then

[12] Search run by the authors on 11 October 2023.

[13] A typical back-and-forth set of proposal-critique-and-rebuttal papers is the one that begins with Bababekov et al. (2019), which was followed by five papers in the "Special Feature: The Post Hoc Power Controversy" (LeMaire. 2021; Althouse 2021; Carle 2021; Griffith 2021; Chang & Stapleton 2021), all published in the *Journal of Surgical Research*. See also Bababekov et al. (2018) followed by Gelman (2019), Helminen & Reito (2019b), Bababekov & Chang (2019), and Helminen & Reito (2019a) in the *Annals of Surgery*; and Lutermann et al. (2022) followed by Ruxton & Colegrave (2023) in *Frontiers in Ecology and the Environment*.

[14] Bababekov & Chang (2019) agree that retrospective power and p-values carry the same information but argue that just because they are redundant does not make reporting both of them incorrect. They accurately point out the difficulty with interpreting p-values and suggest that retrospective power is more easily understood by their target audience (surgeons). But as Helminen & Reito (2019a) point out, Bababekov & Chang (2019) not only continue to equate retrospective power with a probability but also conflate p-values with the risk of a type-I error. We would add that of the three reasons to eschew reporting retrospective power, its 1:1 relationship with a p-value is the least important.

[15] In a brief footnote, Edwards et al. (1963, p. 217) attributed the origin of the interocular traumatic test to a Bastille Day, 1958 "personal communication" from statistician Joseph Berkson. Berkson, however, appeared to ignore the results of his own test in his continual rejection of the evidence of a strong relationship of lung cancer with

be done—no post hoc analyses are required. That is because, as Edwards et al. (1963, p. 217) wrote, "the interocular traumatic test is simple, commands general agreement, and is often applicable; well-conducted experiments often come out that way." However, as Edwards et al. (1963, p. 217) continued, and in the spirit of this book, "the enthusiast's interocular trauma may be the sceptic's random error. A little arithmetic to verify the extent of the trauma can yield great peace of mind for little cost." The remainder of this section looks into how much it really costs, both conceptually and mathematically, for this "little arithmetic."

A GENTLE REMINDER: BEFORE YOU DO ANY POST HOC ANALYSES, REPORT THE EFFECTS YOU OBSERVED

Before we discuss the post hoc alternatives to retrospective power analyses, we offer a gentle reminder. Always report the observed effects in a clear figure, in a simple table, or in text. At a minimum, the effects that should be reported include: the within-group and total number of samples (observations, individuals in experimental treatment groups, etc.); estimate(s) of the expected value(s) of the key parameter(s) (means, regression slopes, etc.); and estimate(s) of the variances around the parameter estimates (standard deviations, variances, coefficients of variation, 95% confidence intervals, etc.). All these values will be needed in any subsequent post hoc analysis. They will be even more useful to others who will use them as inputs into prospective power analyses for follow-up studies; to define Bayesian priors to replicate or build on the work; or in meta-analysis (see Chapter 10).

9.3.1 Confidence intervals

Beyond reporting effect sizes, parameter estimates, and their variances, the simplest post hoc results to report are confidence intervals (CIs). Reflecting our 5% acceptable type-I error rate, we normally compute and report 95% CIs. As we showed in detail in §2.3.1 of Chapter 2, it is straightforward to estimate a 95% CI around the population mean μ of a normally-distributed random variable:

$$\Pr\left(\bar{Y} - \frac{cs}{\sqrt{N}} \leq \mu \leq \bar{Y} + \frac{cs}{\sqrt{N}}\right) = 0.95, \tag{9.1}$$

where $\bar{Y}$ and s are the sample mean and the sample standard deviation, respectively, and c is the t-score (number of standard deviations from the mean) of the 0.025th percentile of the t-distribution with $N-1$ degrees of freedom.[16]

For a quick, back-of-the-envelope computation of a 95% CI that involves only "a little arithmetic," scientists often use $c = 1.96$ in Equation 9.1, together with the sample mean ($\bar{Y}$) and its standard error ($\text{SEM} = \frac{s}{\sqrt{N}}$):

$$\Pr\left(\bar{Y} - 1.96 \cdot \text{SE} \leq \mu \leq \bar{Y} + 1.96 \cdot \text{SE}\right) = 0.95. \tag{9.2}$$

But Equation 9.2 is correct only for "large" sample sizes ($N \gg 100$), not for smaller ones. If our within-group sample size is < 100, we should use Equation 9.1 and do the computations

smoking tobacco. Ironically, he was head of the Division of Biometry and Medical Statistics at the Mayo Clinic—a position he held for 30 years (1934–1954)—when he twice, first in 1957 and again in 1964 (Anon. 1964) rejected the evidence for the link between smoking and lung cancer. The 1964 report of the Surgeon General's Advisory Committee on Smoking and Health on this link led to the addition, two years later, of the first iteration of the now-ubiquitous "smoking causes cancer" labels on tobacco products.

[16] More precisely, c is the value of the inverse cumulative density function of the (Student or) t-distribution for a random variable x and degrees of freedom df.

using the exact percentile of the *t*-distribution. We can use the R function `qt(x, df)`, where `x` is the percentile of interest and `df` is the degrees of freedom, to obtain the value for *c* in Equation 9.1.[17]

95% CONFIDENCE INTERVALS VERSUS RETROSPECTIVE POWER

What do we learn from a 95% CI that we do not (or cannot) learn from retrospective power computations? Even though the meaning of confidence intervals is routinely misunderstood (Ellison & Dennis 2010), they are still much more informative than any measure of retrospective power.

A post hoc 95% CI plotted along with a parameter estimate (usually the mean, but a confidence interval can be calculated for any statistical estimator) is usually interpreted as "the parameter equals its observed mean $\pm$ 1.96 $\times$ its standard error." This is not, however, the correct meaning or interpretation of a confidence interval. In fact, a 95% (or any *n*%) CI gives the range of values in which the *true* value of the parameter (*not* its estimate) is expected to occur in 95% (or any *n*%) of the cases.[18] For any given sample, we do not know whether the true mean is in the interval or not, but across many samples it should be within similarly constructed confidence intervals 95% of the time. We presented a simulation to show why this is the correct interpretation of a 95% CI earlier in this book (Code Box 2.5 and Figure 2.3 in Chapter 2).

Like prospective power, CIs are an expression of probability and a rate of error. Thus, CIs and prospective power estimates can be compared directly because both are on probability scales. However, retrospective power is not a probability (or a rate), so it is not directly comparable with a CI. A CI tells us (more or less) that if we did the same study over and over again, (approximately) 95% of the time we would "get it right" and 5% of the time we would miss and get a false negative (commit a type-I error).

If the 95% CIs of the two means we are comparing overlap with one another, we can be pretty safe in saying that our treatment did not find an effect and, given our experimental design, we are not likely to find it even if we were to run the experiment again. Similarly, if the 95% CI of the estimated slope of the relationship between our predictor variable and our observed response variable includes zero, we can be pretty safe in saying that we are not likely to see a relationship between these two variables if we collect another similar set of observations. Confidence intervals do not, however, tell us anything about the rate of false positives we would turn up. But remember, retrospective power does not tell us that either (for additional comparisons of CIs and retrospective power, see §3 of Hoening & Heisey 2001).

USE PREDICTIVE CONFIDENCE INTERVALS TO BETTER DETERMINE SAMPLE SIZES AND ANTICIPATE THE RELIABILITY OF POST HOC INFERENCES

There is general agreement among statisticians and researchers who use statistics that we are much better off if we report parameter estimates and CIs in addition to (or indeed, in place of) *p*-values (Wasserstein & Lazar 2016). Similarly, writing for medical researchers,

[17] Note that the percentile we want to use is from one of the two symmetrical tails of a two-tailed *t*-distribution. Thus, for a 90% CI with $N = 50$ (and corresponding $df = 49$) we would use the command `qt(0.95,49)` because 5% of the distribution is in one tail and another 5% of the distribution is in the other tail.

[18] The situation is actually more complex than even this tongue-twisting sentence might suggest. Because the truth is typically not known, an *n*% CI only includes the mean of all the imaginary replicates in *n*% of an infinite number of trials in which similar confidence intervals were computed and all the assumptions used to compute the intervals were identically correct (Greenland et al. 2016). The two statements are essentially equivalent only if the statistical model is asymptotically unbiased, which it is for most commonly used probability distributions and statistical models.

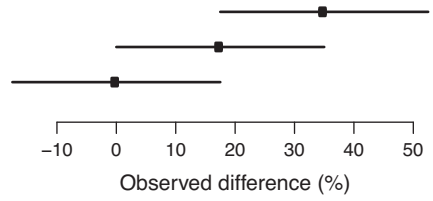


Figure 9.3 Predictive 95% confidence intervals for hypothetical findings of 0, 17.5, and 35% differences when the sample size is identified as $N = 80$ per group to detect a difference of 25% between treatments with 80% power and type-I error rate $\alpha = 0.05$. If we observe no difference, we can be fairly certain that the true effect is $< 25\%$ because the 95% CI does not include 25%.

Bristol (1989) suggested that using consistent methods based on CIs for determining sample size and parameter estimation would have the positive effect of reinforcing the value of parameter estimation over simply reporting p-values. Writing for the same audience, Goodman & Berlin (1994) reiterated this suggestion, focusing attention on the fact that the width of a 95% (or, indeed, any n%) CI and prospective power both depend on sample size. This dependency allows us to determine the sample size we should use to have confidence that we detected a given effect size.

Specifically, Goodman & Berlin (1994) argued that if we are looking for an effect size Δ, we should ask: What is the estimated (predictive or a priori) CI on it, given a sample size, desired power $(1 - \beta)$, and an acceptable type-I error rate α?[19] For the common situation in clinical trials, where the interest is in comparison between two groups (e.g., a control and a therapeutic intervention), we can use a little arithmetic to estimate the predictive CI as:[20]

$$\text{Predictive 95\% CI} = \text{observed difference} \pm 0.7 \cdot \Delta_{0.8}, \tag{9.3}$$

where $\Delta_{0.8}$ is the estimated effect size detectable with 80% power for a sample size determined from standard prospective power calculations.[21]

For example, consider the situation in which the results of a prospective power calculation showed that we would need 80 individuals per group to detect a treatment effect (Δ) of 25% with 80% power and $\alpha = 0.05$. Using Equation 9.3, we would estimate that the result of our experiment would have a predicted precision (i.e., the width of the 95% CI) equal to $0.7 \times 25\% = \pm 17.5\%$ (Fig. 9.3). This result also tells us that 17.5% is the smallest observed difference between our control and our therapeutic intervention that will be statistically significant ($p < 0.05$).

What additional information does a predictive CI give us? Figure 9.3 illustrates how imprecise different plausible results of our trial could be. If we observe no difference between the control and the treatment, we cannot exclude the reasonable possibility that differences between the groups of up to 17.5% could have occurred by chance alone. Even if we observed a 35% difference, we could not reasonably exclude a difference of 17.5%, which is right on the edge of statistical significance at $p = 0.05$. The 95% CI of the borderline-significant result of a 17.5% difference still just includes zero. What about differences outside of the CI? They

[19] Note that this predictive CI is not the same as the prediction interval discussed in §8.5 in Chapter 8.

[20] Equation 9.3 assumes that the random variables are normally distributed, the observations are i.i.d, and the within-group variances are homogeneous.

[21] Equation 9.3, from Goodman & Berlin (1994), is accurate for per-group sample sizes > 20, as is their equation for Δ detectable with 90% power: Predicted 95% CI = observed difference $\pm 0.6 \cdot \Delta_{0.9}$. The exact formulas for normally distributed and binomial data of any sample size are given in Bristol (1989).

are not impossible; there simply would be less supporting evidence for them than for values within the CI (Goodman & Berlin 1994).

What is really important, though, is the difference that a clinician or a regulatory panel cares about. If it is 25%—the effect size we used in Figure 9.3 to determine our sample size—then our predictive CIs tell us that if we observed a difference of $< 20\%$ we could be pretty confident that our result was not a false negative. However, if the clinically meaningful difference is only 17.5%, then even observing a 25% difference could be a false positive (Figure 9.3). Equation 9.3 can be modified easily for different sample sizes and the data replotted. Such an exercise would give a sense of how much (or how little) the precision of our results would change with power (Goodman & Berlin 1994). In short, predictive CIs in tandem with prospective power calculations tell us more about what we might learn from our experiment than prospective power calculations alone.

This is an interesting idea, but it appears to have been used only once in the last 30 years (Kapp et al. 2006). Kapp et al. (2006) used Goodman and Berlin's (1994) exact procedure to compute predictive CIs and determine the necessary sample size for an experiment testing the efficacy of mifepristone and misoprostol on inducing medical abortions. Four other papers used analytical models to compute predictive CIs and estimate observable effects expected by their models. The models were then confronted with actual data on trajectories of particles in ocean currents (Ullman et al. 2006), rainfall during typhoons (Chen 2019, 2021), or marine engine failures (Fu et al. 2023).[22]

9.3.2 Minimum detectable effects (MDEs) and minimum detectable differences (MDDs)

Cohen (1988) described how a detectable effect size can be determined if N, α, and $1 - \beta$ are specified. Although Cohen (1988) thought that these calculations would be less useful prospectively than computing either power or sample size for a given expected effect size, he did suggest that determining the detectable effect size could be used during literature reviews and similar retrospective studies. In particular, Cohen (1988) suggested as a convention that we should use the effect size that could be detectable for $\alpha = 0.05$ and $1 - \beta = 0.5$ as the comparative detectable effect size ("CDES"), which he equated with the inverse of the sensitivity (i.e., the probability of a false positive, or type-I error), albeit expressed in the units of effect size, not on a probability scale. The apparent intent of the CDES was for it to be examined across studies being synthesised in literature reviews and meta-analyses. However, there is little evidence that the CDES has ever been used this way. Rather, it has been repurposed for retrospective analysis of non-significant studies in two ways—as the MDE and the MDD.

THE MINIMUM DETECTABLE EFFECT (MDE)

One of the earliest transformations of the CDES into the MDE was effected by Rotenberry & Wiens (1985). These two ecologists wanted to better quantify the then increasingly common finding that competition (negative effects on survivorship or reproduction) between species was rare among many animals (Strong et al. 1984). But rather than using the 50% power and the $\alpha = 0.05$ criteria that Cohen (1988) had suggested for computing the CDES, Rotenberry & Wiens (1985) used 95% power and discovered that the true effect could not exceed (in their

[22] These results are from a Web of Science Search done by the authors on 12 October 2023 using only one search term—{"predictive confidence interval"}—applied to all indexed fields. The search returned 11 articles, of which only Kapp et al. (2006) used predictive CIs as Goodman & Berlin (1994) intended. The four aforementioned model-based papers did not use the predictive CIs to determine sample sizes (with fixed α and $1 - \beta$). The remaining six papers retrieved by the search incorrectly referred to standard (post hoc) CIs as predictive CIs.

case) $r = 0.48$. This use of what has since come to be called the minimum detectable effect (MDE)—using a pre-specified α and power (now most commonly 80%)—was taken up by subsequent researchers and continues to be used extensively in fields ranging from education and psychology to ecology, conservation, and medicine (e.g., Anderson 2002; Loos et al. 2015; Djimeu & Houndolo 2016; Arend & Schäfer 2019; Karsa et al. 2020; Castañon et al. 2022; Mair et al. 2020; Yildiz et al. 2022).

Determining the MDE is straightforward using any software that can compute power. If these calculations are done prospectively, the result would indicate the minimum effect size that we could determine to be statistically significant for a pre-defined power and a sample size determined by available resources. If we thought that the actual effect was likely to be smaller than this MDE, then either we would need to find more resources or we would have to adjust our expectations of significance and power.

After we have completed our study, however, we are only likely to compute a MDE if our results are not significant (as did Rotenberry & Wiens 1985). If the MDE turns out to be larger than the effect we found, we are back in the same situation as having computed retrospective power—concluding that if only we had had a larger sample, we would have found a significant effect. This has the same set of problems that we identified for retrospective power analysis earlier in this chapter (§9.2.2).

REPACKAGING RETROSPECTIVE POWER INTO MDE SOLVES NO PROBLEM

Hoening & Heisey (2001) identified another problem with the MDE, which they called the power approach paradox ("PAP"; see also Hoening & Heisey 2001; Mair et al. 2020). In brief, the PAP describes the situation we might encounter if we have two experiments, each of which yields a non-significant result (i.e., $p \geq 0.05$). If the observed effects differ, but the 95% CIs are the same size and both include zero, an MDE calculation would correctly identify no effect but our intuition would suggest that the experiment with the smaller effect size (i.e., closer to zero; Case 1, Experiment 2 in Figure 9.4) provides stronger support for the null hypothesis. In fact, the support for the null hypothesis is equivalent in both cases.

Another example of the PAP would be for two non-significant results with equivalent effect sizes, but whose 95% confidence intervals both include zero and one is much larger than the other (Case 2 in Figure 9.4). The MDE is smaller for the experiment with the smaller CI (Experiment 2 in Case 2), but our intuition would again incorrectly suggest that the experiment with the larger CI (Experiment 1 in Case 2) provides stronger support for the null hypothesis.

Finally, if the estimated effect sizes for the two experiments are equal to one another and are both nearly zero (Case 3 in Figure 9.4), there is no PAP. Our intuition agrees with the result of the MDE: the experiment with the smaller 95% CI provides stronger evidence for no effect (Experiment 1 in Figure 9.4).

THE MDD AS A SUBSTITUTE FOR THE MDE

In evaluating potential harmful effects of pesticides and other potential environmental contaminants, ecotoxicologists use the minimum detectable difference (MDD) instead of the MDE to interpret nonsignificant results (i.e., when a potential toxin is found not to have toxic effects; van der Hoeven 2008; Mair et al. 2020). The MDD is defined as "the effect size that would have been *just significant* considering the sample size and variance measured in a completed experiment" (van der Hoeven 2008; Mair et al. 2020; emphasis added). That is, MDD is the effect size that corresponds to the precise critical value of the test statistic (α). Unlike MDE, which is conditional on α, sample size, and a predefined power, MDD is conditional on α, sample size, and the actual power of the experiment. Even though MDD is conditional on the actual power of the experiment, low values of the MDD relative to the

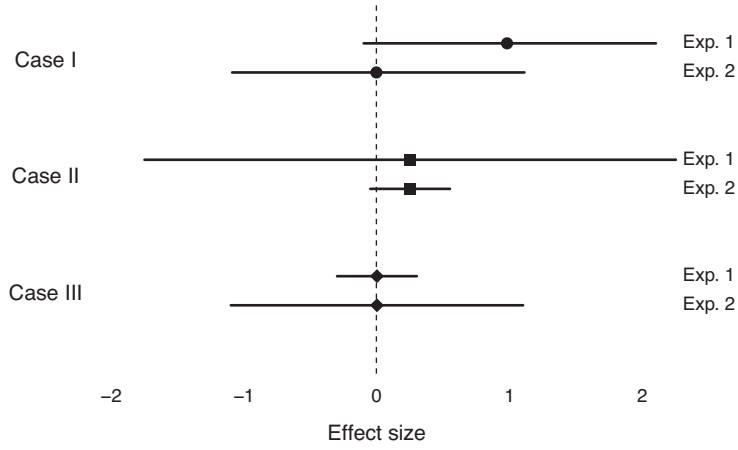


Figure 9.4 Illustration of the power approach paradox (PAP). In Case 1, computation of the minimum detectable difference (MDE) shows that both experiments provide equivalent evidence for an effect size (Δ) equal to zero, but paradoxically our intuition would suggest that Experiment 2, whose estimated effect size is closer to zero, provides stronger evidence for $\Delta = 0$. In Case 2, MDE computations give stronger evidence that $\Delta = 0$ in Experiment 2, but paradoxically our intuition suggests that Experiment 1 provides stronger evidence that $\Delta = 0$ because more of its confidence interval includes zero. There is no paradox in Case 3, where estimates of MDE and our intuition agree: $\Delta = 0$. See Hoening & Heisey (2001) and Mair et al. (2020) for additional discussion.

control mean (i.e., the proportional MDD or pMDD) are interpreted incorrectly to mean that the experiment had higher power to detect a true positive.[23]

For the *t*-test commonly used in two-level toxicological studies, the MDD is equal to the upper bound of the 95% CI minus the estimated effect; it is similar to the CI constructed around the effect size expected under H_0 (typically equal to zero in toxicological studies; Mair et al. 2020). However, the MDD does not have good power; indeed its asymptotic power is only 50%.[24]

Mair et al. (2020) used simulations to determine the relative merits of MDDs, MDEs, and confidence intervals to discriminate between true null effects and false negatives (Figure 9.5). Their goal was to determine how much we should trust that non-significant results are, indeed, indicative of no toxicological effect. For an experiment with two treatments—an experimental application of a toxin and a control—Mair et al. (2020) defined two error rates, the false trust rate and the false mistrust rate, to be:

$$\text{False trust rate} = \frac{\Sigma(\text{Trust a non-significant result AND There is a real effect})}{\Sigma(\text{A real effect exists})}$$

$$\text{False mistrust rate} = \frac{\Sigma(\text{Mistrust a non-significant result AND There is no real effect})}{\Sigma(\text{No effect exists})} \tag{9.4}$$

[23] In a regulatory setting, *p*-values are used to determine the lowest-observed-effect concentrations ("LOEC") of potentially harmful compounds and pMDD values are used as post hoc indicators of the level of evidence for true positives in subsequent measurements with non-significant *p*-values (Mair et al. 2020).

[24] Of course, the MDE has nothing to say about power either, as power has to be pre-specified to compute the MDE.

The investigator uses the proportional values of CI, MDD, and MDE, where pCI, pMDD, and pMDE equal, respectively, the upper bound of the 95% CI of the experimental treatment mean over the upper bound of the 95% CI of the control mean; the MDD of the experimental treatment over the MDD of the control; and the MDE of the experimental treatment over the MDE of the control, assuming 80% power to decide whether they (mis)trust the results. This decision is based on whether the pCI, pMDD, or MDE exceeds an investigator-defined threshold value.

Mair et al. (2020) then simulated 1,000 experiments with no toxicological effects ($\Delta = 0$) and another 1,000 experiments with moderate toxicological effects ($\Delta = 0.5$); sample sizes, control means, and variances varied (3–10, 10–100, and 1–50, respectively) among the experiments (Code Box 9.3). They then plotted the relationship between the False trust rate and False mistrust rate for a range of predefined threshold values (Figure 9.5).

Code Box 9.3: Simulation of experiments and computation of pCI, pMDD, and pMDE

```
# Code reproduced from Mair et al. (2020); CC-BY-SA 4.0
# Available online at http://doi.org/10.5281/zenodo.3951619

# Define number of experiments to be simulated:
size = 2000

# Define "true" parameters:
experiments = data.frame(
 effect = c(rep("no",size/2), rep("yes",size/2)),
 effectsize = c(rep(0,size/2), rep(0.5,size/2)), # 1/2 no effect, 1/2
      effect of 0.5
 theta = runif(size, max = 50, min=1),
 sampleSize = sample(3:10, size, replace = T),
 abund = sample(10:100, size, replace=T)
)
experiments$variance = experiments$theta * experiments$abund

# Prepare empty output dataframe:
out <- data.frame(
 p = rep(NA,size),
 pMDD = rep(NA,size),
 pMDE = rep(NA,size),
 pCI = rep(NA,size)
)

# Simulate and analyze experiments:
for (i in 1:size)
{
 N <- experiments$sampleSize[i]
 theta <- experiments$theta[i]
 abund <- experiments$abund[i]
 effectsize <- experiments$effectsize[i]
```

```
  # Simulate experiments:
   control <- rnbinom(n = N, mu = abund, size = abund/(theta-1))
   tmean <- abund * (1 - effectsize)
   treatment <- rnbinom(n = N, mu = tmean, size = tmean/(theta-1))

  # log-transform data prior to analysis:
   transc <- log((2*control)+1)
   transt <- log((2*treatment)+1)

  # Perform tests and calculations on transformed data:
   test <- t.test(transc, transt, alternative = "greater", var.equal = T)
   mdd.ln <- MDD(N1 = N, N2 = N,
    variance1 = var(c(transc-mean(transc),transt-mean(transt))),
    alpha=0.05, two.sided = F, var.equal = T)
  postpower <- power.t.test(n = N, delta = NULL,
   sd = sd(c(transc-mean(transc),transt-mean(transt))),
   sig.level = 0.05, alternative="one.sided", type = "two.sample",
   power = 0.8)
  upperCI <- mean(transc)-mean(transt) +
    qt(0.05/1, df = 2*N-2, lower.tail = FALSE) *
    sqrt(var(c(transc-mean(transc),transt-mean(transt))))
    * sqrt(2/N)

  # Back-transformation:
   mdd.abu <- (exp(mean(transc))-exp(mean(transc) - mdd.ln$mdd))/2
   mde.abu <- (exp(mean(transc))-exp(mean(transc) - postpower$delta))/2
   upperCI.abu <- (exp(mean(transc))-exp(mean(transc) - upperCI))/2

  # Relation to control mean (compute proportionalities):
   pMDD <- 100 * mdd.abu / ((exp(mean(transc))-1)/2)
   pMDE <- 100 * mde.abu / ((exp(mean(transc))-1)/2)
   pCI <- 100 * upperCI.abu / ((exp(mean(transc))-1)/2)

  # Store results in output dataframe for later use:
   out[i,1] <- test$p.value
   out[i,2] <- pMDD
   out[i,3] <- pMDE
   out[i,4] <- pCI
}
```

In all cases, confidence intervals outperformed both MDDs and MDEs (Figure 9.5). For any threshold value, the False mistrust rate was lower for pCI than for either pMDD or pMDE. Similarly, the False trust rate was lower for pCI than for either pMDD or pMDE for threshold values ≤ 80. False mistrust rates were nearly equivalent for pMDD and pMDE, but their false trust rates were slightly offset (Figure 9.5). As have many other authors (e.g., Goodman & Berlin 1994; Hoening & Heisey 2001; Colegrave & Ruxton 2003), Mair et al. (2020, p. 2121) concluded that "a stronger emphasis on estimated effect size and their confidence intervals instead of hypothesis tests would lead to better decisions and greater transparency."

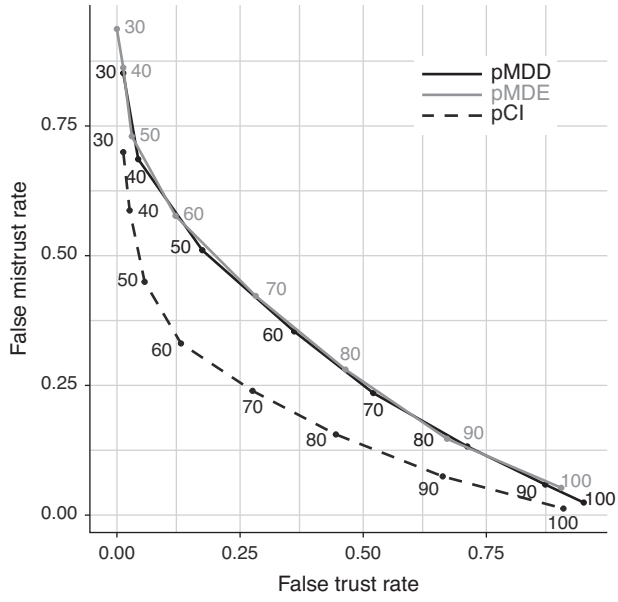


Figure 9.5 Relationship between False trust rates and False mistrust rates (Equation 9.4) across a range of threshold levels of trust (30–100%) for pCI (black), pMDD (dark grey) and pMDE (lighter grey) from a set of 2,000 simulated toxicological experiments with equal number of experiments yielding no significant effect ($\Delta = 0$) or a significant effect ($\Delta = 0.5$). For all threshold values, errors in trust and mistrust are lower when based on pCIs than on pMDDs or pMDEs. The ideal curve would hug the origin. (Figure based on code of Mair et al. (2020), and their Figure 5B.)

9.3.3 Faul et al.'s alternative post hoc analyses

Faul et al. (2007, 2009) proposed four different options for post hoc analyses: post hoc power analysis, compromise power analysis, sensitivity analysis, and criterion analyses. We consider these four options to be only one actual alternative to retrospective power analysis, as each one is simply a rearrangement of terms in the equation that relates statistical power ($1 - \beta$) to sample size, effect size, and the acceptable value of a type-I error (α). These post hoc options are implemented in the free, menu-driven software package, G*Power (Buchner et al. 2020).

POST HOC POWER ANALYSIS

Unlike retrospective power analysis but like prospective power analysis, Faul et al.'s confusingly-named post hoc power analysis (see footnote 2, above) "requires an H1 effect size [i.e., the alternative to the null hypothesis of no effect] specification *for the underlying population*" (emphasis in the original), not that observed (i.e., estimated) from the sample. Prima facie, this seems to be a reasonable proposal, although one could argue that if we knew the effect size for the underlying population, we would have used it to compute power prospectively given our proposed sample size and α. In addition, as far as we can tell, the many researchers who have used G*Power to determine post hoc power simply use the observed effect size and therefore are computing retrospective power, not post hoc power, and, as is all too common, misinterpreting it. An alternative approach to using population-level estimates of effect sizes can be found in Gelman and Carlin's (2014) design analysis (see §9.3.4, below).

COMPROMISE POWER ANALYSIS

Compromise power analysis illustrates the trade-off between α and β that is inherent to classical hypothesis testing. In compromise power analysis, both α and $1 - \beta$ are computed as functions of the effect size (estimated for the population whether the computations are done prospectively or retrospectively), the sample size (N), and the ratio of error probabilities (α/β). The value of this approach is that it forces an explicit acknowledgement that, by convention, we place a much higher value on type-I errors than on type-II errors (see also footnote 13 in Chapter 3). Thus, compromise power analysis shows us how changing our values affects our assertions of statistical significance (see also Gigerenzer et al. 2004).

SENSITIVITY ANALYSIS

In a further instance of nomenclatural confusion, Faul et al. (2007) implement sensitivity analysis in G*Power to estimate the population-level effect size that could have been detected given predefined values of α, $1 - \beta$, and N.[25] This is equivalent to identifying the MDE (see §9.3.2, above) that would be detectable given α, $1 - \beta$, and N. Faul et al. (2007, p. 177) assert that computing this measure of sensitivity "may be particularly useful for evaluating published research [as it] provides answers to questions such as 'What effect size was a study able to detect with a power of $1 - \beta = 0.8$ given its sample size and α as specified by the author?'" It appears, however, that the estimation of sensitivity in G*Power simply fixes α, $1 - \beta$, and N and solves for the minimum effect size Δ_{min} using the standard equation for estimating retrospective power. Thus, this variant of sensitivity analysis is as unreliable as retrospective power analysis, estimates of MDE, or estimates of MDD.

CRITERION ANALYSIS

Finally, criterion analysis completes the rearrangement of terms in the retrospective power equation and solves for α. Here, Faul et al. (2007) refer only to effect size (easily considered by readers to be the observed effect size even though Faul et al. presumably mean the unknown population-level effect size). They suggest (p. 177) that the computation of α "as a function of $1 - \beta$, the effect size, and a given sample size … is an alternative to post hoc power analyses [that] may be reasonable whenever the control of α is less important than the control of β."

HOW SHOULD THE EFFECT SIZE OF THE UNSAMPLED POPULATION BE ESTIMATED?

There are two standard ways that researchers prospectively determine the effect size used in calculations of statistical power: empirically and on the basis of their experimental goals. An empirical determination of effect size assumes that the effect size of our experiment will be approximately equal to that estimated in a previous study, such as a pilot study. Alternatively, a goal-based effect size is chosen based on what we think of as the minimum effect that would be "important," such as might occur in a clinical trial (see §9.3.1, above).

Although G*Power can calculate post hoc (and prospective) power, along with their rearrangements (in compromise, sensitivity, and criterion analyses) for a large number of common statistical models, Faul et al. (2007) provide no guidance on how to estimate the effect size of the larger population from which the sample was drawn. They suggest that

[25] "Sensitivity" is used by scientists in two ways, neither of which relates directly to the effect size being estimated by G*Power. On its own, the sensitivity of a test is the rate at which a test detects a true positive conditional on the effect being, in fact, true (see §3.4.3 in Chapter 3). When paired with "analysis," however, sensitivity analysis normally is used by scientists to determine how the range of possible outputs of a mathematical model, numerical system, or complex system can result from variation in inputs to the model or system. It is used widely in fields ranging from economics to ecology and is particularly important for forecasting the wide range of global climate-change scenarios (e.g., Murphy et al. 2004).

Cohen's categories of small, medium, and large effect sizes would be helpful in specifying the population-level effect sizes (but see footnote 16 in Chapter 3 and Correll et al. 2020 for a critique of this T-shirt size approach); the default values in G*Power generally correspond to medium effect sizes.

Gelman & Carlin (2014), among others, have criticised these kinds of approaches because they tend to overestimate actual effect sizes. They suggest considering the true effect size D that would be observed in a hypothetically infinitely large sample that would completely encompass the entire population of interest (this is analogous to Faul et al.'s [2007] otherwise undefined population-level effect size). Gelman & Carlin (2014) go on to recommend that the estimate of D should come from *external* information (their emphasis), such as data available from other studies, literature reviews (with attention to the well-documented biases in publishing only significant effects that we described in §9.2.2, above), meta-analyses, and (hierarchical) models.

We emphasise the importance of clear, quantitative models in helping us think about the upper and lower bounds of plausible effect sizes. Quantitative models focus our attention on particular parameters or processes of interest, identify what is measurable, define plausible effect sizes, and sharpen hypotheses. In the last decade, large-scale studies in neuroimaging, linguistics, microeconomics, sociology, behaviour, and ecology and evolution have revealed that different investigators analysing the same dataset do not estimate the same effect size—or even the same direction (sign) of the effect (Silberzahn et al. 2018; Botvinik-Nezer et al. 2020; Huntington-Klein et al. 2021; Schweinsberg et al. 2021; Breznau et al. 2022; Coretta et al. 2023; Gould et al. 2023). In many of these studies, however, the question that the different investigators were supposed to answer with the common dataset was determined in retrospect to be ill-framed, poorly worded, or unintentionally vague (Auspurg & Brüderl 2021). The effects of researchers' a priori beliefs or specific methodological choices on different estimates of effect sizes and directions were also quantified in three of these studies (Silberzahn et al. 2018; Breznau et al. 2022; Coretta et al. 2023); Schweinsberg et al. (2021) developed an `R`-based tool (`DataExplained`)[26] to document the methodological choices researchers make when analysing datasets (see also Lerner et al. 2022).

9.3.4 Design analysis

Gelman & Carlin (2014, p. 642) suggested that retrospective analysis—and specifically, their prescription of design analysis—is most appropriate when "apparently strong (statistically significant) evidence for nonnull effects has been found." Note that this is the antithesis of the reason that virtually all researchers use retrospective power analysis or its alternatives, which is to explain away why they fail to find statistically significant effects and could not reject their null hypothesis.

Rather than looking for reasons for rejecting or failing to reject a null hypothesis, Gelman & Carlin (2014) recommended that if an effect is determined to be statistically significant using a conventional hypothesis test, we then retrospectively determine the probability that the observed (estimated) effect was in the wrong direction from the true effect (the type-S [sign] error rate) and the factor by which the size of the effect was overestimated (a type-M [magnitude] error, equal to the absolute value of the expected effect size / observed effect size).

Estimating type-S and type-M errors requires an estimate of the effect size—which they call d^{rep}—that would have been observed in a hypothetical (but not done) study with an

[26] Available from https://dataexplained.net/

Code Box 9.4: Gelman and Carlin's (2014) original `retrodesign` function

```
# Required inputs to retrodesign include the effect size (A) and its standard
    error (s). The critical value (alpha), degrees of freedom (df) and the
    number of simulations (n.sims) have default values that can be overridden
    by the user.

 retrodesign <- function(A, s, alpha=.05, df=Inf, n.sims=10000){
    z <- qt(1-alpha/2, df)
    p.hi <- 1 - pt(z-A/s, df)
    p.lo <- pt(-z-A/s, df)
    power <- p.hi + p.lo
    typeS <- p.lo / power
    estimate <- A + s * rt(n.sims, df)
    significant <- abs(estimate) > s * z
    exaggeration <- mean(abs(estimate)[significant]) / A
    return(list(power=power, typeS=typeS, exaggeration=exaggeration))
}
```

identical design to the one being analysed.[27] In other words, d^{rep} is the effect size that would have been obtained if the study were repeated in the same way and with the same sample size, and the sample was taken from an (infinitely) large population with true effect size D (for further details on estimating D, see our discussion of estimating effect sizes in §9.3.3, above). Gelman & Carlin (2014) also use d^{rep} to estimate power, which they define as the probability that $|d^{\text{rep}}|$ is greater than the critical value that would define statistical significance in the analysis under consideration.

The computation of power and type-S and type-M errors are done easily with the R package `retrodesign` (Timm et al. 2023).[28] The `retrodesign` package includes the function `retro_design_closed_form()` that computes exact values of power, type-S error rate, and type-M error using a closed-form analytical solution, but this solution is only applicable assuming a normal distribution. For non-normal distributions or smaller sample sizes, where we might use a t-distribution, the function `retrodesign()` determines power, the type-S error rate, and the type-M error through simulation (Code Box 9.4).[29]

In general, unless our study is severely underpowered (power $\ll 0.6$; Figure 9.6, right), we should worry much more about type-M errors (overestimating an apparently significant

[27] d^{rep} is similar, but not identical, to the unobserved and unknown population-level effect size required in G*Power's retrospective calculations (§9.3.3; and see also Faul et al. 2007). d^{rep} is estimated for a hypothetical sample of equivalent size, whereas Faul et al.'s (2007) estimated effect would be for the entire population. However, in all subsequent calculations both Faul et al. (2007) and Gelman & Carlin (2014) use the same sample standard deviation, s (i.e., the SEM), which is estimated from the observed data. A similar approach is used by Quach et al. (2022), who compute posterior distributions of retrospective power by repeatedly simulating an effect size drawn from the distribution $\mathcal{N}(\hat{\Delta}_N, \frac{1}{N_1} + \frac{1}{N_2})$, where $\hat{\Delta}_N$ is the observed standardised effect size $\frac{\bar{Y}_1 - \bar{Y}_2}{s}$ for sample size $N = N_1 + N_2$, the sample sizes of the two groups being combined.

[28] Note that `retrodesign` is irregularly maintained and, at the time we are writing this book, has been deprecated from CRAN. It is still available on Andy Timm's GitHub site (https://github.com/andytimm/retrodesign).

[29] We include the complete code here for `retrodesign()` from Gelman & Carlin (2014) for its pedagogical utility.

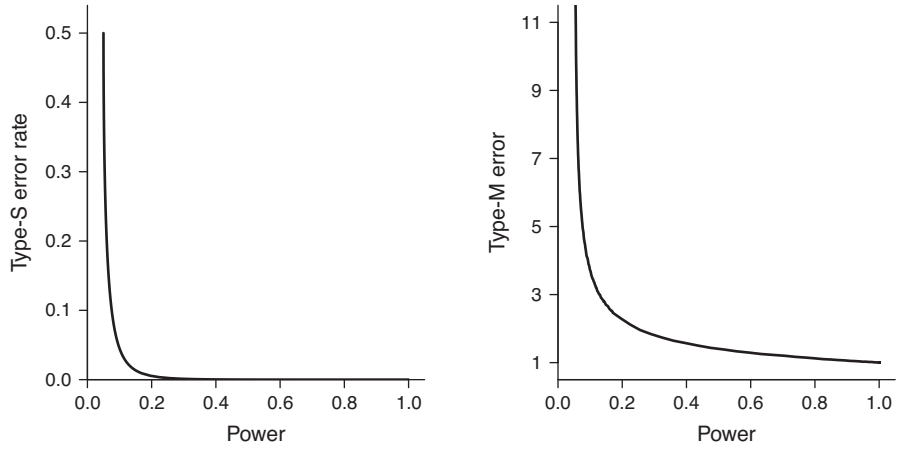


Figure 9.6 Type-S error rate (left panel) and type-M error (right panel) as a function of statistical power for unbiased estimates that are approximately Normally distributed (i.e., *t*-distributed). For studies with reasonably high power ($\gg 0.6$), the type-S error rate is nearly zero and the type-M error is < 1.5. As power declines below 0.6 , the type-M error increases rapidly. When power is unusually low (< 0.2), the probability of a type-S error also increases rapidly. Modified from Figure 2 of Gelman & Carlin (2014).

effect) than the type-S error rate (which rapidly approaches zero when power > 0.2; Figure 9.6, left).

As an example of how we might use a post hoc design analysis, we explored the results of the "Many Analysts, One Data Set" study (Silberzahn et al. 2018). In this widely-discussed study (e.g., Auspurg & Brüderl 2021), the same dataset was given to 29 teams who were tasked with using the data to determine whether soccer referees are more likely to give "red cards" to players with darker skin tones than players with lighter skin tones.[30] The different teams used analytic approaches ranging from ordinary linear regression to complex hierarchical Bayesian models to test their hypotheses. All the teams estimated effects and CIs around them. The effects, expressed as odds-ratios, ranged from 0.89 to 2.93 (median odds-ratio = 1.31); 69% of the teams reported a statistically significant ($p < 0.05$) effect of skin tone on the odds of getting a red card, whereas the remaining teams found no significant effect.

We applied design analysis to the overall results of Silberzahn et al.'s (2018) study. We used the median effect size = 0.15 (calculated from the odds-ratio [OR] = 1.31 using the standard formula $d = \log(\text{OR})\frac{\sqrt{3}}{\pi}$ from Sánchez-Meca et al. 2003) and an estimated standard error of 0.055 (Code Box 9.5).

Figure 9.7 shows that with an estimated $D = 0.15$ and $s = 0.055$, we would have a power to detect an effect equal to 0.78, a type-S error rate $< 2 \times 10^5$, and little likelihood of committing a type-M error (mean M = 1.002).

Applying design analysis to the "Many Analysts, One Dataset" study suggests that the many analysts using a range of different methods were likely to have really detected an effect of skin tone on the odds of getting a red card. Although that was not the actual focus of Silberzahn et al. (2018), who were more interested in understanding the differences among research groups, it is also important to identify similarities among outcomes of similar crowd-sourced studies. For example, we expect that the results of a design analysis applied to two other "many analyst" studies published recently—Breznau et al. (2022) and Gould

[30] A red card signals that a player has broken an important rule of the game. A player given a red card is required to leave the game immediately.

Code Box 9.5: Design analysis of Silberzahn et al.'s (2018) results.

```
# This simulation uses routines built-in to R's retrodesign package. Required
    inputs to retrodesign include the effect size (A) and its standard error
     (s). Default values for other parameters are alpha = 0.05, infinite
    degrees of freedom, and 10,000 simulations.

# Load required libraries
 library(retrodesign)
 library(ggplot2)   # Plotting routines in retrodesign uses ggplot2 if "gg =
     TRUE"

# Compute retrospective power, and type-S and type-M error rates

 A <- 0.15  # Median effect size
 s <- 0.055 # Estimated population standard deviation
 set.seed(123)
 retrodesign(A, s, alpha = 0.05, df = Inf, n.sims = 10000)

| $power
| [1] 0.7785524
|
| $type_s
| [1] 1.77842e-06
|
| $type_m
| [1] 1.001439

# Illustration of probability of type-S and type-M errors in Silberzahn et al.'s
     (2018) results.
 sim_plot(A, s, alpha = 0.05, df = Inf, n.sims = 5000, gg = TRUE)
```

et al. (2023)—would be less sanguine. Breznau et al. (2022) found that about equal numbers of research teams (among a set of 73) reported negative or positive effects of immigration on the degree of public support for social policies, and a majority of the teams could not reject the null hypothesis. Gould et al. (2023) asked 174 teams to evaluate two different hypotheses (each hypothesis tested with a different dataset) in evolutionary ecology and conservation biology. One hypothesis—that there is an inverse relationship between number of sibling nestlings of a small bird and individual nestling growth—was supported by most teams, although there also were apparent type-S errors. The other hypothesis—that there is some (unspecified) relationship between the amount of grass cover and the number of *Eucalyptus* seedlings—was supported in more than 50% of the analyses, but these included negative, positive, and "mixed" responses.

9.4 Summary: Most retrospective analyses should be avoided

If the results of our experiments fail to reject our null hypothesis, we are unlikely to argue convincingly otherwise. To paraphrase Madison Avenue, 99 out of 100 statisticians agree:

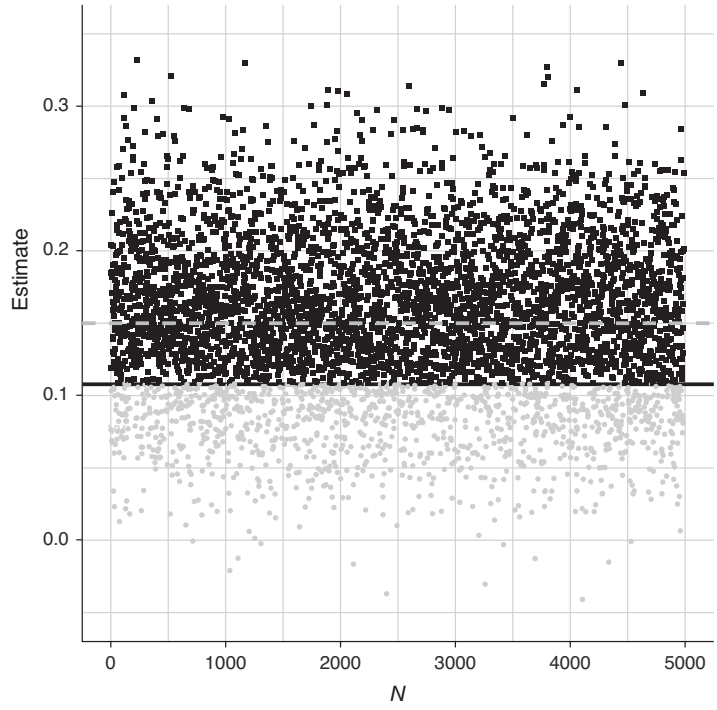


Figure 9.7 Results of a design analysis of the estimated effect of skin tone on the probability of a soccer player getting a red card. We ran 5,000 simulations and each point is the result of a single one of them. The horizontal dashed line at an estimate = 0.15 (equivalent to an odds-ratio of 1.31) indicates the true effect size (the median of the 29 analyses reported in Silberzahn et al. 2018). The solid horizontal line is the smallest effect size for which an estimate would be considered statistically significantly different from zero ($p < 0.05$), given our estimated standard error of 0.055. The grey points indicate the values from simulations for which the effect sizes were not statistically significant and the black squares indicate the values from simulations that would not be considered type-M errors. Because this study had good statistical power (0.78), our simulations did not return any type-S errors, which would have been indicated with black triangles in the plot.

retrospective power analysis and its close cousins, including post hoc power and other rearrangements of the basic prospective power computations applied to results already obtained (Faul et al. 2007), and MDE and MDD (Mair et al. 2020), have no place in our statistical toolbox. Furthermore, justifications for determining retrospective power based on it being easier to interpret than *p*-values that have the exact same information content will lead only to more confusion than to more clarity in interpreting results (cf. Bababekov & Chang 2019; Helminen & Reito 2019a).

