

ANDY FIELD, JOHNNY VAN DOORN
& ERIC-JAN WAGENMAKERS

DISCOVERING STATISTICS USING JASP



| JASP

 Sage



1 Oliver's Yard
55 City Road
London EC1Y 1SP

2455 Teller Road
Thousand Oaks
California 91320

Unit No 323-333, Third Floor, F-Block
International Trade Tower
Nehru Place, New Delhi 110 019

8 Marina View Suite 43-053
Asia Square Tower 1
Singapore 018960

Editor: Jai Seaman
Development editor: Hannah Cooper
Assistant editor: Becky Oliver
Assistant editor, digital: Benedict Hegarty
Production editor: Ian Antcliff
Copyeditor: Richard Leigh
Proofreader: Thea Watson
Marketing manager: Lorna Patkai
Cover design: Wendy Scott
Typeset by: C&M Digital (P) Ltd, Chennai, India
Printed in the UK

© Andy Field, Johnny van Doorn and Eric-Jan
Wagenmakers 2025

Throughout the book, screenshots and images from JASP are reproduced without needing to ask the JASP team for permission. JASP is open-source software, and therefore can be used by anyone, anywhere, for whatever purpose, with no strings attached.

Apart from any fair dealing for the purposes of research, private study, or criticism or review, as permitted under the Copyright, Designs and Patents Act, 1988, this publication may not be reproduced, stored or transmitted in any form, or by any means, without the prior permission in writing of the publisher, or in the case of reprographic reproduction, in accordance with the terms of licences issued by the Copyright Licensing Agency. Enquiries concerning reproduction outside those terms should be sent to the publisher.

Library of Congress Control Number: 2024948439

British Library Cataloguing in Publication data

A catalogue record for this book is available from the British Library

ISBN 978-1-5296-9144-3
ISBN 978-1-5296-9143-6 (pbk)

CONTENTS

EXTENDED CONTENTS	VII
PREFACE	XIII
ABOUT THE AUTHORS	XVII
HOW TO USE THIS BOOK	XIX
THANK YOU	XXIII
SYMBOLS USED IN THIS BOOK	XXVII
A BRIEF MATHS OVERVIEW	XXIX
1 WHY IS MY EVIL LECTURER FORCING ME TO LEARN STATISTICS?	1
2 THE SPINE OF STATISTICS	51
3 THE PHOENIX OF STATISTICS	109
4 THE JASP ENVIRONMENT	151
5 VISUALIZING DATA	179
6 THE BEAST OF BIAS	205
7 CORRELATION	277
8 THE LINEAR MODEL (REGRESSION)	313
9 CATEGORICAL PREDICTORS: COMPARING TWO MEANS	365
10 MODERATION AND MEDIATION	395
11 GLM 1: COMPARING SEVERAL INDEPENDENT MEANS	425
12 GLM 2: COMPARING MEANS ADJUSTED FOR OTHER PREDICTORS (ANALYSIS OF COVARIANCE)	477
13 GLM 3: FACTORIAL DESIGNS	505
14 GLM 4: REPEATED-MEASURES DESIGNS	539
15 NON-PARAMETRIC MODELS	579
16 CATEGORICAL OUTCOMES: CHI-SQUARE	619
17 CATEGORICAL OUTCOMES: LOGISTIC REGRESSION	645
EPILOGUE	689
APPENDIX	693
GLOSSARY	705
REFERENCES	733
INDEX	747

Until next time...

This book, in its various forms, has consumed the past 20 years of my life. With each new edition and adaptation I try not just to make superficial changes but also to rewrite and improve everything (one of the problems with getting older is you look back at your past work and think you can do things better). Each time I get a nice email from someone who found it useful I am reminded that it is the most useful thing I'll ever do with my academic life. It continues to be a labour of love. It still isn't perfect, and I still love to have feedback (good or bad) from the people who matter most: you.

Andy



www.facebook.com/profandyfield



[@profandyfield.bsky.social](https://bsky.app/profile/profandyfield.bsky.social)



www.youtube.com/user/ProfAndyField



www.instagram.com/discovering_catistics/



www.discoverjasp.com



ABOUT THE AUTHORS



Johnny van Doorn is an assistant professor at the University of Amsterdam, where he has enjoyed ten years of Bayesian brainwashing by EJ. Coincidentally, he loves enlightening his students about the beauty of Bayesian inference, and he contributes to JASP by writing R code and helping fellow teachers to adopt it. In his spare time, Johnny gathers data to statistically prove that he is better at Mario Kart and Terraforming Mars than his friends. Beyond his statistical endeavours, he spends his time wondering how he still has friends, playing the guitar, hiking, and rock 'n' roll dancing.