In sum:

- Retrospective power analysis is fatally flawed. Fundamentally, prospective power is an assertion of probability (1− the probability of a type-II error), but retrospective power analysis reflects only a single result: the outcome of the study. The data are what they are, and the result is either significant or not. More succinctly, retrospective power equals 1 if the results are significant and equals 0 if they are not.
- Alternatives to retrospective power analysis include: confidence intervals computed a priori or after the study has been completed; the computation of minimum detectable effects and minimum detectable differences; Faul et al.'s (2007) "post hoc" power analysis; and its close relative Gelman & Carlin's (2014) design analysis. All these

alternatives can be challenging to implement correctly and should be used with caution.

- Confidence intervals are the simplest way to illustrate concisely estimated effect sizes and uncertainty. If hypothesis testing and parameter estimation support the finding of a significant effect, adding in estimates of type-S and type-M errors would add additional confidence to the conclusions.

9.5 Coda: What would a Bayesian do instead?

The conceptual gymnastics we are required to perform when interpreting *p*-values (statistical significance), prospective power, confidence intervals, and the range of retrospective analyses we discussed in the previous sections certainly exercise our minds, but in most cases they do not actually help us learn what we really want to know: given an observed effect Δ, how probable is it that the effect Δ really is different from no effect at all? To answer that question—and more generally, what is the probability of our results (i.e., parameter estimates) given the data we observed?—we are much better off using Bayesian methods (e.g., Ellison 1996; Gelman et al. 2013; Kruschke & Liddell 2018). It is easier to communicate accurately the results from an experiment (i.e., the parameter estimates) and the actual degree of confidence that we have in the outcome, conditional on the data.

Or, to quote Daniel Lakeland,

> Power is a broken idea, the Bayesian version makes more sense. [We can use it to ask] how much data do you need to make the high probability posterior interval of your parameter estimate smaller than some desirable precision or width? This can be estimated by first drawing a parameter value from your prior, then generating data of a certain size *N*, then running the data through your Stan/inference engine [or any other MCMC method] to get a posterior sample. Repeat with different values of the parameters and different *N*. [This] only makes sense if you're using moderately informed priors.
>
> Post hoc power analysis in this context [only] makes sense if you ask the question "how many more data points do I need to get my posterior interval down to an even smaller interval?" [This would be appropriate only if we] use the posterior as the prior in re-doing the same analysis [see also Kruschke & Liddell (2018)]. If you want to decide how small your interval should be, use Bayesian decision theory.[31]

References

Althouse, A. D. (2021). Post hoc power: Not empowering, just misleading. *Journal of Surgical Research*, 259, A3–A6. doi: 10.1016/j.jss.2019.10.049.

Anderson, A. B. (2002). Detecting changes in natural resources using Land Condition Trend Analysis data. *Environmental Management*, 29(3), 428–436. doi: 10.1007/s00267-001-0017-z.

Anonymous (1964). Verdict on cigarets: Guilty as charged. *LIFE*, 56(4), 56A–64.

Arend, M. G. & Schäfer, T. (2019). Statistical power in two-level models: A tutorial based on Monte Carlo simulation. *Psychological Methods*, 24(1), 1–19. doi: 10.1037/met0000195.

Auspurg, K. & Brüderl, J. (2021). Has the credibility of the social sciences been credibly destroyed? Reanalyzing the "Many Analysts, One Data Set" project. *Socius: Sociological Research for a Dynamic World*, 7, 1–14. doi: 10.1177/23780231211024421.

[31] This quote is from a comment he posted on 24 September 2018 in response to Andrew Gelman's (2018) blog post, "Don't calculate post hoc power using observed estimate of effect size," in which he was critiquing Bababekov et al. (2018). The published version of Gelman's blog post appeared the following year (Gelman 2019), along with additional comments from Helminen & Reito (2019b; 2019a) and a reply by Bababekov & Chang (2019).

Aviles-Rosa, E. O., McGlone, J. J., & Hall, N. J. (2020). Use of a habituation-dishabituation paradigm to assess gilt olfaction and sensitivity to the boar pheromone. *Applied Animal Behaviour Science*, 231, 105086. doi: 10.1016/j.applanim.2020.105086.

Bababekov, Y. J. & Chang, D. C. (2019). Post hoc power: A surgeon's first assistant in interpreting "negative" studies. *Annals of Surgery*, 269(1), e11–e12. doi: 10.1097/SLA.0000000000002914.

Bababekov, Y. J., Hung, Y.-C., Hsu, Y.-T., et al. (2019). Is the power threshold of 0.8 applicable to surgical science?—Empowering the underpowered study. *Journal of Surgical Research*, 241, 235–239. doi: 10.1016/j.jss.2019.03.062.

Bababekov, Y. J., Stapleton, S. M., Mueller, J. L., et al. (2018). A proposal to mitigate the consequences of type 2 error in surgical science. *Annals of Surgery*, 267(4), 621–622. doi: 10.1097/SLA.0000000000002547.

Bartlett, M. E., Edmunds, C. E. R., Belpaeme, T., & Thill, S. (2022). Have I got the power? Analysing and reporting statistical power in HRI. *Journal of Human-Robot Interactions*, 11(2), 16. doi: 10.1145/3495246.

Botvinik-Nezer, R., Holzmeister, F., Camerer, C. F., et al. (2020). Variability in the analysis of a single neuroimaging dataset by many teams. *Nature*, 582(7810), 84–88. doi: 10.1038/s41586-020-2314-9.

Breznau, N., Rinke, E. M., Wuttke, A., et al. (2022). Observing many researchers using the same data and hypothesis reveals a hidden universe of uncertainty. *Proceedings of the National Academy of Sciences, USA*, 119(44), e2203150119. doi: 10.1073/pnas.2203150119.

Bristol, D. R. (1989). Sample sizes for constructing confidence intervals and testing hypotheses. *Statistics in Medicine*, 8(7), 803–811. doi: 10.1002/sim.4780080705.

Buchner, A., Erdfelder, E., Faul, F., & Lang, A.-G. (2020). G*Power: Statistical Power Analyses for Mac and Windows. https://www.psychologie.hhu.de/arbeitsgruppen/allgemeine-psychologie-und-arbeitspsychologie/gpower. Accessed 7 January 2024.

Butterworth, J. M. (2016). The Standard Model: How far can it go and how can we tell? *Philosophical Transactions of the Royal Society of London, Series A*, 374, 20150260. doi: 10.1098/rsta.2015.0260.

Carle, A. C. (2021). A response to some criticisms of post hoc power. *Journal of Surgical Research*, 259, A7–A8. doi: 10.1016/j.jss.2020.09.011.

Castañon, E., Sanchez-Arraez, A., Jimenez-Fonseca, P., et al. (2022). Bayesian interpretation of immunotherapy trials with dynamic treatment effects. *European Journal of Cancer*, 161, 79–89. doi: 10.1016/j.ejca.2021.11.002.

Chang, D. C. & Stapleton, S. M. (2021). Response: The proliferation and misinterpretation of "as safe as" statements in surgical science: A call for professional discourse to search for a solution. *Journal of Surgical Research*, 259, A12–A15. doi: 10.1016/j.jss.2020.03.074.

Chen, S.-T. (2019). Probabilistic forecasting of coastal wave height during typhoon warning period using machine learning methods. *Journal of Hydroinformatics*, 21(2), 343–358. doi: 10.2166/hydro.2019.115.

Chen, S.-T. (2021). Probabilistic typhoon rainfall forecasting using a modified fuzzy inference model. *Terrestrial Atmospheric and Oceanic Sciences*, 32(4), 583–596. doi: 10.3319/TAO.2021.07.23.01.

Cohen, J. (1962). The statistical power of abnormal-social psychological research: A review. *Journal of Abnormal and Social Psychology*, 65(3), 145–153. doi: 10.1037/h0045186.

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences*. New York, NY, USA: Academic Press, 2nd edition.

Colegrave, N. & Ruxton, G. D. (2003). Confidence intervals are a more useful complement to nonsignificant tests than are power calculations. *Behavioral Ecology*, 14(3), 446–447. doi: 10.1093/beheco/14.3.446.

Coretta, S., Casillas, J. V., Roessig, S., et al. (2023). Multidimensional signals and analytic flexibility: Estimating degrees of freedom in human-speech analyses. *Advances in Methods and Practices in Psychological Science*, 6(3), 1–29. doi: 10.1177/25152459231162567.

Correll, J., Mellinger, C., McClelland, G. H., & Judd, C. M. (2020). Avoid Cohen's 'small', 'medium' and 'large' for power analysis. *Trends in Cognitive Sciences*, 24(3), 200–207. doi: 10.1016/j.tics.2019.12.009.

Djimeu, E. W. & Houndolo, D.-G. (2016). Power calculation for causal inference in social science: Sample size and minimum detectable effect determination. *Journal of Development Effectiveness*, 8(4), 508–527. doi: 10.1080/19439342.2016.1244555.

Edwards, W., Lindman, H., & Savage, L. J. (1963). Bayesian statistical inference for psychological research. *Psychological Review*, 70(3), 193–242. doi: doi.org/10.1037/h0044139.

Ellison, A. M. (1996). An introduction to Bayesian inference for ecological research and environmental decision-making. *Ecological Applications*, 6(4), 1036–1046.

Ellison, A. M. & Dennis, B. (2010). Paths to statistical fluency for ecologists. *Frontiers in Ecology and the Environment*, 8(7), 362–370. doi: 10.1890/080209.

Faul, F., Erdfelder, E., Buchner, A., & Lang, A.-G. (2009). Statistical power analyses using G*Power 3.1: Tests for correlation and regression analyses. *Behavior Research Methods*, 41(4), 1149–1160. doi: 10.3758/BRM.41.4.1149.

Faul, F., Erdfelder, E., Lang, A. G., & Buchner, A. (2007). G*Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior Research Methods*, 39(2), 175–191. doi: 10.3758/BF03193146.

Fu, C., Lu, K., Li, Q., et al. (2023). Physics-based modelling for on-line condition monitoring of a marine engine system. *Journal of Marine Science and Engineering*, 11(6), 1241. doi: 10.3390/jmse11061241.

Gaskill, B. N. & Garner, J. P. (2020). Power to the people: Power, negative results and sample size. *Journal of the American Association for Laboratory Animal Science*, 9(1), 9–16. doi: 10.30802/AALAS-JAALAS-19-000042.

Gelman, A. (2018). Don't calculate post-hoc power using observed estimate of effect size. https://statmodeling.stat.columbia.edu/2018/09/24/dont-calculate-post-hoc-power-using-observed-estimate-effect-size/. Accessed 7 January 2024.

Gelman, A. (2019). Don't calculate post-hoc power using observed estimate of effect size. *Annals of Surgery*, 269(1), e9–e10. doi: 10.1097/SLA.0000000000002908.

Gelman, A. & Carlin, J. (2014). Beyond power calculations: Assessing type S (sign) and type M (magnitude) errors. *Perspectives on Psychological Science*, 9(6), 641–651. doi: 10.1177/1745691614551642.

Gelman, A., Carlin, J., Stern, H., et al. (2013). *Bayesian Data Analysis*. New York, NY, USA: Chapman & Hall/CRC.

Gigerenzer, G., Krauss, S., & Vitouch, O. (2004). The null ritual: What you always wanted to know about significance testing but were afraid to ask. In D. Kaplan (Ed.), *The SAGE Handbook of Quantitative Methodology for the Social Sciences* (pp. 391–408). Thousand Oaks, CA, USA: Sage Publications.

Goodman, S. N. (1992). A comment on replication, *P*-values and evidence. *Statistics in Medicine*, 11(7), 875–879. doi: 10.1002/sim.4780110705.

Goodman, S. N. & Berlin, J. A. (1994). The use of predicted confidence intervals when planning experiments and the misuse of power when interpreting results. *Annals of Internal Medicine*, 121(3), 200–206. doi: 10.7326/0003-4819-121-3-199408010-00008.

Gould, E., Fraser, H. S., Parker, T. H., et al. (2023). Same data, different analysts: Variation in effect sizes due to analytical decisions in ecology and evolutionary biology. *EcoEvoRxiv*, 2023, X2GG62. doi: 10.32942/X2GG62.

Greenland, S., Senn, S. J., Rothman, K. J., et al. (2016). Statistical tests, *P* values, confidence intervals, and power: A guide to misinterpretations. *European Journal of Epidemiology*, 31(4), 337–350. doi: 10.1007/s10654-016-0149-3.

Griffith, K. N. (2021). Amplifying the noise: The dangers of post hoc power analyses. *Journal of Surgical Research*, 259, A9–A11. doi: 10.1016/j.jss.2019.09.075.

Hedges, L. V. (1984). Estimation of effect size under non-random sampling: The effects of censoring studies yielding statistically insignificant mean differences. *Journal of Educational Statistics*, 9(1), 61–85. doi: 10.3102/10769986009001061.

Helminen, O. & Reito, A. (2019a). Comment on "Post-hoc power: A surgeons's first assistant in interpreting 'negative' studies" and "A proposal to mitigate the consequences of type 2 error in surgical science." *Annals of Surgery*, 270(6), e77–e78. doi: 10.1097/SLA.0000000000003288.

Helminen, O. & Reito, A. (2019b). Letter to Editor: A proposal to mitigate the consequences of type 2 error in surgical science. *Annals of Surgery*, 269(1), e10. doi: 10.1097/SLA.0000000000002907.

Hoening, J. M. & Heisey, D. M. (2001). The abuse of power: The pervasive fallacy of power calulatons for data analysis. *The American Statistician*, 55(1), 19–24. doi: 10.1198/000313001300339897.

Huntington-Klein, N., Arenas, A., Beam, E., et al. (2021). The influence of hidden researcher decisions in applied microeconomics. *Economic Inquiry*, 59(3), 944–960. doi: 10.1111/ecin.12992.

Jobst, L. J., Bader, M., & Moshagen, M. (2023). A tutorial on assessing statistical power and determining sample size for structural equation models. *Psychological Methods*, 28(1), 207–221. doi: 10.1037/met0000423.

Kapp, N., Borgatta, L., Ellis, S. C., & Stubblefield, P. (2006). Simultaneous very low dose mifepristone and vaginal misoprostol for medical abortion. *Contraception*, 73(5), 525–527. doi: 10.1016/j.contraception.2005.12.011.

Karsa, A., Punwani, S., & Shmueli, K. (2020). An optimized and highly repeatable MRI acquisition and processing pipeline for quantitative susceptibility mapping in the head-and-neck region. *Magnetic Resonance in Medicine*, 84(6), 3206–3222. doi: 10.1002/mrm.28377.

Kimmel, K., Avolio, M. L., & Ferraro, P. J. (2023). Empirical evidence of widespread exaggeration bias and selective reporting in ecology. *Nature Ecology & Evolution*, 7(9), 1525–1536. doi: 10.1038/s41559-023-02144-3.

Kroes, A. D. A. & Finley, J. R. (2023). Demystifying omega squared: Practical guidance for effect size in common analysis of variance designs. *Psychological Methods*, in press. doi: 10.1037/met0000581.

Kruschke, J. K. & Liddell, T. M. (2018). The Bayesian new statistics: Hypothesis testing, estimation, meta-analysis, and power analysis from a Bayesian perspective. *Psychonomic Bulletin & Review*, 25, 178–206. doi: 10.3758/s13423-016-1221-4.

Lane, D. M. & Dunlap, W. P. (1978). Estimating effect size: Bias resulting from the significance criterion in editorial decisions. *British Journal of Mathematical and Statistical Psychology*, 31(2), 107–112. doi: 10.1111/j.2044-8317.1978.tb00578.x.

LeMaire., S. A. (2021). A post hoc discussion about post hoc power: Divergent viewpoints on controversial methodology. *Journal of Surgical Research*, 259, A1–A2. doi: 10.1016/j.jss.2021.01.002.

Lenth, R. V. (2007). Statistical power calculations. *Journal of Animal Science*, 85(13 Suppl), E24–E29. doi: 10.2527/jas.2006-449.

Lerner, B., Boose, E., Brand, O., et al. (2022). Making provenance work for you. *R Journal*, 14(4), 141–159. doi: 10.32614/RJ-2023-003.

Loos, J., Hanspach, J., von Wehrden, H., et al. (2015). Developing robust field survey protocols in landscape ecology: A case study on birds, plants and butterflies. *Biodiversity and Conservation*, 24(1), 33–46. doi: 10.1007/s10531-014-0786-3.

Lutermann, H., Butler, K. B., & Bennett, N. C. (2022). Parasite-mediated mate preferences in a cooperatively breeding rodent. *Frontiers in Ecology and Evolution*, 10, 838076. doi: 10.3389/fevo.2022.838076.

Mair, M. M., Kattwinkel, M., Jakoby, O., & Hartig, F. (2020). The Minimum Detectable Difference (MDD) concept for establishing trust in nonsignificant results: A critical review. *Environmental Toxicology and Chemistry*, 39(11), 2109–2123. doi: 10.1002/etc.4847.

Munafò, M. R., Nosek, B. A., Bishop, D. V. M., et al. (2017). A manifesto for reproducible science. *Nature Human Behaviour*, 1(1), 0021. doi: 10.1038/s41562-016-0021.

Murphy, J. M., Sexton, D. M. H., Barnett, D. N., et al. (2004). Quantification of modelling uncertainties in a large ensemble of climate change simulations. *Nature*, 430(7001), 768–772. doi: 10.1038/nature02771.

Neyman, J. & Pearson, E. S. (1933). The testing of statistical hypotheses in relation to probabilities *a priori*. *Mathematical Proceedings of the Cambridge Philosophical Society*, 33(4), 492–510. doi: 10.1017/S030500410001152X.

NIH (2022). G.500—PHS human subjects and clinical trials information. https://grants.nih.gov/grants/how-to-apply-application-guide/forms-h/general/g.500-phs-human-subjects-and-clinical-trials-information.htm. Accessed 7 January 2024.

Nosek, B. A., Hardwicke, T. E., Moshontz, H., et al. (2022). Replicability, robustness, and reproducibility in psychological science. *Annual Review of Psychology*, 73, 719–748. doi: 10.1146/annurev-psych-020821-114157.

Nuzzo, R. L. (2018). Post hoc power. *PM&R (Physical Medicine and Rehabilitation)*, 13(4), 422–424. doi: 10.1002/pmrj.12476.

Park, M. & Snyder, S. A. (2020). Statistical profiling for identifying transformation products in an engineered treatment process. *Chemosphere*, 251, 126401. doi: 10.1016/j.chemosphere.2020.126401.

Percie du Sert, N., Ahluwalia, A., Alam, S., et al. (2020). Reporting animal research: Explanation and elaboration for the ARRIVE guidelines 2.0. *PLoS Biology*, 18(7), e3000411. doi: 10.1371/journal.pbio.3000411.

Quach, N. E., Yang, K., Chen, R., et al. (2022). Post-hoc power analysis: A conceptually valid approach for power based on observed study data. *General Psychiatry*, 35(4), e100764. doi: 10.1136/gpsych-2022-100764.

Ribeiro, J. D., Harris, L. M., Linthicum, K. P., & Bryen, C. P. (2020). Do suicidal behaviors increase the capability for suicide? A longitudinal pretest–posttest study of more than 1,000 high-risk individuals. *Clinical Psychological Science*, 8(5), 890–904. doi: 10.1177/2167702620921511.

Rotenberry, J. T. & Wiens, J. A. (1985). Statistical power analysis and community-wide patterns. *The American Naturalist*, 125(1), 164–168. doi: 10.1086/284335.

Ruxton, G. D. & Colegrave, N. (2023). Commentary: Parasite-mediated mate preferences in a cooperatively breeding rodent. *Frontiers in Ecology and Evolution*, 10, 1108008. doi: 10.3389/fevo.2022.1108008.

Sánchez-Meca, J., Marín-Martínez, F., & Chacón-Moscoso, S. (2003). Effect-size indices for dichotomized outcomes in meta-analysis. *Psychological Methods*, 8(4), 448–467. doi: 10.1037/1082-989X.8.4.448.

Schweinsberg, M., Feldman, M., Staub, N., et al. (2021). Same data, different conclusions: Radical dispersion in empirical results when independent analysts operationalize and test the same hypothesis. *Organizational Behavior and Human Decision Processes*, 165, 228–249. doi: 10.1016/j.obhdp.2021.02.003.

Silberzahn, R., Uhlmann, E. L., Martin, D. P., et al. (2018). Many analysts, one data set: Making transparent how variations in analytic choices affect results. *Advances in Methods and Practices in Psychological Science*, 1(3), 337–356. doi: 10.1177/2515245917747646.

Sterling, T. D. (1959). Publication decisions and ther possible effects on inferences drawn from tests of significance—or vice versa. *Journal of the American Statistical Association*, 54(1), 30–34. doi: 10.1080/01621459.1959.10501497.

Stommes, D., Aronow, P. M., & Sävje, F. (2023). On the reliability of published findings using the regression discontinuity design in political science. *Research & Politics*, 10(2), 1–12. doi: 10.1177/20531680231166457.

Strong, Jr., D. R., Simberloff, D., Abele, L. G., & Thistle, A. B., Eds. (1984). *Ecological Communities: Conceptual Issues and the Evidence*. Princeton, NJ, USA: Princeton University Press.

Timm, A., Gelman, A., Carlin, J., & Weterings, M. (2023). retrodesign: Tools for Type S (Sign) and Type M (Magnitude) errors. https://cran.r-project.org/package=retrodesign.

Ullman, D. S., O'Donnell, J., Kohut, J., et al. (2006). Trajectory prediction using HF radar surface currents: Monte Carlo simulations of prediction uncertainties. *Journal of Geophysical Research—Oceans*, 111(C12), C12005. doi: 10.1029/2006JC003715.

van der Hoeven, N. (2008). Calculation of the minimum significant difference at the NOEC using a non-parametric test. *Ecotoxicology and Environmental Safety*, 70(1), 61–66. doi: 10.1016/j.ecoenv.2007.06.010.

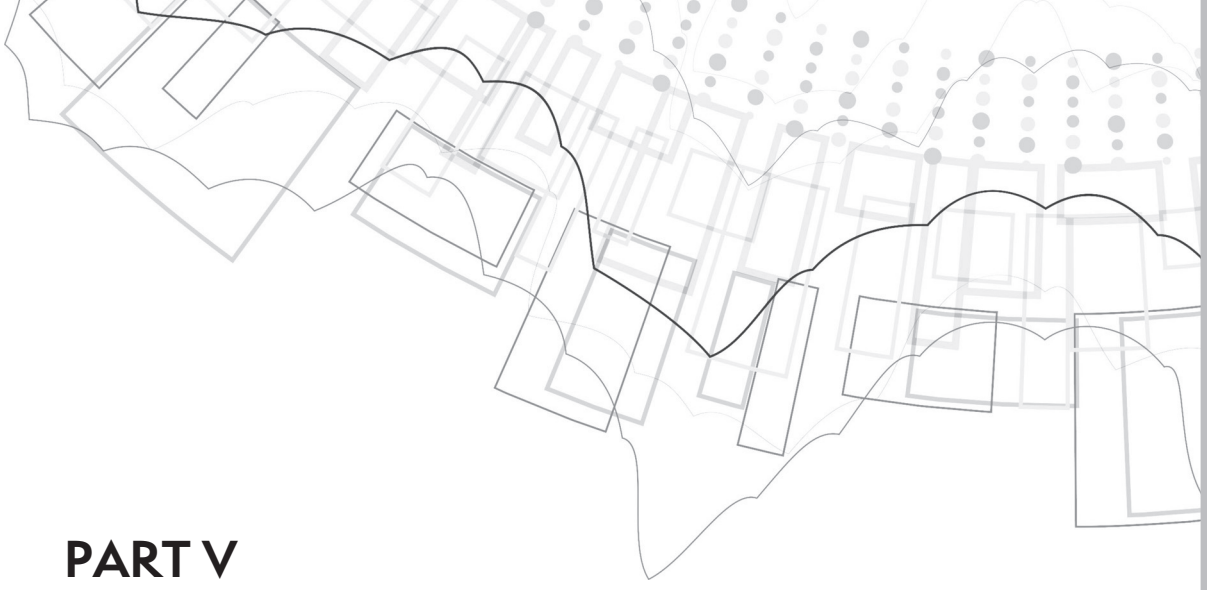
Walker, H. M. & Lev, J. (1953). *Statistical Inference*. New York, NY, USA: Holt, Rinehart & Winston.

Wasserstein, R. L. & Lazar, N. A. (2016). The ASA statement on *p*-values: Context, process, and purpose. *The American Statistician*, 70(2), 129–133. doi: 10.1080/00031305.2016.1154108.

Yang, Y., Sánchez-Tójar, A., O'Dea, R. E., et al. (2023). Publication bias impacts on effect size, statistical power, and magnitude (Type M) and sign (Type S) errors in ecology and evolutionary biology. *BMC Biology*, 21, 71. doi: 10.1186/s12915-022-01485-y.

Yasdiman, M. B., Townsend, E., & Blackie, L. E. R. (2022). Examining the protective influence of posttraumatic growth on interpersonal suicide risk factors in a 6-week longitudinal study. *Frontiers in Psychology*, 13, 998836. doi: 10.3389/fpsyg.2022.998836.

Yildiz, M., Shi, R., & Kara, M. (2022). How to improve the design of experimental studies in computing education: Evidence from the international assessments. *Education and Information Technologies*, 27(4), 5075–5102. doi: 10.1007/s10639-021-10800-z.



PART V

In posterum: Simulations for new methods

Statistical methods are not static. Rather, they are constantly evolving and changing as new ideas and hypotheses arise, new tools and models are developed, and new mathematical and computational methods are brought to bear on classical and cutting-edge problems. In the final three chapters, we look at some of these new statistical methods and models, and illustrate how data simulation is used to test and validate them.

Chapter 10 begins with a discussion of classical meta-analysis, which is often used to synthesise existing research findings before embarking on new research. As data privacy and security issues have come to the fore, federated data analysis has begun to supplant meta-analysis as a method to synthesise multiple studies while securely preserving anonymity. Simulations illustrate the challenges and opportunities in meta-analysis and federated analysis that researchers are actively investigating as these tools continue to evolve. In Chapter 11, we make a turn from specific case-study-based simulations to using simulations to demonstrate the generality of new methods. Only a few fields have yet defined protocols for testing new methods using benchmark datasets and simulations, but such testing and validation increasingly is seen as de rigueur for publication of new statistical models and methods in the most impactful journals. In this context, we also discuss how new methods should be described in multiple ways: verbally, mathematically, and algorithmically to, respectively, enhance intuitive understanding, formal specification, and implementation of new methods. Finally, Chapter 12 returns to where we started this book: using simulations for planning large and very complex research programs that are addressing some of the most difficult questions scientists have posed to date.

10

Combining studies: Meta-analysis and federated analysis

In the previous chapters of this book, we have focused on the standard or classical approach to the design of a research study and the analysis of the resulting data: an individual investigator or research group designs the study, collects the data, and has all the data available for its analysis. Research really is a multi-generational endeavour, however, and we are always extending the work of those who have gone before us.[1] In fact, a substantial part of science is meta-analytic, reanalysing data collected by others and using the results to identify robust underlying principles and extend generalisable scientific theories. This chapter focuses on two classes of methods we use to analyse and synthesise multiple datasets that were collected by others: meta-analysis and federated analysis. We use simulations to illustrate the promises and challenges of these methods.

What you will learn from this chapter

- Five different types of methods for analysing multiple datasets, and the kinds and quality of data used for each of them.
- Many of the same problems that bedevil standard analyses—missing values, biased sampling, correlated predictors, and lack of model fit—also arise in meta-analysis and federated analysis.

10.1 Whence the data?

If our predecessors were working and publishing before data were collected or stored in digital form, we rarely have their raw data available for re-analysis. More often than not, we reconstruct historical datasets by digitising published summary tables or graphs (e.g., Pick et al. 2019). Canonical examples include the data Galileo collected to develop his theories of motion and mechanics (Galilei 1960; Abbott 1998) and the mortality statistics that Florence Nightingale collected to argue for improved sanitary conditions for treating wounded soldiers (Small 1999; Friendly 2008).

On the other hand, most data collected since 2000—and some collected decades before—are available digitally. Even in this era of so-called Big Data (Anderson 2008), however, accessing digital data still presents many challenges. Some datasets are available to anyone with an Internet connection, others are shared only among colleagues and collaborating

[1] To quote Isaac Newton, and Bernard de Chartres long before him, we are "standing on the shoulders of giants." See footnote 3 in Chapter 3.

research teams, and still others are unavailable to anyone other than the individual who collected the data and created the dataset. At one extreme, many datasets collected by government agencies in the USA and Western Europe are freely available to everyone. At the other extreme, data collected on endangered species and human subjects are tightly controlled and guarded carefully.[2] Across the board, however, the granularity of the data varies. Sometimes only high-level summary statistics are available, and these may be little better than what we could get from digitising a published bar chart. Other datasets include only enough of the raw data necessary to reproduce a published analysis. Relatively few datasets include all the data collected by sensors before they were subject to any QA/QC procedures.[3]

10.2 From meta-analysis through federated analysis to complete analysis

We distinguish five different types of methods for analysing multiple datasets (Figure 10.1). At one extreme, a classical meta-analysis (e.g. Glass 1977; Koricheva et al. 2013; Schmid et al. 2021) is done only on published summary statistics—the sample mean, its variance, and the sample size—from which we can estimate a standardised effect size (e.g. Glass 1977). These estimates are known to be biased (see Figure 9.1 and discussion in §9.2.2 in Chapter 9), but even proposed bias-corrections (e.g. Hedges 1984) cannot account for results that were never published because of self-censorship, decisions made by funders,[4] or decisions made by publishers.[5] Self-censorship, funding biases, and publication biases reduce the amount of data available for meta-analysis and introduce biases towards larger effect sizes.

At the other extreme, a complete analysis of the all available data from multiple studies, whether published or not, likely would be as unbiased as possible. Although such a complete analysis would be analogous to an unbiased meta-analysis, it also would be identical

[2] Privacy laws prevent open sharing of data collected about patients by nurses and physicians. Even when these data have been anonymised, however, cross-referencing them with social media posts, census or tax data (which are openly available in some countries), or other publicly-available information may enable others to identify the patient or research subject (Murdoch 2021).

[3] For example, the US National Aeronautics and Space Administration (NASA) distinguishes five levels of data that reflect how much processing has been done to the data since they were collected. Level 0 data are truly raw data at full instrument resolution. Often, Level 0 digital data are signals recorded in (milli)voltages that have to be processed and converted into the numerical values needed by researchers. Level 1 data add time-stamps, georeferencing, and calibration coefficients (Level 1A), are processed to sensor units (Level 1B), and have additional variables to describe any spectral frequencies that were either collected directly by the sensors or computed from other variables collected by the sensors (Level 1C). Level 2 data include geophysical variables that have been derived (computed) from georeferenced sensor data (Level 2A) that are further processed to sensor units (Level 2B). Level 3 data map Level 1 and Level 2 data onto uniform spatiotemporal grids; any missing values are imputed and data gaps are filled (see §6.1 in Chapter 6). Sometimes additional periodic summaries (e.g., weekly, monthly, annual) of Level 2 data also are computed; these are referred to as Level 3A data. Finally, Level 4 data include results of analyses of one or more lower-level datasets and output from models. See NASA (2021) for more details.

[4] Funding bias occurs as a consequence of self-censorship when results do not fit the expectations of the individual, organization, or agency that funded the research. Indeed, one of us was scolded by a colleague working in conservation biology for publishing an evolutionary analysis whose results suggested flexible climate niches in mammals, as they could have been used "inappropriately" by anti-conservationists. Pre-registration of clinical trials and other research projects can ameliorate funding biases (Nosek et al. 2018, but see Rubin 2020).

[5] Publication bias is better understood and typically is linked to journal (and researcher) preferences for positive findings, significant effects, and strong effects (Chalmers 1990; Easterbrook et al. 1991; Egger & Smith 1995; Andrews & Kasy 2019; van Aert et al. 2019; Brodeur et al. 2020). Ironically, there seems to be no publication bias in studies reporting on publication bias (Dubben & Beck-Bornholdt 2005). We regard the oft-mentioned GIGO (garbage-in, garbage-out) problem (Egger et al. 2001; Smith & Hing 2011; Andrade 2020; Ferguson et al. 2020) as largely curable using stringent inclusion criteria and quality checklists (e.g. Borenstein 2009; Koricheva et al. 2013; Mupepele et al. 2016).

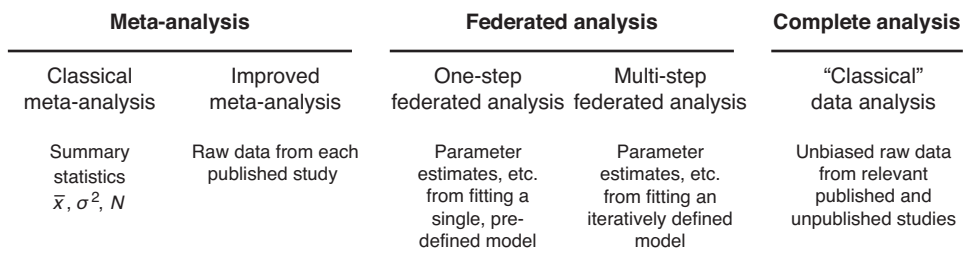


Figure 10.1 Five different types of methods for analysing multiple datasets and the data used for each of them. These types of methods range from classical meta-analysis, for which the analyst has only published summary statistics, to a complete data analysis, for which the analyst has all available data that can be used in a standard, classical way.

methodologically to any standard or classical data analysis for which we had access to all the data needed to fit the model(s) of interest.

We consider three steps between a classical meta-analysis and a complete analysis (Figure 10.1). First, we can improve on a classical meta-analysis if we have access to the raw data from each study included in the meta-analysis. Although access to these raw data cannot reduce or eliminate the (unknown) effects of funder or publication bias, it does ensure that we can do the same analysis for all the datasets. This can help to circumvent the problem that the authors of the original studies may have discarded some predictor variables as irrelevant; when we are analysing the data across many studies it might be useful to retain those predictors. Such improved meta-analyses are referred to in the literature as "individual participant-level meta-analysis" or, in clinical settings, "individual patient-level meta-analysis" (e.g. Riley et al. 2010; Valgimigli et al. 2021).[6]

Second, we can use a "federated analysis."[7] Federated analysis aims to estimate a common model for multiple datasets without the analyst having access to all of the individual datasets. In a federated analysis, the model is distributed to participating partners, each of whom runs the model locally and returns estimates of model parameters to the meta-analyst (Jones et al. 2012; Doiron et al. 2013; Hallock et al. 2021).[8] Federated analysis addresses the privacy concerns intrinsic to statistical analyses of patient data, security concerns of economic data, or, indeed, any reason—good or bad—that individuals or research groups have and use to not share their raw data.

There are two kinds of federated analysis. In a one-step federated analysis, the meta-analyst defines a common model and writes a script for the analysis using an agreed-upon language or platform (e.g., R). The script is then sent to each of the participating research groups. Each group runs the script locally on their dataset and then returns only the results of the analysis to the meta-analyst. Such results normally include parameter estimates, their covariances, autocorrelation statistics, ranges of predictor variables, or other summary

[6] We shall not use the term "integrative data analysis" proposed by Curran & Hussong (2009), as it is too unspecific. Still, that is what it may be called in the field of psychology.

[7] We are indebted to Daniela Zöller at the University Clinic Freiburg for talking us through the different current approaches to federated analysis and the subtleties of the gradient from meta-analysis to complete analysis illustrated in Figure 10.1.

[8] Federated analysis also requires data encryption standards and protocols. The software engineering required for such encryption is known as "federated learning" (e.g. Wei et al. 2020; Froelicher et al. 2021). Although we recognise the importance of federated learning and associated privacy concerns that arise with federated analysis, we are concerned here only with the mechanics and biases of federated statistical analyses.

statistics requested by the meta-analyst that are used to synthesise the results across the different studies.

In a multi-step federated analysis, defining the analysis is a collective activity. All the participating groups work together to determine the common model used to analyse their datasets. This collective activity helps to account for local idiosyncrasies in data collection and management, such as local differences in subject selection, methods, and measurements. Methods have been developed to ensure that the collective model definition and decision making do not lead to tailoring the overall analysis to any particular dataset or subset. These include a limit to the number of iterative changes or improvements (often ≤ 6) to the models or only bilateral, distributed interactions among groups (Li et al. 2020).

10.2.1 Difficulties with combining studies

There are a number of difficulties that arise in meta-analysis and federated analysis. First, the datasets subject to meta-analyses typically are small. Even though any individual study could have had a large sample size, the number of observations used in a meta-analysis is the number of studies, not the total number of observations summed over all the studies included in the meta-analysis. Thus, the statistical model used and fitted in a meta-analysis generally is very simple. It often includes only a single predictor or at most two predictors and their interaction (e.g., Borenstein 2009; Koricheva et al. 2013). Actual cause-and-effect relationships tend to be much more complex, however, and it seems unfortunate to draw general conclusions from such simple models. For example, we might expect the effect of a medication to be dependent on age, gender, ethnicity, and their two- and three-way interactions. Yet, fitting such a model will yield robust results only with hundreds of observations—that is, studies—included in the meta-analysis. Rarely do we have that many replicated studies. Further, any individual study likely used some kind of model selection, leading either to biased estimates or to its reduced utility in a subsequent meta-analysis. Last, we would expect that only statistically significant studies would have been published and available at all for meta-analysis.

Most of the problems with a classical meta-analysis are resolved if each individual research group runs a prescribed model (plus any auxiliary analyses) in a one-step federated analysis. Others, however, may arise. For example, publication bias is not a problem with federated analysis, but it could be replaced by a "network-of-chums" effect, in which only an in-group of investigators are contacted to take part in the federated activity. The prescribed model can be more complex than would be usable in a meta-analysis, but the common model developed by the meta-analyst or core group may be the wrong model, leading to pooling of incorrect or, at best, sub-optimal local models. Even if the prescribed model is correct, however, the estimates returned may be unstable relative to what we would expect from a mixed-effect model of the complete dataset. This result occurs because the regularising effect of the hyper-distribution is not available at the local level. That is, in contrast to a mixed-effect model, where $Y = Xb + Zu + \varepsilon$, the one-step federated analysis will have only estimates of b' reported from local studies of the model $Y' = X'b' + \varepsilon'$. The meta-analyst can derive only the conditional estimate $u|b'$ of the full model.

Multi-step federated analysis goes a long way towards resolving these last two problems. If the different research groups work together to define the model through iterative improvement, the federated analysis will yield estimates that are identical to those returned by a (mixed-model) GLM fit to the complete dataset (Jones et al. 2012). When local analysts negotiate and jointly decide on the common model structure, the incomplete picture presented by any local group can be filled in (somewhat optimistically called "swarm learning" by Warnat-Herresthal et al. 2021).

10.2.2 Our aims for the next sections

In the next five sections, we explore in detail and with simulations each of the five different types of methods for analysing multiple datasets (Figure 10.1). The first four of these methods—classical and improved meta-analysis, and one- and multi-step federated analysis—use some subset of the full dataset. The reference analysis, which we present in the last section, is the analysis of the full dataset, with access to all predictors for each case, access to unpublished data (to avoid self-censorship, and funder and publication biases), and information on the representative sampling underlying the data.

In addition to discussing the details of these five methods, we address a few additional questions that often bedevil meta-analysts and data synthesisers.

Meta-analysis What should we do with missing effect sizes?

One-step federated analysis Should we use only studies that can be fit using the prescribed models, or use all studies with the best model they can fit (i.e., a model using only those prescribed predictors that are available for a local study)?

Complete data analysis What is the effect of not publishing when the target effect size is not statistically significant?

DISCLAIMER

Finally, when we do meta-analyses, federated analyses, or complete analyses ourselves, we reiterate that ***stepwise model selection should not be used*!** Stepwise model selection leads to biased estimates and uninterpretable p-values (see §6.1 in Chapter 6). But for exactly those reasons, and because stepwise model selection still is a common practice in many fields of science, we use it in our simulations deliberately—to create bias! That is, we use it as a simple method to ensure that the analysis of each local study will yield a slightly different result, despite its using the same underlying data-generating model. Thus, we intentionally use stepwise model selection for its otherwise unintended, and statistically negative, side-effects.

10.3 Meta-analysis

We start our walk through the different types of methods for analysing multiple datasets with a classical meta-analysis. For a classical meta-analysis, we need summary statistics, their variances, and the sample sizes. To obtain these, we simulated a large dataset (the overall population), from which we simulated a number of samples (in this case, 21), each of which is assigned to an individual study. Each study analyses its "own" dataset, and the effect sizes from each study are then subject to a meta-analysis. Because we simulated the complete dataset (population) and know the effect sizes (in this case, regression coefficients) of the simulated predictors, the meta-analysis ideally should recover the effect sizes.

Not so fast. In the real world, different studies that are combined in meta-analyses may have had incompletely overlapping sets of predictors. For example, different hospitals testing the same medication or treatment protocol may have different laboratory procedures, recording practices, or confidentiality guidelines. If we combine data from these different hospitals in a meta-analysis to determine the effect of a particular medical treatment, we will end up with a composite dataset that includes some common predictors and some different ones. Similar situations arise in other settings, too. For example, every research group studying the effects of feeding by insects on crop performance is likely to collect data on the population density of the insects and the plants, but different groups also are likely to collect additional predictor variables that reflect local contexts (e.g., rainfall, temperature, soil composition, presence of predators, etc.). Even formal networks of distributed studies with

Code Box 10.1: Simulate a large dataset describing the population of interest

```
# Load required libraries
library(mvtnorm)

# Simulate the dataset
set.seed(123)
S <- matrix(rnorm(20*20, 0, 0.21), 20, 20)
Sigma <- S %*% t(S) # To ensure positive definite covariance matrix
X <- rmvnorm(20000, mean=rep(0, 20), sigma=Sigma)
X <- scale(X) # Scale to get standardised regression coefficients as effect size
range(cor(X)[lower.tri(cor(X))]) # Check the range of the correlations

| [1] -0.5187035  0.5266495

slopes <- runif(20, -2, 2)
Y <- X %*% slopes + rnorm(nrow(X), 0, 5)
fullData <- cbind.data.frame(Y, X)
colnames(fullData) <- c("Y", paste0("X", 1:20))
```

defined designs and experimental protocols (Fraser et al. 2013) can end up with partially overlapping sets of predictor variables.[9]

Our simulation reflects these real-world challenges. Specifically, we are interested in whether estimates of effect sizes derived from meta-analysis are biased if each study is allowed to use a set of common predictors and a set of partially overlapping additional predictors. We go through the four steps of this simulation in detail for a single dataset (Code Boxes 10.1–10.4) and then use 100 simulations to test for bias (Code Box 10.5).

STEP 1: SIMULATE A LARGE DATASET

We start by simulating a dataset of size $N = 20{,}000$ cases with 20 predictors (X) drawn from a multivariate normal distribution and standardised to $\bar{X}_i = 0$ and $\sigma = 1$, so that the fitted models will yield standardised regression coefficients. The simulated standardised predictors are only modestly correlated ($\bar{r} = -0.02 \pm 0.21$ [SD], $r_{\min} = -0.52$, $r_{\max} = 0.53$). Then, the response variable Y is simulated by drawing values for the 20 regression coefficients (= slopes) from $\mathcal{U}(-2, 2)$, multiplying them by the standardised predictor values, and adding Gaussian noise ($\mathcal{N}(0, 5)$). The output is a matrix of simulated data with 20,000 rows of observations $\times$ 21 columns that include the response variable Y and the 20 predictor variables

[9] For example, the Nutrient Network, a global research cooperative of more than 130 sites around the world, uses a standard experimental protocol to address three questions about the relationship between the local environment, species diversity, and grassland productivity (Borer et al. 2014). Because the experimental design and core measurements are carefully prescribed, there is a common set of predictor variables collected at each site by the local research group. However, the formal protocol also encourages each local group to consider additional sampling that might include pitfall trapping for insects, live-traps for mammals, chemical composition of plant tissues or soils, etc. (Borer et al. 2014). Because these measurements are optional, the end result is a set of partially overlapping predictor variables over the entire network, and some cross-site analyses can only use data from a subset of sites (e.g., Orrock et al. 2015).

Code Box 10.2: Distribute observations from the large dataset from Code Box 10.1 into 21 individual studies

```
# This code uses the "fullData" object created in Code Box 10.1

# Set the number of studies and the sample sizes in each of them
 Nstudies <- 21
 Ns <- sample(10:100, Nstudies) * 10 # between 100 and 1000
 index <- sample(20000, sum(Ns)) # pick data from full simulation

 studyList <- list() # Empty object in which to put the output
 for (i in 1:Nstudies){
 studyList[[i]] <- fullData[index[1:Ns[i]], sort(c(1, 2, 3, sample(3:21, 8)))]
     # pick 10 predictors, 8 randomly
  index <- index[-c(1:Ns[i])] # Remove data already assigned
 }
```

(the `fullData` object in Code Box 10.1). We consider this large matrix to be a complete description of our population of interest.

STEP 2: DISTRIBUTE THE DATA OVER MULTIPLE STUDIES

To simulate 21 individual (local) studies with sample sizes between 100 and 1,000 subjects, we first define samples sizes $N_i : i \in \{1, 2, \ldots, 21\}$ for each study. Then we randomly pick N_i subjects per study from the 20,000 cases generated in Code Box 10.1 (these are equivalent to row numbers in `fullData`). For each of the 21 studies, we always keep the two predictors X_1 and X_2, but then randomly select another eight predictors from the 18 remaining $X_c = \{X_3, X_4, \ldots, X_{20}\}$. This yields 21 datasets with non-overlapping subjects and partly overlapping predictors (Code Box 10.2).

STEP 3: ANALYSIS OF INDIVIDUAL STUDIES AND COMPUTATION OF EFFECT SIZES

We run a multiple regression with backward model selection using BIC for each individual "study."[10] We then use the estimates of the model coefficients as the effect sizes. This is appropriate because the estimates are standardised regression coefficients and hence directly comparable to one another within and among studies (Code Box 10.3).

STEP 4: META-ANALYSIS OF THE INDIVIDUAL STUDIES

We do the simplest meta-analysis: we compute the mean effect size (i.e., of the standardised regression coefficients) for each predictor X_i across all studies. To do this, we use the weighted mean, where the standardised effect size of each study is weighted by its sample size (Code Box 10.4).[11]

[10] As we noted in the Disclaimer at the end of §10.2, above, we are well aware that such model simplification introduces bias (§6.1 in Chapter 6), but in many areas of science this is still the default approach and we want our simulations to show the consequences of such improper practice.

[11] This weight is appropriate because $1/\sigma^2$ of an estimate scales linearly with N, and σ^2 is the optimal weight when computing best linear unbiased estimates (BLUE) for a weighted regression (Aitken 1936).

Code Box 10.3: Analyses of each of the 21 studies

```
# This code uses the "studyList" object from Code Box 10.2

# Identical analysis of each study
 fmList <- lapply(studyList, \(x) step(lm(Y ~ ., data=x),
                  scope=list("lower"=Y~X1+X2, "upper"=Y ~ .),
                  k=log(nrow(x)), trace=F))

# Compute effect sizes = standardised regression coefficients
 effectSizes <- matrix(0, Nstudies, 20)
 colnames(effectSizes) <- colnames(fullData)[-1]
 for (i in 1:Nstudies){
   cs <- summary(fmList[[i]])$coefficients[-1, 1]
   effectSizes[i, names(cs)] <- cs
 }
```

THE FIRST EMERGENT QUESTION: WHAT SHOULD BE DONE WHEN SOME EFFECT SIZES OF INTEREST ARE MISSING FROM AN INDIVIDUAL STUDY?

We have to decide how to handle the non-reported effects resulting from the partially overlapping set of predictors among studies. These non-reported effects arise either because a predictor is not in the random selection for an individual study (Step 2; Code Box 10.2) or because it is dropped from the model during model simplification (Step 3; Code Box 10.3). We identify two ways to deal with these non-reported effects—setting them to NA or setting them equal to 0—and compare the effects of these decisions on the results of our meta-analysis.

When we set an unreported effect size to NA, we are asserting that we have no information on this predictor. Alternatively, when we set an unreported effect size to 0, we are asserting that the predictor would have been reported if it had been important at all.[12] After implementing each of these two scenarios, we proceed to estimate the effect sizes and compute the relative bias of the estimate of the regression coefficient $\hat{b}_i,\ i \in \{1, 2, \ldots, 20\}$ (the number of coefficients) as the scaled difference between its estimate $\hat{b}_i$ and its true value b_i^*: relative bias $= \dfrac{\hat{b}_i - b_i^*}{b_i^*}$ (Code Box 10.4).

MANY SIMULATION RUNS LATER

Running this simulation many times (Code Box 10.5) shows that the estimates of the regression coefficients determined by our meta-analysis are biased (Figure 10.2). The bias is more pronounced when we represented missing effect sizes as zeros, which leads to a consistent underestimation of the true effect sizes. On the other hand, dropping missing predictors (i.e., setting them equal to NA) from the meta-analysis yields slightly overestimated effect sizes, as we would have expected from the bias introduced by model selection in the individual studies (§6.1 in Chapter 6).

[12] If the predictor is missing due to Step 2, this represents a decision by the local study analyst to not have collected data on this predictor. Alternatively, the predictor could have been lost following model selection in Step 3. Either of these are plausible scenarios.

Code Box 10.4: Meta-analysis of the 21 simulated studies

```
# This code uses the "effectSizes" object from Code Box 10.3

# Handling un-reported effects two ways
# For unreported effects = NA
 effectSizesNAs <- effectSizes
 effectSizesNAs[effectSizesNAs==0] <- NA
 effectSizesNAs.wmeans <- apply(effectSizesNAs, 2,
                                \(x) weighted.mean(x, w = Ns,
                                na.rm = T)) # Sample-size weighted means
 relbiasNAs <- (effectSizesNAs.wmeans - slopes) / slopes # Relative bias

# For unreported effects = 0
 effectSizes.means <- apply(effectSizes, 2,
                            \(x) weighted.mean(x, w = Ns,
                            na.rm = T)) # weighted means
 relbiasAll <- (effectSizes.means - slopes)/ slopes # Relative bias
```

Code Box 10.5: Repeated simulations of the complete meta-analysis

```
# Load required libraries
 library(mvtnorm)

# Full simulation, includes all steps from Code Boxes 10.1 - 10.4
 Nstudies <- 21
 set.seed(123)
 effectSizes.list <- list()
 for (k in 1:100){ # 100 repetitions
  # Create the dataset and apportion observations among Nstudies
   S <- matrix(rnorm(20*20, 0, 0.21), 20, 20)
   Sigma <- S %*% t(S)
   X <- rmvnorm(20000, mean=rep(0, 20), sigma=Sigma)
   X <- scale(X)
   slopes <- runif(20, -2, 2)
   Y <- X %*% slopes + rnorm(nrow(X), 0, 5)
   fullData <- cbind.data.frame(Y, X)
   colnames(fullData) <- c("Y", paste0("X", 1:20))
   Ns <- sample(10:100, Nstudies) * 10 # Sample size per study is 100 - 1000
   index <- sample(20000, sum(Ns)) # Observations from full simulation
   studyList <- list()
    for (i in 1:Nstudies){
      studyList[[i]] <- fullData[index[1:Ns[i]], sort(c(1, 2, 3,
           sample(3:21, 8)))] # Select predictors: 2 fixed, 8 at random
      index <- index[-c(1:Ns[i])] # remove data already assigned
    }
  # Local analyses, always keeping X1 and X2
   fmList <- lapply(studyList, \(x) step(lm(Y ~ ., data=x),
```

```
                    scope=list("lower"=Y~X1+X2, "upper"=Y ~ .),
                    k=log(nrow(x)), trace=F))
  # Compute effect sizes = standardised regression coefficients
  effectSizes <- matrix(0, Nstudies, 20)
  colnames(effectSizes) <- colnames(fullData)[-1]
   for (i in 1:Nstudies){
     cs <- summary(fmList[[i]])$coefficients[-1, 1]
     effectSizes[i, names(cs)] <- cs
   }
  # meta-analysis
   effectSizesNAs <- effectSizes
   effectSizesNAs[effectSizesNAs==0] <- NA
   effectSizesNAs.wmeans <- apply(effectSizesNAs, 2,
                                  \(x) weighted.mean(x, w = Ns,
                                  na.rm = T))
   relbiasNAs <- (effectSizesNAs.wmeans  - slopes) / slopes
   effectSizes.means <- apply(effectSizes, 2,
                              \(x) weighted.mean(x, w = Ns,
                              na.rm=T))
   relbiasAll <- (effectSizes.means - slopes) / slopes
   effectSizes.list[[k]] <- list(effectSizes, relbiasNAs, relbiasAll)
 }
relbiasNAs100 <- sapply(effectSizes.list, \(x) x[[2]])
relbiasAll100 <- sapply(effectSizes.list, \(x) x[[3]])
```

INTERLUDE: MEDIAN RETROSPECTIVE POWER ANALYSIS

As a brief interlude, we link this result to Chapter 9 on post-hoc power.

Stanley et al. (2022) suggested that meta-analyses should be subjected to a form of retrospective power analysis, which they called median retrospective power ("MRP"), to determine if the findings are credible. Assuming the standard critical value $\alpha = 0.05$, MRP is computed as

$$\text{MRP} = 1 - \Phi\left(1.96 - \frac{|\text{UWLS}|}{SE_m}\right), \tag{10.1}$$

where UWLS is the unrestricted weighted least squares estimate of the slope of the standardised effect sizes (d_i/SE_i) of the i studies included in the meta-analysis regressed (with intercept set equal to zero) on their precision ($1/SE_i$); SE_m is the median of the standard errors of the i studies; and $\Phi(\cdot)$ is the cumulative standard normal probability evaluated at $1.96 - \frac{|\text{UWLS}|}{SE_m}$. Stanley et al. (2022) proposed that if MRP $\geq 50\%$, then the results of the meta-analysis should be considered to be credible. In contrast, if MRP $< 50\%$, any positive findings of the meta-analysis (i.e., statistically significant effects) should be interpreted cautiously or discounted completely.

Although we are not fond of this approach (see next paragraph), we can quickly compute this value for one of the present simulations (Code Box 10.6).

Because the computed value of MRP is larger than 50%, Stanley et al. (2022) would have a stronger belief in this meta-analysis than if the value of MRP were less than 50%. Since we know there is a weak true effect of X_1 (because we simulated the data: it is 0.12), we also know that the MRP ought to be higher than 50%. In Code Box 10.7, we repeat this procedure for X_2, which has a much stronger simulated effect (-1.9):

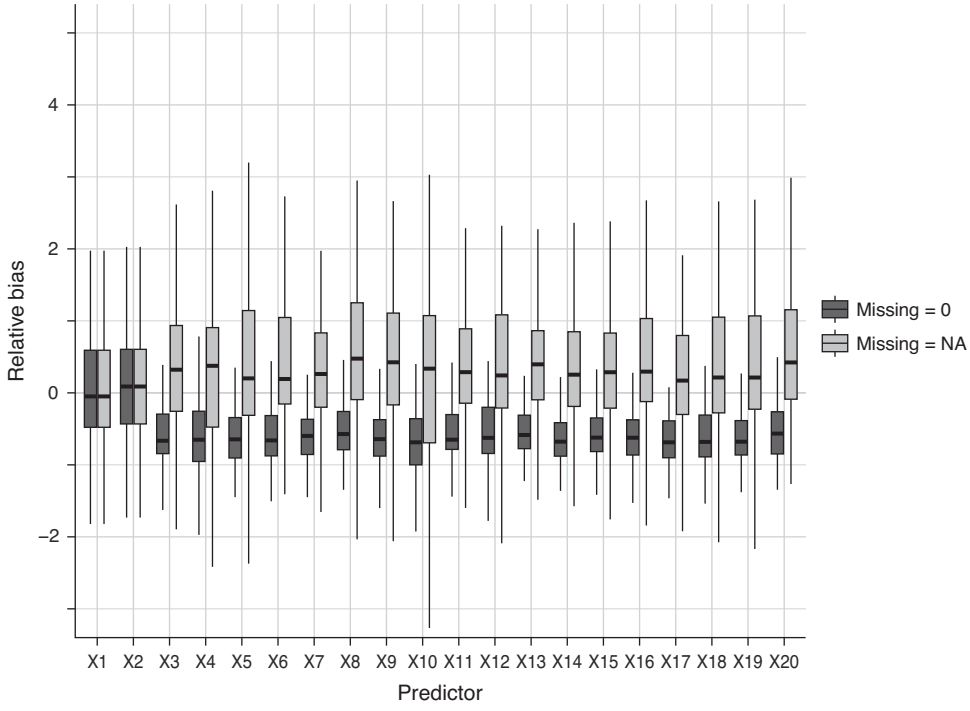


Figure 10.2 Biases in estimates of effect sizes determined from a meta-analysis of 21 simulated studies aimed at estimating the effects of 10 predictors: two fixed (X_1 and X_2) and eight chosen at random for each study from an overall set of 18 (X_3–X_{20}) (Code Box 10.5). To estimate the overall effects of each predictor, either we use only those reported to have an effect by each study and set the others for that study equal to NA (light grey boxes); or we set effect sizes for all non-reported predictors equal to zero (dark grey boxes). Note that in the first set of estimates, NAs could have resulted in two ways: (1) the predictor was not available in the dataset; or (2) the predictor was dropped during stepwise model selection.

Code Box 10.6: Median retrospective power of meta-analysis

```
# This code uses object fmList from Code Box 10.3

# focus here: coefficient estimate for X1:
 ds <- sapply(fmList, \(x) summary(x)$coefficients[2,1])
 ses <- sapply(fmList, \(x) summary(x)$coefficients[2,2])
 lm(ds/ses ~ I(1/ses) - 1)

| lm(formula = ds/ses ~ I(1/ses) - 1)
|
| Coefficients:
| I(1/ses)
|   0.7095

 UWLS <- unname(coef(lm(ds/ses ~ I(1/ses) - 1)))
 (1 - pnorm(1.96 - abs(UWLS)/median(ses))) *100

| [1] 52.49123
```

Code Box 10.7: Median retrospective power of meta-analysis

```
# This code uses object fmList from Code Box 10.3

# focus here: coefficient estimate for X2:
 ds <- sapply(fmList, \(x) summary(x)$coefficients[3,1])
 ses <- sapply(fmList, \(x) summary(x)$coefficients[3,2])
 UWLS <- unname(coef(lm(ds/ses ~ I(1/ses) - 1)))
 (1 - pnorm(1.96 - abs(UWLS)/median(ses))) *100

| [1] 99.99931
```

So the weak effect of X_1 is much less reliable than the strong effect of X_2, which at least is consistent with our simulations.[13]

However, MRP appears to suffer from many of the same problems associated with retrospective power analysis (Chapter 9). Stanley et al. (2022) extrapolate Equation 10.1 from the equation for the retrospective power of the ith study:

$$\text{Power}_i = 1 - \Phi(Z_\beta) = 1 - \Phi\left(1,96 - \frac{|\delta|}{\sigma_i}\right), \tag{10.2}$$

where δ is the effect size, σ_i is the standard deviation of the sampling distribution, and $\Phi(\cdot)$ is the cumulative normal probability. As with Faul et al.'s (2007; 2009) definition of post hoc power, δ and σ_i are considered to be population parameters that need to be estimated from the data, and standard ways of estimating these tend to overestimate actual effect sizes (Gelman & Carlin 2014, and see §9.3.3). Whether their UWLS from a finite number of studies is a reasonable estimator of Gelman and Carlin's (2014) d^{rep} (§9.3.4) remains an open question.

CONCLUSIONS

Assuming local studies are imperfect subsets of the overall population, and after employing model selection, we found a small overestimation of true effect sizes. Estimated effect sizes of the two target predictors X_1 and X_2 included in all 21 of the studies were practically unbiased, suggesting that meta-analyses would benefit from an "all-studies all-predictors" approach. Median retrospective power analysis appears to suffer from the same issues as retrospective power analyses applied to individual studies.

10.4 Individual participant-level meta-analysis

The previous section exposed one limitation of classical meta-analysis: its reliance solely on reported effect sizes, their variances, and their sample sizes. We could do better if we had access to the full dataset from each study in our meta-analysis. Instead of setting unreported effect sizes equal to zero or NA (Figure 10.2), we could combine the full datasets and analyse all of them together. In the medical literature, such an analysis is referred to as "individual patient-level (or participant-level) meta-analysis" (IPL meta-analysis for short;

[13] Repeating this analysis across many simulations (as in Code Box 10.5), we could evaluate sensitivity and specificity of this 50% threshold. Because of the fundamental problems attached to retrospective power analysis, we will not do that; it would only be a coding practice.

Code Box 10.8: Individual participant level meta-analysis of the 21 simulated studies

```
# This code uses the "fullData" object from Code Box 10.1 and the "Ns" object
    from Code Box 10.2

# Load required libraries
 library(lmerTest)

# Dataset
 allStudies <- fullData[1:sum(Ns), ]
 allStudies$ID <- rep(LETTERS[1:21], times=Ns)

# Mixed-effect model
 f <- as.formula(paste("Y~", paste0("X", 1:20, collapse=" + "), "+(1|ID)"))
 fmer <- lmer(f, data=allStudies)
 summary(fmer)$coefficients[-1, 1]

|         X1         X2         X3         X4         X5
| -0.6433734 -0.5697783 -0.3621669  0.1719555 -1.4209145
|         X6         X7         X8         X9        X10
| -0.2333437 -1.9939860 -1.1494683 -0.5254225 -0.3982715
|        X11        X12        X13        X14        X15
| -0.2147995  0.8117574  0.9183250  1.8344980 -0.3761892
|        X16        X17        X18        X19        X20
|  1.0502763  1.9703120 -0.3695758  0.3161516 -2.2427062
```

reviewed in Riley et al. 2010). It lies far away from a classical meta-analysis (which has access only to reported effect sizes) and is actually a complete data analysis (which also has access to unpublished data; see §10.7, below).[14]

The combination of different studies in either a classical or an IPL meta-analysis is not likely to be a set of random samples of the population, even though individually they may be representative of different subsets of data space. Thus, "the statistical implementation of an individual participant data meta-analysis crucially must preserve the clustering of patients within studies; it is inappropriate to simply analyse individual participant data as if they all came from a single study" (Riley et al. 2010, p. 1). Finally, IPL meta-analysis reduces the reported effects bias of published studies (Lane & Dunlap 1978), but it does not eliminate funder bias, publication bias, or bias resulting from self-censorship.

SIMULATION

We can easily extend the simulation that we used for the classical meta-analysis in §10.3 to an IPL meta-analysis (Code Box 10.8). For the IPL meta-analysis, we fit a mixed-effect model to the combined dataset and assume that the analyst has access to all 20 of the predictors. Thus, we avoid the emergent question of the classical meta-analysis—how to handle unreported

[14] It is too late to argue against this label now, but it is really misleading to call IPL-MA a meta-analysis. With the original data on our computer, there is nothing meta about this analysis. Such fancy yet confusing naming does not help to endear statistics to a novice.

effects—and any differences we observe between the results of the classical and the IPL meta-analyses are a consequence of the completeness of the predictors.

We repeat the simulations and analysis many times, and then calculate their relative biases (Code Box 10.9).

For illustrative purposes, we compare the results from the classical meta-analysis in which we set the missing effect sizes to NA (light grey boxes in Figures 10.2 and 10.3) with the results of the mixed-model used in the IPL meta-analysis (dark grey boxes in Figure 10.3). In both cases, the estimates of the effect sizes of the common predictors X_1 and X_2 were unbiased, but because we had access to all the predictors in our mixed-model IPL meta-analysis, the estimates of the remaining 18 predictor variables also were nearly unbiased in the IPL meta-analysis (Figure 10.3).

10.5 One-step federated analysis

The "one" in one-step federated analysis refers to the practice of a single meta-analyst coordinating the federated analysis and defining the model. This model is then sent to the other research groups, who apply it to their local data and return the model output to the central coordinating group. On first glance, such a single prescribed model may seem to be unnecessarily restrictive; there is an element of chance in how representative this model will be for all other studies. However, there is a substantial benefit to this approach: the potential effects of spurious predictors, which in a meta-analysis would be applied to every included study, only enter once in a one-step federated analysis. In addition, *p*-hacking and other questionable research practices (see §3.5 in Chapter 3) are more of a problem for classical meta-analysis, in which multiple models can be examined until statistical significance is found (e.g., Friese & Frankenbach 2020), than it is for federated analysis where one model is prescribed and used. Although one-step federated analysis, especially in concert with strict data-security frameworks such as in DataSHIELD (Jones et al. 2012), must be seen as a very good development from a reproducibility point of view, multi-step federated analysis has the potential to re-introduce arbitration and hence re-open the door to *p*-hacking.

STEPS 1 AND 2. SIMULATE THE DATASET AND DISTRIBUTE IT TO MULTIPLE RESEARCH GROUPS

We use the same 21 simulated data and studies we used to illustrate a basic meta-analysis in the preceding section (Code Boxes 10.1 and 10.2). We then pick one of the 21 studies at random and assign it the role of the central coordinator. The model identified in the analysis of the central coordinator's dataset then is used as a common model that is prescribed to, and used by, the other 20 studies.

STEP 3. IDENTIFY THE COMMON MODEL TO BE PRESCRIBED

In practice, the coordinating research group will identify an appropriate model based on previous research and published data, their own experience, and new data they have already collected. For our simulation, however, we identify the prescribed model only from the data of the randomly chosen coordinating research group. As we did in the meta-analysis (Code Box 10.3), we used stepwise backward model selection to identify the common model that is then applied to the remaining 20 studies (Code Box 10.10).

STEP 4. APPLYING THE PRESCRIBED MODEL TO ALL STUDIES REVEALS A SECOND EMERGENT QUESTION

Each of the studies should run the prescribed model (`formulaCommon` in Code Box 10.10). However, because of local idiosyncrasies in data collection (simulated in Code Box 10.2 by assigning random subsets of predictors to each of the studies), not every local

Code Box 10.9: Repeated simulations of the IPL meta-analysis

```
# The comparison of relative biases uses the "effectSizes.list" object from Code
    Box 10.5

# Load required libraries
library(mvtnorm)
library(lmerTest)

# Full simulation, includes all steps from Code Boxes 10.1 (to create the
    datset), 10.2 (to distribute the data among studies), and 10.8
    (mixed-model analysis)
set.seed(123)
Nreps <- 100
IPLMAsimu <- replicate(Nreps, {
 S <- matrix(rnorm(20*20, 0, 0.21), 20, 20)
 Sigma <- S %*% t(S) # to ensure positive definite covariance matrix
 X <- rmvnorm(20000, mean=rep(0, 20), sigma=Sigma)
 X <- scale(X) # scale to get standardised regression coefficients as effect
     size
 slopes <- runif(20, -2, 2)
 Y <- X %*% slopes + rnorm(nrow(X), 0, 5)

 fullData <- cbind.data.frame(Y, X)
 colnames(fullData) <- c("Y", paste0("X", 1:20))

 Nstudies <- 21
 Ns <- sample(10:100, Nstudies) * 10 # between 100 and 1000
 allStudies <- fullData[1:sum(Ns), ]
 allStudies$ID <- rep(LETTERS[1:21], times=Ns)

 f <- as.formula(paste("Y~", paste0("X", 1:20, collapse=" + "), "+(1|ID)"))
 fmer <- lmer(f, data=allStudies)
 (summary(fmer)$coefficients[-1, 1] - slopes) / slopes
})

# Compute relative bias of IPL meta-analyses and compare it with relative bias
    from classical meta-analysis
relbiasNAs100 <- sapply(effectSizes.list, \(x) x[[2]]) # Relative bias from the
    classical meta-analysis

dats <- cbind(rbind(stack(as.data.frame(t(relbiasNAs100))),
              stack(as.data.frame(t(IPLMAsimu)))),
              "MAtype"=rep(c("meta analysis", "IPL-MA"), each=20*100))

# Plot the comparison (code not shown); output in Figure 10.3
```

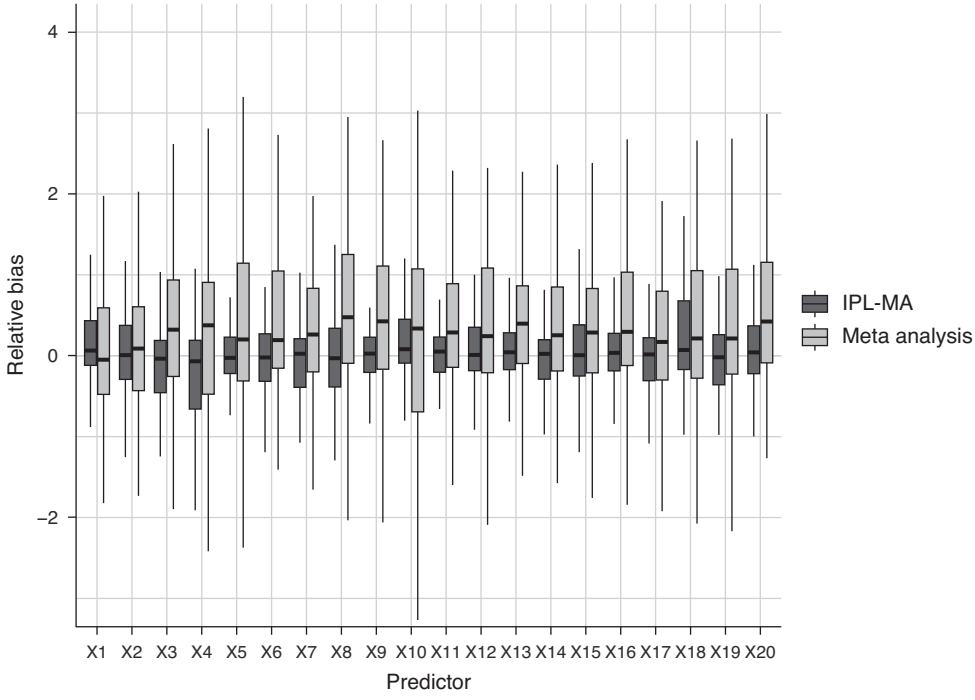


Figure 10.3 Biases in estimates of effect sizes determined from a classical meta-analysis (light grey) and individual participant-level meta-analysis (IPL-MA; dark grey) of 21 simulated studies aimed at estimating the effects of 10 predictors: two fixed (X_1 and X_2) and eight chosen at random for each study from an overall set of 18 (X_3–X_{20}). The reduction in bias in the IPL meta-analysis is a consequence of the analyst having access to the full set of predictors for each study.

Code Box 10.10: Identify the prescribed model for the one-step federated analysis

```
# This code uses the "studyList" object from Code Box 10.2

# Simulation
 set.seed(123)
 coordinatingSite <- sample(length(studyList), 1) # Choose the coordinating study
      at random

# Fit the model to be prescribed
 fmCommon <- step(lm(Y ~ (.)^2, data=studyList[[coordinatingSite]]),
                  scope=list("lower"=Y~X1+X2, "upper"=Y ~ (.)^2),
                  k=log(nrow(studyList[[coordinatingSite]])), trace=F)
 formulaCommon <- fmCommon$call$formula
 predictorsCommon <- attr(terms(fmCommon) , "term.labels")
```

Code Box 10.11: Apply the prescribed model to all studies

```
# This code uses the "studyList" object from Code Box 10.2 and the
    "predictorsCommon" object from Code Box 10.10.

# Identify the studies that contain all the predictors of the common model
fullPredictorSetAvailable <- rep(FALSE, length(studyList))
onestep.coefficients.mat <- matrix(NA, nrow=length(studyList),
    ncol=length(predictorsCommon))
colnames(onestep.coefficients.mat) <- predictorsCommon

# Run the common model on all the studies in studyList, using either the full
    predictor set or the set of available predictors
for (i in seq_along(studyList)){
 onOffer <- colnames(model.matrix(~(.)^2, data=studyList[[i]][1:2,-1]))
 studyPredictors <- intersect(onOffer, predictorsCommon)
 if (length(studyPredictors) == length(predictorsCommon))
     fullPredictorSetAvailable[i] <- TRUE
 fmone <- lm(formula(paste("Y~", paste(studyPredictors, collapse="+"))),
     data=studyList[[i]])
 onestep.coefficients.mat[i, match(studyPredictors, colnames(onestep.
     coefficients.mat))] <- coef(fmone)[-1]
}
```

study has the full set of predictors demanded by the prescribed model. As in the classical meta-analysis, we have two choices of what to do, but here the choices are different from setting missing predictors to either NA or zero. In the federated analysis, we could use only those studies in the analysis that are able to fit the full model as prescribed by the central coordinator. Alternatively, we could fit a local subset of the prescribed model. The first option would reduce the overall number of studies used in the analysis, whereas the second option would be contrary to the idea of a coordinated analysis. We explore the consequences of both choices with our simulations, and compare the resulting estimates (Code Box 10.11).

STEP 5. COMBINE THE RESULTS OF THE LOCAL MODELS

The final step is to collate the local study estimates and combine them, just as in a classical meta-analysis. As in Step 4 of the meta-analysis (in §10.3), we weight each study by the square root of its sample size (Code Box 10.12).

MANY SIMULATION RUNS LATER

We repeat the above procedure 100 times (Code Box 10.13) and present the results in exactly the same way as we did in Figure 10.2 for the meta-analysis (Figure 10.4).

The results (Figure 10.4) suggest that the one-step federated analysis yields more variable estimates than the classical meta-analysis (compare Figures 10.2 and 10.4). This is presumably because the bias of the prescribed common model cannot be improved upon by changing the local analyses. In the classical meta-analysis, every study may be slightly biased, but the studies are likely to be biased in different directions. In contrast, because the coordinating study determined which model to run, its bias (here resulting from model selection) in formulating the model will be amplified by the contributing studies. However,

Code Box 10.12: Compute weighted means of the local effect sizes

```
# This code uses the "onestep.coefficients.mat" and the
  "fullPredictorSetAvailable" objects from Code Box 10.9

# Compute the weighted means of all the studies
  apply(onestep.coefficients.mat, 2, \(x) weighted.mean(x, w=Ns, na.rm=T))

# Use only studies with full set of predictors
  apply(onestep.coefficients.mat[fullPredictorSetAvailable,, drop=F], 2, \(x)
       weighted.mean(x, w = sqrt(Ns[fullPredictorSetAvailable]), na.rm=T))
```

the difference in relative bias was small regardless of how we solved the emergent question for one-step federated analysis: using all the studies, even if they could fit only a subset of model terms (dark grey boxes in Figure 10.4; overall median relative bias = 0.19), or using only those studies that had the full set of variables necessary for the prescribed model (light grey boxes in Figure 10.4; overall median relative bias = 0.31).[15]

10.6 Multi-step federated analysis

Federated analysis was developed to address issues of data privacy and ownership.[16] One of the basic tenets of data privacy is to not collect sensitive, personal data in the first place (Anderson & von Seck 2020; Shanmugam et al. 2022). Any federated analysis initiated after collecting the data will therefore be incomplete, as only the predictors relevant for the original study purpose and pre-registration would be available.[17] Because local studies collect (and store) different (sub)sets of potential predictors, any model proposed by a single central coordinator (as in the one-step federated analysis) likely will be sub-optimal for the other participating sites. The idea behind a multi-step federated analysis is that by including iterative feedback between the coordinating and the participating sites, a better model will be used in the final analysis.

In practice, such feedback is sophisticated and multi-layered. Auxiliary analyses could be carried out (such as investigating correlations among predictors or providing summary statistics of the predictors), reported back to the coordinating site, and iterated to a more nuanced meta-analysis of the contributions of the individual studies (Deeks et al. 2023). Such sophistication is beyond the scope of our illustrative simulation, however. Instead, we simulate only a two-step federated analysis and we restrict the feedback from local studies to the coordinating sites to include only the identification of the predictors that were retained in the local analysis. Across all sites, these may well be all possible predictors, so we arbitrarily set a threshold: only predictors retained in at least one-third of the participating sites are used in the centrally prescribed model (Code Box 10.14). To continue with our comparisons, we again use the full dataset from both meta-analyses and the one-step federated analysis.

[15] We used the median here primarily for consistency with the box plots shown in Figure 10.4, but also because a few, but extremely large (negative), biases result in large discrepancies between medians and means.

[16] Federated analysis also neatly solves the issue of multiple versions of the same data floating around. Versioning would do the same trick, but federation forces the analyst to use only a single existing version per dataset. Any local correction will automatically be enforced in all future analyses.

[17] Although this may be annoying to statisticians, it is an obvious ways to ameliorate concerns about maintaining data privacy: data not collected cannot be misused.

Code Box 10.13: Repeated simulation of the one-step federated analysis

```
# Load required libraries
 library(mvtnorm)

# Full simulation, includes all steps from Code Boxes 10.1, 10.2, and 10.10 -
    10.12
MM <- model.matrix(~(.)^2, data=fullData[1:2,-1]) # for names
reps <- 100
Nstudies <- 21
set.seed(123)
onestep.simu <- replicate(reps, {
 # Set up the data and split over studies
  S <- matrix(rnorm(20*20, 0, 0.21), 20, 20)
  Sigma <- S %*% t(S)
  X <- rmvnorm(20000, mean=rep(0, 20), sigma=Sigma)
  X <- scale(X)
  slopes <- runif(20, -2, 2)
  Y <- X %*% slopes + rnorm(nrow(X), 0, 5)
  fullData <- cbind.data.frame(Y, X)
  colnames(fullData) <- c("Y", paste0("X", 1:20))
  Ns <- sample(10:100, Nstudies) * 10 # between 100 and 1000
 # Pick data from full simulation:
  index <- sample(20000, sum(Ns))
  studyList <- list()
  for (i in 1:Nstudies){
   studyList[[i]] <- fullData[index[1:Ns[i]], sort(c(1, 2, 3, sample(3:21, 8)))]
        # pick 10 predictors, 8 randomly
   index <- index[-c(1:Ns[i])]
  }
 # Identify the common model based on one site (chosen at random)
  coordinatingSite <- sample(length(studyList), 1)
  fmCommon <- step(lm(Y ~ (.)^2, data=studyList[[coordinatingSite]]),
                   scope=list("lower"=Y~X1+X2, "upper"=Y ~ (.)^2),
                   k=log(nrow(studyList[[coordinatingSite]])), trace=F)
  formulaCommon <- fmCommon$call$formula
  predictorsCommon <- attr(terms(fmCommon) , "term.labels")
 # Run the common model (or subset) on data from all sites
  fullPredictorSetAvailable <- rep(FALSE, length(studyList)) # list of studies
      that contain all predictors of the common model
  onestep.coefficients.mat <- matrix(NA, nrow = length(studyList), ncol =
      ncol(MM)-1) #without intercept
  colnames(onestep.coefficients.mat) <- colnames(MM)[-1]
  for (i in seq_along(studyList)){
   onOffer <- colnames(model.matrix(~(.)^2, data=studyList[[i]][1:2,-1]))
   studyPredictors <- intersect(onOffer, predictorsCommon)
   if (length(studyPredictors) == length(predictorsCommon))
        fullPredictorSetAvailable[i] <- TRUE
```

```
    fmone <- lm(formula(paste("Y~", paste(studyPredictors, collapse="+"))), data=
        studyList[[i]])
    onestep.coefficients.mat[i, match(studyPredictors, colnames(onestep.
        coefficients.mat))] <- coef(fmone)[-1] #without intercept
    rm(fmone, onOffer, studyPredictors)
  }
 # Return relative bias of estimates
  rbind(
   (apply(onestep.coefficients.mat[, 1:20], 2, \(x) weighted.mean(x, w=Ns,
        na.rm = T)) - slopes)/slopes,
   (apply(onestep.coefficients.mat[fullPredictorSetAvailable, 1:20, drop = F],
        2, \(x) weighted.mean(x, w = sqrt(Ns[fullPredictorSetAvailable, drop =
        F]), na.rm = T)) - slopes)/slopes)
 })

# For each replicate and coefficient, compute the median
 apply(onestep.simu[1, 1:10,], 1, median, na.rm=T)

|          X1          X2          X3          X4          X5
| -0.08922942  0.09889962  0.08893776  0.31362231  0.40845359
|          X6          X7          X8          X9         X10
|  0.26860638 -0.24141336  0.12033967  0.16429371  0.34990238

# Use only studies with full set of predictors
 apply(onestep.simu[2, 1:10,], 1, median, na.rm=T)

|          X1          X2          X3          X4          X5
|  0.15639821 -0.01924646  0.50045030  0.45837319  0.39258824
|          X6          X7          X8          X9         X10
|  0.82571210  0.11384747  0.34527025  0.06608803  0.49818583
```

SIMULATION

Once we have the common model, the analysis proceeds exactly the same as in the one-step federated analysis. Each participating study runs the agreed-upon model with the available predictors available and reports their parameter estimates as the effect sizes (Code Box 10.15).

We repeat this many times (Code Box 10.16), but because the modeling procedure of the two-step federated analysis is very similar in our simple simulations to the one-step federated analysis, we show the results only in the final section of the chapter (§10.7) when we compare all five approaches for analysing multiple datasets (Figure 10.6).

10.7 Complete data analysis

The final method for analysing multiple datasets is a complete analysis of all the available data across studies; this is also known as "coordinated data analysis" (e.g., Graham et al. 2022)). A complete data analysis can only improve on the other four methods—especially the classical and IPL meta-analyses—if it removes or at least substantially reduces biases associated with funders, publishers, and the researchers themselves. In the simulations we used

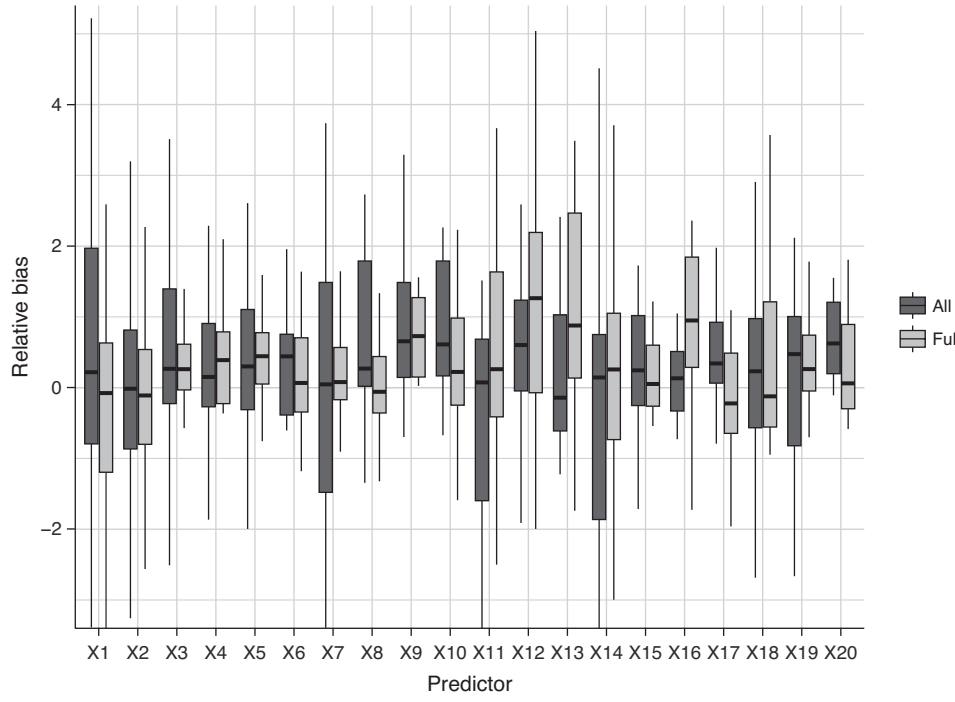


Figure 10.4 Biases in estimates of effect sizes determined from a one-step federated analysis of 21 simulated studies aimed at estimating the effects of 20 predictors, $\{X_1, X_2, \ldots, X_{20}\}$, where each study had two predictors in common (X_1 and X_2) and eight chosen at random from the remaining 18 ($X_3 - X_{20}$). Effect sizes were computed either by including all studies that provided any estimate (dark grey boxes) or only those where the full prescribed model could be fitted (light grey boxes).

Code Box 10.14: Identify the predictors retained in at least one-third of the individual studies

```
# This code uses the "studyList" object from Code Box 10.2 and the "fmList"
    object from Code Box 10.3.

# Identify the frequency with which each predictor occurs in the different
    possible models
frequencyOfVariables <- sort(table(unlist(sapply(fmList, \(x)
    names(x$coefficients)[-1]))))

# Select only those used in more than 7 studies (i.e., remove the least-frequent
    two thirds):
commonThird <- which(frequencyOfVariables > 7)
predictorsCommon <- sort(names(frequencyOfVariables)[commonThird])
commonModel <- as.formula(paste("Y ~", paste(predictorsCommon, collapse="+")))
```

Code Box 10.15: Fit the agreed-upon model to each participating study

```
# This code uses the "studyList" object from Code Box 10.2 and the "
    predictorsCommon" object from Code Box 10.14

twostep.coefficients.mat <- matrix(NA, nrow=length(studyList), ncol=ncol(MM)-1)
colnames(twostep.coefficients.mat) <- colnames(MM)[-1]
for (i in seq_along(studyList)){
 onOffer <- colnames(model.matrix(~(.)^2, data=studyList[[i]][1:2,-1]))
 studyPredictors <- intersect(onOffer, predictorsCommon)
 fmone <- lm(formula(paste("Y~", paste(studyPredictors, collapse="+"))), data=
     studyList[[i]])
 twostep.coefficients.mat[i, match(studyPredictors,
     colnames(twostep.coefficients.mat))] <- coef(fmone)[-1]
 rm(fmone, onOffer, studyPredictors)
}
```

in the preceding sections, we assumed that the 21 studies (generated in Code Boxes 10.1 and 10.2) either were published (or at least publishable) or were part of the federated consortium. In this section we expand on this simulation in two ways. First, we add some additional studies. Then, we simulate publication bias by removing from initial consideration those studies for which the main effects of predictors X_1 and X_2 are not statistically significant. The key comparison here is between the analysis of the complete dataset and the dataset simulating publication bias.

SIMULATION

We start by simulating the data in the same way as we did for the previous four methods, but this time we add a study identifier (ID) label to the full dataset.

For the 40 simulated studies generated in Code Box 10.17, we fit the same model as we did in Code Box 10.3 and then identify those studies for which the effect sizes of the parameter estimates for neither X_1 nor X_2 is statistically significant ($p > 0.05$; Code Box 10.18). We (somewhat arbitrarily, but conditioned by our own experiences) consider that such studies would be unlikely to be published, and perhaps not even submitted. By removing them from the complete dataset, we are left with a dataset that reflects publication bias.

Now, we fit the same linear model to each of the two datasets: the complete dataset (Code Box 10.17) and the publication-bias dataset (Code Box 10.18). As we have done in previous sections, we use stepwise backward model selection to identify the best model. Next, we use the differences between the parameter estimates (slopes) and the true values used to simulate the data to compute relative bias (Code Box 10.19).

Finally, we run the simulation multiple times and extract and organise all the relative biases for plotting (Code Box 10.20; Figure 10.5).

As we would have expected, the complete data analysis is largely unbiased, despite the step-wise backward selection, because the number of data points available for estimation allows for the identification of a model (dark grey bars in Figure 10.5). However, it turns out that for the very large dataset we simulated, our imposed publication bias did not change the bias very much.

Code Box 10.16: Repeated simulation of the two-step federated analysis

```
# Load required libraries
 library(mvtnorm)

# Full simulation, includes all steps from Code Boxes 10.1, 10.2, 10.14, and
    10.15
 reps <- 100
 set.seed(123)
 twostep.simu <- replicate(reps, {
  S <- matrix(rnorm(20*20, 0, 0.21), 20, 20)
  Sigma <- S %*% t(S) # To ensure positive definite covariance matrix
  X <- rmvnorm(20000, mean=rep(0, 20), sigma=Sigma)
  X <- scale(X) # Scale to get standardised regression coefficients as effect
      size
  slopes <- runif(20, -2, 2)
  Y <- X %*% slopes + rnorm(nrow(X), 0, 5)

  fullData <- cbind.data.frame(Y, X)
  colnames(fullData) <- c("Y", paste0("X", 1:20))

  Nstudies <- 21
  Ns <- sample(10:100, Nstudies) * 10 # Between 100 and 1000
  index <- sample(20000, sum(Ns)) # Pick data from full simulation
    studyList <- list()
  for (i in 1:Nstudies){
    studyList[[i]] <- fullData[index[1:Ns[i]], sort(c(1, 2, 3, sample(3:21, 8)))]
         # pick 10 predictors, 8 randomly
    index <- index[-c(1:Ns[i])] # remove data already assigned
  }

# Run local analyses
  fmList <- lapply(studyList, \(x) step(lm(Y ~ (.)^2, data=x),
                    scope=list("lower"=Y~X1+X2, "upper"=Y ~ .),
                    k=log(nrow(x)), trace=F))

# Variables present in ANY study
  (frequencyOfVariables <- sort(table(unlist(sapply(fmList, \(x) names(
      x$coefficients)[-1])))))
  commonThird <- which(frequencyOfVariables > 7)
  predictorsCommon <- sort(names(frequencyOfVariables)[commonThird])
  commonModel <- as.formula(paste("Y ~", paste(predictorsCommon, collapse="+")))

# Run this common model on all studies (akin to one-step, just different
      model):
  twostep.coefficients.mat <- matrix(NA, nrow=length(studyList), ncol=ncol(MM)-1)
  colnames(twostep.coefficients.mat) <- colnames(MM)[-1]
  for (i in seq_along(studyList)){
    onOffer <- colnames(model.matrix(~(.)^2, data=studyList[[i]][1:2,-1]))
```

```
    studyPredictors <- intersect(onOffer, predictorsCommon)
    fmone <- lm(formula(paste("Y~", paste(studyPredictors, collapse="+"))), data=
        studyList[[i]])
    twostep.coefficients.mat[i, match(studyPredictors, colnames(twostep.
        coefficients.mat))] <- coef(fmone)[-1]
    rm(fmone, onOffer, studyPredictors)
  }

  # Compute and return relative bias of estimates
  (apply(twostep.coefficients.mat[, 1:20], 2, \(x) weighted.mean(x, w=Ns,
       na.rm=T)) - slopes)/slopes
})
```

Code Box 10.17: Simulate the complete dataset and include study identifiers

```
# Load required library
library(mvtnorm)

# Simulate the complete dataset following procedures in Code Boxes 10.1 and 10.2
set.seed(123)
Nstudies <- 40 # Nearly double the number of studies relative to Code Box 10.2
Ns <- sample(10:100, Nstudies) * 10 # Number of observations per study is
    between 100 and 1000

S <- matrix(rnorm(20*20, 0, 0.21), 20, 20)
Sigma <- S %*% t(S) # To ensure positive definite covariance matrix
X <- rmvnorm(sum(Ns), mean=rep(0, 20), sigma=Sigma)
X <- scale(X) # Scale to get standardised regression coefficients as effect size
slopes <- runif(20, -2, 2)
Y <- X %*% slopes + rnorm(nrow(X), 0, 5)

fullData <- cbind.data.frame(Y, X)
colnames(fullData) <- c("Y", paste0("X", 1:20))

fullData$ID <- rep(paste0("study", 1:40), times=Ns) # Add ID label to each
    study

index <- sample(sum(Ns), sum(Ns)) # Pick data from full simulation
studyList <- list()
for (i in 1:Nstudies){
studyList[[i]] <- fullData[index[1:Ns[i]], sort(c(1, 2, 3, sample(3:21, 8)))]
    # pick 10 predictors, 8 randomly
 index <- index[-c(1:Ns[i])] # Remove data already assigned
}
```

Code Box 10.18: Identify individual studies that would be suppressed by publication bias

```
# This code uses the "fullData" and "studyList" objects from Code Box 10.17

# Fit the model
 fmList <- lapply(studyList, \(x) step(lm(Y ~ ., data=x),
                 scope=list("lower"=Y~X1+X2, "upper"=Y ~ .),
                 k=log(nrow(x)), trace=F))

# Identify those studies for which the coefficients of X_1 and X_2 are not
    significantly different from zero
 notSignif <- which(sapply(fmList, \(x) (any(summary(x)$coefficients[2:3, 4] >
     0.05))))

# Produce a dataset with only published studies
 publishedData <- fullData[!(fullData$ID %in% paste0("study", notSignif)), ]
```

Code Box 10.19: Fit linear models to the complete and publication-bias datasets, and estimate their relative biases

```
# This code uses the "fullData" and "studyList" objects from Code Box 10.16, and
     the "publishedData" object from Code Box 10.18

# Fit the model to the complete dataset
 flmfull <- step(lm(Y ~ ., data=fullData),
            scope=list("lower"=Y~X1+X2, "upper"=Y ~ .),
            k=log(nrow(fullData)), trace=F)

# Fit the same model to the publication-bias dataset
 flmpublished <- step(lm(Y ~ ., data=publishedData),
                 scope=list("lower"=Y~X1+X2, "upper"=Y ~ .),
                 k=log(nrow(publishedData)), trace=F)

# Compute slopes
# Slopes for the complete dataset (only the first five estimates are shown
    below)
 (coef(flmfull)[-1] - slopes[as.numeric(gsub("X", "",
        names(coef(flmfull)[-1])))]) /
       slopes[as.numeric(gsub("X", "",
        names(coef(flmfull)[-1])))]

|           X1           X2           X3           X4           X5
|  0.375685858 -0.184092974  0.230822958  0.425125749  0.023293604

# Slopes for the publication-bias dataset (only the first five estimates are
    shown below)
(coef(flmpublished)[-1] - slopes[as.numeric(gsub("X", "",
```

```
        names(coef(flmpublished)[-1]))]) /
       slopes[as.numeric(gsub("X", "",
        names(coef(flmpublished)[-1])))]
```

```
|          X1           X2           X4           X5           X6
| 0.007355101  3.679548334  1.622909829 -0.241065526  0.004764971
```

Code Box 10.20: Full simulation and relative biases of the models fit to the complete and publication-biased datasets

```
# Load required libraries
 library(mvtnorm)

# Full simulation, includes all steps from Code Boxes 10.17 - 10.19
 reps <- 100
 set.seed(123)
 full.simu <- replicate(reps, { # Simulate 40 studies, remove those with non
      -significant effects for X1 or X2
  Nstudies <- 40
  Ns <- sample(10:100, Nstudies) * 10 # Observations per study between 100 and
      1000

  S <- matrix(rnorm(20*20, 0, 0.21), 20, 20)
  Sigma <- S %*% t(S) # To ensure positive definite covariance matrix
  X <- rmvnorm(sum(Ns), mean=rep(0, 20), sigma=Sigma)
  X <- scale(X) # Scale to get standardised regression coefficients as effect
      size
  slopes <- runif(20, -2, 2)
  Y <- X %*% slopes + rnorm(nrow(X), 0, 5)

  fullData <- cbind.data.frame(Y, X)
  colnames(fullData) <- c("Y", paste0("X", 1:20))
  fullData$ID <- rep(paste0("study", 1:40), times=Ns)

  index <- sample(sum(Ns), sum(Ns)) # Pick data from full simulation

  studyList <- list()
  for (i in 1:Nstudies){
  studyList[[i]] <- fullData[index[1:Ns[i]], sort(c(1, 2, 3, sample(4:21, 8)))]
      # Pick 10 predictors, 8 randomly
    index <- index[-c(1:Ns[i])] # Remove data already assigned
  }

  # Remove "non-significant" studies to simulation publication bias
   fmList <- lapply(studyList, \(x) step(lm(Y ~ ., data=x),
                                         scope=list("lower"=Y~X1+X2, "upper"=
                                              Y ~ .),
                                         k=log(nrow(x)), trace=F))
```

```
  notSignif <- which(sapply(fmList, \(x) (any(summary(x)$coefficients[2:3, 4] >
      0.05))))

 # Create a second dataset with only published studies:
  publishedData <- fullData[!(fullData$ID %in% paste0("study", notSignif)), ]

 # Fit the model to the complete and publication-biased datasets
  flmfull <- step(lm(Y ~ ., data=fullData),
                  scope=list("lower"=Y~X1+X2, "upper"=Y ~ .),
                  k=log(nrow(fullData)), trace=F)

  flmpublished <- step(lm(Y ~ ., data=publishedData),
                       scope=list("lower"=Y~X1+X2, "upper"=Y ~ .),
                       k=log(nrow(publishedData)), trace=F)

 # Extract model coefficients
  list("full"=(coef(flmfull)[-1] - slopes[as.numeric(gsub("X", "",
      names(coef(flmfull)[-1])))])/slopes[as.numeric(gsub("X", "",
      names(coef(flmfull) [-1])))],
   "published"=(coef(flmpublished)[-1] - slopes[as.numeric(gsub("X", "",
      names(coef(flmpublished)[-1])))])/slopes[as.numeric(gsub("X", "",
      names(coef(flmpublished)[-1])))] )
})

# Organize the coefficients into a dataframe for plotting
 dats <- data.frame()
 for (i in 1:reps){
  onedat <- data.frame(
    "coef"=names(full.simu[1, i][[1]]),
    "estimate"=full.simu[1, i][[1]],
    "type"=rep(names(full.simu[1, i]), length(full.simu[1, i][[1]])))
  )
  twodat <- data.frame(
    "coef"=names(full.simu[2, i][[1]]),
    "estimate"=full.simu[2, i][[1]],
    "type"=rep(names(full.simu[2, i]), length(full.simu[2, i][[1]])))
  )
  dats <- rbind(dats, onedat, twodat)
 }
 dats$coef <- factor(dats$coef, levels=paste0("X", 1:20))
```

10.8 Conclusions and outlook

Our overall aim in this chapter was to illustrate and discuss the five different methods of combining multiple datasets in a synthetic analysis to identify consistent effects (Figure 10.1). We emphasise that our simulations are not representative of all possible scenarios in these methods and do not identify or deal with the currently most pressing issues of each of them, nor were they designed to identify which method is the best. Indeed, it is unlikely that there is a

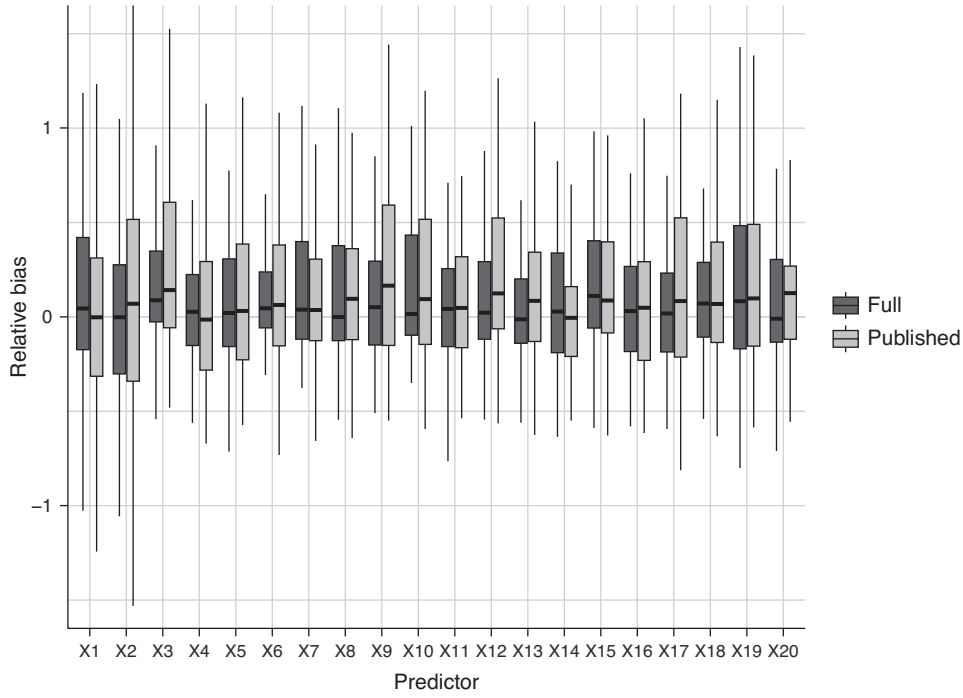


Figure 10.5 Biases in estimates of effect sizes determined from the complete data analysis aimed at estimating the effects of 20 predictors, $\{X_1, X_2, \ldots, X_{20}\}$. Effect sizes were computed including either all data (dark grey boxes) or only those where estimates of both X_1 and X_2 were significantly different from zero ($p \leq 0.05$; light grey boxes).

universally best method for data synthesis. Meta-analysis was developed without concerns for data ownership or privacy and confidentiality. Federated analysis, on the other hand, explicitly addresses these challenges, which are unrelated to methods of statistical inference. Because our overall focus in this chapter (as throughout this book) is on statistical inference, it is unsurprising that the results of our classical and IPL meta-analysis outperform both one- and multi-step federated analysis (Figure 10.6). Simulations are specific to particular questions and datasets (§1.2 in Chapter 1), and our results clearly reflect our aim and how we defined and devised our simulations.

Based on these simulations, we draw two key conclusions. First, the more the fitted model resembles the complete (full) model (i.e., without model selection and reporting on all parameters fitted), the lower is its relative bias. That is, analysing the complete dataset is the least-biased approach (black boxes in Figure 10.6), followed closely by the structurally very similar IPL meta-analysis (lighter grey boxes in Figure 10.6). The classical meta-analysis (lightest grey bars in Figure 10.6) is more biased than the IPL meta-analysis because it depends on what individual studies did before publishing their effect sizes or handing them over to a meta-analyst.

Second, the parameter estimates become more and more variable as the model structure becomes more tightly prescribed. In the classical meta-analysis, we allowed each local study to fit whatever model they wanted to, and each study made use of their subset of predictors to maximum effect. Regardless of whether a model was prescribed centrally (one-step federated analysis) or after some iteration (multi-step federated analysis), the prescription itself amplified any misspecification and reinforced any existing biases in the data or the model

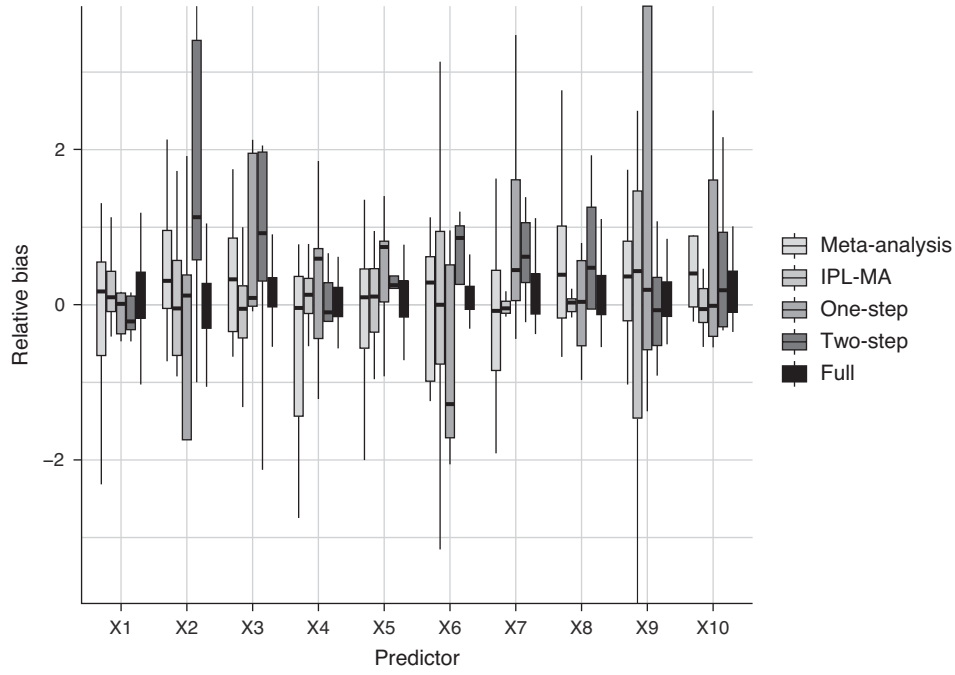


Figure 10.6 Biases in estimates of effect sizes of 10 predictors, $\{X_1, X_2, \ldots, X_{10}\}$, for the five data synthesis methods illustrated in Figure 10.1 and applied to identically simulated datasets (Code Boxes 10.5, 10.9, 10.13, 10.16, and 10.20).

(darker grey bars in Figure 10.6). In our simulations, both types of federated analysis suffered from incomplete sets of predictors. We did this in an attempt to mimic the situation in which different studies record different predictors (e.g., because they were set up for different purposes). Without wanting to over-interpret our simulation results, it seems evident that designing studies jointly before data are collected will reduce bias and variance of estimates of parameters and effect sizes.

The development of new and more robust methods for data synthesis is an active area of research. From early meta-analyses through federated analysis to network meta-analysis (Rücker 2012; Cipriani et al. 2018; Freeman 2022), many different challenges have been overcome, but others remain unsolved or unaddressed (e.g., Borenstein 2009; Koricheva et al. 2013; Higgins et al. 2023). No matter how good the method, however, the quality of the contributing studies remains the most crucial element. Because of pre-registration, establishment and enforcement of international protocols (e.g. Altman et al. 2012; Sauerbrei et al. 2014; Lang & Altman 2015),[18] and the vital interest of the public in correct outcomes, synthesis of clinical trials continually demonstrates how high-quality results can be achieved (Higgins et al. 2023).[19] If privacy requirements allow it and confidentiality can be maintained, storing all predictors for future data synthesis would be an important advancement.

[18] A more complete list is available from the National Library of Medicine (2023).

[19] Of course, biomedical research and associated synthesis are not immune to inadequate data quality, scientific misconduct, or outright fraud. But methodologically, biomedical research clearly leads all other fields in its ability to take full advantage of advancements in applied statistics.

It remains to be seen whether federated analyses will be used more commonly outside of medical, psychological, and sociological research fields (and any others that routinely use human subjects). Federated analysis has particular utility for collaborations that need to use embargoed datasets (e.g., data collected by active [post]graduate students who are working on individual datasets for their theses and dissertations) or that include institutions that are adamantly opposed to open data policies. Another possible use of federated analysis would be the development of an open analytical platform with specified models that any researcher or research group could use on their own data provided they committed to reporting their individual (local) analysis to the consortium. Until then, however, meta-analyses are here to stay and will deliver nearly unbiased insights as long as publication bias is low (e.g., Andrews & Kasy 2019; Azizi et al. 2023; Yang et al. 2023).

10.9 Summary

- There are five different types of methods for analysing multiple datasets, and each uses different types of data with different characteristics and qualities:

  Classical meta-analysis summarizes and synthesizes published means, variances, and sample sizes to determine standardised effect sizes of specific phenomena or experimental treatments. The estimates are known to be biased due to unpublished results and social biases such as self-censorship, funding biases, and publication biases.

  Improved meta-analysis (a.k.a. individual participant-level meta-analysis) uses the raw data from the available studies instead of the published summary statistics. Although it cannot overcome social biases, improved meta-analysis can take advantage of archived but otherwise unpublished data and does ensure that identical analyses are done on all available datasets.

  Federated analysis applies a common statistical model to multiple datasets without the analyst having access to any or all of the individual datasets. Federated analysis addresses privacy and security concerns associated with datasets that for whatever reason cannot be shared with others. Federated analysis comes in two forms: one-step and multi-step federated analysis.

  In **one-step federated analysis**, the analyst defines the model, writes the script, and sends it to participating research groups who run the script on their own data and return the results to the analyst for synthesis. One-step federated analysis avoids publication bias but can produce biased results because of lack of representation of a diversity of research groups or unstable results because of regularising effects.

  In **multi-step federated analysis**, the analysis is defined collectively. Multi-step federated analysis accounts for differences in data collection, management, and quality among the different participating groups better than one-step federated analysis. Multi-step federated analysis resolves virtually all of the problems and biases of meta-analysis and one-step federated analysis. However, multi-step federated analysis does re-open the door to potential *p*-hacking, a door which was closed by one-step federated analysis.

  A complete analysis of all available data from multiple studies on a specific question, hypothesis, or topic would be as unbiased as possible. Methodologically, however, it would be identical to any standard analysis for which we had all the data we needed to fit the model(s) of interest.

- Our simulations suggest two general results about combining results from multiple studies:

1. The more the model fitted to multiple studies resembles the complete model with all parameters and without model selection, the lower is its relative bias. Thus, it is best to analyse all the raw data from all the studies rather than to only fit models to collections of summary statistics.
2. The variability of parameter estimates in federated analysis increases as the model fitted to any one of a group of studies becomes more tightly prescribed.

- Jointly and cooperatively designing multiple studies before data are collected and sharing the results afterwards in a complete analysis or multi-step federated analysis will reduce bias and variance of estimates of parameters and effect sizes.

References

Abbott, A. (1998). Galileo's manuscripts go on the Internet. *Nature*, 393(6685), 501. doi: 10.1038/31049.

Aitken, A. C. (1936). IV.—On least squares and linear combination of observations. *Proceedings of the Royal Society of Edinburgh*, 55, 42–48. doi: 10.1017/S0370164600014346.

Altman, D. G., McShane, L. M., Sauerbrei, W., & Taube, S. E. (2012). Reporting recommendations for tumor marker prognostic studies (REMARK): Explanation and elaboration. *PLoS Medicine*, 9(5), e1001216. doi: 10.1371/journal.pmed.1001216.

Anderson, C. (2008). The end of theory: The data deluge makes the scientific method obsolete. *WIRED*, 16(7). doi: https://www.wired.com/2008/06/pb-theory/.

Anderson, D. & von Seck, R. (2020). The GDPR and its impact on the web. In G. Carle, S. Günther, & J. B. (Eds.), *Proceedings of the Seminar Innovative Internet Technologies and Mobile Communications (IITM), Summer Semester 2020* (pp. 1–5). Munich, Germany: Chair of Network Architectures and Services.

Andrade, C. (2020). Understanding the basics of meta-analysis and how to read a forest plot: As simple as it gets. *The Journal of Clinical Psychiatry*, 81(5), 21858. doi: 10.4088/JCP.20f13698.

Andrews, I. & Kasy, M. (2019). Identification of and correction for publication bias. *American Economic Review*, 109(8), 2766–2794. doi: 10.1257/aer.20180310.

Azizi, Z., Lindner, S., Shiba, Y., et al. (2023). A comparison of synthetic data generation and federated analysis for enabling international evaluations of cardiovascular health. *Scientific Reports*, 13(11), 11540. doi: 10.1038/s41598-023-38457-3.

Borenstein, M. (2009). Effect sizes for continuous data. In H. Cooper, L. V. Hedges, & J. C. Valentine (Eds.), *Handbook of Research Synthesis and Meta-Analysis*, 2nd edition (pp. 221–235). New York, NY, USA: Russell Sage Foundation.

Borer, E. T., Harpole, W. S., Adler, P. B., et al. (2014). Finding generality in ecology: A model for globally distributed experiments. *Methods in Ecology and Evolution*, 5(1), 65–73. doi: 10.1111/2041-210X.12125.

Brodeur, A., Cook, N., & Heyes, A. (2020). Methods matter: *p*-hacking and publication bias in causal analysis in economics. *American Economic Review*, 110(11), 3634–3660. doi: 10.1257/aer.20190687.

Chalmers, I. (1990). Underreporting research is scientific misconduct. *Journal of the American Medical Association*, 263(10), 1405–1408. doi: 10.1001/jama.1990.03440100121018.

Cipriani, A., Furukawa, T. A., Salanti, G., et al. (2018). Comparative efficacy and acceptability of 21 antidepressant drugs for the acute treatment of adults with major depressive disorder: A systematic review and network meta-analysis. *The Lancet*, 391(10128), 1357–1366. doi: 10.1016/S0140-6736(17)32802-7.

Curran, P. J. & Hussong, A. M. (2009). Integrative data analysis: The simultaneous analysis of multiple data sets. *Psychological Methods*, 14(2), 81–100. doi: 10.1037/a0015914.

Deeks, J. J., Higgins, J. P. T., & Altman, D. G. (2023). Chapter 10: Analysing data and undertaking meta-analyses. In J. P. T. Higgins, J. Thomas, J. Chandler, M. Cumpston, T. Li, M. G. Page, & V. A. Welch (Eds.), *Cochrane Handbook for Systematic Reviews of Interventions*. Version 6.4, 2nd edition (Updated August 2023). Chichester, UK: John Wiley & Sons.

Doiron, D., Burton, P., Marcon, Y., et al. (2013). Data harmonization and federated analysis of population-based studies: The BioSHaRE project. *Emerging Themes in Epidemiology*, 10(1), 12. doi: 10.1186/1742-7622-10-12.

Dubben, H.-H. & Beck-Bornholdt, H.-P. (2005). Systematic review of publication bias in studies on publication bias. *British Medical Journal*, 331(7514), 433–434. doi: 10.1136/bmj.38478.497164.F7.

Easterbrook, P. J., Berlin, J. A., Gopalan, R., & Matthews, D. R. (1991). Publication bias in clinical research. *The Lancet*, 337(8746), 867–872. doi: 10.1016/0140-6736(91)90201-y.

Egger, M. & Smith, G. D. (1995). Misleading meta-analysis. *British Medical Journal*, 310(6982), 752–754. doi: 10.1136/bmj.310.6982.752.

Egger, M., Smith, G. D., & Sterne, J. A. C. (2001). Uses and abuses of meta-analysis. *Clinical Medicine*, 1(6), 478–484. doi: 10.7861/clinmedicine.1-6-478.

Faul, F., Erdfelder, E., Buchner, A., & Lang, A.-G. (2009). Statistical power analyses using G*Power 3.1: Tests for correlation and regression analyses. *Behavior Research Methods*, 41(4), 1149–1160. doi: 10.3758/BRM.41.4.1149.

Faul, F., Erdfelder, E., Lang, A. G., & Buchner, A. (2007). G*Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior Research Methods*, 39(2), 175–191. doi: 10.3758/BF03193146.

Ferguson, C. J., Copenhaver, A., & Markey, P. (2020). Reexamining the findings of the American Psychological Association's 2015 Task Force on Violent Media: A meta-analysis. *Perspectives on Psychological Science*, 15(6), 1423–1443. doi: 10.1177/1745691620927666.

Fraser, L. H., Henry, H. A. L., Carlyle, C. N., et al. (2013). Coordinated distributed experiments: An emerging tool for testing global hypotheses in ecology and environmental science. *Frontiers in Ecology and the Environment*, 11(3), 147–155. doi: 10.1890/110279.

Freeman, S. C. (2022). Individual patient data meta-analysis and network meta-analysis. *Methods in Molecular Biology*, 2345, 279–298. doi: 10.1007/978-1-0716-1566-9_17.

Friendly, M. (2008). The golden age of statistical graphics. *Statistical Science*, 23(4), 502–535. doi: 10.1214/08-STS268.

Friese, M. & Frankenbach, J. (2020). p-Hacking and publication bias interact to distort meta-analytic effect size estimates. *Psychological Methods*, 25(4), 456–471. doi: 10.1037/met0000246.

Froelicher, D., Troncoso-Pastoriza, J. R., Raisaro, J. L., et al. (2021). Truly privacy-preserving federated analytics for precision medicine with multiparty homomorphic encryption. *Nature Communications*, 12(11), 5910. doi: 10.1038/s41467-021-25972-y.

Galilei, G. (1960). *On Motion and On Mechanics: Comprising De Motu (ca. 1590), Translated with Introduction and Notes by I. E. Drabkin; and Le Meccaniche (ca. 1600), Translated with Introduction and Notes by Stillman Drake*. Madison, WI, USA: University of Wisconsin Press.

Gelman, A. & Carlin, J. (2014). Beyond power calculations: Assessing type S (sign) and type M (magnitude) errors. *Perspectives on Psychological Science*, 9(6), 641–651. doi: 10.1177/1745691614551642.

Glass, G. V. (1977). Integrating findings: The meta-analysis of research. *Review of Research in Education*, 5, 351–379. doi: 10.2307/1167179.

Graham, E. K., Willroth, E. C., Weston, S. J., et al. (2022). Coordinated data analysis: Knowledge accumulation in lifespan developmental psychology. *Psychology and Aging*, 37(1), 125–135. doi: 10.1037//pag0000612.

Hallock, H., Marshall, S. E., 't Hoen, P. A. C., et al. (2021). Federated networks for distributed analysis of health data. *Frontiers in Public Health*, 9, 712569. doi: 10.3389/fpubh.2021.712569.

Hedges, L. V. (1984). Estimation of effect size under non-random sampling: The effects of censoring studies yielding statistically insignificant mean differences. *Journal of Educational Statistics*, 9(1), 61–85. doi: 10.3102/10769986009001061.

Higgins, J. P. T., Thomas, J., Chandler, J., et al., Eds. (2023). *Cochrane Handbook for Systematic Reviews of Interventions.* Version 6.4, 2nd edition (updated August 2023). Chichester, UK: John Wiley & Sons.

Jones, E. M., Sheehan, N. A., Masca, N., et al. (2012). DataSHIELD—shared individual-level analysis without sharing the data: A biostatistical perspective. *Norsk Epidemiologi*, 21(2), 231–239. doi: 10.5324/nje.v21i2.1499.

Koricheva, J., Gurevitch, J., & Mengersen, K. L., Eds. (2013). *Handbook of Meta-analysis in Ecology and Evolution*. Princeton, NJ, USA: Princeton University Press.

Lane, D. M. & Dunlap, W. P. (1978). Estimating effect size: Bias resulting from the significance criterion in editorial decisions. *British Journal of Mathematical and Statistical Psychology*, 31(2), 107–112. doi: 10.1111/j.2044-8317.1978.tb00578.x.

Lang, T. A. & Altman, D. G. (2015). Basic statistical reporting for articles published in biomedical journals: The "Statistical Analyses and Methods in the Published Literature" or the SAMPL guidelines. *International Journal of Nursing Studies*, 52(1), 5–9. doi: 10.1016/j.ijnurstu.2014.09.006.

Li, T., Sahu, A. K., Talwalkar, A., & Smith, V. (2020). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3), 50–60. doi: 10.1109/MSP.2020.2975749.

Mupepele, A.-C., Walsh, J. C., Sutherland, W. J., & Dormann, C. F. (2016). An evidence assessment tool for ecosystem services and conservation studies. *Ecological Applications*, 26(5), 1295–1301. doi: 10.1890/15-0595.

Murdoch, B. (2021). Privacy and artificial intelligence: Challenges for protecting health information in a new era. *BMC Medical Ethics*, 22(1), 122. doi: 10.1186/s12910-021-00687-3.

NASA (2021). Data Processing Levels. https://www.earthdata.nasa.gov/engage/open-data-services-and-software/data-information-policy/data-levels. Accessed 17 January 2024.

National Library of Medicine (2023). Research Reporting Guidelines and Initiatives: By Organization. https://www.nlm.nih.gov/services/research_report_guide.html. Accessed 17 January 2024.

Nosek, B. A., Ebersole, C. R., DeHaven, A. C., & Mellor, D. T. (2018). The preregistration revolution. *Proceedings of the National Academy of Sciences, USA*, 115(11), 2600–2606. doi: 10.1073/pnas.1708274114.

Orrock, J. L., Borer, E. T., Brudvig, L. A., et al. (2015). A continent-wide study reveals clear relationships between regional abiotic conditions and post-dispersal seed predation. *Journal of Biogeography*, 42(4), 662–670. doi: 10.1111/jbi.12451.

Pick, J. L., Nakagawa, S., & Noble, D. W. A. (2019). Reproducible, flexible and high-throughput data extraction from primary literature: The metaDigitise R package. *Methods in Ecology and Evolution*, 10(3), 426˘431. doi: 10.1111/2041-210X.13118.

Riley, R. D., Lambert, P. C., & Abo-Zaid, G. (2010). Meta-analysis of individual participant data: Rationale, conduct, and reporting. *British Medical Journal*, 340, c221. doi: 10.1136/bmj.c221.

Rubin, M. (2020). Does preregistration improve the credibility of research findings? *The Quantitative Methods for Psychology*, 16(4), 366–390. doi: 10.20982/tqmp.16.4.p376.

Rücker, G. (2012). Network meta-analysis, electrical networks and graph theory. *Research Synthesis Methods*, 3(4), 312–324. doi: 10.1002/jrsm.1058.

Sauerbrei, W., Abrahamowicz, M., Altman, D. G., et al. (2014). STRengthening Analytical Thinking for Observational Studies: The STRATOS initiative. *Statistics in Medicine*, 33(30), 5413–5432. doi: 10.1002/sim.6265.

Schmid, C. H., Stijnen, T., & White, I., Eds. (2021). *Handbook of Meta-Analysis*. Boca Raton, FL, USA: Chapman & Hall/CRC.

Shanmugam, D., Diaz, F., Shabanian, S., et al. (2022). Learning to limit data collection via scaling laws: A computational interpretation for the legal principle of data minimization. In *FAccT '22: Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency* (pp. 839–849). New York, NY, USA: Association for Computing Machinery.

Small, H. (1999). *Florence Nightingale: Avenging Angel*. London, UK: Palgrave Macmillan.

Smith, T. O. & Hing, C. B. (2011). "Garbage in, garbage out"—The importance of detailing methodological reasoning in orthopaedic meta-analysis. *International Orthopaedics*, 35(2), 301–302. doi: 10.1007/s00264-010-1171-9.

Stanley, T. D., Doucouliagos, H., & Ioannidis, J. P. A. (2022). Retrospective median power, false positive meta-analysis and large-scale replication. *Research Synthesis Methods*, 13(1), 88–108. doi: 10.1002/jrsm.1529.

Valgimigli, M., Gragnano, F., Branca, M., et al. (2021). P2y12 inhibitor monotherapy or dual antiplatelet therapy after coronary revascularisation: individual patient level meta-analysis of randomised controlled trials. *British Medical Journal*, 373, n1332. doi: 10.1136/bmj.n1332.

van Aert, R. C. M., Wicherts, J. M., & van Assen, M. A. L. M. (2019). Publication bias examined in meta-analyses from psychology and medicine: A meta-meta-analysis. *PLoS ONE*, 14(4), e0215052. doi: 10.1371/journal.pone.0215052.

Warnat-Herresthal, S., Schultze, H., Shastry, K. L., et al. (2021). Swarm learning for decentralized and confidential clinical machine learning. *Nature*, 594(7862), 265–270. doi: 10.1038/s41586-021-03583-3.

Wei, K., Li, J., Ding, M., et al. (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15, 3454–3469. doi: 10.1109/TIFS.2020.2988575.

Yang, Y., Sánchez-Tójar, A., O'Dea, R. E., et al. (2023). Publication bias impacts on effect size, statistical power, and magnitude (Type M) and sign (Type S) errors in ecology and evolutionary biology. *BMC Biology*, 21, 71. doi: 10.1186/s12915-022-01485-y.

11

Putting it through its paces: Does this new method work?

Research and its strategic partner, development, colloquially R&D, together are the "creative and systematic work undertaken in order to increase the stock of knowledge—including knowledge of humankind, culture and society—and to devise new applications of available knowledge" (OECD 2015, p. 28). Although we tend to think of the research disciplines and subdisciplines as divinely ordained with fixed residences in their specific academic departments and programs, the actual ongoing and evolving creative work of R&D occasionally gives birth to new fields of research (Rudwick 2016). New subdisciplines arise more frequently, particularly when technological advances make an existing laboratory or field-based method much more efficient or inexpensive (think: DNA sequencing) or when new inventions lead to new discoveries (think: remote sensing or fMRI imaging). As these inventions and methods are tested, used, and compared, some are found to be more useful; others, less so.

Statistics and data analysis also are research disciplines, and new methods for data analysis are developed in similar ways as they are in the laboratory, the surgery, or the field. New algorithms, estimators, indices, and workflows (collectively, "methods") regularly are introduced with the avowed intent of being better able to represent a specific aspect of data.[1] To be useful to researchers and practitioners, any such method should be a step forward and improve on existing ones: it should be (for example) faster, quicker to converge, more sensitive (Belmont et al. 2022), or efficient (Chung et al. 2013), handle larger data than before (Reshef et al. 2011), be less biased (Strona et al. 2014), have less variance (Bried & Vilmi 2022), or tell us something entirely new (Fay et al. 2022). In all cases, the new method should be sensitive (do whatever it does accurately), specific (not be fooled by noise), and easy to interpret (see §3.4.3 in Chapter 3).

Although it is relatively easy to propose a new statistical or data-analytical method, it is much harder to convince our colleagues of its value. Indeed, peer-reviewed journals routinely reject manuscripts proposing new methods unless they deliver one of the aforementioned boons. The burden is on the inventor to demonstrate its quality, utility, and novelty.[2] Ideally, any paper introducing such a new method would use data simulation to

[1] As part of advertising and merchandising, publication of new methods routinely are heralded as new dawns, breakthroughs, or panaceas. Keeping their feet to the fire, for example, James Heathers runs the X account @justsaysinmice that retweets the "hottest" new medical findings to call out the hype that is made of a potential human therapy that so far has been tested only in mice (Chakradhar 2019).

[2] For two reasons, the inclusion of the word "potentially" in a paper or book can suggest that the proposed method is not yet fully developed. First, if a method is "potentially" able to improve the accuracy or precision of a measurement, estimate, or prediction, then it may or may not. That is, the claimed improvement has not been

demonstrate its applicability and robustness for a wide range of data and in light of typical problems encountered in relevant (sub)disciplines. Guidelines on how to do simulations for statistical and data-analytic methods exist or are being developed in many fields, with medicine (Boulesteix et al. 2017; Morris et al. 2019) and ecology (e.g. Drake 2023; Ellison 2023) leading the way.

In this chapter we review the steps required to convince ourselves and and our colleagues of a new statistical or data-analytic method (see also Morris et al. 2019). Largely, these steps use data simulation to test the proposed method, put it through its paces, see if it breaks under pressure, and enable others to reproduce our results or extend our tests. The steps and methods presented in this chapter differ from most of the ones presented in other chapters. The simulations are not restricted to a specific dataset because the new method has to work in general, not for only a single dataset or specific system. At the same time, it is impossible to simulate any or all of the conditions to which someone else may want to apply the new method. Still, there are common strategies to test the robustness of a method to "normal," extreme, or inane inputs, its asymptotic behaviour, and its relative quality. We teach by example: this is not a complete or general how-to guide to testing and validating any new statistical or data-analytic method.

We are also aware that virtually all journals that publish new statistical or data-analytic methods require accessible (open-access) source code that will be permanently and publicly archived should the paper be accepted. For example, "where contributions concern statistical methodology, the authors should develop the knowledge of the statistical performance of the technique to the greatest extent possible. For instance, simulation studies might be used to quantify statistical power, bias and variance. To be generally accessible, statistical contributions should be accompanied by high quality, open access source code" (Drake 2023, p. 1645). In that spirit, we remind you that all of the code we have developed for this can be downloaded from the online code repository associated with this book (Dormann & Ellison 2024). The online repository also includes additional examples and associated code that we have developed since writing the book, and we encourage our readers and others interested in data simulation to contribute their own examples and code to the repository.

What you will learn from this chapter

- The key steps, focusing on those using data simulation, that are required to convince ourselves and and our colleagues of a new statistical or data-analytic method.

11.1 Unit testing

New statistical and data-analytic methods are written (coded, programmed) in a computer language. After we have written a piece of code, the first step is to use unit testing to determine whether it works correctly. Such unit tests have two levels: verification of the code itself, and functional verification. The former asks whether the code actually runs and returns the expected type of output (e.g., an integer, a character, a matrix). The latter asks whether the code works correctly and returns accurate values (within the expected precision). Unit testing is a necessary, but not sufficient, test of a new method, because unit testing does not address a method's value—its utility, quality, or novelty. We return to assessing the value of new methods in §11.3–11.5.

demonstrated conclusively and indisputably. Second, when a method identifies "potential" candidates, e.g., of a new pathway in a biochemical signalling reaction, then it actually is unclear whether or not the method has produced a new and useful insight. Throwing sand into the eyes of the reader does not instil trust in the new method.

11.1.1 Code verification

At the very least, the code should run without crashing the computer or sending it off into an infinite loop. For code written in a relatively "low"-level programming language (e.g., assembly language or a compiled language such as C), memory overflow, deletion of pointer references, etc. are common problems. "Higher"-level languages, including `R`, `Python`, and `Matlab`, that are very abstracted from the internal computational details of assembly language or machine code are usually written in a more natural language and often are interpreted (executed line-by-line) rather than batch-compiled. They are not perfect, however. They can be slow to run, are difficult to optimise, and are used easily (perhaps too easily) by inexperienced programmers. Because it is relatively easy to write and hack together functions in higher-level languages, the resulting code may give unpredictable results or be brittle and break when exposed to unanticipated inputs.

An example of a straightforward unit test for code verification is simply to check that the output of the new function is of the correct type (e.g., always a matrix). This same test also would check that the function handles any type of input, either by converting it to the type required for the function or by aborting the operation and returning a useful error message.

11.1.2 Functional verification

Of course, a new method should do more than simply run. It should actually return consistent and accurate results with defined precision. At its most basic, functional verification should illustrate logical or mathematical consistency. For example, if the result of our proposed method is a probability, its value must be in the interval $[0, 1]$. If the result of our code is in the right range, we can move on to testing its accuracy.

For many functions, even complicated ones, there are toy examples for which we can compute the results analytically or by hand and compare them to the values returned by our code. In such cases, our unit testing should go through many such cases of known inputs and outputs. For example, Nakagawa et al. (2023) developed a method to guesstimate missing standard deviations that would enable meta-analysts to retain more studies in a meta-analysis than if all included studies were required to report standard deviations. Nakagawa et al. (2023) then used simulation to show that their method worked as they had proposed. Similarly, Korthauer et al. (2019) investigated bias in false-discovery rates across several methods using data simulation. Such demonstration by simulation is by now the de facto standard for method development in statistical and computational fields (e.g., Simpkins et al. 2022).

It also is not sufficient that the method works well within typical ranges. Rather, it should be robust to applications well beyond anticipated typical inputs (e.g., as inputs or sizes of objects $\rightarrow \pm\infty$). For example, the number of possible configurations of a binary matrix of size $m \times n$ with a proportion p of 1s and predefined (fixed) row and column sums can be generated by random simulation (Carstens et al. 2018); such matrices often are used in inferring the total number of species (i.e., species richness) in one or more sites (Smith & van Belle 1984; Ugland et al. 2003; Mao & Colwell 2005). Will this proposed method work for large values of m and n? The short answer is no. Barvinok (2010) compared the simulation approach with an analytical, asymptotic estimate of the number of such matrices (i.e., its cardinality). Because the possible number of configurations of such binary matrices is enormous, the proportion of simulations that can cover this configurational space is too small for reliable extrapolation for even moderate values (e.g., $m, n > 10$).

Finally, unit testing can reveal computational limits. For example, writing a recursive function to compute the Fibonacci sequence ($F_n = F_{n-1} + F_{n-2}$, with $F_0 = 0;\ F_1 = 1$) will work fine for n of reasonable size, but would soon freeze the computer (value out-of-bounds)

or send it into an infinite loop if there is no built-in stopping rule. Similarly, a method that works well for the analysis of a few hundreds of nodes in a simple network may not scale well to millions of nodes.

AN ILLUSTRATIVE EXAMPLE: ANALYTICAL ERROR PROPAGATION FOR THE MEAN OF TWO RANDOM VARIABLES

Functional verification also ensures that we have understood the mathematics correctly, so that our code returns the correct results. This is of particular concern for error propagation in workflows (see §7.7 in Chapter 7).

As an illustrative example, we consider the problem of averaging distributions, which occurs when we average the predictions of two models. In the simplest case, let both such predictions be normally distributed and that we also have estimates of $\bar{y}_i$ and their associated standard errors: $\hat{y}_1 \pm s_{y_1}$ and $\hat{y}_2 \pm s_{y_2}$.

The average of the two means is simply their mean: $(\hat{y}_1 + \hat{y}_2)/2$. But what is the average of their standard errors? It turns out that the variances are additive but their standard deviations are not, and the variances also need to be weighted (Wikipedia Contributors 2023a):

$$\mathrm{var}(aA + bB) = a^2\mathrm{var}(A) + b^2\mathrm{var}(B) + 2ab\,\mathrm{cov}(A, B). \tag{11.1}$$

The standard deviation is simply the square root of the variance computed using Equation 11.1.

In our case, $a = b = 0.5$, and in the benign case of no correlation between A and B, the third term (the covariance) vanishes. This leaves us with $\mathrm{var}((\hat{y}_1 + \hat{y}_2)/2) = \frac{1}{4}\mathrm{var}(\hat{y}_1) + \frac{1}{4}\mathrm{var}(\hat{y}_2)$. Now this is a bit unexpected. The variance of the average is the sum of only the one-quarter of the variances of the contributing distributions.

Code Box 11.1: Computing the variance of an average

```
set.seed(123)
x1 <- rnorm(1000, mean=-3, sd=3)   # First Gaussian random variable
x2 <- rnorm(1000, mean=3, sd=5)    # Second Gaussian random variable

# Mean of the two Gaussian random variables (expectation = 0)
mean.sim <- 0.5 * sample(x1, 1000, replace=T) + 0.5 * sample(x2, 1000,
    replace=T)
mean(mean.sim)

| [1] 0.02755365

sd(mean.sim) # Standard deviation of simulation

| [1] 2.879558

(sdmean <- sqrt(0.5^2 * var(x1) + 0.5^2 * var(x2))) # SD according to equation
    11.1

| [1] 2.929897
```

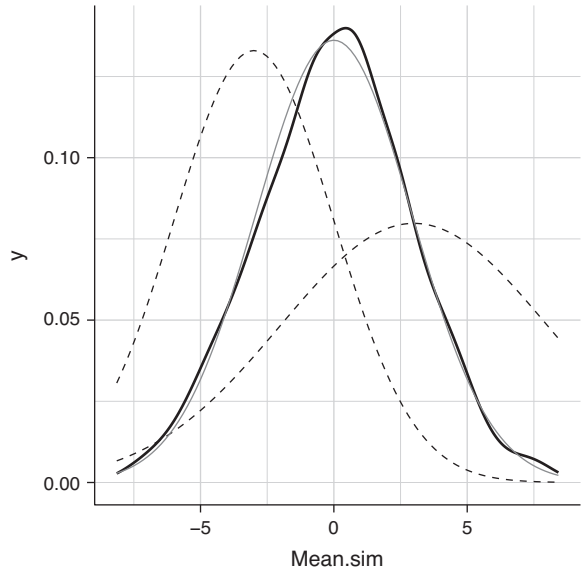


Figure 11.1 Simulated averaging of two distributions (dashed lines). The simulation (black line) and the analytical solution (grey line) using Equation 11.1 yield nearly identical results.

We can use a simple simulation to check whether we correctly coded the computation of the propagation of uncertainty (Code Box 11.1).

Indeed, we did understand the propagation of uncertainty correctly (Figure 11.1). The point where our intuition could have led us astray was that we had to do our computations on the variance, not the standard deviations themselves. Because we are interested in the standard deviation, which is the square root of the variance, we might have noted that $\sqrt{\frac{1}{4}} = \frac{1}{2}$, which looks suspiciously like the division by 2 that yields the average of the means. However, directly averaging the standard error would be wrong (as it would yield a value for the variance of the average of the sum of the means of ≈ 4, rather than ≈ 3).

11.2 Dimensional analysis

An additional useful check on a computation is to ask whether the units on the left-hand side and right-hand side of an equation are the same. If they aren't, something went wrong in the derivation of the equation and it must be corrected. This check is called dimensional analysis (e.g. Gibbings 2011) and is a key practice of physicists. Dimensional analysis could (or should) be part of unit testing, but since dimensional analysis is used rarely by researchers in fields other than physics (but see Chapter 3 of Legendre 2022 and Chapter 6 in Schneider 2009), it is not a standard element of unit testing.

Although there are some intricacies to dimensional analysis (such as when to use Rayleigh's or Buckingham's methods), the principle is straightforward and is based on treating units as if they were variables and ignoring all constants. Then the correct equation can be rearranged such that eventually the same units are on both sides of the equation and hence cancel each other out.

A simple illustration of dimensional analysis is illustrated with the relationship between enzyme activity v (in units of mol/s) and substrate concentrations $[S]$ (in units of mol)

expressed by Michaelis-Menten kinetics:

$$v = \frac{V_{\max}[S]}{K_m + [S]}, \text{ with } \left[\frac{mol}{s} = \frac{\frac{mol}{s} \cdot mol}{mol + mol} = \frac{\frac{mol^2}{s}}{mol} = \frac{mol}{s} \right]. \tag{11.2}$$

When computing the quotient $\frac{V_{\max}[S]}{K_m+[S]}$, some units cancel, and the result of the cancellation is in mol/s, the same as the units of v.

Similarly, when we want to define a new indicator variable (i.e., an index), we use dimensional analysis to determine the units of the index itself (e.g., the Standardised Potential Evapotranspiration Index, or SPEI; Vicente-Serrano et al. 2010). In many cases we work to design an index that is dimensionless (e.g., specific leaf mass is measured in g/g, so there is no need to report the units, which cancel each other). On the other hand, if our proposed index has units that would be difficult to interpret (e.g., mol/m), it might suggest to us that our index might not be as useful as we initially thought.

11.3 Comparisons

Is our proposed new method better than existing ones? As we noted at the beginning of this chapter, new methods are more likely to be taken up if they are of higher quality and utility, where these values are often expressed in terms of speed, efficiency, accuracy and precision, sensitivity, specificity, and ease of interpretation. To demonstrate any of these values, we compare our new method to existing ones.

Comparisons between methods lend credence to the new one if it first shows its equivalence to known correct methods, and second, if it shows improvements in specific attributes of interest (e.g., improved performance for smaller sample sizes or lower computational effort). We most commonly use two types of comparisons: across benchmarks and with respect to simulated data. Each of these comparisons has different utility.

11.3.1 Benchmark datasets

In some fields, standard benchmark datasets exist for which any new method has to demonstrate an improvement over existing ones. Benchmark datasets include real-world data that were collected to answer real-world questions or solve real-world problems of interest. They are used to show that a new method that worked fine on made-up toy examples will not falter when exposed to an actual dataset. Such benchmarks are not without problems, because we rarely know the absolute truth and because there may be a non-negligible amount of incorrect labels or other errors in the benchmark datasets (Northcutt et al. 2021).

Many machine-learning competitions[3] are run using benchmark datasets, and the University of California at Irvine maintains a repository of such datasets that can be used for virtually any newly developed method for machine-learning models.[4] For example, unless your new machine-learning model cannot recognise handwritten digits (Figure 11.2) better than the current standard method, you can leave your new method on the shelf to gather dust.

Note that no single benchmark dataset is sufficient to demonstrate that a method is useful. Typically dozens to hundreds of such benchmarks are run to get a more comprehensive evaluation. And, of course, some methods may perform better on a subset of those benchmarks than on others. When comparing, for example, the performance of `R` and `Python`,

[3] E.g., https://www.kaggle.com/competitions.

[4] https://archive.ics.uci.edu/.

Figure 11.2 Example subset of the Modified National Institute of Standards and Technology (MNIST) handwritten digits database. Reproduced from Steppan (2017)—own work, CC BY-SA 4.0.

they are on a par when it comes to algebraic tasks (since those actually are carried out in compiled C/C++-coded libraries), but vary greatly for many other tasks and with a user's skill in programming (Hothorn et al. 2005; Couto et al. 2017; Moura 2021; Pereira et al. 2021).

11.3.2 Comparing methods using simulated datasets

Instead of comparing different methods to a single benchmark dataset, we can also compare our new method directly with its competitors. After all, what is the point of a new method if it does not improve on existing ones? There are several attributes for which one can compare methods. From a statistical standpoint, accuracy (or unbiasedness) and statistical efficiency (least variance of estimator) often are the most desirable properties.

Comparisons focused on accuracy and bias often use simulated data: when we know the truth—because we defined it with our data simulation—we actually can see which method works best to approximate it (as long as we are neutral a priori about the outcome of such comparisons; Boulesteix et al. 2017). Accurate and unbiased estimators are particularly relevant for inferential statistics, whereas low prediction error is more important for prediction. The number of studies reporting on such method comparisons are legion (e.g., Binder et al. 2013; Borji et al. 2015; John et al. 2019; Korthauer et al. 2019; Hopf & Reifenrath 2021; Tran et al. 2020; Yang et al. 2020).

We have illustrated the standard way to evaluate unbiasedness several times in this book. In all of them, we simulated a range of conditions, ran the method to estimate the target statistic, and compared the estimate to the known truth. Mean squared prediction error (MSPE) is the sum of the squared bias of the prediction and its variance (Hastie et al. 2009, Equation 2.25):

$$\text{MSPE} = \text{bias}_{\hat{y}}^2 + \text{var}(\hat{y}). \tag{11.3}$$

Accordingly, if a little bit of bias reduces predictive variance substantially, the net effect is a reduction of prediction error. This is the foundation of all regularisation approaches, such as ridge regression, the lasso, and elastic net shrinkage (Tibshirani 1996; Zou & Hastie 2005; Hastie et al. 2020), and model selection in a predictive setting (van Houwelingen & Sauerbrei 2013; Dormann et al. 2018).

From a computational point of view, runtime (in CPU time), ease of use ("user experience"), and computational efficiency (energy consumption, scalability) feature highly (e.g. Worley & Foster 1994; Lee & Skadron 2005; Valeriani 2020; Speich et al. 2021). Of course, there are likely to be trade-offs among these different dimensions, and between the results

of statistical and computational comparisons. The best method to use is likely to depend on a specific context.

11.4 Intellectual advancement

New methods may arise de novo or by tinkering with existing ones, but in the scientific context what counts is whether they can contribute substantially to the generation of new scientific knowledge. Unless a new method allows us to improve our ability to test hypotheses derived from theory, it contributes nothing to the scientific method (e.g., McKeon 1966; Ford 2000; Ladyman 2001; Popper 2002). This means that if a method is to have a broad impact across many fields (and be published in the most impactful journals), it should be motivated by the general pursuit of scientific advancement, not simply technological improvement, personal preferences, or visual fireworks.

Such a statement warrants some qualification. In statistics and many other fields, there are dozens of good journals that constantly publish methods that improve on existing approaches. The production of such incremental improvements has become a subdiscipline in its own right, and one that often has little consideration for whether the new approach actually leads to new scientific insights. We usually take for granted that the use of a less-biased estimator or a method to handle larger datasets will yield better science.

This statement is generally true if by better we mean more accurate estimators with higher precision. But in practice, such improvements may be irrelevant. For example, we have known since the 1950s and can effectively demonstrate, using current measurements and statistical methods, that 1.6–4.8 million people die every year from diseases caused by particulate dust (Daly 1959; Lelieveld et al. 2015; GBD 2015 Risk Factors Collaborators 2016). Of course, we can develop even more sensitive $PM_{2.5}$ sensors and more accurate or precise mortality estimates. But doing so really is moot from both scientific and public-health perspectives.

Similarly, progress in technology and statistics in most cases does not translate one-to-one into scientific advances. The availability of point-and-click statistical software did not turn all users into competent statisticians (in fact, it may have made us less so), and scientists now may spend too much time thinking about clever analyses instead of coming up with smarter experiments or surveys to address crucial hypotheses (e.g., Houlahan et al. 2017).

11.5 Intuitive understanding

A new method has a higher chance of proliferation and acceptance if it is correct (§11.3), helpful (§11.4), and easy to understand (intuitive). What we mean by intuitive comprehension of a new method has at least two facets: an understanding of how the method works and an understanding of what the results mean. That is, a new method may be correct even if we do not understand it, but it certainly helps us to consider using it if we can get the gist of it, intuit its workings, and relate it to current thinking.

One element that facilitates an intuitive grasp of a method is recognition of familiar elements. A modification of an existing method is easier to understand than a genuinely de novo one. Even surprising combinations of known approaches are easier to accept than formally equivalent methods that are densely described.[5] For example, when Schapire (1990) introduced boosting to statistics, the general idea was easy to understand: run a model, see

[5] When James Clerk Maxwell formulated his four equations of electrodynamics in 1861–1862, they were met with resistance and resentment by experimental physicists because they required substantial mathematical skill to understand (at the time, coupled partial differential equations were not taught in universities). However, two

which points were fitted poorly, fit another model to the residuals with poorly fit points receiving higher weight, and sum the two steps; continue until no further improvement is made (Freund & Schapire 1997). The fundamental elements of boosting are weighted regression and iterative regression on residuals, both of which were already known, widely used, and easy to grasp. A decade later, Friedman et al. (2000) developed the mathematical, not the algorithmic, description for this approach, allowing them to improve computation by an order of magnitude. Their work is formal, less intuitive (except to mathematically adept statisticians), and terse. Had Friedman et al. (2000) came up with boosting before Schapire (1990), we doubt it would have caught on as readily as it did.

11.5.1 Multiple descriptions enhance intuitive understanding

In today's world of inter- and transdisciplinary approaches to science, a new statistical method ideally ought to be described in three ways: verbally, mathematically, and algorithmically (for a nice example, see Reshef et al. 2011). Verbal descriptions trigger an intuitive understanding, mathematics are used for formal specification, and algorithms implement the method. Not all three will be possible in every case, but their complementary roles matter when communicating to multiple or broad audiences.[6]

To illustrate the point of introducing a new method and presenting it verbally, mathematically, and algorithmically, consider the idea of a recurrent neural network (RNN). It was the first of a range of time-series approaches within artificial neural networks; we specifically describe the simplest RNN, the Elman network.

To avoid copyright infringement, we use the verbal definition of an Elman network from Wikipedia Contributors (2023b):

> An Elman network is a three-layer network (arranged horizontally as x, y, and z in the illustration [Figure 11.3]) with the addition of a set of context units (u in the illustration). The middle (hidden) layer is connected to these context units fixed with a weight of one. At each time step, the input is fed forward and a learning rule is applied. The fixed back-connections save a copy of the previous values of the hidden units in the context units (since they propagate over the connections before the learning rule is applied). Thus the network can maintain a sort of state, allowing it to perform such tasks as sequence-prediction that are beyond the power of a standard multilayer perceptron.

Some terms in this description may be unfamiliar, such as calling predictors "context," or labelling output in this case with z, while the unobserved latent ("hidden") states are labelled y. But by looking at Figure 11.3 while reading the text, we can intuit the idea: there is a time-series input, x, and a time-series output, z. In the Elman network, there is a variable computed using information not only from x_t, but also from an internal latent state y_t, and

elements led to their rapid adoption. First, they combined four known equations (Gauss' Law, Gauss' Law of Magnetism, Faraday's Law of Induction, and Ampère's Circuital Law, the latter two with additions and corrections by Maxwell) in a surprising but intuitive way. Second, they worked marvellously in explaining and predicting electric, optical, and magnetic phenomena. (We are glossing over the somewhat more convoluted nature of Maxwell's equations that were later simplified by Oliver Heaviside in the notation that we now use.) Interestingly, only the term "electric potential" remains from Maxwell's textual description; all other terms have since changed. Yet the equations have remained the same, as does the meaning of the terms.

[6] Indeed, in domain-specific publications (i.e., those in fields that use statistics but are not widely read or published in by statisticians), the verbal description usually is foregrounded in the main text, the mathematics are in appendices or supplements, and the algorithms and code are posted in public, open-access archives. See Fawcett & Higginson (2012) for additional discussion.

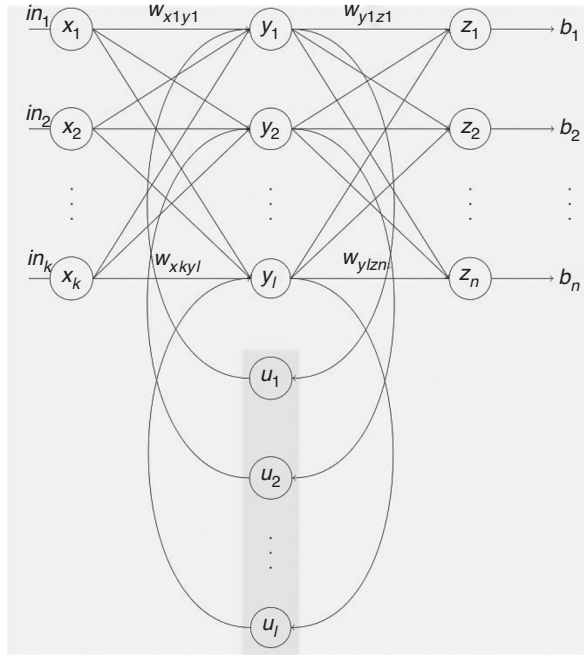


Figure 11.3 The Elman Simple Recurrent Neural Network. The context layer ($u_1 \ldots u_l$) provides a temporal memory to the network, allowing it to "remember" previous hidden layer states. (Image by Fyedernoggersnodden, reproduced by Wikipedia Contributors (2023b); CC-BY 3.0)

both are used to predict the output z_t. The latent variable y_t is itself informed by what happened earlier (in x_{t-k}); changes to y_t are stored in u. As a result, the value of output z_t includes information from previous time steps as coded ("remembered") in $z_{t-1}, z_{t-2}, \ldots, z_{t-k}$.

Mathematically, this looks a bit more mundane:

$$\begin{aligned} y_t &= \sigma_y(W_y x_t + u_y y_{t-1} + b_y), \\ z_t &= \sigma_z(W_z h_t + b_z). \end{aligned} \tag{11.4}$$

In artificial neural networks, the non-adept are likely to be confused because $\sigma(\cdot)$ is a function, not a standard deviation. The function $\sigma(\cdot)$ describes how the input to the neural network is related to the output, for example, using a sigmoidal or a ramp function (and there are two, potentially different, such "activation functions" in the RNN described in Equation 11.4: one for y and one for z). In Equation 11.4, the hidden layer y_t is a function of the input x_t and u_{t-1} plus a constant intercept b_y. That is, the equation for y_t is akin to that of a regression model, with a slope W_y for input x, an offset (coefficient set to 1) $u_y y_{t-1}$, and the intercept. The function $\sigma(\cdot)$ is then equivalent to a link function in a GLM (and may indeed be exactly the logit function of a binomial GLM).

The second line in Equation 11.4 is also similar to a regression expression, but without the offset; the outcome z_t is related directly to the latent predictor y_t.

The third description of the RNN is its implementation in code. Since practically all efficient artificial neural networks are coded in C++, this implementation is, too (Code Box 11.2; this specific snippet is taken from IBM's Open Source tutorial [Jones 2017]). It represents a single evaluation ("forward step") for given parameters W (in the code: `w_h_i/c1/c2`, `w_o_h`), and $b = 0$:

Code Box 11.2: C++ code representing a single step in an Elman RNN

```
void RNN_feed_forward( void )
{
   int i, j, k;

   // Stage 1: Calculate hidden layer outputs
   for ( i = 0 ; i < HIDDEN_NEURONS ; i++ )
   {
      hidden[ i ] = 0.0;

      // Incorporate the input.
      for ( j = 0 ; j < INPUT_NEURONS+1 ; j++ )
      {
         hidden[ i ] += w_h_i[ i ][ j ] * inputs[ j ];
      }

      // Incorporate the recurrent hidden.
      hidden[ i ] += w_h_c1[ i ] * context1[ i ];
      hidden[ i ] += w_h_c2[ i ] * context2[ i ];

      // apply tanh activation function.
      hidden[ i ] = tanh( hidden[ i ] );
   }

   // Stage 2: Calculate output layer outputs
   for ( i = 0 ; i < OUTPUT_NEURONS ; i++ )
   {
      outputs[ i ] = 0.0;

      for ( j = 0 ; j < HIDDEN_NEURONS+1 ; j++ )
      {
         outputs[ i ] += ( w_o_h[ i ][ j ] * hidden[ j ] );
      }

      // apply sigmoid activation function.
      outputs[ i ] = sigmoid( outputs[ i ] );
   }

   // Stage 3: Save the context hidden value
   for ( k = 0 ; k < HIDDEN_NEURONS+1 ; k++ )
   {
      context2[ k ] = context1[ k ];
      context1[ k ] = hidden[ k ];
   }

   return;
}
```

This code uses different symbols and labels for the same things as the verbal and mathematical description. `HIDDEN_NEURONS` in the code is the same as y in Equation 11.4 and `INPUT_NEURONS` is the same as x. Context u only remembers two steps, so there are two variables, `context1` and `context2`, which are equivalent to u_t and u_{t-1} (in this respect the code deviates from the conceptual and mathematical description, where only u_t is included). Hence, the first line of the mathematical description (Equation 11.4) is broken down into lines 10–22 of the code, and the second line of Equation 11.4 is in lines 24–36 of the code. The closing lines (41 and 42) update the lagged hidden variable y. Note that the activation functions are $\sigma_y(\cdot) = \tanh(\cdot)$ and $\sigma_z(\cdot) = \text{logit}(\cdot)$.

The best description of a new method is whichever of the three descriptions—verbal, mathematical, algorithmic—we may find easiest to understand, or at least to use to develop an intuitive understanding of the method. Since anyone may differ in how they best learn and understand new concepts and ideas, presenting a new method with all three descriptions improves the likelihood that others will use it.

11.5.2 Intuitive presentation of results

Not every new method yields results that are easily understood. This is particularly the case for new indices, which may exhibit all sorts of weird behaviours. Two simple and well-known classical cases are ratios and the Shannon-Weiner diversity index. Ratios ($\frac{a}{b}$) suffer from the problem that when $a \geq b$, $\frac{a}{b} \in [1, \infty)$, but if $b > a$, $\frac{a}{b} \in [0, 1)$. Plots of such ratios will be dominated by the region where $a > b$, which can easily lead to misrepresentation and misinterpretation. A solution is to use and plot the log-ratio, i.e., $\log \frac{a}{b} = \log a - \log b$, which is symmetric: if $a > b$, $\frac{a}{b} \in (0, \infty)$, whereas if $b > a$, $\frac{a}{b} \in (-\infty, 0)$ (e.g., Elston et al. 1996).

Shannon entropy (Shannon 1948) is often used in ecology to quantify "diversity:"

$$H' = -\sum_{i=1}^{R} p_i \ln p_i = -\sum_{i=1}^{R} \ln p_i^{p_i}, \tag{11.5}$$

where R is the number of species, and p_i is the relative abundance of the ith species (with all relative abundances summing to 1).[7] As the second part of the equation shows, the Shannon-Weiner index equals the geometrically-weighted mean of species richness. Its interpretation is straightforward: larger values of H' are better because they represent a larger number of species of similar abundance. But there is a catch: counterintuitively, losing a common species increases H' (because it increases the relative abundance of all others [Code Box 11.3, samples `c1` and `c2`]). The reverse also leads to a counterintuitive result. Gaining a rare species lowers H', because it increases the dissimilarity in the weights (Code Box 11.3, samples `d1` and `d2`).

Although the behaviour of H' is in line with the logic behind Shannon's definition of entropy, it is not what most ecologists want (or expect) to see in a diversity index. Still, despite repeated clarifications and calls to adapt different and more intuitive diversity indices (Spellerberg & Fedor 2003; Jost 2006; Chao et al. 2014; Roswell et al. 2021), H' as defined

[7] Ecologists usually refer to H' as "Shannon" diversity or "Shannon-Weiner" diversity. For many years, it was erroneously referred to as "Shannon-Weaver" diversity (Spellerberg & Fedor 2003). Its use now is being superseded by Hill numbers for measuring diversity (Chao et al. 2014); the Hill number of order 1 equals the exponential of H': ${}^1D = e^{H'}$.

Code Box 11.3: Shannon entropy is a counterintuitive diversity index

```
# Load required libraries
 library(vegan) # Includes a built-in Shannon diversity function

 c1 <- c(1000, 10, 10, 10, 1)
 c2 <- c(10, 10, 10, 1) # abundant species lost
 d1 <- c(10, 10, 10, 10, 1)
 d2 <- c(10, 10, 10, 10, 1, 1000) #abundant species gained
 round(sapply(list(c1, c2, d1, d2), diversity), 2)

| [1] 0.17 1.21 1.47 0.22
```

in Equation 11.5 is still far and away the most commonly employed index in ecology and environmental science.

11.5.3 New wine in old bottles can taste better

If a new method improves on an existing one but yields the same output, the former's output can be interpreted in the same way as the latter's. For example, before we had rapid computational methods to solve nonlinear equations or fit linear models to non-Gaussian or discrete data, we would first transform the response variables so we could fit a linear model, then back-transform the output to the nonlinear scale. As computational methods improved and were implemented in accessible software, it took only a relatively small intuitive leap to fit GLMs to response variables with a range of distributions (e.g., binary, integer count, percentage) using appropriate link functions instead of fitting an ordinary least-squares model to some weirdly transformed response variable (e.g., the folkloric arcsin square-root transformation applied to proportional data between 0 and 1; Warton & Hui 2011).

Two more recent examples in applied statistics that have overcome software limitations are illustrated with the model diagnostics package `DHARMa` (Hartig 2022) and the `brms` interface (Bürkner 2017) to the Bayesian software Stan (Stan Development Team 2023). Although we are familiar with standard regression diagnostic plots (e.g., Figure 8.6 in Chapter 8), comparable diagnostic residual plots of the results of a GLM fit to binomial or Bernoulli data are nearly useless (Figure 11.4, left). The patternless residuals we idealise for normally distributed data do not exist in this case. Whereas studentised and deviance residuals at least take into account the expected variance-mean relationship, the obvious and strong pattern is not helpful for diagnosis of non-linear relationships or overdispersion.

Then, around 2015, Florian Hartig developed and coded a simulation-based routine for model diagnostics (explained in detail in §8.4 in Chapter 8). His `DHARMa` package immediately became the de facto standard for GLM diagnostics in `R` and is applicable not only to classical (frequentist) GLMs but also to machine-learning and Bayesian models. The idea behind `DHARMa` had been around for decades (Pierce & Schafer 1986), but it had not been implemented in widely used software (Hartig 2022). Because the output is fully in line with model diagnostic plots for normal data, it is intuitive to use (Code Box 11.4) and understand (Figure 11.4, centre and right).

Code Box 11.4: DHARMa-based analysis of residuals from a binomial GLM

```
# Load required library
 library(DHARMa)

# Read in data (local copy)
 bolger <- read.delim("../data/bolger.txt")

# Fit the model
fm <- glm(RODENTSP ~  PERSHRUB, family=binomial, data=bolger)

# Standard residual plot (Fig. 11.4, Left)
 plot(predict(fm), rstudent(fm), las=1, pch=16, xlab="link-scale prediction",
      ylab="studentised residuals")
 abline(h=0, col="grey")

# DHARMa residual plots (Fig. 11.4, Center and Right)
 sim <- simulateResiduals(fm)
 plot(sim)
```

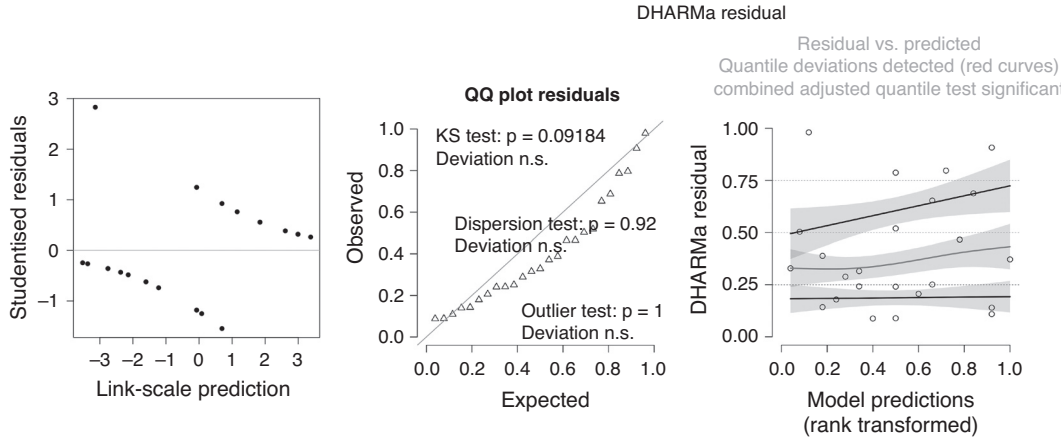


Figure 11.4 Model diagnostic plot for a binomial GLM fit to presence/absence of a rodent as a function of habitat area and distance to the nearest other habitat patch. *Left*: Studentised residuals as function of link-scale fitted values. *Centre* and *Right*: quantile-quantile and residual plots resulting from DHARMa-based diagnostics. Although no one would argue that this model fits well, at least the residuals plots are interpretable. Data from Bolger et al. (1997).

In a similar way, two elements in `brms` (Bürkner 2017) led to its success and accessibility by Bayesians and non-Bayesians alike. First, it provides an easy-to-use interface to the state-of-the-art Bayesian modelling software Stan (Stan Development Team 2023). Second, it also produces outputs and plots that are familiar and similar to a standard GLM output.[8]

[8] Although we are unable to recover the original source of the quote that "Paul Bürkner's `brms` changed the uptake of Stan single-handedly more than all our activities in the past years," we think it was probably written or spoken by Andrew Gelman. If you know for sure, or know differently, please let us know!

Similar examples are the intuitive measure of heterogeneity (I^2) used widely in meta-analysis (Higgins & Thompson 2002; Higgins et al. 2003, but see Borenstein et al. 2017); the MCMC convergence diagnostic $\hat{R}$ (Gelman et al. 2013, which goes back to an idea informally communicated at Bell Labs in 1990: Gelman 2019); and Spiegelhalter et al.'s (2002) DIC as the Bayesian equivalent to the omnipresent AIC. Other intuitive methodological concepts and ideas that were taken up rapidly but are much more controversial include: the ecological "footprint" of a person or product (e.g. Rees 1992), which has been criticised for being limited to a single object (e.g., the water, CO_2, or land footprint) and for making consumers the culprit of problems generated by industry (Turner 2014); and the metabolic scaling theory of ecology (West et al. 1997), which is now used nearly universally in ecosystem science but suffers from the "minor" problem of being physically, biologically, and evolutionarily implausible (Kozlowski & Konarzewski 2004; Apol et al. 2008).

11.6 Model-agnostic number of parameters: Generalised degrees of freedom

Our final example in this chapter illustrates in detail how a new method can be made accessible through linking it to existing methods and by generating an intuitive understanding of how the method works and what its output means.

The paradigmatic use of information-theoretic approaches, particularly Akaike's AIC for model comparison, selection, and averaging ($\text{AIC} = -2\cdot$ log-likelihood $+2\cdot p$, where p is the number of model parameters), requires a quantification of a model's complexity. For (generalised) linear models, model complexity is simply the number of parameters in the model. But for semi-parametric models, such as generalised additive models (GAMs) or machine-learning models, including random forests and neural networks, the number of parameters fitted is a poor measure of model complexity.[9]

One approach to quantifying complexity in semi-parametric models that is analogous to the number of parameters in a linear model is the idea of generalised degrees of freedom ("GDF": Efron 1986; Ye 1998; Elder 2003). The GDF overcomes the main problems of effective degrees of freedom used in GAMs that were identified by Janson et al. (2015). GDFs also are used implicitly in the computation of model complexity in Bayesian MCMCs and the definition of the DIC (Gelman et al. 2013).

Intuitively, we deem a model to be richer in parameters if it responds more strongly to noise added to the response variable (data perturbation; see §2.5 in Chapter 2). Fits of a parameter-poor, stiff model will hardly be affected by such perturbations, whereas a parameter-rich model will fit to the noise and thus have fitted values altered by perturbing the response.

The mathematics of GDFs (following the presentation in Ye 1998) start with the so-called "hat" or projection matrix $H = X(X'X)^{-1}X'$, which relates the observations $\{Y\}$ of a random variable y to the fitted values $\hat{Y}$: $\hat{Y} = HY$. The hat matrix H thus represents the influence that each Y_i value has on the fitted value. The sum of the partial derivatives of the diagonal of this matrix (i.e., only the influences of Y_i on $\hat{Y}_i$ but not those on $\hat{Y}_{j\neq i}$) defines the degrees of

[9] The intuitive reasoning for why the number of fitted parameters is a poor measure of complexity for semi-parametric models is that non-linear functions are fitted and parameter estimates are highly correlated in most such models. Thus, adding another parameter does not increase model complexity by one degree of freedom, but only by a fraction of 1. Further, when multiple models are averaged, as in a random forest, the effective variability of the total model is reduced. Similarly, the decay rate of the back-propagation in neural networks determines the actual stiffness of the resulting fit, and hence the effective complexity.

freedom in a GLM, p:

$$p = \sum_i h_{ii} = \sum_i \frac{\partial \hat{Y}}{\partial Y} \tag{11.6}$$

Accordingly, we define the GDF, D, of a model M as

$$D(M) = \sum_{i=1}^{N} h_i^M(Y) \tag{11.7}$$

with

$$h_i^M(Y) = \frac{\partial E_{\hat{Y}}(\hat{Y}_i(Y))}{\partial Y_i} = \lim_{\delta \to 0} E_{\hat{Y}} \left(\frac{\hat{Y}_i(Y + \delta\varepsilon_i) - \hat{Y}_i(Y)}{\delta} \right), \tag{11.8}$$

where $\hat{Y}_i = \hat{Y}_i(Y)$ is supposed to make explicit that the fitted values $\hat{Y}$ depend on the observed Y. In words, adding noise ε to the observed Y will yield fitted values that differ from the unperturbed fitted values proportional to the GDF.

We can continue with this equation and express it in more statistical terms (Efron 1986), such that

$$h_i^M = \frac{1}{\sigma^2} \mathrm{cov}(\hat{Y}_i(Y), \hat{Y}_i - Y_i). \tag{11.9}$$

In other words, the (normalised) covariance of fitted and fitted-perturbed values is an estimate for the diagonal entries of the hat matrix and hence an estimate of the GDF.

Algorithmically, we start with a fitted model, extract its fitted values, and re-fit it with noise added to the original response. We do this repeatedly ($N \gg 100$ times) and save both perturbed values ($Y + \delta\varepsilon_i$) and fitted-perturbed values ($\hat{Y}' = \hat{Y}_i(Y + \delta\varepsilon_i)$; Equation 11.8) in an array. Finally, we regress all N fitted-perturbed on the fitted values of each datum, yielding estimates of h_i^M, which are then summed to yield our estimate of D.[10] Instead of pseudo-code, the R code for this procedure is presented in Code Box 11.5.

11.6.1 Testing the GDF approach

Demonstrating that this approach works turns out to be difficult. We have no analytical way of computing the effective number of parameters, in, for example, a neural-network or a random-forest model. All we can do is fit GLMs of different complexity and compare their rank (i.e., the number of parameters in a GLM)[11] to the GDF estimate. Although data simulation is not strictly necessary to accomplish this comparison, it does allow us to set parameters equal to zero, thereby enabling us to distinguish between parameters fit in the model and parameters used to generate the data. It is the former that are counted as the number of parameters for computing the AIC.

In this example, we simulate data and fit models that include spurious predictors. We also fit machine-learning models and quantify their GDFs. We use the results of the simulations to compute the AICs of these models and compare them to one another. The basic steps of each simulation and its analysis are:

1. Simulate data from predictors x_1–x_3 and spurious effects x_4–x_6 (Code Box 11.6);
2. Fit a GLM to the data (responses as a function of predictors and spurious effects) and compute its GDF (Code Box 11.7);

[10] We could have opted to estimate D for each perturbation and then average. However, the approach of first estimating h_i^M is more stable (Elder 2003).

[11] The rank of a GLM fit in R is stored in the variable `rank` in the model summary.

Code Box 11.5: Algorithm for computing GDF

```
# Create a function to compute the GDF (following Elder 2003)
 gdf.lm <- function(object, N=200, noise=0.1){
   yextracted <- model.frame(object)[,1]
   n <- length(yextracted)
   preds.prime <- replicate(N, {
     LHS <- all.vars(object$call)[1]
     RHS <- paste(names(coef(object)[-1]), collapse = ' + ')
     yprime <- yextracted + rnorm(n, sd=noise*sd(yextracted))
     newfm <- update(object, yprime ~ ., data=cbind.data.frame("yprime"=yprime,
         object$model))
     cbind(yprime, "yhat"=fitted.values(newfm))
     }
   )
  # Output to an array of n x 2 x N
   elasticities <- apply(preds.prime, 1, function(x) coef(lm(x[2,] ~ x[1,])))
   c("GDF"=unname(rowSums(elasticities)[2]))
 }
```

Code Box 11.6: GDF Step 1: Simulated dataset

```
 set.seed(123)
# Actual predictors

 x1 <- 0:100
 x2 <- sample(0:100)
 x3 <- x2 + rnorm(101, sd=20) # x3 is highly correlated with x2
 cor(x2, x3)

| [1] 0.8026195

# Spurious predictors variables
 x4 <- sample(0:100)
 x5 <- sample(0:100)
 x6 <- sample(0:100)

# Response variable
 y <- rnorm(101, mean=2 + 2*x1 -0.5*x2 + 1*x3, sd=30)

# Create the data frame for the model
 dats <- cbind.data.frame(y, x1, x2, x3, x4, x5, x6)
```

Code Box 11.7: GDF Step 2: Fit GLM and compute its GDF

```
# This code uses the predictor and response variables simulated in Code Box
   11.5

# Fit and summarise the GLM
 summary(fm <- lm(y ~ x1 + x2 + x3 + x4 + x5 + x6, data=dats))

| <... snip ...>
|
| Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept)  9.15528   12.90504   0.709   0.4798
| x1           2.13028    0.10535  20.221  < 2e-16 ***
| x2          -0.32496    0.16979  -1.914   0.0587 .
| x3           0.66549    0.15381   4.327 3.77e-05 ***
| x4           0.02428    0.10580   0.229   0.8190
| x5           0.01913    0.10182   0.188   0.8514
| x6          -0.14334    0.10129  -1.415   0.1603
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 29.43 on 94 degrees of freedom
| Multiple R-squared:  0.8341,  Adjusted R-squared:  0.8235
| F-statistic: 78.75 on 6 and 94 DF,  p-value: < 2.2e-16

#Run the GDF function on the GLM
 set.seed(123)
 gdf.lm(fm)

|     GDF
| 6.95014
```

3. Fit two GAMs with different spline types to the data and compute their GDFs (Code Box 11.8);
4. Fit a random forest model to the data and compute its GDF (Code Box 11.9);
5. Fit two simple neural network models, each with a different number of nodes, and compute their GDFs (Code Box 11.10);
6. For all models, compute their fits using log-likelihoods and from them and their GDFs, the associated AICs (Code Box 11.11).

The GDF value returned in Code Box 11.7 is approximately correct. There are seven parameters being fitted by the GLM: the six slope estimates and the intercept (i.e., the rank of this GLM [in the object `fm$rank`] equals 7).

The GAM routine in R also estimates its degrees of freedom but the values are different from the GDF estimates (Code Box 11.8). Since the GAM-estimated degrees of freedom also are only an approximation, this does not mean that the GDF computations are wrong. Indeed, there is another GAM-based estimate, called the "reference degrees of freedom,"

Code Box 11.8: Step 3: Fit two GAMs and compute their GDFs

```
# This code uses the predictor and response variables simulated in Code Box 11.6
      and the gdf.lm function created in Code Box 11.5

# Load required libraries
 library(mgcv)

# Fit two GAMs
 fgam1 <- gam(y ~ s(x1) + s(x2) + s(x3) + s(x4) + s(x5) + s(x6))
 fgam2 <- gam(y ~ s(x1, bs="cs") + s(x2, bs="cs") + s(x3, bs="cs") + s(x4,
      bs="cs") + s(x5, bs="cs") + s(x6, bs="cs"))

 fgam1 # The first GAM

| <... snip ...>
| Estimated degrees of freedom:
| 1.00 1.00 1.00 1.00 2.58 4.64  total = 12.21

 fgam2 # The second GAM

| <... snip ...>
| Estimated degrees of freedom:
| 1.92 0.90 1.30 0.00 2.63 4.62  total = 12.38

 gdf.lm(fgam1) # The first GAM

|    GDF
| 15.55471

 gdf.lm(fgam2) # The second GAM

|     GDF
| 17.73244
```

which for our two models are 13 and 54, respectively. These varying results highlight the fact that estimating degrees of freedom for semi-parametric models is a non-trivial task (Wood 2017).

We fit a random-forest model using the fast C++-implementation in the `ranger` library of R (Code Box 11.9; Wright & Ziegler 2017).

For our last model (Code Box 11.10), we fit two simple old-but-fast single-hidden-layer neural-network models (Venables & Ripley 2002). The key tuning parameters are the number of nodes in the hidden layer and the decay rate (of the back-propagation optimisation).

These different models (Code Boxes 11.7–11.10) result in a very clear sequence of model fits; the most complex model (the neural network with 30 nodes) clearly outperforms all the other approaches (Code Box 11.11). Note that this assessment is based on model fit, not on

Code Box 11.9: GDF Step 4: Fit a random forest model and compute its GDF

```
# This code uses the predictor and response variables simulated in Code Box 11.6

# Load required libraries
 library(ranger)

# First, modify function gdf.lm from Code Box 11.5 to work with ranger
 gdf.rf <- function(object, N=50, noise=0.1, data, type="response"){
  # Data are NOT stored in the object (by default), thus have to be handed over
      as an argument;
  # Add some noise to the y and re-run original model, N times, storing both
      perturbed y (=yprime) and fitted yhat:
   split1 <- strsplit(as.character(object$call), "-", fixed=T)
   split2 <- strsplit(split1[[2]], " ~ ", fixed=TRUE)
   RHS <- split2[[1]][2]
   yextracted <- data[, split2[[1]][1]]

   n <- length(yextracted)
   preds.prime <- replicate(N, {
    yprime <- yextracted + rnorm(n, sd=noise*sd(yextracted))
    dats <- cbind.data.frame("yprime"=yprime, data)
    newfrf <- ranger(as.formula(paste("yprime ~ ", RHS)), data=dats)
    fitted.values.rf <- predict(newfrf, data=dats, type=type)
    cbind(yprime, "yhat"=fitted.values.rf$predictions)
   }) # an array of n x 2 x N

   elasticities <- apply(preds.prime, 1, function(x) coef(lm(x[2,] ~ x[1,])))
   c("GDF"=unname(rowSums(elasticities)[2])) # GDF!
 }

# Fit the random forest model
 fra <- ranger(y ~ x1 + x2 + x3 + x4 + x5 + x6, data=dats)

#Estimate the generalised degrees of freedom
 gdf.rf(fra, data=dats)

|     GDF
| 55.10918
```

its predictive performance on independent data. The point here was to demonstrate that an AIC can be computed when using GDFs.

11.6.2 Is this approach similar to other approaches?

Two (related) alternatives have been proposed to quantify the effective number of parameters in a model. The first is the log-likelihood loss on cross-validation, i.e., $D_\ell = \ell_{\text{train}} - \overline{\ell_{\text{test}}}$ (Hauenstein et al. 2018). This is a similarly intuitive measure as it basically says: the more

Code Box 11.10: GDF Step 5: Fit two neural network models and compute their GDFs

```
# This code uses the predictor and response variables simulated in Code Box 11.6

# Load required library
 library(nnet)

# First, modify function gdf.lm from Code Box 11.5 to work with nnet
gdf.nnet <- function(object, N=200, noise=0.1, data){
  # computes GDF for a nnet neural network

  # some sanity checks:
   if (!any(class(object) %in% "nnet")) stop("This function currently only
       handles nnet objects!")
   if (!object$call$linout) stop("This approach to GDF only makes sense for
       regressions. Since linout=F, this is not the case here.")
   if (object$convergence != 0) stop("Seems like this neural net did not
       converge!")

   yextracted <- model.frame(object)[,1]
   n <- length(yextracted)
   preds.prime <- replicate(N, {
    LHS <- all.vars(object$call)[1]
    RHS <- strsplit(as.character(object$call$formula), "~", fixed=T)[[3]]
    yprime <- yextracted + rnorm(n, sd=noise*sd(yextracted))
    dats <- cbind.data.frame("yprime"=yprime, data)
    newfnn <- update(object, yprime ~ ., data=dats, trace=F)
    cbind(yprime, "yhat"=fitted.values(newfnn))
   }
   ) # an array of n x 2 x N
   elasticities <- apply(preds.prime, 1, function(x) coef(lm(x[2,] ~ x[1,])))
   c("GDF"=unname(rowSums(elasticities)[2])) # GDF!
 }

# Fit the models
 set.seed(123)
 fnn1 <- nnet(y ~ x1 + x2 + x3 + x4 + x5 + x6, data=dats,
  linout=TRUE, size=7, decay=.1, maxit=1E5, trace=F)
 fnn2 <- nnet(y ~ x1 + x2 + x3 + x4 + x5 + x6, data=dats,
  linout=TRUE, size=30, decay=.1, maxit=1E5, trace=F)

# Compute their GDFs
 (gdf.fnn1 <- gdf.nnet(fnn1, data=dats)) # Model 1

|     GDF
| 51.75229

 (gdf.fnn2 <- gdf.nnet(fnn2, data=dats)) # Model 2

|     GDF
| 99.29928
```

Code Box 11.11: Step 6: Compute AICs for all models in Code Boxes 11.7–11.10

```
# This code uses the GDF functions and outputs from models fit in Code Boxes
    11.5-11.10

# First, we write a function to compute the log likelihoods
 ell <- function(obs, preds){
  sum(dnorm(obs, mean=preds, sd=sd(obs-preds), log=T))
 }

# Next, compute AICs for each

 -2*ell(dats$y, predict(fm)) + 2*gdf.lm(fm) # GLM

| 976.5711

 -2*ell(dats$y, predict(fgam1)) +2*gdf.lm(fgam1) # First GAM

| 980.2936

 -2*ell(dats$y, predict(fgam2)) +2*gdf.lm(fgam2) # Second GAM

| 982.823

 -2*ell(dats$y, predict(fra, data=dats)$prediction) +2*gdf.rf(fra, data=dats) #
     Random forest

| 970.4631

 -2*ell(dats$y, predict(fnn1)) +2*gdf.fnn1  # First Neural Network

| 1026.828

 -2*ell(dats$y, predict(fnn2)) +2*gdf.fnn2  # Second Neural Network

| 418.0308
```

a model overfits, the greater the number of effective parameters it has. Overfitting is then quantified as the difference between fit and cross-validation match. Although this measure of effective number of parameters (and hence model complexity) has been mentioned diffusely in various publications (including Hastie et al. 2009; Wood 2015), it has not yet gained traction in the machine- or deep-learning communities.[12] What is computationally relevant is that k-fold cross-validation only requires k runs, whereas a stable estimate of GDF may

[12] Instead they define other measures of model complexity that create logically flawed and confusing artefacts such as the "double-descent" of Belkin et al. (2019). See Pichler & Hartig (2023) for further discussion.

easily require several thousand. Thus, computing the mean cross-validation log-likelihood $\bar{\ell}_{\text{test}}$ is computationally much less intensive than computing the GDF. Even if repeated cross-validation would be necessary to stabilise the estimate of D_ℓ, it is not clear what GDF has to additionally offer.

The second approach similar to GDF is used frequently in Bayesian statistics to quantify the number of parameters in the model. When Spiegelhalter et al. (2002) introduced the DIC into Bayesian statistics, they needed a stand-in estimate for the number of parameters in the comparable AIC. Spiegelhalter et al. (2002) call this value pD and estimate it based on the deviances computed during the MCMC fitting of the model. Let d_i be the deviance of model M for parameter set θ_i in the MCMC sequence. Then the effective number of parameters $pD = \overline{d(\theta)} - d(\bar{\theta})$ (Spiegelhalter et al. 2002). Celeux et al. (2006) provides some other ways this pD can be computed, of which one gained most traction: $pD = \frac{1}{2}\text{var}(d(\theta))$ (Gelman et al. 2013). Intuitively, this measure means that the more stable the deviance of a model is, the fewer effective parameters it has. In a sense, this is the opposite of the perturbation approach of the GDF: instead of perturbing y and evaluating changes in fit, pD disturbs the model parameters and evaluates variance in fit. Since the MCMC chains typically will run for hundreds to hundreds of thousands of iterations, pD is expensive to compute; but it is available at no additional cost when MCMC is used anyway.

11.6.3 Conclusion

The perturbation approach of GDF offers an intuitive approach to quantifying the effective number of parameters in a model, and thereby its complexity. It works, in the sense that for known problems (GLM) the estimates are what we would expect.[13] Why, then, has it not made it into mainstream machine-learning (cf. Zhang et al. 2012)?

For one thing, the approach is very computer intensive, requiring thousands of perturbations to yield stable estimates of D. For another, although the approach demonstrated here that is used for normally distributed data works well, the situation is different for categorical responses. In that case, a perturbation is a mis-classification, and every such mis-classification not only changes the values, it also changes the mean, which affects the model estimates. Hence, for binary data the GDF is less reliable (Hauenstein et al. 2018).

Most importantly, however, is the apparent lack of improvement on the existing cross-validation-based estimate D_ℓ and the MCMC-based pD. Both are also variable and require expensive computation. So GDF is just another one of those, with no obvious improvement in computation or interpretation.

11.7 Know your limits

Finally, a key piece of information is commonly, and regrettably, missing in publications that introduce new methods: what are the limits of their applicability? When should we not use them? There are bad and good reasons why this information is often missing. The bad reason is that revealing the limits of a method may lead to the perception that the method is restricted to special cases or is not as good as it is claimed to be. On the other hand, at least one good reason for revealing limits is that we cannot know all the limits a method

[13] On mathematical grounds, Janson et al. (2015) fundamentally questioned the utility of Efron's 1986 "effective" degrees of freedom (Equation 11.9). They showed that the estimated value of D can be larger than the number of parameters in the model, for example, in a best-subset regression. Their argument was mathematically rather involved and, to us, inaccessible. Their simulation code does not seem to wrap the best-subset regression into the GDF computation, however, and does not convince us as a strong argument against the idea of GDF. Still, it illustrates that there may be mathematical problems with GDFs.

may or may not have. Still, the limits we do know about should be clearly communicated. For example, do not use the χ^2-test when any expected value is < 5. Do not use ANOVA when groups are heteroscedastic. Do not use the variance-to-mean ratio as a measure of randomness (Hurlbert 1990). Threshold-based indices may be inappropriate for predominantly smoothly-changing natural phenomena (including effects of alcohol consumption and ecosystem transitions: Schröder et al. 2005; Wood et al. 2018; Hillebrand et al. 2020). Variance inflation factors are great as guidance for collinearity problems, but they also can mislead (O'Brien 2007).

As we discussed extensively in Chapter 6, not all folkloric limits to a method are valid. It thus falls onto the creator of a new method to determine and judge whether communicating its limits is prudent, could unnecessarily stifle its use, or prevent its inappropriate use. *Caveat venditor!*

11.8 Summary

- The key steps that are required to convince ourselves and our colleagues of a new statistical or data-analytic method include:
  - Unit testing, including both code verification and functional verification.
  - Dimensional analysis.
  - Demonstration of improvement over existing methods, especially in terms of accuracy, performance with smaller sample sizes, or computational efficiency.
  - Substantiated assertion that the new method could contribute to the generation of new scientific knowledge.
  - A comprehensible, often intuitive, description of the new method.
- Whenever possible, new statistical methods ought to be described verbally, mathematically, and algorithmically, and clearly commented code should always accompany its descriptions.
- The best descriptions of a new method includes an assessment of the limits of its applicability.

References

Apol, M. E. F., Etienne, R. S., & Olff, H. (2008). Revisiting the evolutionary origin of allometric metabolic scaling in biology. *Functional Ecology*, 22(6), 1070–1080. doi: 10.1111/j.1365-2435.2008.01458.x.

Barvinok, A. (2010). On the number of matrices and a random matrix with prescribed row and column sums and 0–1 entries. *Advances in Mathematics*, 224(1), 316–339. doi: 10.1016/j.aim.2009.12.001.

Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias-variance trade-off. *Proceedings of the National Academy of Sciences, USA*, 116(32), 15849–15854. doi: 10.1073/pnas.1903070116.

Belmont, J., Miller, C., Scott, M., & Wilkie, C. (2022). A new statistical approach for identifying rare species under imperfect detection. *Diversity and Distributions*, 28(5), 882–893. doi: 10.1111/ddi.13495.

Binder, H., Sauerbrei, W., & Royston, P. (2013). Comparison between splines and fractional polynomials for multivariable model building with continuous covariates: A simulation study with continuous response. *Statistics in Medicine*, 32(13), 2262–2277. doi: 10.1002/sim.5639.

Bolger, D. T., Alberts, A. C., Sauvajot, R. M., et al. (1997). Response of rodents to habitat fragmentation on coastal Southern California. *Ecological Applications*, 7(2), 552–563. doi: 10.1890/1051-0761(1997)007[0552:RORTHF]2.0.CO;2.

Borenstein, M., Higgins, J. P. T., Hedges, L. V., & Rothstein, H. R. (2017). Basics of meta-analysis: I^2 is not an absolute measure of heterogeneity. *Research Synthesis Methods*, 8(1), 5–18. doi: 10.1002/jrsm.1230.

Borji, A., Cheng, M.-M., Jiang, H., & Li, J. (2015). Salient object detection: A benchmark. *IEEE Transactions on Image Processing*, 24(12), 5706–5722. doi: 10.1109/TIP.2015.2487833.

Boulesteix, A.-L., Wilson, R., & Hapfelmeier, A. (2017). Towards evidence-based computational statistics: Lessons from clinical research on the role and design of real-data benchmark studies. *BMC Medical Research Methodology*, 17(1), 138. doi: 10.1186/s12874-017-0417-2.

Bried, J. T. & Vilmi, A. (2022). Improved detection of mass effect species assembly for applied metacommunity thinking. *Journal of Applied Ecology*, 59(4), 921–926. doi: 10.1111/1365-2664.14115.

Bürkner, P.-C. (2017). brms: An R package for Bayesian multilevel models using Stan. *Journal of Statistical Software*, 80(1), 1–28. doi: 10.18637/jss.v080.i01.

Carstens, C. J., Berger, A., & Strona, G. (2018). A unifying framework for fast randomization of ecological networks with fixed (node) degrees. *MethodsX*, 5, 773–780. doi: 10.1016/j.mex.2018.06.018.

Celeux, G., Forbes, F., Robert, C. P., & Titterington, D. M. (2006). Deviance information criteria for missing data models. *Bayesian Analysis*, 1(4), 651–673. doi: 10.1214/06-BA122.

Chakradhar, S. (2019). It's just in mice! This scientist is calling out hype in science reporting. https://www.statnews.com/2019/04/15/in-mice-twitter-account-hype-science-reporting/. Accessed 25 December 2023.

Chao, A., Chiu, C.-H., & Jost, L. (2014). Unifying species diversity, phylogenetic diversity, functional diversity and related similarity/differentiation measures through Hill numbers. *Annual Review of Ecology, Evolution, and Systematics*, 45, 297–324. doi: 10.1146/annurev-ecolsys-120213-091540.

Chung, Y., Rabe-Hesketh, S., Dorie, V., et al. (2013). A nondegenerate penalized likelihood estimator for variance parameters in multilevel models. *Psychometrika*, 78(4), 685–709. doi: 10.1007/s11336-013-9328-2.

Couto, M., Pereira, R., Ribeiro, F., et al. (2017). SBLP '17: Towards a green ranking for programming languages. In *Proceedings of the 21st Brazilian Symposium on Programming Languages* (pp. 1–8). New York: Association for Computing Machinery.

Daly, C. (1959). Air pollution and causes of death. *British Journal of Preventive & Social Medicine*, 13(1), 14–27. doi: 10.1136/jech.13.1.14.

Dormann, C. & Ellison, A. M. (2024). Data ex machina. https://osf.io/3skv4/.

Dormann, C. F., Calabrese, J. M., Guillera-Arroita, G., et al. (2018). Model averaging in ecology: A Review of Bayesian, information-theoretic and tactical approaches for predictive inference. *Ecological Monographs*, 88(4), 485–504. doi: 10.1002/ecm.1309.

Drake, J. M. (2023). How to publish a 'Method' article in Ecology Letters. *Ecology Letters*, 26(10), 1645–1646. doi: 10.1111/ele.14304.

Efron, B. (1986). How biased is the apparent error rate of a prediction rule? *Journal of the American Statistical Association*, 81(394), 461–470. doi: 10.1080/01621459.1986.10478291.

Elder, J. F. (2003). The generalization paradox of ensembles. *Journal of Computational and Graphical Statistics*, 12(4), 853–864. doi: 10.1198/1061860032733.

Ellison, A. M. (2023). *Plus ça change, plus c'est la même chose*: On our quattuordecennial, a good Methods paper still is not about my friend the dolphin. *Methods in Ecology and Evolution*, 14(12), 2904–2906. doi: 10.1111/2041-210X.14232.

Elston, D. A., Illius, A. W., & Gordon, I. J. (1996). Assessment of preferences among a range of options using log ratio analysis. *Ecology*, 77(8), 2538–2548. doi: 10.2307/2265752.

Fawcett, T. W. & Higginson, A. D. (2012). Heavy use of equations impedes communication among biologists. *Proceedings of the National Academy of Sciences, USA*, 109(29), 11735–117399. doi: 10.1073/pnas.1205259109.

Fay, R., Martin, J., & Plard, F. (2022). Distinguishing within- from between-individual effects: How to use the within-individual centring method for quadratic patterns. *Journal of Animal Ecology*, 91(1), 8–19. doi: 10.1111/1365-2656.13606.

Ford, E. D. (2000). *Scientific Method for Ecological Research*. Cambridge, UK: Cambridge University Press.

Freund, Y. & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1), 119–139. doi: 10.1006/jcss.1997.1504.

Friedman, J. H., Hastie, T., & Tibshirani, R. J. (2000). Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 28(2), 337–407. doi: 10.1214/aos/1016218223.

GBD 2015 Risk Factors Collaborators (2016). Global, regional, and national comparative risk assessment of 79 behavioural, environmental and occupational, and metabolic risks or clusters of risks, 1990–2015: A systematic analysis for the Global Burden of Disease Study 2015. *The Lancet*, 388(10053), 1659–1724. doi: 10.1016/S0140-6736(16)31679-8.

Gelman, A. (2019). The first version of my "inference from iterative simulation using parallel sequences" paper! https://statmodeling.stat.columbia.edu/2012/05/09/the-first-version-of-my-inference-from-iterative-simulation-using-parallel-sequences-paper/. Accessed 25 December 2023.

Gelman, A., Carlin, J., Stern, H., et al. (2013). *Bayesian Data Analysis*. New York, NY, USA: Chapman & Hall/CRC.

Gibbings, J. (2011). *Dimensional Analysis*. London, UK: Springer.

Hartig, F. (2022). DHARMa: Residual diagnostics for hierarchical (multi-level/mixed) regression models, version 0.4.5. https://cran.r-project.org/package=DHARMa.

Hastie, T., Tibshirani, R., & Wainwright, M. (2020). *Statistical Learning with Sparsity*. New York: Routledge.

Hastie, T., Tibshirani, R. J., & Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edition. Berlin, Germany: Springer.

Hauenstein, S., Dormann, C. F., & Wood, S. N. (2018). Computing AIC for black-box models using generalised degrees of freedom: A comparison with cross-validation. *Communications in Statistics—Simulation and Computation*, 47(5), 1382–1396. doi: 10.1080/03610918.2017.1315728.

Higgins, J. P. T. & Thompson, S. G. (2002). Quantifying heterogeneity in a meta-analysis. *Statistics in Medicine*, 21(11), 1539–1558. doi: 10.1002/sim.1186.

Higgins, J. P. T., Thompson, S. G., Deeks, J. J., & Altman, D. G. (2003). Measuring inconsistency in meta-analyses. *British Medical Journal*, 327(7414), 557–560. doi: 10.1136/bmj.327.7414.557.

Hillebrand, H., Donohue, I., Harpole, W. S., et al. (2020). Thresholds for ecological responses to global change do not emerge from empirical data. *Nature Ecology & Evolution*, 4(11), 1502–1509. doi: 10.1038/s41559-020-1256-9.

Hopf, K. & Reifenrath, S. (2021). Filter methods for feature selection in supervised machine learning applications—Review and benchmark. *arXiv*, 2111.12140, 1–38. doi: 10.48550/arXiv.2111.12140.

Hothorn, T., Leisch, F., Zeileis, A., & Hornik, K. (2005). The design and analysis of benchmark experiments. *Journal of Computational and Graphical Statistics*, 14(3), 675–699. doi: 10.1198/106186005X59630.

Houlahan, J. E., Mckinney, S. T., Anderson, T. M., & McGill, B. J. (2017). The priority of prediction in ecological understanding. *Oikos*, 126(1), 1–7. doi: 10.1111/oik.03726.

Hurlbert, S. H. (1990). Spatial distribution of the montane unicorn. *Oikos*, 58(3), 257–271. doi: 10.2307/3545216.

Janson, L., Fithian, W., & Hastie, T. J. (2015). Effective degrees of freedom: A flawed metaphor. *Biometrika*, 102(2), 479–485. doi: 10.1093/biomet/asv019.

John, B., Raiturkar, P., Le Meur, O., & Jain, E. (2019). A benchmark of four methods for generating 360° saliency maps from eye tracking data. *International Journal of Semantic Computing*, 13(3), 329–341. doi: 10.1142/S1793351X19400142.

Jones, M. T. (2017). Recurrent neural networks deep dive: Build your own RNN. https://developer.ibm.com/learningpaths/supervised-deep-learning/recurrent-neural-networks/introduction-recurrent-neural-networks/. Accessed 25 December 2023.

Jost, L. (2006). Entropy and diversity. *Oikos*, 113(2), 363–375. doi: 10.1111/j.2006.0030-1299.14714.x.

Korthauer, K., Kimes, P. K., Duvallet, C., et al. (2019). A practical guide to methods controlling false discoveries in computational biology. *Genome Biology*, 20(1), 118. doi: 10.1186/s13059-019-1716-1.

Kozlowski, J. & Konarzewski, M. (2004). Is West, Brown and Enquist's model of allometric scaling mathematically correct and biologically relevant? *Functional Ecology*, 18(2), 283–289. doi: 10.1111/j.0269-8463.2004.00830.x.

Ladyman, J. (2001). *Understanding Philosophy of Science*. London: Routledge.

Lee, K.-J. & Skadron, K. (2005). Using performance counters for runtime temperature sensing in high-performance processors. In *19th IEEE International Parallel and Distributed Processing Symposium* (pp. 8 pages).

Legendre, P. (2022). lmodel2: Model II regression, version 1.7-3. https://cran.r-project.org/package=lmodel2.

Lelieveld, J., Evans, J. S., Fnais, M., et al. (2015). The contribution of outdoor air pollution sources to premature mortality on a global scale. *Nature*, 525(75697569), 367–371. doi: 10.1038/nature15371.

Mao, C. X. & Colwell, R. K. (2005). Estimation of species richness: Mixture models, the role of rare species, and inferential challenges. *Ecology*, 86(5), 1143–1153. doi: 10.1890/04-1078.

McKeon, R. (1966). Philosophy and the development of scientific methods. *Journal of the History of Ideas*, 27(1), 3–22. doi: 10.2307/2708305.

Morris, T. P., White, I. R., & Crowther, M. J. (2019). Using simulation studies to evaluate statistical methods. *Statistics in Medicine*, 38(11), 2074–2102. doi: 10.1002/sim.8086.

Moura, D. (2021). R vs. Python vs. Julia: How easy it is to write efficient code? https://towardsdatascience.com/r-vs-python-vs-julia-90456a2bcbab. Accessed 17 January 2024.

Nakagawa, S., Noble, D. W. A., Lagisz, M., et al. (2023). A robust and readily implementable method for the meta-analysis of response ratios with and without missing standard deviations. *Ecology Letters*, 26(2), 232–244. doi: 10.1111/ele.14144.

Northcutt, C. G., Athalye, A., & Mueller, J. (2021). Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv*, 2103.14749, http://arxiv.org/abs/2103.14749. doi: 10.48550/arXiv.2103.14749.

O'Brien, R. M. (2007). A caution regarding rules of thumb for variance inflation factors. *Quality & Quantity*, 41(5), 673–690. doi: 10.1007/s11135-006-9018-6.

OECD (2015). *Frascati Manual 2015: Guidelines for Collecting and Reporting Data on Research and Experimental Development, The Measurement of Scientific, Technological and Innovation Activities*. Paris, France: OECD Publishing.

Pereira, R., Couto, M., Ribeiro, F., et al. (2021). Ranking programming languages by energy efficiency. *Science of Computer Programming*, 205, 102609. doi: 10.1016/j.scico.2021.102609.

Pichler, M. & Hartig, F. (2023). Machine learning and deep learning—A review for ecologists. *Methods in Ecology and Evolution*, 14(4), 994–1016. doi: 10.1111/2041-210X.14061.

Pierce, D. A. & Schafer, D. W. (1986). Residuals in generalized linear models. *Journal of the American Statistical Association*, 81(396), 977–986. doi: 10.1080/01621459.1986.10478361.

Popper, K. (2002). *The Logic of Scientific Discovery*, 2nd edition. London, UK: Routledge.
Rees, W. E. (1992). Ecological footprints and appropriated carrying capacity: What urban economics leaves out. *Environment and Urbanization*, 4(2), 121–130. doi: 10.1177/095624789200400212.
Reshef, D. N., Reshef, Y. A., Finucane, H. K., et al. (2011). Detecting novel associations in large data sets. *Science*, 334(6062), 1518–1524. doi: 10.1126/science.1205438.
Roswell, M., Dushoff, J., & Winfree, R. (2021). A conceptual guide to measuring species diversity. *Oikos*, 130(3), 321–338. doi: 10.1111/oik.07202.
Rudwick, M. J. S. (2016). Constructive controversy and the growth of knowledge. In A. Blum, K. Gavroglu, C. Joas, & J. Renn (Eds.), *Shifting Paradigms: Thomas S. Kuhn and the History of Science* (pp. 181–189). Berlin, Germany: Edition Open Access and the Max Planck Institute for the History of Science.
Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2), 197–227. doi: 10.1007/BF00116037.
Schneider, D. C. (2009). *Quantitative Ecology: Measurement, Models and Scaling*, 2nd edition. Amsterdam, The Netherlands: Academic Press.
Schröder, A., Persson, L., & De Roos, A. M. (2005). Direct experimental evidence for alternative stable states: A review. *Oikos*, 110(1), 3–19. doi: 10.1111/j.0030-1299.2005.13962.x.
Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3), 379–423. doi: 10.1002/j.1538-7305.1948.tb01338.x.
Simpkins, C. E., Hanÿ, S., Spangenberg, M. C., et al. (2022). spectre: an R package to estimate spatially-explicit community composition using sparse data. *Ecography*, 2022(12), e06272. doi: 10.1111/ecog.06272.
Smith, E. P. & van Belle, G. (1984). Nonparametric estimation of species richness. *Biometrics*, 40(1), 119–129. doi: 10.2307/2530750.
Speich, M., Dormann, C. F., & Hartig, F. (2021). Sequential Monte-Carlo algorithms for Bayesian model calibration—A review and method comparison. *Ecological Modelling*, 455, 109608. doi: 10.1016/j.ecolmodel.2021.109608.
Spellerberg, I. F. & Fedor, P. J. (2003). A tribute to Claude Shannon (1916–2001) and a plea for more rigorous use of species richness, species diversity and the 'Shannon-Wiener' Index. *Global Ecology and Biogeography*, 12(3), 177–179. doi: 10.1046/j.1466-822X.2003.00015.x.
Spiegelhalter, D. J., Best, N. G., Carlin, B. P., & van der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, 64(4), 583–639. doi: 10.1111/1467-9868.00353.
Stan Development Team (2023). *Stan Modeling Language Users Guide and Reference Manual, 2.33*. NumFOCUS, Austin, TX, USA.
Steppan, J. (2017). File:MnistExamples.png. https://commons.wikimedia.org/w/index.php?curid=64810040). Accessed 17 January 2024.
Strona, G., Nappo, D., Boccacci, F., et al. (2014). A fast and unbiased procedure to randomize ecological binary matrices with fixed row and column totals. *Nature Communications*, 5(11), 4114. doi: 10.1038/ncomms5114.
Tibshirani, R. J. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267–288. doi: 10.1111/j.2517-6161.1996.tb02080.x.
Tran, H. T. N., Ang, K. S., Chevrier, M., et al. (2020). A benchmark of batch-effect correction methods for single-cell RNA sequencing data. *Genome Biology*, 21(1), 12. doi: 10.1186/s13059-019-1850-9.
Turner, J. M. (2014). Counting carbon: The politics of carbon footprints and climate governance from the individual to the global. *Global Environmental Politics*, 14(1), 59–78. doi: 10.1162/GLEP_a_00214.
Ugland, K. I., Gray, J. S., & Ellingsen, K. E. (2003). The species-accumulation curve and estimation of species richness. *Journal of Animal Ecology*, 72(5), 888–897. doi: 10.1046/j.1365-2656.2003.00748.x.
Valeriani, A. S. (2020). Runtime performances benchmark for knowledge graph embedding methods. *arXiv*, 2011.04275, 1–10. doi: 10.48550/arXiv.2011.04275.
van Houwelingen, H. C. & Sauerbrei, W. (2013). Cross-validation, shrinkage and variable selection in linear regression revisited. *Open Journal of Statistics*, 3(2), 79–102. doi: 10.4236/ojs.2013.32011.
Venables, W. N. & Ripley, B. D. (2002). *Modern Applied Statistics with S*. New York, NY, USA: Springer, 4th edition.
Vicente-Serrano, S. M., Beguería, S., & López-Moreno, J. I. (2010). A multiscalar drought index sensitive to global warming: The Standardized Precipitation Evapotranspiration Index. *Journal of Climate*, 23(7), 1696–1718. doi: 10.1175/2009JCLI2909.1.
Warton, D. I. & Hui, F. K. C. (2011). The arcsine is asinine: The analysis of proportions in ecology. *Ecology*, 92(1), 3–10. doi: 10.1890/10-0340.1.
West, G. B., Brown, J. H., & Enquist, B. J. (1997). A general model for the origin of allometric scaling laws in biology. *Science*, 276(5309), 122–126. doi: 10.1126/science.276.5309.122.
Wikipedia Contributors (2023a). Propagation of uncertainty—Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Propagation_of_uncertainty&oldid=1174273326. Accessed 25 December 2023.

Wikipedia Contributors (2023b). Recurrent neural network—Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Recurrent_neural_network&oldid=1189029233. Accessed 25 December 2023.

Wood, A. M., Kaptoge, S., Butterworth, A. S., et al. (2018). Risk thresholds for alcohol consumption: Combined analysis of individual-participant data for 599,912 current drinkers in 83 prospective studies. *The Lancet*, 391(10129), 1513–1523. doi: 10.1016/S0140-6736(18)30134-X.

Wood, S. N. (2015). *Core Statistics*. Cambridge, UK: Cambridge University Press.

Wood, S. N. (2017). *Generalized Additive Models: An Introduction with R*, 2nd edition. New York, NY, USA: Chapman & Hall/CRC.

Worley, P. H. & Foster, I. T. (1994). Parallel spectral transform shallow water model: A runtime-tunable parallel benchmark code. In *Proceedings of IEEE Scalable High Performance Computing Conference* (pp. 207–214).

Wright, M. N. & Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1), 1–17. doi: 10.18637/jss.v077.i01.

Yang, J., Lu, J., Chen, Y., et al. (2020). Large underestimation of intraspecific trait variation and its improvements. *Frontiers in Plant Science*, 11, 53. doi: 10.3389/fpls.2020.00053.

Ye, J. (1998). On measuring and correcting the effects of data mining and model selection. *Journal of the American Statistical Association*, 93(1), 120–131.

Zhang, B., Shen, X., & Mumford, S. L. (2012). Generalized degrees of freedom and adaptive model selection in linear mixed-effects models. *Computational Statistics and Data Analysis*, 56(3), 574–586. doi: 10.1016/j.csda.2011.09.001.

Zou, H. & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67, 301–320. doi: 10.1111/j.1467-9868.2005.00503.x.

12

Outroduction: How far should we push simulations?

This book has introduced and advocated for using data simulation as a tool when planning ahead, checking assumptions, interpreting results, and introducing new methods. In virtually all cases, data simulations were very close to the analysis performed. As the sophistication of analysis increases, so does the complexity of data simulation. For costly projects, simulations should be considered that are much more far-reaching than what we have covered.

The sky is not even the limit for such far-reaching simulations. For example, scientists and engineers responsible for any satellite mission will simulate the data the mission expects to collect so that they can develop and test the analytical pipelines before the start of the mission and process the real data as it arrives. Such simulation and testing may take decades before the mission is launched (e.g., the James Webb Space Telescope; see footnote 1 in Chapter 4). Similarly, hundreds of data analysts worked for years before the rocket launches to write routines for the European Space Agency's Sentinel missions.

The target of these and similar data simulations is the interface between theory and experiment. We may expect some developments resulting from the analysis and interpretation of a scientific theory, for example, an increase in inflation following the collapse of a stock market as predicted by macroeconomic principles. However, the measurements of such developments may be difficult or biased, so data simulations are required to explore whether or when the expected outcome would be detectable. Continuing with this example, if the consumer price index were to indicate inflation but consumers had substantially shifted their purchasing habits, the index could be inaccurate and inflation could be misdiagnosed or detected too late. Simulations allow analysts to explore the effect of behavioural changes on inflation detection.

What you will learn from this chapter

- As theory becomes more involved, so do the associated data simulations.

We illustrate this single point with three concrete examples.

12.1 Stochastic weather forecasting

In the early 2010s, the European Centre for Medium-range Weather Forecasting (ECMWF) started to add a new kind of stochasticity into their weather predictions. In addition to the standard uncertainty about initial conditions (which understandably result from not knowing the conditions of temperature, humidity, and wind speed in every modelled voxel),

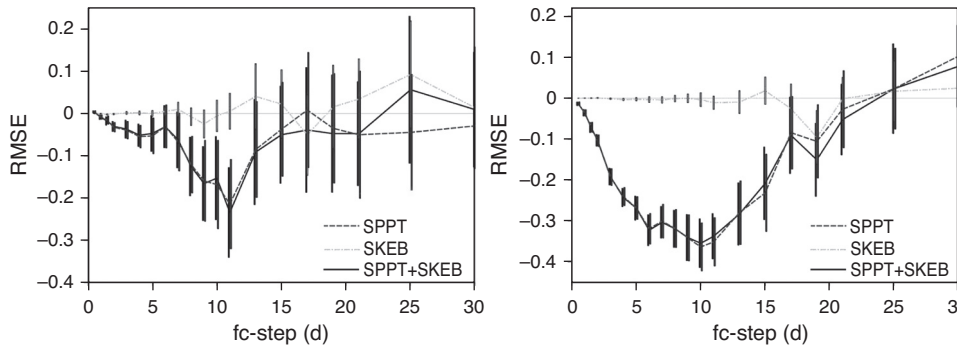


Figure 12.1 Reduction in forecasting error (RMSE) for the northern extratropics (left) and tropics (right) after including two stochastic process schemes (SPPT and SKEB) into the ensemble, relative to ensembles with only initial condition uncertainty (0 line). (Figure modified from Figure 4 of Leutbecher et al. 2017).

the ECMWF started to include in every simulation run additional uncertainty about the model parameters, and thus its structure (Leutbecher et al. 2017).

At first glance, it may seem like a bad strategy to add noise to reduce errors in weather forecasts, but it actually led to the hoped-for better forecasts. Indeed, and in line with intuition, the different model runs really differed more from one another, which led to a wider spread of predictions. However, these different model runs also had better coverage of the true uncertainty (see Figure 2.3 in Chapter 2 for an explanation of coverage). The competing forecasting models were more confident (i.e., had less uncertainty), but their forecasts were biased. The ECMWF's new stochastic-process approach was less biased because it averaged over a more complete set of plausible predictions. The net result was better weather forecasts that have steadily improved as more information becomes available from the combination of successive model runs and more observations.

The improvement has been particularly strong for tropical regions (Figure 12.1). One measure of uncertainty, the root mean square error (RMSE), in the ECMWF forecasts is 2–10% lower than in the deterministic models (i.e., models with uncertainty only in initial conditions). Although the computational requirements are larger for the ECMWF models, the reward is well worth the computational burden: better forecasts with a much more complete propagation of initial and process uncertainty into the prediction. Unsurprisingly, this approach is now being picked up by other weather forecasting institutions (Bengtsson et al. 2019).

12.2 Infusing fake signals to test the workflow at LIGO

Experiments with multi-year run-ups, such as satellite launches and astrophysical measurements, also include in their planning (and funding requests) all data processing and analytical steps. More recently, and especially in the physical sciences, "blind analyses" have become more prominent. In such analyses, the idea is that the analyst does not know whether or not there is a signal in the data (MacCoun & Perlmutter 2015). Every step of the workflow—filtering, transformation, data correction, and computation—is prepared in ignorance of whether there is a signal. In fact, there is a working group in the project responsible for making such "fake data" (i.e., simulated data) to test the pipeline by manipulating the data unbeknownst to the analysts. This is the equivalent to double-blind studies in a medical setting, where neither the administering physician nor the patient knows whether the treatment is the real thing or a placebo.

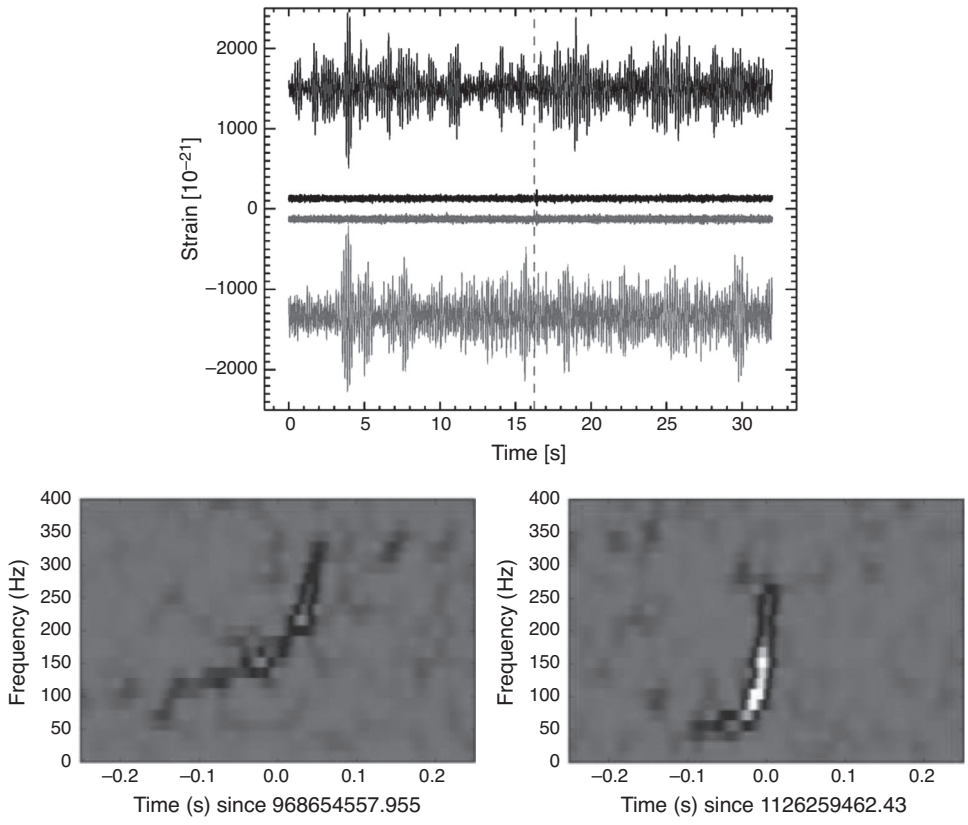


Figure 12.2 *Top*: 32 seconds worth of data from LIGO's two sites (Hanford in black and Livingston in grey). The top and bottom time series are raw data, the middle two time series are the result of data processing using 100-fold amplification. The dashed vertical line in the plot indicates the detected black-hole merger (Image from the arXiv version 2 of Creswell et al. 2017, and used under terms of the arXiv.org—Non-exclusive license to distribute version 1.0.) *Bottom*: Fake (left) and real (right) signal of a gravitational wave. The fake signal was simulated five years before the first detection of such a wave. (Public Domain Image by The LIGO Open Science Center team and the LIGO Scientific Collaboration.)

One such example is the workflow for detecting gravitational waves by the Laser Interferometer Gravitational-Wave Observatory (LIGO),[1] which is based at complementary sites in Hanford, Washington, and Livingston, Louisiana, USA. The signal of gravitational waves is almost unfathomably small (the merger of two black holes causes a spatial distortion of 10^{-18} m, half the diameter of a proton), and virtually any phenomenon occurring anywhere on Earth shakes and moves the detector far more than that in an erratic, non-stationary and non-Gaussian way (Figure 12.2 and Abbott & The LIGO Scientific Collaboration and Virgo Collaboration 2016). Thus, the analytical pipeline for the data collected by LIGO is very sophisticated, and needs to find a very good representation of the noise that can be removed from any (potential) signal.

Will such a hyper-sensitive analysis continue to be reliable? To find out, analysts work with the continuous raw data together with occasional and unannounced spiked data

[1] https://www.ligo.caltech.edu.

(Figure 12.2). This is a stringent test of the analysis, and only data that are similar to those that are expected should be cause for celebration.[2]

12.3 Virtual LIDAR scanning

Similar to simulating data and anticipating results from a satellite mission is the simulation of laser scans in environmental research. For example, researchers may want to predict the consequences of forest management for forest structure and the theoretically linked species richness of forest-dwelling organisms. But which of the many indices and statistics that can be constructed on the basis of a LIDAR scan should be used? Developing predictive models with neural networks, particularly when based on image data, requires many thousands of scenes recorded painstakingly in the forest. Or does it?

As an alternative, virtual LIDAR scanning simulates forest stands from a mixture of catalogued tree scans and generic tree-growth algorithms (Winiwarter et al. 2022). It then uses ray-tracing to simulate the deployment of terrestrial or aerial laser scanning and the resulting data. This allows researchers to train their algorithms and networks before using the few and expensive field recordings for validation and hypothesis testing.

12.4 Advanced simulation may be neither possible nor desirable

Large collaborative research projects at national and international scales (e.g., Fraser et al. 2013; Borer et al. 2014) should be expected to have demonstrated their usefulness before being funded. However, such demonstration does not always involve simulation. For example, no planning-ahead simulations were done when the US National Science Foundation developed and argued successfully for funding the National Ecological Observatory Network (NEON)—a distributed set of instrumentation and measurement systems in over 80 locations across 20 ecoclimatic zones. The lack of advance simulations made good sense, and not only because such an extensive sampling endeavour would have been extremely difficult to simulate. In addition, NEON was not designed to answer one single question or test a specific hypothesis. Rather, it is meant to be useful for many questions, many hypotheses, and many investigators, most of which are as yet unknown or may become apparent only after years of environmental monitoring (Doak et al. 2008; Filbee-Dexter et al. 2017). Simulations also could introduce unintended biases. For example, would simulations for biodiversity monitoring have included that colonisations (immigration, "invasion") are being detected earlier than local extinctions (emigration), which are biasing trends in estimates of species richness (Kuczynski et al. 2023)?

This brief example reminds us that scientific research is a multi-faceted process and that leaving room for serendipity is a good thing. Still, providing room for serendipity is not the same as sloppy thinking or asking vague questions. Given the ease of data simulations and the familiarity of all areas of science with process models and experiments, it is actually surprising how few prior power analyses, much less simulations, can be found either in the primary literature or in project reports.

The Future

Statistics remains the basis of analysis, modelling, and prediction, and we should become comfortable using as many tools as possible in the statistical toolbox. As quantitative skills

[2] See Hartsfield (2016) for a more dramatic description of the 2010 signal injection event and its nearly leading to a journal submission.

become more widespread across the sciences, and experiments shift from lab and field to in silico assessments (e.g., Kollmann & Sourjik 2007), it seems only appropriate to use data-generating models and simulations systematically before and after data collection, prospectively and retrospectively. We think that simulations help with developing comfort with a wide range of statistical tools, and hope that our book has provided inspiration for all researchers to think more about the specific data they will generate with their projects and whether the anticipated analyses will deliver interpretable and meaningful results. Gaining intuition where we had none and being forced to be specific about measurement and noise reduce the time we might otherwise spend being frustrated by sample sizes that turn out to be too small or time series that turn out to be too short. The real world is messy and uncertain; simulation will help you prepare for it and for what and how you will measure to learn about it!

References

Abbott, B. P. & The LIGO Scientific Collaboration and Virgo Collaboration (2016). Observation of gravitational waves from a binary black hole merger. *Physical Review Letters*, 116(6), 061102. doi: 10.1103/PhysRevLett.116.061102.

Bengtsson, L., Dias, J., Gehne, M., et al. (2019). Convectively coupled equatorial wave simulations using the ECMWF IFS and the NOAA GFS cumulus convection schemes in the NOAA GFS model. *Monthly Weather Review*, 147(11), 4005–4025. doi: 10.1175/MWR-D-19-0195.1.

Borer, E. T., Harpole, W. S., Adler, P. B., et al. (2014). Finding generality in ecology: A model for globally distributed experiments. *Methods in Ecology and Evolution*, 5(1), 65–73. doi: 10.1111/2041-210X.12125.

Creswell, J., von Hausegger, S., Jackson, A. D., et al. (2017). On the time lags of the LIGO signals. *Journal of Cosmology and Astroparticle Physics*, 2017(8), 013. doi: 10.1088/1475-7516/2017/08/013.

Doak, D. F., Estes, J. A., Halpern, B. S., et al. (2008). Understanding and predicting ecological dynamics: Are major surprises inevitable? *Ecology*, 89(4), 952–961. doi: 10.1890/07-0965.1.

Filbee-Dexter, K., Pittman, J., Haig, H. A., et al. (2017). Ecological surprise: Concept, synthesis, and social dimensions. *Ecosphere*, 8(12), e02005. doi: 10.1002/ecs2.2005.

Fraser, L. H., Henry, H. A. L., Carlyle, C. N., et al. (2013). Coordinated distributed experiments: An emerging tool for testing global hypotheses in ecology and environmental science. *Frontiers in Ecology and the Environment*, 11(3), 147–155. doi: 10.1890/110279.

Hartsfield, T. (2016). Faking Data for a Good Cause. https://www.realclearscience.com/blog/2016/01/ligos_fascinating_trial_by_fake_data.html. Accessed 17 January 2024.

Kollmann, M. & Sourjik, V. (2007). In silico biology: From simulation to understanding. *Current Biology*, 17(4), R132–R134. doi: 10.1016/j.cub.2006.12.034.

Kuczynski, L., Ontiveros, V. J., & Hillebrand, H. (2023). Biodiversity time series are biased towards increasing species richness in changing environments. *Nature Ecology & Evolution*, 7(77), 994–1001. doi: 10.1038/s41559-023-02078-w.

Leutbecher, M., Lock, S.-J., Ollinaho, P., et al. (2017). Stochastic representations of model uncertainties at ECMWF: State of the art and future vision. *Quarterly Journal of the Royal Meteorological Society*, 143(707), 2315–2339. doi: 10.1002/qj.3094.

MacCoun, R. & Perlmutter, S. (2015). Blind analysis: Hide results to seek the truth. *Nature*, 526(7572), 187–189. doi: 10.1038/526187a.

Winiwarter, L., Esmorís Pena, A., Weiser, H., et al. (2022). Virtual laser scanning with HELIOS++: A novel take on ray tracing-based simulation of topographic full-waveform 3D laser scanning. *Remote Sensing of Environment*, 269, 112772. doi: 10.1016/j.rse.2021.112772.

Appendix A

Useful R functions for data simulations

A.1 Drawing random values from a distribution

Throughout the R software system, distribution-generating functions provide similar functionality. For the normal (Gaussian) distribution, for example, the probability density function (PDF) is given by the function `dnorm()` and random numbers drawn from it can be generated using `rnorm()`.

Similarly, we can draw random values from the Poisson, negative binomial, uniform, Cauchy, Weibull, and other distributions using `r...` (e.g., `rpois()`, `rnbinom()`, `runif()`, `rcauchy()`, `rweibull()`). Contributed packages, including `extraDistr`, `MASS`, `VGAM`, and `mvtnorm`, provide additional distribution-generating functions that follow the same naming and syntactical conventions. The specific name for the function can be found with the built-in help, e.g., `??cauchy`.

We explicitly assume that distribution-generating functions in all core packages of R are implemented correctly and work as expected. For most univariate distributions, the internal algorithm draws from a uniform distribution using a pseudo-random number generator ("PRNG"; Press et al. 2007; Templ 2016).[1] These uniform values then are transformed into random numbers of the desired distribution using the inverse of the cumulative density function (a.k.a. the quantile function, e.g., `qpois()` in Code Box A.1; results in Figure A.1). Note that if we initialise the "seed" for the PRNG using the command `set.seed()` with a specific number (our favourite is 123), when we re-run the code we will get the same set of (pseudo)random numbers.

Generating *multivariate* random numbers from uniform PRNGs is a bit more involved and requires finding a decomposition of the covariance matrix Σ (i.e., finding A such that $AA^\intercal = \Sigma$).[2] The decomposition method employed in R to find A from Σ has a barely perceptible effect on the resulting random numbers (Code Box A.2; results in Figure A.2).

[1] For a deep dive into the history of "random number generation" and to convince yourself that all these functions are *pseudo*-random number generators, a good starting place is Wikipedia Contributors (2024).

[2] Then, together with mean vector $\boldsymbol{\mu}$ and n standard normally distributed $\mathbf{z} = \{z_1, z_2, \ldots z_n\}^\intercal$, the requested n multivariate normal $\mathbf{x} = \boldsymbol{\mu} + A\mathbf{z}$.

Code Box A.1: Convert random draws from uniform to any distribution

```
set.seed(123)   # Ensures repeatability
x <- runif(1e5)  # We need many samples to show this
yindirect <- qpois(x, lambda=2.1) # Indirect method using the quantile function
ydirect <- rpois(1e5, lambda=2.1) # Direct method calling the random number
    function

plot(table(yindirect), lwd=2)  #Plot it
points(as.numeric(names(table(ydirect)))+.1, table(ydirect), type="h",
     col="green", lwd=2) # +.1 to offset the bars a bit
```

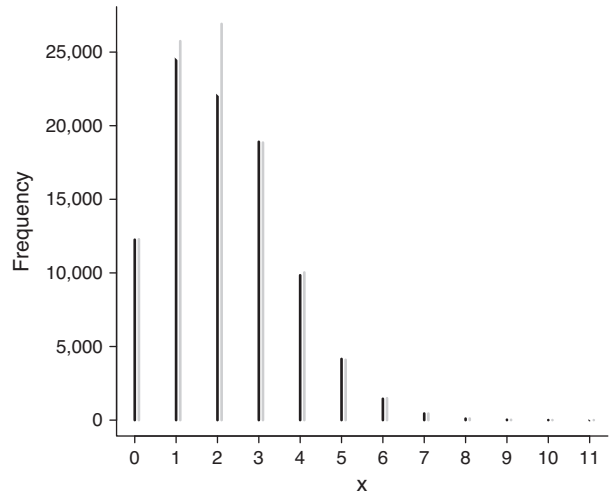


Figure A.1 Frequency of 100,000 random values drawn from a Poisson distribution with mean = $\lambda = 2.1$: Black lines are converted from uniformly drawn random values using the quantile function of the Poisson distribution. Grey bars are "directly" drawn from the purpose-built function `rpois()`.

A.2 Doing things repeatedly: `for`-loops and `replicate`

A.2.1 `for`-loops

The canonical way to code the repeated execution of a computational operation is to use a `for`-loop. The `for`-loop, like the similarly-structured `if-then-else` statement, is a central building block of computation, and programming without it is unthinkable (Dahl et al. 1972). We use `for`-loops extensively in the book, since they are the most obvious and easy-to-read way to structure a repeated operation. Its syntax includes a declaration of a counter, often `i` (but the counter can take on any name, including non-numeric values; Code Box A.3), and a scope, which is the range of values that the counter takes.

For example, the typical declaration of a `for`-loop,

```
for (i in 1:10) {...},
```

Code Box A.2: Effect of changing the decomposition method of multivariate normal random variates

```
# Load required libraries
 library(mvtnorm)
 library(corprcor)

 set.seed(123)
 A <- matrix(runif(100), 10, 10)
 Sigma <- A %*% t(A)
 corpcor::is.positive.definite(Sigma)   # Note here the call to the function
     "is.positive.definite()" in the ''corpcor'' library. There are several R
     libraries with a function of this name

 xChol <- rmvnorm(n=1e4, mean=rep(0, 10), sigma=Sigma) # Choleski decomposition
     (default)
 xSVD <- rmvnorm(n=1e4, mean=rep(0, 10), sigma=Sigma, method="svd") # Singular
     value decomposition
 xeigen <- rmvnorm(n=1e4, mean=rep(0, 10), sigma=Sigma, method="eigen")
     # Eigenvalue decomposition

# Plots
 hist(xChol, col="grey20", angle=30, density=0, freq=F, las=1)
 hist(xSVD, add=T, col="grey20", angle=-30, density=10, freq=F)
 hist(xeigen, add=T, col="grey", angle=30, density=10, freq=F)
```

sets `i` equal to 1, then works through whatever expression or sets of commands is in `{...}`, and then increments the counter and executes `{...}` again. If you have never seen a `for`-loop before, you may find it confusing that the value of the counter does not have to play a role in whatever happens in the loop (although it can be used within the `for`-loop, too; see Code Box A.4). It need only be a counter.

A common expression in a `for`-loop is the assignment of a computation result to a list or vector or matrix (Code Box A.4).

There are a few things to unpack and clarify in the code in Code Box A.4. First, we cannot use `levelOfExperiment` as our scope: there would be no information in rows 4, 5, 6, etc. Thus, we generate a scope "on the fly" that has the same length as `levelOfExperiment` but is consecutively numbered 1, 2, 3, 4, 5, ... using the `seq_along()` function. Second, we access the relevant value of `levelOfExperiment` by indexing it with the counter: `levelOfExperiment[i]`. Then we do our computations, analyses, or whatever else we want to do. Finally, we store the value of `levelOfExperiment[i]` and the results of the computation in the `i`th columnn of the predefined `results` matrix (which must be defined outside of the loop).[3] Finally, we add a progress indicator using `cat()`, which simply returns

[3] Everything within the loop not defined before will be forgotten! We can get around that by assigning an object "globally," using the "double-headed" global assignment operator «– instead of <–, as we did in Code Box 3.1 in Chapter 3. The idea is as follows: define a variable, then, in a for-loop or function call, assign a value to that variable as a side effect of the function:

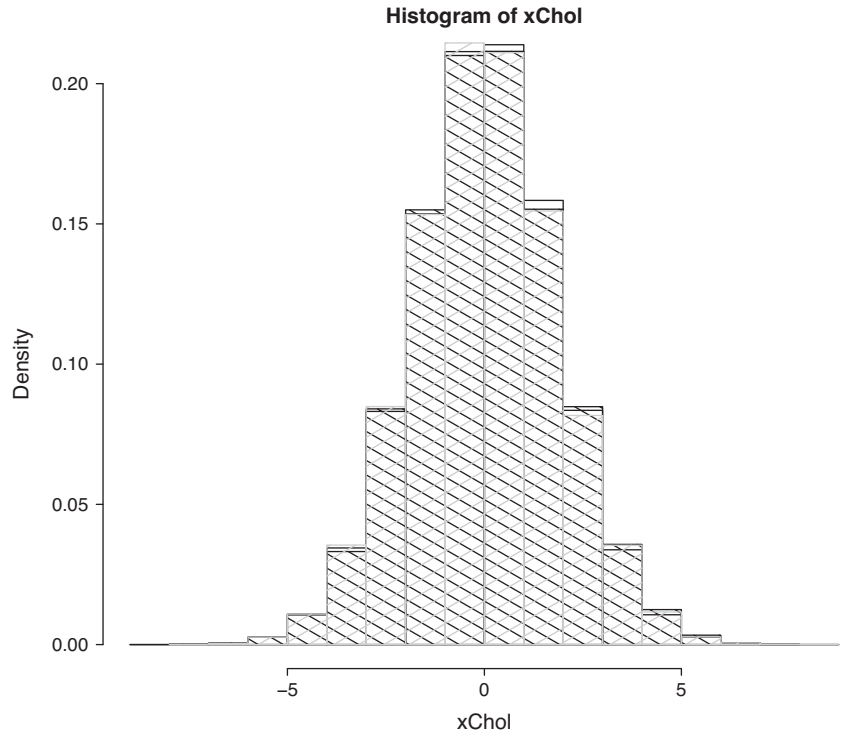


Figure A.2 Density of 10,000 random values drawn from a 10-dimensional multivariate normal distribution, with means = 0 and covariance matrix Σ derived from a 10-column matrix of random values drawn from a standard uniform distribution. Three different methods were used to find the decomposition of Σ: by Cholesky (white bars), by singular-value (dark grey sloping down), and by eigenvalue decomposition (light grey sloping up).

Code Box A.3: A for-loop to check if a word contains the letter "a"

```
namesOfThings <- c("desk", "chair", "table", "glass")
for (whatever in namesOfThings){
 # check if name contains an "a", and if so print "contains an a"
  if(length(grep("a", whatever))>0) print(paste(whatever, "contains an a"))
}
| [1] "chair contains an a"
| [1] "table contains an a"
| [1] "glass contains an a"
```

`x <- 0` # set x to value 0
`f <- function() x «- 1` # defines a function "f"
`x`
`| [1] 0`
`f()` # running "f" globally assigns to x the value 1
`x`
`| [1] 1`
Now x has the value 1, despite being assigned within a function.

Code Box A.4: A for-loop to compute a value and store it in a matrix

```
levelsOfExperiment <- c(1,2,3,8,16,32)
results <- matrix(NA, nrow=length(levelsOfExperiment), ncol=2) # Predefine the
    output object
for (i in seq_along(levelsOfExperiment)){
 out <- 1/(1 - exp(-0.1*levelsOfExperiment[i])) # Do some computation or
     analysis
 results[i,] <- c(levelsOfExperiment[i], out)
 cat(levelsOfExperiment[i], " ") # Our indicator of progress
}

| 1  2  3  8  16  32

results

|      [,1]     [,2]
| [1,]    1 10.508332
| [2,]    2  5.516656
| [3,]    3  3.858296
| [4,]    8  1.815966
| [5,]   16  1.252970
| [6,]   32  1.042494
```

(prints to the screen) the current value of `levelsOfExperiment[i]`, followed by a space. Progress indicators are useful for error-trapping and debugging, and also let you know if your code is caught in an infinite loop.

A.2.2 `apply()` and its relatives

In R, unlike many other languages, there exists a group of functions that make the `for`-loop more efficient. These `*apply()` functions (`apply()`, `lapply()`, `sapply()`, `tapply()`, `mapply()`, …) can be used in many different ways; we illustrate here a few of the simplest cases.

`sapply()` AND `lapply()`

`sapply()` and `lapply()` apply a function to a list (note that a vector is also a list). The only difference is that `sapply()` tries to put its output in a simple format if possible (e.g., a vector), whereas `lapply()` always returns a list. We can re-write the `for`-loops in Code Boxes A.3 and A.4 using `sapply()` (Code Box A.5).

In words, this application of `sapply(...)` means "apply to every element in the vector `levelsOfExperiment` the function `function(x) ...`" (in Code Box A.5, the function applied to each element of the vector is `1/(1 - exp(-0.1*x))`). In general, this function is called an "anonymous function" (or "lambda function"), as it is defined "on the fly" without a name and without being available outside this line of code. An even shorter way to write an anonymous function is to use `\(x)` instead of `function(x)`.

Using `*apply()` is elegant and computationally more efficient, but it is more difficult to read than a `for`-loop. It may take some getting used to, as we need to keep the abstract idea

Code Box A.5: Replacing a for-loop with sapply()

```
# This example uses the object levelsOfExperiment from Code Box A.4.

 sapply(levelsOfExperiment, function(x) 1/(1 - exp(-0.1*x)))

| [1] 10.508332  5.516656  3.858296  1.815966  1.252970  1.042494
```

Code Box A.6: Computing row- and column-wise statistics using apply()

```
m <- matrix(1:81, 9, 9)
m
|      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
| [1,]    1   10   19   28   37   46   55   64   73
| [2,]    2   11   20   29   38   47   56   65   74
| [3,]    3   12   21   30   39   48   57   66   75
| [4,]    4   13   22   31   40   49   58   67   76
| [5,]    5   14   23   32   41   50   59   68   77
| [6,]    6   15   24   33   42   51   60   69   78
| [7,]    7   16   25   34   43   52   61   70   79
| [8,]    8   17   26   35   44   53   62   71   80
| [9,]    9   18   27   36   45   54   63   72   81

apply(m, 1, var) # Row-wise: index=1

| [1] 607.5 607.5 607.5 607.5 607.5 607.5 607.5 607.5 607.5

apply(m, 2, var) # Column-wise: index=2

| [1] 7.5 7.5 7.5 7.5 7.5 7.5 7.5 7.5 7.5
```

of the function in our head while simultaneously thinking about to what we will apply it. In the book, we typically use `sapply()` to collate results of many repetitions stored in a list.

`apply()`
The function `apply()` applies a function to every row or column of a matrix. It is a very convenient way to summarise information (Code Box A.6).

`apply()` can also be used on arrays of higher dimensions, with indexing then becoming a bit more tricky. For means and sums there are dedicated (faster) implementations of `apply()`, such as `colMeans` and `rowSums()`.

`tapply()`
`tapply()` ("table apply") applies a function to data grouped by (a list of) factors. Using `tapply()` is the simplest way to make a pivot table in R. That is, it averages over all non-specified variables in a table. So we can summarise the values of one column in a table by each value of (a combination of) levels in another (Code Box A.7).

Code Box A.7: Create a pivot table with tapply()

```
#Load required libraries
 library(faraway) # For the example dataset

 data(mammalsleep)
 summary(mammalsleep) # Note: only columns 5, 8, 9 are shown.

|        sleep         predation        exposure
|  Min.   : 2.60   Min.   :1.000   Min.   :1.000
|  1st Qu.: 8.05   1st Qu.:2.000   1st Qu.:1.000
|  Median :10.45   Median :3.000   Median :2.000
|  Mean   :10.53   Mean   :2.871   Mean   :2.419
|  3rd Qu.:13.20   3rd Qu.:4.000   3rd Qu.:4.000
|  Max.   :19.90   Max.   :5.000   Max.   :5.000
|  NA's   :4

 with(mammalsleep, tapply(sleep, predation, mean, na.rm=T))

|         1         2         3         4         5
| 12.050000 12.720000  9.120000 10.228571  7.383333
```

Code Box A.8: Pivot table with R, *continued*

```
with(mammalsleep, tapply(sleep, list(predation, exposure), mean, na.rm=T))

|          1        2    3    4        5
| 1 12.99000 10.30000  6.2 12.0       NA
| 2 14.87143 11.01429  9.6   NA       NA
| 3 10.46667 10.60000   NA   NA 3.600000
| 4 13.00000       NA 10.3  9.7 6.200000
| 5 13.80000 12.80000  8.2 12.5 4.071429
```

Regrettably, none of the `*apply()` functions accepts a data object directly as an argument, so we have to specify the data we want to use either for each variable directly (`mammalsleep$sleep`, `mammalsleep$predation`), or use the `with()` function to create a local environment within which the `*apply`-function is executed.

The second argument of `tapply`, `INDEX`, also can be a list. If it is, then we will get a pivot-table averaging over all non-specified variables, but differentiated by rows (as named first; e.g., `predation`) and columns (e.g., `exposure`) (Code Box A.8).

The `NA`s indicate the missing combinations, not `NA`s within the groups (which we ignored using the argument `na.rm=T` passed on to the function `mean` within `tapply`).

`mapply()`

Finally, `mapply()` is like `apply`, but it allows us to pass on more than one argument. This is useful when, for example, we want to compare two lists of the same length (Code Box A.9).

Again, this is an elegant, short bit of code, but it is much less intuitive than the `for`-loop.

Code Box A.9: Computations on two or more lists using mapply()

```
set.seed(123)
A <- as.list(1:10) # 1:10 as a list
B <- as.list(sample(1:10))
res <- 1:10
for (i in 1:10){
 res[i] <- (A[[i]] - B[[i]])^2 # Compute the Euclidean distance between all
      pairs of points
}
res

| [1]  4 64  1 16  1  9 36  1 16 36

mapply(\(x1, x2) (x1-x2)^2, A, B) # anonymous function of two variables

| [1]  4 64  1 16  1  9 36  1 16 36
```

REPRISE: `for{}` VERSUS `*apply()`
`for`-loops are easy to read and will stop the moment an error occurs so we can investigate what is going on for the current value of the counter. In contrast, `*apply()` functions generally are faster than `for`-loops, but they have the serious disadvantage of not showing where something goes wrong when it does. Although `*apply()` statements are difficult to read and interpret initially, as we become used to using them, they become as readable and interpretable as `for`-loops.

A.2.3 `replicate()`

The `replicate()` function is useful for stochastic expressions: code that returns a different result every time it is run because of built-in randomness. All our data simulations, bootstraps, data perturbations, and resampling routines have built-in randomness because we draw randomly from some distributions. Using `replicate()` is the simplest and computationally most efficient way to run the same stochastic expression many (n) times. However, no arguments can be handed over to `replicate()`, so we cannot use it for computing something on inputs that change with each iteration (for example, when the counter in a `for`-loop is also used within the expression that is executed repeatedly).

Often we first do a single run of a particular operation, so that we can see the result of the operation on the actual dataset. For our repeated stochastic simulations, we use the same code within `replicate()`. An example is shown in Code Box A.10, where we first run a regression of the effect of lifespan on sleep in mammals (a single run on the original data) and then show how to use both a `for`-loop and `replicate()` to bootstrap the resulting estimate of the slope.

Note that there is virtually no difference or advantage to using `replicate()` or `for(){...}` in terms of readability or space. `replicate()` only saves us the bother of pre-defining the output object for the results. If the expression returns a complicated object, `replicate()` automatically stores it as a list. There is little difference in computational speed between using a `for`-loop and using `replicate()` unless we need to parallelise the computations. In that case, the `future_replicate()` function in the `future.apply`

Code Box A.10: Using a for-loop and replicate() to boostrap a linear model

```
# Load required libraries
 library(faraway)

 data(mammalsleep)

# Run the model once, on the actual data, and return the estimated slope
 coef(lm(sleep ~ lifespan, data=mammalsleep))[2] # The index [2] specifies the
      slope estimate

# One bootstrap (see Code Box A.10 for an example of using the function "sample
     ()"):
 coef(lm(sleep ~ lifespan, data=mammalsleep[sample(nrow(mammalsleep),
      replace=T),]))[2]

 set.seed(123)
# Use a for-loop for 1000 bootstraps:
 bs <- NULL # initalise the object ("bs") for the results, in this case a vector
 for (i in 1:1000){
  bs[i] <- coef(lm(sleep ~ lifespan, data=mammalsleep[sample(nrow(mammalsleep),
       replace=T),]))[2]
 }
 summary(bs)

|     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
| -0.22560 -0.12631 -0.10119 -0.10771 -0.08284 -0.04662

# Do it again, but with replicate()
 bs <- replicate(1000, {
  coef(lm(sleep ~ lifespan, data=mammalsleep[sample(nrow(mammalsleep), replace=T
       ),]))[2]
 })
 summary(bs)

|     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
| -0.23146 -0.13375 -0.10962 -0.11236 -0.08789 -0.04833
```

package is easier and much more efficient to use than the parallelised `for`-loop in the `foreach` package.

A.3 Shuffling, resampling, and bootstrapping: `sample()`

Although randomisation techniques are mostly outside the scope of this book (but see §2.5 in Chapter 2), we do occasionally use them for comparison or illustration. In those cases, we make use of the `sample()` function. By default, `sample(x)` shuffles (i.e., randomises) a vector x of length n. Besides the object `x` to be sampled, `sample()` has two other arguments that turn the randomisation into a resampling function:

`size` defines the size of the resulting output; if `size` $< n$, `sample(x, size=k)` draws a sub-sample of length k from the data. If `size` $= n$, the default, it only shuffles the values of x. If `size` $> n$ (and `replace=T`, see the next point), x is sampled *with replacement*, leading to some values being represented multiple times, and others not at all.

`replace` determines whether x is sampled with or without replacement. The default is to sample without replacement, but sampling with replacement (using the argument `replace=T`) and `size` $= n$ is called *bootstrapping* (Efron & Tibshirani 1993). Fun fact: the proportion of x that, on average, is part of the sampled bootstrap equals $1 - 1/e = 0.632$.

Nonparametric statistics heavily rely on resampling tests for inference and on bootstrapping for uncertainty estimation (Carsey & Harden 2014; Manly 2017). For example, if we want to test whether a medication given to individuals in one of two `groups` has an effect on the extra hours of sleep a subject gets, we can use a classical t-test or bootstrap it with `sample` (Code Box A.11).

In this first run, the results differ. The randomisation-based p-value of 0.039 is actually significant, while that of the t-test is not. Side note: the result turns out to be the same whether we shuffle the group labels or the response values.

However, in this first run, we glossed over the fact that this is a paired design, where each subject has been allocated to both groups sequentially. Thus, neither the t-test nor the resampling was done correctly. For the shuffling, we have to randomly allocate the two measurements per subject to the groups, i.e., shuffle `group` *within* subject (Code Box A.12).

The repeated sampling of individuals forced us to permute the group label in a per-subject way. Although doing this is a bit more tedious, taking account of this pairing of data leads to a higher power of the test (both the resampling version and the t-test itself). In addition, because the analysis was done correctly for the experimental design, the results of the paired t-test and the bootstrapping were similarly statistically significant.

A.4 Little helpers

There are a handful of functions that are used very often simply to quickly provide us with some information about how well our code works or to make our coding lives easier.

A.4.1 Convenient functions

`system.time()` **AND** `Sys.time()`

To compare the speed of different algorithms or to estimate the total runtime based on a single execution of some code, we can use `system.time(...)`. The code in `(...)` has no length restrictions, although it should be enclosed by {. . .} if it extends for more than a line.

For timing of long code chunks it is often easier to use `Sys.time()` as in Code Box A.13.

`simulate()`

For some model types (notably, `lm` and `glm`), the generic `simulate()` function offers a simple way to directly simulate data from a fitted model. Some packages provide a class-specific simulate function for their models, such as `lme4, glmmTMB, MCMCglmm, mgcViz, metafor,` and `VGAM`. In those cases, the predictor data are kept as they are in the original, and the response is computed as a random draw from the distribution implied by the model and based on its expectation (e.g., the `fitted.values` of the model).

In this book, we do not use the `simulate` function, as it hides exactly the step that is the target of our book. In real life, `simulate` is our friend.

Code Box A.11: Resampling as an alternative to a *t*-test

```
# Load required libraries
 library(faraway)

 data(sleep)
 with(sleep, diff(tapply(extra, group, mean))) # Computes difference between
      group means

 with(sleep, diff(tapply(extra, sample(group), mean))) # Same, but with shuffled
       group labels

# Do this many times (see Code Box A.10 for the "replicate()" function):
 set.seed(123)
 resamp <- replicate(1000, with(sleep, diff(tapply(extra, sample(group), mean))))
 sum(resamp > 1.58) / length(resamp)

| [1] 0.039

# Compare the results to a t-test

 t.test(extra ~ group, data=sleep)

|     Welch Two Sample t-test
|
| data:  extra by group
| t = -1.8608, df = 17.776, p-value = 0.07939
| alternative hypothesis: true difference in means between group 1 and group 2
     is not equal to 0
| 95 percent confidence interval:
| -3.3654832  0.2054832
| sample estimates:
| mean in group 1 mean in group 2
|           0.75            2.33
```

`update()`

The generic `update()` function is a convenience function that allows us to re-fit a model after changing some of its specifications (as used in §11.6 in Chapter 11). For example, we could do bootstrapping using `update` in such a way that the fitted model (`fm`) is re-fit in the same way, but on bootstrapped data:

```
update(fm, .~., data=mydata[sample(1:nrow(mydata), replace=T), ]).
```

For `update()` to work, the model type needs to be able to extract the model's call, related commands, parameters, and outputs. Therefore we should not be surprised if it does not work for every operation or algorithm for which we would like to use it. Notably, `update()` rarely works with machine-learning models because they have different object structures.

Code Box A.12: Resampling as an alternative to paired-*t*-test

```
groupPrime <- 1:20
for (i in 1:10){
 groupPrime[c(i, i+10)] <- sample(sleep$group[c(i, i+10)])
}
cbind(sleep$group, groupPrime) # random swapping of group within ID
```

```
|         groupPrime
|  [1,] 1          2
|  [2,] 1          1
|  [3,] 1          2
|  [4,] 1          1
|  [5,] 1          2
|  [6,] 1          1
|  [7,] 1          1
|  [8,] 1          2
|  [9,] 1          2
| [10,] 1          2
| [11,] 2          1
| [12,] 2          2
| [13,] 2          1
| [14,] 2          2
| [15,] 2          1
| [16,] 2          2
| [17,] 2          2
| [18,] 2          1
| [19,] 2          1
| [20,] 2          1
```

```
resamp2 <- replicate(1000, {
 groupPrime <- 1:20
 for (i in 1:10){
   groupPrime[c(i, i+10)] <- sample(sleep$group[c(i, i+10)])
 }
 cbind(sleep$group, groupPrime) # random swapping of group within ID
 diff(tapply(sleep$extra, groupPrime, mean))

})

#Plot it (plots not shown)
hist(resamp2)
abline(v=1.58)

sum(resamp2 > 1.58) / length(resamp2)
```

```
| [1] 0
```

```
t.test(extra ~ group, data=sleep, paired=T)
```

```
|   Paired t-test
|
| data:  extra by group
| t = -4.0621, df = 9, p-value = 0.002833
| alternative hypothesis: true mean difference is not equal to 0
| 95 percent confidence interval:
|  -2.4598858 -0.7001142
| sample estimates:
| mean difference
|           -1.58
```

Code Box A.13: How long does resampling take?

```
p1 <- Sys.time() # start of computation
   set.seed(123)
   resamp <- replicate(1000, with(sleep, diff(tapply(extra, sample(group), mean)
       )))
p2 <- Sys.time() # end of computation
p2-p1

| Time difference of 0.06743789 secs
```

A.4.2 Data wrangling

The current standard package for data wrangling, i.e., reformatting, merging, sorting, aggregating, and newly defining data objects is `dplyr`. For simplicity, however, we mostly use `base` functions, such as `[...]` for indexing, `stack()` and `unstack()` for changing a data frame from wide to long format and vice-versa, `aggregate()` or `*apply()` for aggregating data, etc. To access specific values of a vector, `which(variable == condition)` returns the indices of the positions of a vector that meet the condition specified, with `which.max()` and `which.min()` being specific common cases.

A.4.3 Logical operators

Using logical operators is a fundamental programming skill and one that we use frequently. The most relevant logical operators are `&` (and), `==` (is equal to), `!=` (is not equal to), and `|` (or). Logical operators are used extensively in `if-then-else` statements, and their evaluation should always lead to (a vector of) `TRUE` or `FALSE` values, unless the expression cannot be logically evaluated, in which case they return an error. We also must take care to ensure that logical operations can return only a single `TRUE` or `FALSE` when used with an `if` statement. Otherwise, the execution will return a warning.

A.5 Dedicated simulation packages

In most of the examples in this book, we use simulation for specific questions and datasets. For such specific operations, it is difficult to generalise them with a single function or

package. However, for developing new general methods or addressing more generalisable or generic questions, there are some data-simulation packages available in R.

First and foremost there is the `simstudy` package (Goldfeld & Wujciak-Jens 2020). Its functionality is described in various vignettes, papers, and tutorials provided within the package itself. From our perspective, it is particularly useful for generating multiply nested data, as we do in §5.6 in Chapter 5. The logic and style of the functions in `simstudy` take some getting used to, but it is well worth the effort to learn them. `simstudy` is extremely flexible and allows for all types of structure, distributions, and effects.

An incomplete and alphabetical list of other potentially interesting simulation packages is:

- `aniMotum`, simulating animal movement in ecology;
- `lsasim`, for simulation of large-scale assessments, e.g. in education;
- `Mediana`, specifically for clinical trail simulation;
- `NCC`, for non-concurrent control simulations in medicine;
- `parSim`, generic, fast but limited in scope;
- `SimInf`, for stochastic disease-spread simulation in epidemiology;
- `slimr`, for population genetic simulations;
- `survsim`, for simulation of survival data in medicine;
- `virtualspecies`, for simulating species distribution data in ecology;
- `sim.data()` in the `WIN` package for time-to-event simulation.

References

Carsey, T. M. & Harden, J. J. (2014). *Monte Carlo Simulation and Resampling Methods for Social Science*. Los Angeles, CA, USA: Sage.

Dahl, O.-J., Dijkstra, E. W., & Hoare, C. A. R. (1972). *Structured Programming*. New York, NY, USA: Academic Press.

Efron, B. & Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. London: Chapman & Hall.

Goldfeld, K. & Wujciak-Jens, J. (2020). simstudy: Illuminating research methods through data generation. *Journal of Open Source Software*, 5(54), 2763. doi: 10.21105/joss.02763.

Manly, B. F. J. (2017). *Randomization, Bootstrap and Monte Carlo Methods in Biology*, 3rd edition. New York, NY, USA: Chapman and Hall/CRC.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical Recipes: The Art of Scientific Computing*, 3rd edition. Cambridge, UK: Cambridge University Press.

Templ, M. (2016). *Simulation for Data Science with R: Effective Data-Driven Decision Making*. Birmingham, UK: Packt Publishing.

Wikipedia Contributors (2024). Pseudorandom number generator—Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Pseudorandom_number_generator&oldid=1220718809. Accessed 1 May 2024.

Index

Italic pagination indicates figures and tables, "n" indicates footnotes

Adams, D., 137
Addor, N., 252
advanced simulations, 12–13; bias and, 403–6; covariance and, 409, *412*; data-generating model (DGM) and, 407; errors and, 404; future of, 406–7; Gaussian distribution and, 405; hypothesis testing and, 406; LIDAR, 406; LIGO, 404–6; manipulation and, 404; NEON, 406; neural networks and, 406; noise and, 404–7; parameter values and, 404, 419; pipelines and, 403–5; sample size and, 407; signals and, 404–6; simulated data and, 404; stochastic weather forecasting, 403–4; uncertainty and, 403–4
AIC: Akaike weight and, 155, 389; assumptions of statistical analysis and, 148; errors and, 254–55; evaluation and, 296, 389–97; GLS analysis and, 36; missing data and, 217–19; model selection and, 155, 157–59; predictor variables and, 238–39
Akaike weight, 155, 389
Alice's restaurants, 271–273, *274*
American Statistical Association, xii
analysis of variance (ANOVA): assumptions of statistical analysis and, 105, 116–23, 127, 145; bias and, 314; degrees of freedom and, 398; effect sizes and, 314; evaluation and, 398; frequency distribution and, 20; *F*-statistic and, 20–21, 116; *F*-values and, 20–21, *23*, 117, *122*; heteroscedasticity and, 20–21, 118–22, 154; homoscedasticity and, *13*, 105–6, 116–23; normality and, 18; null hypothesis and, 118; predictor variables and, 179; residuals and, 18, 116–17; retrospective power analysis and, 314–15, *316*; robustness and, 118–22; rules-of-thumb and, 154, 179; simulation basics and, 3, 17–22, *23*; study designs and, 60
Anderson-Darling test, 106
`apply`, 413–16
artificial intelligence (AI): Big Data and, 9–10, 201; evaluation and, 307; large language models (LLMs) and, 9, 14; machine learning and, 201, 205
assumptions of statistical analysis, 12; AIC and, 148; analysis of variance (ANOVA) and, 105, 116–23, 127, 145; autocorrelated data and, 106, 149; Bayesian models and, 103, 131–32, 149; bias and, 105, 112, 114–15, *120*, 134, 137, 145; confidence interval (CI) and, 103; correlation and, 103n1, 106, 114, 137–46, 149; degrees of freedom and, 116–17, 148; errors and, 103, 105–13, 116–17, *122*, 123–28, 138, 147, 149; false discovery rate (FDR) and, 126–31; false nulls, 128–31; false positives, 105, 123–31, 145; family-wise rates and, 123–28; fixed effects and, 133–34, 138; frequentist analysis and, 103; *F*-statistic and, 116; Gaussian distribution and, 110–14, 131–32, 137n25; general linear models (GLMs) and, 103, 145, 147; hierarchical data and, 133–41; hyper-distributions and, 131–37; hypothesis testing and, 103; Kolmorogov-Smirnov (K-S) test and, 106, 135n24; machine learning and, 103, 105, 111; matrices and, 128–32; mixed-effect models and, 131–37; model structure and, 133, 149; multiple simulation analysis and, 115–16; multiple testing and, 123–31; noise and, 108–15; normal distribution and, 106–9, 116, 131–32, 135, 137, *140*, *142*; normality and, 105–9, 116, 132–37, 142, 145, 148; null hypothesis and, 118, 126–28; parameter values and, 103, 106, 107n2, 110, 114–16, 119, 131–35, 139–40, *142*, 145, 148; post hoc analysis and, 118, 123, 145; predictor variables and, 137–45; principal component analysis (PCA) and, 142, 144–46; probability and, 123–24, 126–27, 145; *p*-values and, 107, 123, 126–27, 145, 148; replicates and, 108–9, 112, *113*, 124–30, 134, 139, 141, 146; residuals and, 3, 106–9, 116–17, 121, 132–35, 138–41, 145, 147–48; robustness

assumptions of statistical analysis (*continued*) and, 105, 118, 123; sample size and, 106–10, 119, 124–25; seeds and, 107–11, 119–25, 128–30, 146–49; simulated data and, 107, *113*, 127, 133–35, 143; single simulation analysis and, 115; standard deviation and, 107n3, 117, 131, *140*; true positives and, 123–24; *t*-test and, 103, *120*, 124, 127, 138; uncertainty and, 115, 147; violating, xi
atrazine, 227–31
autocorrelated data: assumptions of statistical analysis and, 106, 149; censored observations and, 226; change-point detection and, 196n27; errors and, 252; evaluation and, 281; machine learning and, 203–4; meta-analysis and, 343; Monte Carlo simulation and, 181, 183; noise and, 34, 86–87, 91; phylogenetic, 191n25; power of statistical analysis and, 86–92; residuals and, 33, 87–88, 91–92, 183, *193*, 204, 281; sample size and, 3; simulation basics and, 3, 33–36; spatial, 3, 33, 191n25, 204; temporal, 34–36, 86–88, 91, 106, 181, 183, 203–4, 226; time series and, 191, *193*, 196n27, 281
Aydemir, A., 110

base rate fallacy, 128–29
Bayesian models: assumptions of statistical analysis and, 103, 131–32, 149; censored observations and, 226n9; didactic example of, 271–73; errors and, 246, 249–51; evaluation and, 269–72, *274*, 283, 306n23, 387–89, 397; frequentist analysis and, 10, 103, 182, 306n23, 307, 312, 387; improving estimation in, 93, 98; learning from prior, 270–73; machine learning and, 203n33; Monte Carlo simulation and, 182, 187–89; power of statistical analysis and, 93, 98; predictor variables and, 232; recipes and, xi; retrospective power analysis and, 312, 319, 331, 334; simulation basics and, 10, *13*, 22n6; study designs and, 45, 49
before-after (BA) design, 99–100
before-after control-impact (BACI), 99–100
Benjamini-Hochberg procedure, 126–30
Bennedsen, M., 182, 184, 187–90
Berger, R. L., 11
Berlin, J. A., 314–16, 321–22
Bernoulli data, 52n5, 55n8, 202, 236, 295, 387
bias: advanced simulations and, 403–6; ANOVA and, 314; assumptions of statistical analysis and, 105, 112, 114–15, *120*, 134, 137, 145; complete data analysis and, 222–32, 360–68, 370; crime rate reporting and, 6–7; effect size and, 314; errors and, 245–62; evaluation and, 273, *277*, 375–77, 381–82; federated analysis and, 341–45, 357–58, *361*, 362, 368–71; machine learning and, 201–3; meta-analysis and, 341–58, *369*, 370; missing data and, 212–22; model selection and, 154n2, 159–60, 171; "network of chums" effect and, 344; parameter values and, 4–5, 69, 81, 115, 145, 159, 171–73, 211, 213, 222, 226, 232, 235, *242*, 252–59, 362, 368–69, 371; power of statistical analysis and, 74, 78–85, 88, *89*, 97, 101; predictor variables and, 171–73, 232–42; publication, 342–45, 353, 362, 365–67, 370; regression analysis and, 171, 242–45; relative, 238, 240–41, *242*, 257, *262*, 348–49, *351*, 354–71; retrospective power analysis and, 314, 318, 320n18, 329, *331*; simulation basics and, 3–8, 22n7, 34; study designs and, 45–48, 58n10, 65, 69; uncertainty and, 4, 47–48, 69, 115, 171, 211, 252, 254, 260, 273, *277*, 404
Big Data: artificial intelligence (AI) and, 9–10; era of, 341; machine learning and, 153, 200–4; meta-analysis and, 341; rules-of-thumb and, 153, 200–4; workflow and, 211
binomial distribution: evaluation and, 286, 295, *297*; Pólya's urn and, 18–20
Bioclim, 146
bipartite networks, 9
BLOOM, xii
bootstrapping: confidence interval (CI) and, 39–40, 43, 65–66, 69, *81*, *83*, 204, 288, 300–2, *304*; data perturbation and, 39–40, 43, 416; errors and, 245–46, 251–59, *260*; evaluation and, 273, 275–77, 288, 294n16, 299–303, *304*, *306*; general linear models (GLMs) and, 30; jackknife and, 17, 299–303, *304*; likelihood and, 39–40, 43, 204, 273; machine learning and, 204; Monte Carlo simulation and, 181n14; power of statistical analysis and, 78–79, *81*, *83*; regression analysis and, 29, *31*, 40, 64–66, 300–2, *304*, 416; resampling and, 17, 40, 43, 416–18; R functions and, 416–19; study designs and, 48, 64–66, 69; uncertainty and, 29–30, *31*, 39, 69, 245, 251–54, *260*, 273, 275, *277*, 288, *306*, 418
Borjas, G. J., 110
Boulesteix, A.-L., 381
breakpoint: Monte Carlo simulation and, 182n15, 189; randomness and, 37–42; residuals and, 40, 42; simulated data and, 37–42. *See also* change-point methods
breast cancer, 235–38, 288
Breznau, N., 331–32
Broe, M., xii
Broennimann, O., 142
Buil-Gil, D., 6, *7*

carbon dioxide (CO_2): analyzing trends in, 87–88; ecological footprint and, 389; FLUXNET and, 7–8; monitoring atmospheric, 85–95, *96*, 99n22; Monte Carlo simulation and, 181–90; power of statistical analysis and, 85–87, 89n15, 90–96, 99n22; predictor variables and, 177; simulation parameters and, 86–87; time series and, 85–87, 89n15, 90–91
Carlin, J., 327–34
Casella, G., 11
categorisation: effect size and, 62n17; predictor variables and, 154, 172–80, 205
C/C++, 166n8, 381, 384–85, 393

censored observations: autocorrelated data and, 226; Bayesian models and, 226n9; correlation and, 222, 226; covariance and, 363–66; degrees of freedom and, 230–31; detection limit and, 222–33, *235*; errors and, 228n11, 230; estimating values for, 222–32; false negatives and, 228n11; false positives and, 228n11; *F*-statistic and, 230; Gaussian distribution and, 231; imputation and, *224*, 225, 232; left-censored data and, 222, 226–27, 230; likelihood and, 229; machine learning and, 226, 232; matrices and, 223; model structure and, 232; parameter values and, 226, 232; probability and, 225, 228n11; random effects and, 229; replicates and, 223, 225; residuals and, 230; right-censored data and, 224, *227*; sample size and, 223, *224*; seeds and, 223, 225; signals and, 228n11; truncated data and, 224n8
Central Limit Theorem, 131
Çetinkaya-Rundel, M. 10
change-point methods, 192, 194–98. *See also* breakpoint
ChatGPT, 9
chemistry, xii, 1, 276, 318
"Chinese restaurant" distribution, 20
classification: automatic, 8; evaluation and, 288, 297, 302–3, 397; generative adversarial networks (GANs) and, 8–9; K-S test and, 28; machine learning and, 202; Monte Carlo simulation and, 29n17
classification and regression trees (CARTs), 202
cliques, 9
code verification, 377. *See also* evaluation; functional verification; unit testing
Cohen, J., 313, 314n7, 322, 329
Cohen's *d*, 58n10, 62n17, 314–15, *316*
collinearity: errors and, 252, 255, 258; evaluation and, 398; model selection and, 165–72; "pick one" approach and, 171–72; predictor variables and, 238; rules-of-thumb and, 154, 165–72; simulation basics and, *13*
complete data analysis: bias and, 222–32, 360–68, 370; as coordinated data analysis, 360; effect size and, 360–68; federated analysis and, 341–45, *361*, 362–63, 368; matrices and, 362–66; meta-analysis and, 341–45; parameter values and, 360, 362, 368; predictor variables and, 360–68; randomness and, 222, *224*, 225, *226*, 229, *361*, 363–64, 366; repeated simulation and, 362–64; replicates and, 363, 366; seeds and, 363–66; stepwise model selection and, 362; study identifiers and, 364–65; unbiased data and, 362
compromise power analysis, 328
conditional plots, 276–80
confidence interval (CI): assumptions of statistical analysis and, 103; bootstrapping and, 39–40, 43, 65–66, 69, *81*, *83*, 204, 288, 300–2, *304*; errors and, 249; evaluation and, 270, *286*, 288, 290–307; hypothesis testing and, 334; machine learning and, 203–4, 295–305; misinterpretation of, 21–22; missing data and, *220*; model variability and, 28; Monte Carlo simulation and, *182*, 293; narrow, 212; non-Gaussian data and, 294–95; post hoc analysis and, 319–22; power of statistical analysis and, 78, 80, *81*, *83*; prediction interval and, 203–4, 270, 288–307; predictor variables and, *174*, 320–22; regression analysis and, 243; retrospective power analysis and, 311n1, 313, 319–26, 333–34; R functions and, 419, 421; simulation basics and, 28, *35*, 39–40, 43; study designs and, 63, 65–66, 69; true meaning of, 17, 21–26
consistency, xii, 211, 358n15, 377
control-impact (CI) design, 99–100
convolutional neural networks (CNNs), 8
Cook's distance, 281, *282*
CoPilot, 9
correlation: assumptions of statistical analysis and, 103n1, 106, 114, 137–46, 149; censored observations and, 222, 226; coefficient of, 34, 57–58, 74–78, *81*, *83, 143*, 166, *285*, corridor of stability of, 78–82, *83;* errors and, *248*; 249, 252–53, 257, 260; evaluation and, 270–71, *285*, 378; federated analysis and, 358; machine learning and, 203–4; meta-analysis and, 343, 346; missing data and, 214–15, 222; model selection and, 164–67; Monte Carlo simulation and, 181, 183; number of data points for, 76–80; Pearson's correlation coefficient, 34, 74, 76; point of stability of, 78–79, *81*, *83* power of statistical analysis and, 74–82, *83*, 86–88, 91–95, 101; predictors and, 137–46; 149, 235–42; regression models and, 33, 58, 106, 153, 164–65, 181, 183, 215, 243; retro power analysis and, 312; simulation basics and, 3–4 (*see also* autocorrelated data); spatial, 3; Spearman's coefficient of, *143*, *285*; study design and, 57–58; time series and, 191, *193*, 196n27. *See also* collinearity; covariance
corridor of stability: 78–82, *83. See also* correlation
covariance: advanced simulations and, 409, *412*; censored observations and, 363–66; covariance matrix, 243, 272, 346, 355, 363–64, 366, 409; eddy, 7–8; evaluation and, 272, 293, 378, 390; meta-analysis and, 343, 346, 355; regression analysis and, 243–50; variance-covariance matrix, 33n22, 34, 243n17, 248–50, 293
COVID-19 pandemic, 198
Cramér-von Mises test, 106
Crime Survey for England and Wales (CSEW), 6–7
criterion analysis, 328

Daszkiewicz, M., 106–8
data gaps, 7–8, 215, 342n3. *See also* imputation; missing data
data-generating model (DGM): advanced simulations and, 407; meta-analysis and, 345; power of statistical analysis and, 80; predictor variables and, 236; simulation

data-generating model (DGM) (*continued*)
basics and, 4, 13, 17, 43; study designs and, 50, 69
data mining, 68. *See also* questionable research practices (QRPs)
data perturbation: bootstrapping and, 39–40, 43, 416; evaluation and, 389–90, 394, 397; randomness and, 40–43; R functions and, 416; sample size and, 40
data snooping, 259. *See also* questionable research practices (QRPs)
data wrangling, 10, 238, 421
Dawkins, R., 8
Dechêne, A., 48, *51*, 55, 61–62
DeepMind, 304
degrees of freedom: ANOVA and, 398; assumptions of statistical analysis and, 116–17, 148; censored observations and, 230–31; Efron's effective, 397n13; errors and, 254n29, 261; evaluation and, 272, 275, 278, 291–92, 296, 389–97; generalised (GDF), 389–97; missing data and, 217–19; model selection and, 155, 157–58, 160; Monte Carlo simulation and, 183, 185–86; parameter values and, 389–97; power of statistical analysis and, 90; predictor variables and, 176, 238–39; regression analysis and, 244–45; retrospective power analysis and, 314n7, 319–20, 330, 332; similarity to other approaches, 394–97; simulation basics and, 23, *32*, 36–39; study designs and, 58n10, 67; testing, 390–94
detrending: before modelling, 195–200; change-point methods and, 192, 194–98; differencing and, 191–92, *193*; errors and, 245; time series and, 154, 191–99, 205, 245
diabetes, 213–22, 295
diagnostics: assumption violations and, 105; evaluation and, 267 (*see also* evaluation); generalised, 282–87; markers for, 235; model fit and, 275, 281–82; observed deviations and, 149 simulated random effect and, 135n24
differencing, 191–92, *193*
dimensional analysis, 379–80, 398
domain scientists, xi
dormouse monitoring, 96–98
dredging, 68, 154–55, 158, 235n13. *See also* questionable research practices (QRPs)
Dunlap, W. P., 314–15

echo chambers, 47–48
eddy-covariance technique, 7–8
Edwards, W., 318–19
effect size: ANOVA and, 314; anticipating, 57–58; bias in, 314; categorisation and, 62n17; complete data analysis and, 360–68; errors and, 58–59, 247, *248*; estimating, 57–58; expressions of, 57; federated analysis and, 344, 360–61, 369, 371; hypothesis testing and, 58–59; meta-analysis and, 342, 345–58, 370; post hoc analysis and, 328–29; power of statistical analysis and, 73–76, 83, 95, 100–1; retrospective power analysis and, 311–34; simulation basics and, *13*; standard deviation and, 24, 57–62, 76, *316*, 330n27, 352; study designs and, 47, 49, 53, 57–62, 67, 69
ELISA, 228n11
Elman network, 383–85
empirical cumulative density (ECD) plot, 284, *285*
entropy, 202, 386–87
errors: adjusting tests and, 124–26; advanced simulations and, 404; AIC and, 254–55; assumptions of statistical analysis and, 103, 105–13, 116–17, *122*, 123–28, 138, 147, 149; autocorrelated data and, 252; Bayesian models and, 246, 249–51; Benjamini-Hochberg procedure and, 126–30; bias and, 245–62; bootstrapping and, 245–46, 251–59, *260*; censored observations and, 228n11, 230; collinearity and, 252, 255, 258; confidence interval (CI) and, 249; correlation and, *248*, 249, 252–53, 257, 260; data entry, 260n34; degrees of freedom and, 254n29, 261; detrending and, 245; effect size and, 58–59, 247, *248*; evaluation and, 272–78, 287–88, 291–92, 294–96, 299, 302, 304, 377–81, 392; false discovery rate (FDR), 126–31; false negatives, 21, 57–59, 66, 73, 176, 228n11, 311, 318–24; false nulls, 69, 128–31; false positives, 34–35, 57–60, 67, 105, 123–24, 129n18, 145, 154, 159–65, 228n11, 320, 322; family-wise, 123–28; federated analysis and, 252n23; Gaussian distribution and, 245n18, 249; hallucinations, 9–10, 14; HARKing and, 67–68; hypothesis testing and, 58–59; imputation and, 253, 255, 258; large language models (LLMs) and, 9–10; M, 159–63, *165*, 331; machine learning and, 4, 33, 111, 202–4, 246, 249, 287, 294–95, 299n19, 302; matrices and, 246–50, 251n22, 255; mean squared, 116–17, 381; meta-analysis and, 350; missing data and, 217–20; model selection and, 154–71; model structure and, 256, 258; Monte Carlo simulation and, 180–86, 189, 249–51; multiple statistical steps and, 251–59; noise and, 245; normal distribution and, 246–47, 249n20; power of statistical analysis and, 73–74, 88, 90, 92–93; predictor variables and, 110–12, 174–80, 233, 238–39; probability and, 245, 249n20, 254; propagation of, 4, 103, 181n13, 211, 245–51, 378; questionable research practices (QRPs) and, 67–69; randomness and, 247–49, 253–55; recipes and, 252; regression analysis and, 110–12, 243–45; replicates and, 247, 256–57, 261; residual, 4, 33, 36, 38, 88, 90, 116–17, 176, 183–86, 204, 230, 244–48, 252, 261, 272, 275, 278, 287–88, 291–92, 392; retrospective power analysis and, 311–24, *327*, 328–34; R functions and, 413, 416, 421; rules-of-thumb and, 153–71, 174–86, 189,

192, 202–7; S, 159–62, *165*, 331; sample size and, 247, *248*, 259–60; seeds and, 246–47, 250, 253–56, 261–62; simulated data and, 252, 254; simulation basics and, 3–6, 10, 21–38; stepwise model and, 254–55; study designs and, 49, 57n9, 58–62, 66–67; time series and, 192; unbiased data and, 252, 257; uncertainty and, 245–60; variance-covariance matrix and, 33n22, 34
European Centre for Medium-range Weather Forecasting (ECMWF), 403–4
evaluation, 12; AIC and, 296, 389–97; analysis of variance (ANOVA) and, 398; artificial intelligence (AI) and, 307; autocorrelated data and, 281; Bayesian models and, 269–72, *274*, 283, 306n23, 387–89, 397; benchmark datasets and, 380–81; bias and, 273, *277*, 375–77, 381–82; binomial distribution and, 286, 295, *297*; bootstrapping and, 273, 275–77, 288, 294n16, 299–303, *304*, *306*; chemistry and, 276; classification and, 288, 297, 302–3, 397; code verification and, 377; collinearity and, 398; complex effects and, 276–81; conditional plots and, 276–81; confidence interval (CI) and, 270, *286*, 288, 290–307; Cook's distance and, 281, *282*; correlation and, 270–71, *285*, 378; covariance and, 378, 390; coverage comparison and, 301–2; data perturbation and, 389–90, 394, 397; degrees of freedom and, 272, 275, 278, 291–92, 296, 389–97; DHARMa and, 387–88; didactic example of, 271–73; dimensional analysis and, 379–80, 398; empirical cumulative density (ECD) plot and, 284, *285*; errors and, 272–78, 287–88, 291–92, 294–96, 299, 302, 304, 377–81, 392; frequentist analysis and, 306n23, 307, 387; *F*-statistic and, 272, 278, 292, 392; functional verification and, 377–78; Gaussian distribution and, 273, 281–83, 294–95, *297*, 307, 378, 382n5, 387; general linear models (GLMs) and, 295–96, 299, 384, 387–92, 396–97; homoscedasticity and, 281; information gleaned from model, 273–77; intellectual advancement and, 382; intuitive understanding and, 339, 382–89; iterative learning and, 305–6; Kolmorogov-Smirnov (K-S) test and, 286–87, 289; learning from prior, 270–73; likelihood and, 269, 273, 288, 307, 386, 389, 392–97; limits of, 397–98; machine learning and, 270, 276, 280, 283–84, 287–88, 294–305, 307, 380, 387, 389–90, 397; manipulation and, 307; marginal plots and, 276–81; matrices and, 273, 291, 293, 299, 376–77, 389–90; method comparison and, 381–82; model diagnostics and, 281–88; model-fitting methods and, 267; model structure and, 273, 301; Monte Carlo simulation and, 270, 292–96; neural networks and, 295n18, 302–4, 383–84, 389–96; noise and, 284, 288–91, 295, 299–301, 375, 389–91, 394–95; non-Gaussian data and, 294–95; normal distribution and, 287n10; normality and, 281, *283*; parameter values and, 267–76, 282n6, 294–97, 299n19, 301–2, 306, 384, 389–97; partial plots and, 276–81; peer review and, 68, 375; prediction interval and, 270, 288, 290–98, 302–7; predictor variables and, 288–305; probability and, 270–71, *274*, 285n9, 286, 288n11, 289, 290n13, 295, *297*, *301*, 305, 377; quantile residuals and, 286–87; R&D, 375; randomness and, 272, 282n6, 288–90, 293, 295–96, 299–304, 377–78, 389–98; recurrent neural network (RNN) and, 383–85; replicates and, 271, 277, 284, 288–89, 300, 303–5, *306*, 391, 394–95; residuals and, 267, 272, 275, 278–96, 283–88, 307, 383, 387–88, 392; robustness and, 376–77; sample size and, 302, 380, 398; seeds and, 272, 277–78, 284, 289, 296–99, 303, 305, 378, 391–92, 395; Shannon diversity and, 386; signals and, 375n2; simulated data and, 270, 282–84, 288, 299, 301, 380–82, 391; standard deviation and, *285*, 301, 377–79, 384; stopping rules and, 378; testing data and, 275; training data and, 275; unbiased data and, *277*, 381; uncertainty and, 269–75, *277*, 288–91, 294–96, *297*, 302, 305–7, 379; unit testing and, 376–79

fake data, 4n1, 47–48, 404. *See also* simulated data
false discovery rate (FDR), 126–31
false negatives: censored observations and, 228n11; errors and, 21, 57–59, 66, 73, 176, 228n11, 311, 318–24; power of statistical analysis and, 73; predictor variables and, 176; retrospective power analysis and, 311, 318–24; study designs and, 57–59, 66. *See also* type-II errors
false nulls, 69, 128–31
false positives: assumptions of statistical analysis and, 105, 123–31, 145; avoiding, 35; censored observations and, 228n11; critical level of, 58; errors and, 34–35, 57–60, 67, 105, 123–24, 129n18, 145, 154, 159–65, 228n11, 320, 322; inflation of, 105, 123–31; model selection and, 154, 159–64, *165*; multiple testing and, 123–31; retrospective power analysis and, 320, 322; study designs and, 57–60, 67. *See also* type-I errors
Faul, F., 327–29, 330n27, 333
Fazio, L. K.: bootstrapping and, 64; illusory truth effect and, 48–52, 60–65; plausibility and, 61; 68; study designs and, 48–55, 60–65, 66n22, 68
federated analysis: bias and, 341–45, 357–58, *361*, 362, 368–71; common model identification and, 354; complete data analysis and, 341–45, *361*, 362–63, 368; correlation and, 358; data distribution and, 354; effect size and, 344, 360–61, 369, 371; embargoed datasets and, 370; errors and, 252n23; matrices and, 359; meta-analysis and, 12,

federated analysis (*continued*)
339, 341–45, 369; model application and, 354–57; multi-step, 358–60, 368, 370; one-step, 354–58, *361*, 370; parameter estimates and, 343, 360, 368–69, 371; predictor variables and, 341–45, 354, 357–62, 368–69; replicates and, 359–60; seeds and, 359; two-step, 353, 363
Fibonacci sequence, 377
Fieberg, J. R., 10
fire ants, 142–49
fitted-model methods, 28–29
Fitzpatrick, M. C., 142
fixed effects: assumptions of statistical analysis and, 133–34, 138; randomness and, 132–33, 153, *351*, *356*; rules-of-thumb and, 153
FLUXNET, 7–8
fMRI imaging, 3, 375
`for`-loops, 410–13
Freckleton, R. P., 243
frequency distribution, 20, 50
frequentist analysis: assumptions of statistical analysis and, 103; Bayesian models and, 10, 103, 182, 306n23, 307, 312, 387; evaluation and, 306n23, 307, 387; Monte Carlo simulation and, 182, 188; retrospective power analysis and, 312; simulation basics and, xii, 10; study designs and, 49
F-statistic: ANOVA and, 20–21, 116; assumptions of statistical analysis and, 116; censored observations and, 230; evaluation and, 272, 278, 292, 392; homoscedasticity and, 116; Monte Carlo simulation and, 183, 185–86; regression analysis and, 244–45
functional verification, 377–78. *See also* code verification; evaluation; unit testing
F-values, 20–21, *23*, 117, *122*

Galileo, 341
gamma-ray bursts, 37–42
Gaussian distribution: advanced simulations and, 405; assumptions of statistical analysis and, 110–14, 131–32, 137n25; complete data analysis and, 231; errors and, 245n18, 249; evaluation and, 273, 281–83, 294–95, *297*, 307, 378, 382n5, 387; machine learning and, 202; meta-analysis and, 346; Monte Carlo simulation and, 185, 249; noise and, *113*, 114, 185, 346; power of statistical analysis and, 74n3, 92; random, 22–23, 51, 52n5, 74n3, 202, 378, 409; R functions and, 409; simulation basics and, 23, 29–33; standard deviation and, 23; study designs and, 51, 52n5
Gelman, A., 327–34
general additive mixed model (GAMM), 88
generalized least squares (GLS), 34–36
general linear models (GLMs): assumptions of statistical analysis and, 103, 145, 147; bootstrapping and, 30; evaluation and, 295–96, 299, 384, 387–92, 396–97; hypothesis testing and, 3, 10, 14; machine learning and, 203; meta-analysis and, 344; missing data and, 217–25; model selection and, 155–56, 158, 160, 163, 166; Monte Carlo simulation and, 29; predictor variables and, 3, 174, 176, 178, 232–41; principal component analysis (PCA) and, 3; R functions and, 418; simulation basics and, xii, 3, 29–30; threshold effect and, 103; uncertainty in, 30
generative adversarial networks (GANs), 8–9
Gijbels, I., 224n8
González-Moreno, P., 142
Goodman, S. N., 314–16, 321–22
Google, 304
Gould, E., 331–32
G*Power, 327–29, 330n27
GraphCast model, 304

hallucinations, 9–10, 14
happiness, 127
Hardin, J., 10
HARKing (Hypothesizing After the Results are Known), 67–68. *See also* questionable research practices (QRPs)
Hartig, F., 387
Heisey, D. M., 316, 323
Hessian matrices, 28
heteroscedasticity: ANOVA and, 20–21, 118–22, 154; assumptions of statistical analysis and, 116, 118, 121–23, 145; Monte Carlo simulation and, 180–90, 205; normality and, 116; post hoc analysis and, 123, 145
hierarchical data: assumptions of statistical analysis and, 133–41; multiple analysis of, 134, 139–41
Hoening, J. M., 316, 323
homoscedasticity: ANOVA and, *13*, 105–6, 116–23; assumptions of statistical analysis and, *13*; evaluation and, 281; *F*-statistic and, 116; mixed-effect models and, 118; normality and, 9, 116, 281; testing effects of, 118–23

hypothesis testing: advanced simulations and, 406; assumptions of statistical analysis and, 103; compromise power analysis and, 328; confident interval (CI) and, 334; errors in, 58–59; GLM and, 3, 10, 14; machine learning and, 203; PCA and, 3; reliable, 59; replicates for, 59; retrospective power analysis and, 328, 334; study designs and, 58–60. *See also* type-I errors; type-II errors

`if-then-else statements`, 410, 421
illusory truth effect: adequate replication and, 56–60; Fazio and, 48–52, 60–65; frequency distribution and, 50; hypotheses definitions and, 49–55; null expectation and, 52, 55; perceived accuracy and, 51–55, *56*; prior expectations and, 331–32; processing fluency and, 54–55, 62; quantifying variables of interest, 50–55; reach of, 48; repeated statements and, 48; sample size and, 60–66;

study designs and, 48–52, 55–56, 60–66; testing hypotheses and, 55–67
immigration, 110–11, *113*, 332, 406
imputation: censored observations and, *224*, 225, 232; errors and, 253, 255, 258; issues with, 214–15; missing data and, *13*, 103, 155n5, 156, 160, 175, 211–22, *224*, 225, 232–34, 253–55, 258; model selection and, 155n5, 156, 160; predictor variables and, 175, 213–22, 232–34; qualitative comparison and, 215–19; retrospective power analysis and, 311; simulation basics and, *13*. *See also* data gaps; missing data
individual participant-level (IPL) meta-analysis, 352–55, *356*, 360, 368. *See also* meta-analysis
interocular traumatic test, 318–19intuitive understanding: evaluation and, 339, 382–89; Monte Carlo simulation and, 187n18; multiple descriptions and, 383–86; recurrent neural network (RNN) and, 383–85; result presentation and, 386–87
invented data, 4n1. *See also* fake data; simulated data
iterative learning, 305–6

jackknife, 17, 299–303, *304*. *See also* bootstrapping
James Webb Space Telescope, 73n1
Jena Experiment, 113–14, 119, *120*
`Julia`, xii, 10

Keeling, C. D., 85
Kolmorogov-Smirnov (K-S) test: assumptions of statistical analysis and, 106, 135n24; evaluation and, 286–87, 289; model variability and, 28; normality and, 106; quantile residuals and, 286–87; robustness of, 26–28
Kreyling, J., 179
Kriegeskorte, N., 3
Kruskal-Wallis test, 120

Lane, D. M., 314–15
large language models (LLMs), 9, 14
Laser Interferometer Gravitational-Wave Observatory (LIGO), 404–6
Law of Small Numbers, 81
left-censored data, 222, 226–27, 230
Lev, J., 313
LIDAR, 406
likelihood: bootstrapping and, 39–40, 43, 204, 273; censored observations and, 229; curvature of, 28; evaluation and, 269, 273, 288, 307, 386, 389, 392–97; log, 38–41, 202, 204, 273, 288, 389, 392–97; machine learning and, 202, 204; maximum, 28, 190, 314; Monte Carlo simulation and, 187, 189; null hypothesis and, 39, 41; retrospective power analysis and, 314, 331; simulation basics and, 10, 14; study designs and, 45, 50; time series and, 190
likelihood-ratio test (LRT), 38–40, 43
Lilliefors test, 26–28, 106
Llama, xii, 9
lmodel2, 111
Locally Interpretable Model-agnostic Explanations (LIME), 307
logical operators, 421
log-normal distribution, 31–32, 139n27
Lyell, C., 138n26

machine learning: artificial intelligence (AI) and, 201, 205; assumptions of statistical analysis and, 103, 105, 111; autocorrelated data and, 203–4; Bayesian models and, 203n33; bias and, 201–3; Big Data and, 153, 200–4; bootstrapping and, 204; censored observations and, 226, 232; chemistry and, 276; classical statistics and, *13*; classification and, 202; confidence interval (CI) and, 203–4, 295–305; correlation and, 203–4; distributional assumptions and, 201–3; errors and, 4, 33, 111, 202–4, 246, 249, 287, 294–95, 299n19, 302; evaluation and, 270, 276, 280, 283–84, 287–88, 294–305, 307, 380, 387, 389–90, 397; extrapolation and, 203–4; Gaussian distribution and, 202; general linear models (GLMs) and, 203; hypothesis testing and, 203; likelihood and, 202, 204; missing data and, 212, 214; model selection and, 165–72; non-independence and, 201–3; non-linear functions and, 203; predictor variables and, 165–72, 203–4, 232, 238; probability and, 202; *p*-values and, 203; randomness and, 201–4; regression analysis and, 165–72; residuals and, 204; R functions and, 419; robustness and, 201–2; rules-of-thumb and, 153–54, 165–720, 200–5; sample size and, 202; simulation basics and, 17, 33; unbiased data and, 201; uncertainty and, 203; workflow and, 4
Mair, M. M., 324–25
manipulation: advanced simulations and, 404; assumptions of statistical analysis and, 112–16; categorisation and, 177–80; combinations of, 9; evaluation and, 307; predictor variables and, 112–16, 177; simulation basics and, 9; study designs and, 45, 49
"Many Analysts, One Data Set" 331–32
marginal plots, 276–80
Markov chain Monte Carlo (MCMC) simulation, 50n4, 249–50, 389
Mathematica, 10
mathematicians, 4, 6, 18, 29n17
`Matlab`, 10, 377
matrices: assumptions of statistical analysis and, 128–32; binary, 377; censored observations and, 223; complete data analysis and, 362–66; covariance, 243, 272, 346, 355, 363–64, 366, 409; errors and, 246–50, 251n22, 255; evaluation and, 273, 291, 293, 299, 376–77, 389–90; federated analysis and, 359; hat, 389–90; Hessian, 28; meta-analysis and, 346–50, 355, 357; missing data and, 220–21; model selection and, 161–63, 166;

matrices (*continued*)
power of statistical analysis and, 79–88, 95; predictor variables and, 175, 236, 240–41; random effects and, 132; regression analysis and, 243–44; R functions and, 409, 411–14; study designs and, 61, 66; time series and, 198; variance-covariance, 33n22, 34, 243n17, 248–50, 293; workflow and, 212n1, 220–23, 236, 240–50, 251n22, 255
mean squared errors, 116–17
mean squared prediction error (MSPE), 381
median retrospective power (MRP), 350–52
M-error: retrospective power analysis and, 329–31; rules-of-thumb and, 159–60, 162–63, *165*
meta-analysis: autocorrelated data and, 343; bias and, 341–58, 369– 370; Big Data and, 341; classical, 341–45, 352–57, 360, 368, 370; complete data analysis and, 341–45; correlation and, 343, 346; covariance and, 343, 346, 355; data distribution and, 347; data-generating model (DGM) and, 345; data sources for, 341–42; distributed studies and, 345–46; effect size and, 342, 345–58, 370; errors and, 350; federated analysis and, 12, 339, 341–45, 369; Gaussian distribution and, 346; general linear models (GLMs) and, 344; improved, 343, 345, 370; individual participant-level (IPL), 352–55, *356*, 360, 368; individual studies and, 347–48; large dataset and, 346–47; matrices and, 346–50, 355, 357; median retrospective power (MRP) and, 350–52; mixed-effect models and, 344, 353; model structure and, 344, 368; noise and, 346; normal distribution and, 346; parameter values and, 343, 352; post hoc analysis and, 350, 352; predictor variables and, 343–60; privacy and, 368; probability and, 350, 352; *p*-values and, 345; randomness and, 347–50, *351*, 353–56; repeated simulations and, 348–50; replicates and, 344, 355; retrospective power analysis and, 350–52; robustness and, 341, 344, 369; sample mean and, 342; sample size and, 342–49, 352, 357, 370; seeds and, 346, 349, 355–56; signals and, 342n3; simulated data and, 346, 354, *369*; standard deviation and, 352; stepwise model selection and, 345, *351*, 354; unbiased data and, 342, *343*, 347n11, 352, 354, 370
Michaelis-Menten equation, 114–15, 179, 380
minimum detectable differences (MDDs), 322–26, 333
minimum detectable effects (MDEs), 322–26, 333
missing at random (MAR) data, 213, 215, 222, 225, *226*. *See also* data gaps; imputation; missing data
missing completely at random (MCAR) data, 213, 215, 220–23, *224*, 253. *See also* data gaps; imputation; missing data
missing data: AIC and, 217–19; bias and, 212–22; confidence interval (CI) and, *220*; correlation and, 214–15, 222; degrees of freedom and, 217–19; errors and, 217–20; general linear models (GLMs) and, 217–25; imputation of, *13*, 103, 155n5, 156, 160, 175, 211–22, *224*, 225, 232–34, 253–55, 258; machine learning and, 212, 214; matrices and, 220–21; missing at random (MAR), 213, 215, 222, 225, *226*; missing completely at random (MCAR), 213, 215, 220–23, *224*, 253; missing not at random (MNAR), 213, 222; parameter values and, 213, 217–22; predictor variables and, 213–19; probability and, 212–13, 222; *p*-values and, 215; qualitative comparison and, 215–19; random values and, 215; residuals and, 217–19; sample size and, 212, 219–20; seeds and, 220; types of, 212–13; unbiased data and, 213n2, 219
missing not at random (MNAR) data, 213, 222. *See also* data gaps; imputation; missing data
mixed-effect models: assumptions of statistical analysis and, 131–37; homoscedasticity and, 118; hyper-distributions and, 131–37; linear (LMMs), 131–34; meta-analysis and, 344, 353; randomness and, 153; rules-of-thumb and, 153
model-fitting methods: conditional plots and, 276–81; Cook's distance and, 281, *282*; diagnostics and, 281–88; evaluation and, 267–77 (*see also* evaluation); GLMs and, 10 (*see also* general linear models (GLMs)); marginal plots and, 276–81; obscuring of, 277–78; partial plots and, 276–81; robustness and, 39; subset regression and, 154; *t*-test and, 10, 14, 24, 29–32, 57n9, 59–62, 74–77, 103, *120*, 124, 127, 138, 324, 418–21; uncertainty and, 28–30; workflow and, 211; χ^2 statistics, 10, 14
model selection: AIC and, 155, 157–59; bias and, 154n2, 159–60, 171; collinearity and, 165–72; correlation and, 164–67; degrees of freedom and, 155, 157–58, 160; dredging and, 154–55, 158; effect size and, 154, 159–64, *165*; errors and, 154–71; false positives and, 154, 159–64, *165*; general linear models (GLMs) and, 155–56, 158, 160, 163, 166; improved interpretation and, 154–64; imputation and, 155n5, 156, 160; interpretability and, 164; machine learning and, 165–72; matrices and, 161–63, 166; model structure and, 155–60; Monte Carlo simulation and, 154; parameter values and, 157, 159–60, 165–66, 171–72, 371; predictor variables and, 165–72; principal component analysis (PCA) and, 165n7; probability and, 155, 164, 171; *p*-values and, 160, 163; randomness and, 166–72; replicates and, 160, 163, 170; residuals and, 155–58, 168–69; sample size and, 164; seeds and, 160, 167, 170; simulated data and, 165–67, *172*; stepwise, 154–55, 157, 159–60, 164–65, 236, 254–55, 345, *351*, 354, 362; structural correctness and, 155–60; true positives and,

159–64, *165*; unbiased data and, 159, 171; uncertainty and, 171
model structure: assumptions of statistical analysis and, 133, 149; censored observations and, 232; definition of, 159; errors and, 256, 258; evaluation and, 273, 301; meta-analysis and, 344, 368; model selection and, 155–60; predictor variables and, 240
Monte Carlo simulation: autocorrelated data and, 181, 183; Bayesian models and, 182, 187–89; bootstrapping and, 181n14; breakpoint and, 182n15, 189; carbon dioxide (CO_2) and, 181–90; classification and, 29n17; confidence interval (CI) and, *182*, 293; correlation and, 181, 183; degrees of freedom and, 183, 185–86; errors and, 180–86, 189, 249–51; evaluation and, 270, 292–96; fitted-model methods and, 29; frequentist analysis and, 182, 188; *F*-statistic and, 183, 185–86; Gaussian distribution and, 185, 249; general linear models (GLMs) and, 29; heteroscedasticity and, 180–90, 205; intuitive understanding and, 187n18; likelihood and, 187, 189; Markov chain (MCMC), 50n4, 249–50, 389; model selection and, 154; noise and, 181–86; normal distribution and, 184, 187; normality and, 182–83; null hypothesis and, 184; observed trends and, 184–87; parameter values and, 181n13, 190; plausibility and, 188–89; probability and, *181*, 184, 187; randomness and, 181, 212–22; reference distribution and, 184–87; regression analysis and, 182–84; replicates and, 181, 185–86; residuals and, 183–86; sample size and, 180; seeds and, 185–86; simulated data and, 184–86; standard deviation and, 184, 187; time series and, 182–84; uncertainty and, 181–84, *188*
Mueller, D. K., 228n11
multiple testing: HARKing and, 67; inflation of false positives, 123–31

Nakagawa, S., 377
NASA (National Aeronautics and Space Administration), 342n3
NEON (National Ecological Observatory Network), 406
Neptune, 98
"network of chums" effect, 344
neural networks: advanced simulations and, 406; convolutional (CNN), 8; data augmentation and, 8–9; evaluation and, 295n18, 302–4, 383–84, 389–96; generative adversarial (GAN), 8–9; recurrent (RNN), 383, *384*
neurobiology, xii, 1
Neyman, J., 22n6, 313
niche shifts: assumptions of statistical analysis and, 142–49; demonstrating spurious, 143–44; species invasion and, 144–45
Nightingale, F., 341
noise: advanced simulations and, 404–7; assumptions of statistical analysis and, 108–15; autocorrelated data and, 34, 86–87, 91; convolutional neural networks (CNNs) and, 8; errors and, 245; evaluation and, 284, 288–91, 295, 299–301, 375, 389–91, 394–95; generative adversarial networks (GANs) and, 8–9; meta-analysis and, 346; Monte Carlo simulation and, 181–86; power of statistical analysis and, 73, 86–87, 91–95, 100–1; random, 40–41, 100, 301; signals and, 5, 87, 405; study designs and, 49; time series and, *197*, 198
normal distribution: assumptions of statistical analysis and, 106–9, 116, 131–32, 135, 137, *140*, *142*; Central Limit Theorem and, 131; errors and, 246–47, 249n20; evaluation and, 287n10; Gaussian, 22 (*see also* Gaussian distribution); log, 31–32, 139n27; meta-analysis and, 346; Monte Carlo simulation and, 184, 187; power of statistical analysis and, 74–78; random values and, 22–23; regression analysis and, 243–44; retrospective power analysis and, 315, 317, 330; R functions and, 412; simulation basics and, 22–33; study designs and, 53
normality: ANOVA and, 18; assumptions of statistical analysis and, 105–9, 116, 132–37, 142, 145, 148; Cook's distance and, 281, *282*; evaluation and, 281, *283*; heteroscedasticity and, 116; homoscedasticity and, 9, 116, 281; K-S test and, 26–28; Monte Carlo simulation and, 182–83; OLS and, 182; testing, 135–37; testing for, 106–9; *t*-test and, 29
notational conventions, 4, 11–12, *143*, 382n5
NSF (National Science Foundation), 406
null expectation, 52, 55
null hypothesis: ANOVA and, 118; assumptions of statistical analysis and, 118, 126–28; false discovery rate (FDR) and, 126–31; likelihood and, 39, 41; Monte Carlo simulation and, 184; randomness and, 39; retrospective power analysis and, 311, 313, 323, 327, 329, 332; study designs and, 56–61, 69
null models, 9, *56*, 64
Nutrient Network, 346n9

OpenIntro project, 10
ordinary least squares (OLS): normality and, 182; regression analysis and, 34–36, 111, 115n12, 254–55, 285; residuals and, 36, 285

parameter values: advanced simulations and, 404, 419; assumptions of statistical analysis and, 103, 106, 107n2, 110, 114–16, 119, 131–35, 139–40, *142*, 145, 148; bias and, 4–5, 69, 81, 115, 145, 159, 171–73, 211, 213, 222, 226, 232, 235, *242*, 252–59, 362, 368–69, 371; censored observations and, 226, 232; complete data analysis and, 360, 362, 368; degrees of freedom and, 389–97; dispersion, 7, 148, 157, 217–19, 238–39, 296; evaluation and, 267–76, 282n6, 294–97, 299n19, 301–2, 306, 384, 389–97; meta-analysis and, 343,

parameter values (*continued*)
352; missing data and, 213, 217–22; model selection and, 157, 159–60, 165–66, 171–72, 371; Monte Carlo simulation and, 181n13, 190; notational conventions for, 11; population-level, 11; power of statistical analysis and, 74n3, 79–87, 90–93; predictor variables and, 173, 176, 234–42; regression, *7*, 110, *142*, 301; regression analysis and, 243–46, 249–61; retrospective power analysis and, 315, 319–21, 325, 329, 332, 334; simulation basics and, 4–5, *7*, 11–12, 18–21, 24–31, 33n22, 35–36, 40–42; study designs and, 49–50, 55n7, 56, 60–61, 68–69, 369; time series and, 191; workflow and, 211, 213, 217–19, *220*, 222, 226, 232–46, 249–61
partial plots, 276–80
Pearson, E. S., 313
Pearson's correlation coefficient, 34, 74, 76
peer review, 68, 375
perceived accuracy, 51–55, *56*
Perugini, M., 77–79, 81, *83*
Pima Indian Diabetic Disease dataset, 214
pipelines: advanced simulations and, 403–5; artefacts and, 211, 228n11, 245, 251–58; simulation basics and, 8, 12; workflow and, 8, 12, 211, 228n11, 245, 251–58, 403–5
Planet 9, 98–99
plausibility: Fazio and, 61; Monte Carlo simulation and, 188–89; study designs and, 50–56, 61–62, 66
Pluto, 98n20
pollinators, 9
Pólya, G., 18–20
post hoc analysis: assumptions of statistical analysis and, 118, 123, 145; compromise power analysis, 328; confidence intervals (CI) and, 319–22; criterion analysis and, 328; design analysis and, 329–32; effect size and, 328–29; federated analysis and, 341–45, 354, 357–62, 368–69; G*Power and, 327–29, 330n27; heteroscedasticity and, 123, 145; interocular traumatic test and, 318–19; meta-analysis and, 350, 352; minimum detectable differences (MDDs) and, 322–26, 333; minimum detectable effects (MDEs) and, 322–26, 333; need for, 318–19; predictor variables and, 179, 320–22; retrospective power analysis and, 312–15, 318–34; sensitivity analysis, 328; simulation basics and, 13, *13*; study designs and, 67, 69
posterior probability distribution, 271, 305, 312, 334. *See also* Bayesian models
power of statistical analysis: autocorrelated data and, 86–92; Bayesian models and, 93, 98; bias and, 74, 78–85, 88, *89*, 97, 101; bootstrapping and, 78–79, *81*, *83*; carbon dioxide (CO_2) and, 85–87, 89n15, 90–96, 99n22; confidence interval (CI) and, 78, 80, *81*, *83*; correlation and, 74–82, *83*, 86–88, 91–95, 101; data-generating model (DGM) and, 80; data points for correlation, 76–80; degrees of freedom and, 90; effect size and, 73–76, 83, 95, 100–1; errors and, 73–74, 88, 90, 92–93; false negatives and, 73; Gaussian distribution and, 74n3, 92; impact assessment and, 99–100; improving estimates and, 91–100; matrices and, 79–88, 95; monitoring atmospheric carbon dioxide (CO_2) and, 85–95, *96*, 99n22; noise and, 73, 86–87, 91–95, 100–1; normal distribution and, 74–78; parameter values and, 74n3, 79–87, 90–93; probability and, 73–74, 84, 91, 97; puerperal fever deaths study and, 81–84; *p*-values and, 81, 84, 95, *96*; randomness and, 74–76, 100; "recruit until significant" and, 80–85; reliability and, 78; replicates and, 77, 80, 82, 86–89, 94–95; residuals and, 87–92; R functions and, 75; sample size and, 73–88, 98, 101; seeds and, 76–77, 80–95; signals and, 87; simple group comparisons and, 74–76; simulated data and, 76, 83n8, 87; stability and, 78; standard deviation and, 74–76, 96; stopping rules and, 74, 81–85, *87*, *89*, 95, 101; time series and, 85–91; *t*-test and, 74–77; type-II errors and, 73–74; unbiased data and, 74, 79, 83–85, 97; uncertainty and, 78, 91–93. *See also* type-II errors
prediction interval: confidence interval (CI) and, 203–4, 270, 288–307; evaluation and, 270, 288, 290–98, 302–7; retrospective power analysis and, 321n19
predictor variables: AIC and, 238–39; analysis of variance (ANOVA) and, 179; assumptions of statistical analysis and, 137–45; Bayesian models and, 232; bias and, 171–73, 232–42; carbon dioxide (CO_2) and, 177; categorisation of, 154, 172–80, 205; collinearity and, 238; complete data analysis and, 360–68; confidence interval (CI) and, *174*, 288–305, 320–22; correlations among, 137–46, 149, 235–42; data-generating model (DGM) and, 236; degrees of freedom and, 176, 238–39; errors and, 110–12, 174–80, 233, 238–39; evaluation and, 288–305; false negatives and, 176; general linear models (GLMs) and, 3, 174, 176, 178, 232–41; imputation and, 175, 213–22, 232–34; machine learning and, 165–72, 203–4, 232, 238; manipulation and, 112–16, 177; matrices and, 175, 236, 240–41; mean squared prediction error (MSPE), 381; meta-analysis and, 343–60; missing data and, 213–19; model selection and, 165–72; model structure and, 240; non-Gaussian data and, 294–95; parameter values and, 173, 176, 234–42; post hoc analysis and, 179; pre-selecting, 232–42; randomness and, 236, 238–41, *242*; random values and, 112; regression analysis and, 110–12; replicates and, 178–79, 234, 241; residuals and, 176, 238–39; retrospective power analysis and, 320–22; robustness and, 174, 238; sample size and, 236; seeds and, 178,

233–37, 241; simulated data and, 232, 239; stepwise model selection and, 236; true positives and, *180*; unbiased data and, 172, *235*, 238, *242*
pre-registration, 66–69, 267, 311, 314n8, 342n4, 358, 369
principal component analysis (PCA): assumptions of statistical analysis and, 142, 144–46; general linear models (GLMs) and, 3; model selection and, 165n7; niche shifts and, 142, 144–46
prior probability distribution, 22n6, 45, 50, 270, 313n5. *See also* Bayesian models
probability: assumptions of statistical analysis and, 123–24, 126–27, 145; censored observations and, 225, 228n11; detection, 97; errors and, 245, 249n20, 254; evaluation and, 270–71, *274*, 285n9, 286, 288n11, 289, 290n13, 295, *297*, *301*, 305, 377; improving estimation and, 91–100; likelihood and, 14 (*see also* likelihood); machine learning and, 202; meta-analysis and, 350, 352; missing data and, 212–13, 222; model selection and, 155, 164, 171; Monte Carlo simulation and, *181*, 184, 187; Pólya urn and, 18–20; power of statistical analysis and, 73–74, 84, 91, 97; retrospective power analysis and, 311–22, 329–34; R functions and, 409; selection, 97; simulation basics and, *7*, 10, 12, 14, 17, 18n1, 20, 22n6, 26, 29n17, 33n20, 37; study designs and, 45, 50, 56, 58, 66–67, 69; time series and, 190, *197*
processing fluency, 54–55, 62. *See also* illusory truth effect
Programme for International Student Assessment (PISA), 132
prospective power analysis, 67, 73, 312–13, 327
puerperal fever deaths, 81–84, Semmelweis and, *81*
p-values: assumptions of statistical analysis and, 107, 123, 126–27, 145, 148; machine learning and, 203; meta-analysis and, 345; missing data and, 215; model selection and, 160, 163; power of statistical analysis and, 81, 84, 95, *96*; retrospective power analysis and, 34n23, 312–16, 318n14, 320–21, 333–34; simulation basics and, xii, 5, 20, 27–28; study designs and, 58n12, 67; workflow and, 211, 215. *See also* hypothesis testing; type-I errors;
`Python`, xii, 10, 377, 380

QA/QC (quality assurance and quality control) procedures, *8*, 342
Qucit, 295, 299–300, 302
questionable research practices (QRPs), 67–69

R&D (Research & Development), 375
random effects: assumptions of statistical analysis and, 132–33, 135, 137–38, 141, *142*; censored observations and, 229; matrices and, 132; rules-of-thumb and, 153
random-forest models: assumptions of statistical analysis and, 145; errors and, 255; evaluation and, 288–90, 295n18, 299–304, 390–94; model selection and, 166–72; predictor variables and, 238–41
randomness: assumptions of statistical analysis and, 105, 107, 112, 118, 131–38, 141–47; breakpoint and, 37–42; complete data analysis and, 222, *224*, 225, *226*, 229, *361*, 363–64, 366; data perturbation and, 40–43; errors and, 247–49, 253–55; evaluation and, 272, 282n6, 288–90, 293, 295–96, 299–304, 377–78, 389–98; fixed effects and, 132–33, 153, *351*, *356*; Gaussian, 22–23, 51, 52n5, 74n3, 202, 378, 409; machine learning and, 201–4; meta-analysis and, 347–50, *351*, 353–56; missing data and, 212–22; mixed-effect models and, 153; model selection and, 166–72; Monte Carlo simulation and, 181; noise and, 40–41, 100, 301; null hypothesis and, 39; power of statistical analysis and, 74–76, 100; predictor variables and, 236, 238–41, *242*; resampling and, 35, 37, 40n28, 41, 43, 416–18; retrospective power analysis and, 311n1, 319, 321n20; R functions and, 409–11, *412*, 416–20; seeds and, 4 (*see also* seeds); simulated data vs., 35–43; simulation basics and, 3–5, 8–11, 17–19, 22–24, 27–43; study designs and, 51–52, 55n8, 57, 61; uncertainty and, 29, *31*, 39, 249, 254
random values: Gaussian distribution and, 22–23; missing data and, 215; Monte Carlo simulation and, 181; multivariate, 409; predictor variables and, 112; R functions and, 409–10, *412*, 418
reality, 5, *58*
recipes, xi, 1, 153, 252
recurrent neural network (RNN), 383–85
registered reports, 68
regression analysis: bias and, 171, 242–45; bootstrapping and, 29, *31*, 40, 64–66, 300–2, *304*, 416; confidence interval (CI) and, 243; correlation and, 33, 58, 106, 153, 164–65, 181, 183, 215, 243; covariance and, 243–50; degrees of freedom and, 244–45; detrending and, 190–200; dilution of, 110–15; errors and, 110–12, 243–45; *F*-statistic and, 244–45; generalized least squares (GLS), 34–36; machine learning and, 165–72; matrices and, 243–44; Monte Carlo simulation and, 182–84; multiple regression and, xii, 153, 212, 243–47, *248*, 291, 347; normal distribution and, 243–44; ordinary least squares (OLS), 34–36, 111, 115n12, 254–55, 285; parameter values and, 243–46, 249–61; predictor variables and, 110–12; residuals and, 242–45; sample size and, 243; seeds and, 244; segmented, 37–41; subset, 154–55, 397n13; time series and, 182–84, 190–200; unbiased data and, 243
replicate function, 416–17
replicates: analysis of, 89–91; assumptions of statistical analysis and, 108–9, 112, *113*, 124–30, 134, 139, 141, 146; censored

replicates (*continued*)
observations and, 223, 225; complete data analysis and, 363, 366; errors and, 247, 256–57, 261; evaluation and, 271, 277, 284, 288–89, 300, 303–5, *306*, 391, 394–95; federated analysis and, 359–60; meta-analysis and, 344, 355; model selection and, 160, 163, 170; Monte Carlo simulation and, 181, 185–86; power of statistical analysis and, 77, 80, 82, 86–89, 94–95; predictor variables and, 178–79, 234, 241; retrospective power analysis and, 319, 320n18; R functions and, 410, 416–21; simulation data and, 4, 19n2, 22–32, 40–43, 89–91; study designs and, 56–59, 65, 73; time series and, 194, 197–98
resampling: bootstrapping and, 17, 40, 43, 416–18; jackknife and, 17, 299–303, *304*; random, 35, 37, 40n28, 41, 43, 416–18; R functions and, 416–21; signals and, 40
residuals: ANOVA and, 18, 116–17; assumptions of statistical analysis and, 3, 106–9, 116–17, 121, 132–35, 138–41, 145, 147–48; autocorrelated data and, 33, 87–88, 91–92, 183, *193*, 204, 281; breakpoint and, 40, 42; censored observations and, 230; data perturbation and, 41; empirical cumulative density (ECD) plot and, 284, *285*; errors and, 4, 33, 36, 38, 88, 90, 116–17, 176, 183–86, 204, 230, 244–48, 252, 261, 272, 275, 278, 287–88, 291–92, 392; evaluation and, 267, 272, 275, 278–96, 307, 383, 387–88, 392; machine learning and, 204; missing data and, 217–19; model selection and, 155–58, 168–69; Monte Carlo simulation and, 183–86; OLS, 36, 285; partial plots and, 279; power of statistical analysis and, 87–92; predictor variables and, 176, 238–39; quantile, 283–88; regression analysis and, 242–45; rules-of-thumb and, 155–58, 168–69, 176, 183–86, 191, *193*, 198–99, 204; simulated data and, 4; skewed, 29–32; standard deviation and, 40, 117, 283, *285*; time series and, 191, *193*, 198–99; workflow and, 212, 217–19
retrospective power analysis, 12; analysis of variance (ANOVA) and, 314–15, *316*; Bayesian models and, 312, 319, 331, 334; bias and, 314, 318, 320n18, 329, *331*; chemistry and, 318; compromise power analysis, 328; concept of, 313–18; confidence interval (CI) and, 311n1, 313, 319–26, 333–34; correlation and, 312; criterion analysis, 328; degrees of freedom and, 314n7, 319–20, 330, 332; design analysis and, 329–32; discredited standard approach and, 313; effect size and, 311–34; errors and, 311–24, *327*, 328–34; false negatives and, 311, 318–24; false positives and, 320, 322; frequentist analysis and, 312; G*Power and, 327–29, 330n27; hypothesis testing and, 328, 334; imputation and, 311; interocular traumatic test and, 318–19; lack of recommendation of, 312; likelihood and, 314, 331; median retrospective power (MRP) and, 350–52; meta-analysis and, 350–52; minimum detectable differences (MDDs) and, 322–26, 333; minimum detectable effects (MDEs) and, 322–26, 333; normal distribution and, 315, 317, 330; null hypothesis and, 311, 313, 323, 327, 329, 332; parameter values and, 315, 319–21, 325, 329, 332, 334; post hoc analysis and, 312–15, 318–34; prediction interval and, 321n19; predictor variables and, 320–22; probability and, 311–22, 329–34; problems with, 312–18, 332–34; prospective power analysis and, 67, 312–13, 327; *p*-values and, 34n23, 312–16, 318n14, 320–21, 333–34; randomness and, 311n1, 319, 321n20; replicates and, 319, 320n18; sample mean and, 319; sample size and, 311–13, 316, 319–30; seeds and, 315, 332; sensitivity analysis, 328; signals and, 331n30; standard deviation and, 314n7, 315, *316*, 319, 330n27, 332; true positives and, 324, 328n25; *t*-test and, 324; type-II errors and, 311–15, 328, 333; unbiased data and, 320n18, *331*; uncertainty and, 334
R functions: apply, 413–16; bootstrapping and, 416–19; confidence interval (CI) and, 419, 421; convenient, 418–21; data perturbation and, 416; data wrangling, 10, 238, 421; dedicated simulation packages, 421–22; drawing random values from distribution, 409–10; errors and, 413, 416, 421; `for`-loops and, 410–13; Gaussian distribution and, 409; general linear models (GLMs) and, 418; `if`-`then`-`else` statements, 410, 421; `lapply`, 413; logical operators, 421; machine learning and, 419; `mapply`, 415; matrices and, 409, 411–14; normal distribution and, 412; power of statistical analysis and, 75; probability and, 409; randomness and, 409–11, *412*, 416–20; random values and, 409–10, *412*, 418; repetition, 410–17; `replicate`, 416; replicates and, 410, 416–21; resampling and, 416–21; `sample`, 417; sample mean and, 419, 421; `sapply`, 413; seeds and, 409–11, 416–21; shuffling, 417–18; `simulate`, 418; simulation basics and, 38n26; study designs and, 62n17, 65; `system.time`, 418–21; `Sys.time`, 418–21; `tapply`, 414; *t*-test and, 418–21; uncertainty and, 418; `update`, 419
right-censored data, 224, *227*
RMS *Titanic*, 173–75; Birkenhead drill and, 173n9
robustness: ANOVA and, 118–22; assumptions of statistical analysis and, 105, 118, 123; evaluation and, 376–77; hypothesis tests and, 47; Kolmorogov-Smirnov test and, 26–28; machine learning and, 201–2; meta-analysis and, 341, 344, 369;

model-fitting methods and, 39; predictor variables and, 174, 238; rules-of-thumb and, 153–54, 174, 199–202; simulation basics and, 3–4, 12, 18, 20, 25–29, 39; time series and, 199
Rotenberry, J. T., 322
R packages: `brms`, 272, 387–88; `DHARMa`, 135n24, 283–84, 387–88; `dplyr`, 30, 421; `faraway`, 38, 121, 154n3, 156, 160, 214n3, 214n6, 215, 276, 296, 298–99, 415, 417, 419; `ggplot2`, 30, 63, 189, 332; `lmerTest`, 138, 353, 355; `mgcv`, 92, 276, 280n4, 298, 393, 418; `mice`, 156, 160, 175, 215, 233, 255; `mvtnorm`, 30, 79–80, 82, 244, 246–47, 250, 272, 293, 296, 346, 349, 355, 359, 363–64, 366, 409, 411; `pwr`, 61, 62n17, 75n4, 313n4; `ranger`, 149, 166n8, 167, 170, 239, 241, 255, 289, 295n18, 300, 393, 394
R programming language, xii, 1, 10, 12, 377, 380
rules-of-thumb, xii, 12; abstract concepts and, 1; analysis of variance (ANOVA) and, 154, 179; avoiding mistakes and, xi; Big Data and, 153, 200–4; collinearity and, 154, 165–72; concept of, 153n1; errors and, 153–71, 174–86, 189, 192, 202–7; fixed effects and, 153; machine learning and, 153–54, 165–72, 200–5; mixed-effect models and, 153; model selection and, 154–72; Monte Carlo simulation and, 180–90; predictor variables and, 172–80; random effects and, 153; recipes and, 153; residuals and, 155–58, 168–69, 176, 183–86, 191, *193*, 198–99, 204; robustness and, 153–54, 174, 199–202; standard deviation and, 24; time series and, 190–200

`sample`, 417–18
sample mean: meta-analysis and, 342; notation for, 11; retrospective power analysis and, 319; R functions and, 419, 421; standard deviation and, 23, 26, 28, 31, *60*, 74, *316*, 319; study designs and, 54, 57, 60
sample size: advanced simulations and, 407; assumptions of statistical analysis and, 106–10, 119, 124–25; autocorrelated data and, 3; censored observations and, 223, *224*; computational cost and, 9; data perturbation and, 40; errors and, 247, *248*, 259–60; evaluation and, 302, 380, 398; illusory truth effect and, 60–66; Kolmorogov-Smirnov test and, 26–28; machine learning and, 202; meta-analysis and, 342–49, 352, 357, 370; missing data and, 212, 219–20; model selection and, 164; Monte Carlo simulation and, 180; normal distributions and, 23n8; notational conventions and, 11; Pólya urn and, 19–20; power of statistical analysis and, 73–88, 98, 101; predictor variables and, 236; regression analysis and, 243; retrospective power analysis and, 311–13, 316, 319–30; R functions and, 418; standard deviation and, 24, 28, 40, 58, *60*, 62, 75, *316*, 330n27; stopping rules and, 74, 81; study designs and, 47, 49, 57n9, 58–62, 66, 69; weak effects and, 12
Schapire, R. E., 382–83
Schönbrodt, F. D., 77–79, 81, *83*
seeds: ANOVA and, 22; assumptions of statistical analysis and, 107–11, 119–25, 128–30, 146–49; censored observations and, 223, 225; complete data analysis and, 363–66; errors and, 246–47, 250, 253–56, 261–62; evaluation and, 272, 277–78, 284, 289, 296–99, 303, 305, 378, 391–92, 395; federated analysis and, 359; meta-analysis and, 346, 349, 355–56; missing data and, 220; model selection and, 160, 167, 170; Monte Carlo simulation and, 185–86; Pólya's urn and, 18–20; power of statistical analysis and, 76–77, 80–95; predictor variables and, 178, 233–37, 241; regression analysis and, 244; reproducibility and, 18, 54, 76; retrospective power analysis and, 315, 332; R functions and, 409–11, 416–21; simulation basics and, 4, 18–31, 34, 40–43; study designs and, 54, 65; time series and, 192, 195, 197–98
Semmelweis, I., 81
sensitivity analysis, 328
S-error, 159–62, *165*, 329–30, 331
Shannon, C., 386
Shannon-Weiner diversity index, 386
Shapiro-Wilk statistic, 106–9
signals: advanced simulations and, 404–6; censored observations and, 228n11; evaluation and, 375n2; meta-analysis and, 342n3; noise and, 5, 87, 405; power of statistical analysis and, 87; resampling, 40; retrospective power analysis and, 331n30; study designs and, 55n6; time series and, *195*; truth and, 55n6
Silberzahn, R., 331
`simstudy`, 133
simulated data: advanced simulations and, 404; assumptions of statistical analysis and, 107, *113*, 127, 133–35, 143; comparing, 21, *23*, 381–82; computational competence and, 17–43; concept of, 4–5; errors and, 252, 254; evaluation and, 270, 282–84, 288, 299, 301, 380–82, 391; generalized least squares (GLS) and, 34–35; hallucinations and, 9–10, 14; improving estimation in, 91–100; K-S test and, 28n12; likelihood-ratio test (LRT) and, 41; meta-analysis and, 346, 354, *369*; model selection and, 165–67, *172*; Monte Carlo simulation and, 184–86; power of statistical analysis and, 76, 83n8, 87; predictor variables and, 232, 239; randomisation techniques and, 35–43; reasons for, 9–10; replicates and, 4, 19n2, 22–32, 40–43; residuals and, 4; specific, 5–6; time series and, 196; workflow and, 211, 232, 239, 252, 254. *See also* fake data; simulated data
simulation basics: abstract concepts and, 1, 148, 377, 413–14; analysis of variance (ANOVA) and, 3, 17–18, 20–22, *23*; assumptions of statistical analysis, 3 (*see also* assumptions of

simulation basics (*continued*)
statistical analysis); autocorrelated data and, 3, 33–36; Bayesian models and, 10, *13*, 22n6; bias and, 3–8, 22n7, 34; chemistry and, xii, 1; collinearity and, *13*; computational competence and, 17–43; confidence interval (CI) and, 17, 21–26, 28, *35*, 39–40, 43; data-generating model (DGM) and, 4, 13, 17, 43; degrees of freedom and, 23, *32*, 36–39; effect size, *13*; errors, 3–6, 10, 21–38; frequentist analysis and, xii, 10; Gaussian distribution and, 23, 29–33; general linear models (GLMs), xii, 3, 29–30; imputation and, *13*; Kolmorogov-Smirnov (K-S) test and, 26–28; machine learning and, 17, 33; manipulation and, 9; model variability and, 28–29; motivations for, 1; nine steps for, 17–18; normal distribution and, 22–33; notational conventions, 4, 11–12, *143*, 382n5; parameter values and, 4–5, *7*, 11–12, 18–21, 24–31, 33n22, 35–36, 40–42; pipelines and, 8, 12; Pólya's urn and, 18–20; post hoc analysis and, *13*; probability and, *7*, 10, 12, 14, 17, 18n1, 20, 22n6, 26, 29n17, 33n20, 37; *p*-values and, xii, 5, 20, 27–28; randomness and, 3–5, 8–11, 17–19, 22–24, 27–43; reality and, 5, *58*; recipes and, 1; retrospective power analysis and, xii; R functions and, 38n26; robustness and, 3–4, 12, 18, 20, 25–29, 39; seeds and, 4, 18–31, 34, 40–43; skewed residuals and, 29–32; *t*-test and, 10, 14, 24, 29–32; type-II errors and, 21, 28n12; unbiased data and, 3–5, 22n7
standard deviation: effect size and, 24, 57–62, 76, *316*, 330n27, 352; evaluation and, *285*, 301, 377–79, 384; Gaussian distribution and, 23; log-normal distribution and, 31–32; meta-analysis and, 352; Monte Carlo simulation and, 184, 187; power of statistical analysis and, 74–76, 96; residuals and, 40, 117, 283, *285*; retrospective power analysis and, 314n7, 315, *316*, 319, 330n27, 332; rules-of-thumb and, 24; sample mean and, 23, 26, 28, 31, *60*, 74, *316*, 319; sample size and, 24, 28, 40, 58, *60*, 62, 75, *316*, 330n27; study designs and, 52–53, 57–58, *60*, 62
Standardised Potential Evapotranspiration Index (SPEI), 380
Standard Model, 95–96, 311
Stan Development Team, 388
Stanley, T. D., 350, 352
StarCoder, xii
stepwise model selection: complete data analysis and, 362; errors and, 254–55; meta-analysis and, 345, *351*, 354; predictor variables and, 236; rules-of-thumb and, 154–55, 157, 159–60, 164–65
stochastic weather forecasting, 403–4
stopping rules: evaluation and, 378; power of statistical analysis and, 74, 81–85, *87*, *89*, 95, 101; principle of, 81; puerperal fever deaths study and, 81–84; sample size and, 74, 81; study designs and, 66. *See also* censored observations
study designs: adequate replication and, 56–60; analysis of variance (ANOVA) and, 60; Bayesian models and, 45, 49; bias and, 45–48, 58n10, 65, 69; bootstrapping and, 48, 64–66, 69; confidence interval (CI) and, 63, 65–66, 69; correlation and, 57–58; data-generating model (DGM) and, 50, 69; degrees of freedom and, 58n10, 67; effect size and, 47, 49, 53, 57–62, 67, 69; errors and, 49, 57n9, 58–62, 66–67; false negatives and, 57–59, 66; false positives and, 57–60, 67; frequency distribution and, 50; frequentist analysis and, 49; Gaussian distribution and, 51, 52n5; HARKing and, 67–68; hypotheses definitions and, 49–55; hypothesis testing and, 58–60; illusory truth effect and, 47–52, 55–56, 60–66; likelihood and, 45, 50; manipulation and, 45, 49; matrices and, 61, 66; noise and, 49; normal distribution and, 53; null expectation and, 52, 55; null hypothesis and, 56–61, 69; parameter values and, 49–50, 55n7, 56, 60–61, 68–69, 369; perceived accuracy and, 51–55, *56*; plausibility and, 50–56, 61–62, 66; post hoc analysis and, 67, 69; power of, 45, 47, 49, 57–62, 66–69; prior expectations and, 331–32; probability and, 45, 50, 56, 58, 66–67, 69; processing fluency and, 54–55, 62; *p*-values and, 58n12, 67; quantifying variables of interest, 50–55; questionable research practices (QRPs) and, 67–69; randomness and, 51–52, 55n8, 57, 61; replicates and, 56–59, 65, 73; R functions and, 62n17, 65; sample mean and, 54, 57, 60; sample size and, 47, 49, 57n9, 58–62, 66, 69; seeds and, 54, 65; signals and, 55n6; standard deviation and, 52–53, 57–58, *60*, 62; stopping rules and, 66; testing hypotheses and, 55–67; true positives and, 59, 66; *t*-test and, 57n9, 59–62; type-II errors and, 58, 59n13, 66; uncertainty and, 47–48, 69
subset regression, 154–55, 397n13
synthetic data, 4n1
`system.time`, 418–21
`Sys.time`, 418–21

Thurman, E. M., 228n11
tidyverse, 10
time series: autocorrelated data and, 191, *193*, 196n27, 281; carbon dioxide (CO_2) and, 85–87, 89n15, 90–91; change-point methods and, 192, 194–98; correlation and, 191, *193*, 196n27; detrending and, 154, 190–200, 205, 245; differencing and, 191–92, *193*; errors and, 192; length of, 85–91; likelihood and, 190; matrices and, 198; Monte Carlo simulation and, 182–84; noise and, *197*, 198; parameter values and, 191; probability and,

190, *197*; regression analysis and, 182–84, 190–200; replicates and, 194, 197–98; residuals and, 191, *193*, 198–99; robustness and, 199; seeds and, 192, 195, 197–98; signals and, *195*; simulated data and, 196
true negatives, 35, 67. *See also* type-II errors
true positives: assumptions of statistical analysis and, 123–24; model selection and, 159–64, *165*; predictor variables and, *180*; retrospective power analysis and, 324, 328n25; study designs and, 59, 66. *See also* type-I errors
truncated data, 224n8
t-test: assumptions of statistical analysis and, 103, *120*, 124, 127, 138; normality and, 29; power of statistical analysis and, 74–77; retrospective power analysis and, 324; R functions and, 418–21; simulation basics and, 10, 14, 24, 29–32; skewed residuals and, 29–32; study designs and, 57n9, 59–62; Welch's generalisation of, 120
type-I errors: assumptions of statistical analysis and, 105–6, 117, *122*, 123–27; inflated, 252; model selection and, 154, 164; Neyman and, 313; Pearson and, 313; power of statistical analysis and, 74; retrospective power analysis and, 311–22, 327–28; simulation basics and, 21, 26, 28n12, 29, 31–34; study designs and, 58, 59n13, 60, 66–67. *See also* hypothesis testing
type-II errors: assumptions of statistical analysis and, 105, 126; inflated, 252; missing data and, 219; Neyman and, 313; Pearson and, 313; predictor variables and, 174, 176, 178; retrospective power analysis and, 311–15, 328, 333; simulation basics and, 21, 28n12; study designs and, 58, 59n13, 66. *See also* power of statistical analysis

unbiased data: assumptions of statistical analysis and, 115, 134; complete data analysis and, 362; errors and, 252, 257; evaluation and, *277*, 381; machine learning and, 201; meta-analysis and, 342, *343*, 347n11, 352, 354, 370; missing data and, 213n2, 219; model selection and, 159, 171; power of statistical analysis and, 74, 79, 83–85, 97; predictor variables and, 172, *235*, 238, *242*; regression analysis and, 243; retrospective power analysis and, 320n18, *331*; simulation basics and, 3–5, 22n7; workflow and, 211, 213n2, 219, *235*, 238, *242*, 243, 252, 257
uncertainty: advanced simulations and, 403–4; assumptions of statistical analysis and, 115, 147; bias and, 4, 47–48, 69, 115, 171, 211, 252, 254, 260, 273, *277*, 404; bootstrapping and, 29–30, *31*, 39, 69, 245, 251–54, *260*, 273, 275, *277*, 288, *306*, 418; communication of, 47–48, 69, 275, 302; errors and, 245–60; evaluation and, 269–75, *277*, 288–91, 294–96, *297*, 302, 305–7, 379; improving estimation and, 91–100; machine learning and, 203; model-fitting methods and, 28–30; model selection and, 171; Monte Carlo simulation and, 181–84, *188*; power of statistical analysis and, 78, 91–93; propagation of, 247–49; quantification of, *8*, 245, 250, 254, 269, 302; randomness and, 29, *31*, 39, 249, 254; retrospective power analysis and, 334; R functions and, 418; study designs and, 47–48, 69; workflow and, 211–12, 245–60
unit testing, 376–79. *See also* code verification; evaluation; functional verification; unit testing

van Marle, M. J. E., 180–90
variance-covariance matrices, 33n22, 34, 243n17, 248–50, 293. *See also* correlation
Vasishth, S., xii

Walker, H., 313
W boson, 95–96
Weisser, W. W., 113–15
Wiens, J. A., 322
Wilcox, R. R., 29n18, 117
workflow: bias and, 211 (*see also* bias); Big Data and, 211; machine learning and, 4; matrices and, 212n1, 220–23, 236, 240–50, 251n22, 255; model-fitting methods and, 211; parameter values and, 211, 213, 217–19, *220*, 222, 226, 232–46, 249–61; pipelines and, 8, 12, 211, 228n11, 245, 251–58, 403–5; *p*-values and, 211, 215; random-forest models and, 166–72; residuals and, 212, 217–19; simulated data and, 211, 232, 239, 252, 254; unbiased data and, 211, 213n2, 219, *235*, 238, *242*, 243, 252, 257; uncertainty and, 211–12, 245–60, 254–56

χ^2-statistics, 10, 14

Yates, K. L., 145

Zayed, A. A., 211
Z-score, 58n11, 75