Eric-Jan ('EJ') Wagenmakers is a mathematical psychologist and a fundamentalist Bayesian. His lab at the University of Amsterdam guides the development of the JASP open-source statistics program. EJ is a staunch advocate of Open Science and the preregistration of analysis plans. His book *Bayesian Thinking for Toddlers* is a must-have for any toddler with even a passing interest in Ockham's razor and the prequential principle.

EJ's hobbies are chess, squash, and watching arm-wrestling videos on YouTube. He lives in Hilversum, The Netherlands, together with his wife Nataschja and their children Theo and Leanne.

HOW TO USE THIS BOOK

When the publishers asked me to write a section on ‘How to use this book’ it was tempting to write ‘Buy a large bottle of Olay anti-wrinkle cream (which you’ll need to fend off the effects of ageing while you read), find a comfy chair, sit down, fold back the front cover, begin reading and stop when you reach the back cover.’ I think they wanted something more useful. 😊

What background knowledge do I need?

In essence, I assume that you know nothing about statistics, but that you have a very basic grasp of computers (I won’t be telling you how to switch them on, for example) and maths (although I have included a quick overview of some concepts).

Do the chapters get more difficult as I go through the book?

Yes, more or less: Chapters 1–9 are first-year non-STEM undergraduate degree level and Chapters 10–17 move into second- or possibly final-year degree level. However, my aim is to tell a statistical story rather than worry about what level a topic is at. Many books teach different tests in isolation and never really give you a grasp of the similarities between them; this, I think, creates an unnecessary mystery. Most of the statistical models in this book are the same thing expressed in slightly different ways. I want the book to tell this story, and I see it as consisting of five parts:

- Part 1 (Doing research): Chapters 1–4.
- Part 2 (Exploring data and fitting models): Chapters 5–6.
- Part 3 (Linear models with continuous predictors): Chapters 7–8.
- Part 4 (Linear models with continuous or categorical predictors): Chapters 9–15.
- Part 5 (Linear models with categorical outcomes): Chapter 16–17.

This structure might help you to see the method in my mind. To help you on your journey I’ve also coded sections within chapters with a letter icon that gives you an idea of the difficulty of the material. It doesn’t mean you can skip the sections, but it might help you to contextualize sections that you find challenging. It’s based on a wonderful categorization system using the letters A to D to indicate increasing difficulty. Each letter also conveniently stands for a word that rhymes with and describes the likely state of your brain as you read these sections:

- **A** is for brain **Attain**: I’d hope that everyone can get something out of these sections. These sections are aimed at readers just starting a non-STEM undergraduate degree.
- **B** is for brain **Bane**. These sections are aimed at readers with a bit of statistics knowledge, but they might at times be a source of unhappiness for your brain. They are aimed at readers in the second year of a non-STEM degree.

- © is for brain **Complain**. These topics are quite difficult. They are aimed at readers completing the final year of a non-STEM undergraduate degree or starting a master's degree.
- Ⓓ is for brain **Drain**. These are difficult topics that will drain many people's brains, including my own. The 'D' might instead stand for **Dogbane** because if your brain were a dog, these sections would kill it. Anything that hurts dogs is evil, so draw your own conclusions about these sections.

Why do I keep seeing silly faces everywhere?



Brian Haemorrhage: Brian is a really nice guy, and he has a massive crush on Jane Superbrain. He's seen her around the university campus and whenever he sees her, he gets a knot in his stomach. He soon realizes that the only way she'll date him is if he becomes a statistics genius (and changes his surname). So, he's on a mission to learn statistics. At the moment he knows nothing, but he's about to go on a journey that will take him from statistically challenged to a genius, in 792 pages. Along his journey he pops up and asks questions, and at the end of each chapter he flaunts his newly found knowledge to Jane in the hope that she'll notice him.



Confusius: The great philosopher Confucius had a lesser-known brother called Confusius. Jealous of his brother's great wisdom and modesty, Confusius vowed to bring confusion to the world. To this end, he built the confusion machine. He puts statistical terms into it, and out of it come different names for the same concept. When you see Confusius he will be alerting you to statistical terms that mean (essentially) the same thing.



Correcting Cat: This cat lives in the ether and appears to taunt the Misconception Mutt by correcting his misconceptions. He also appears when I want to make a bad cat-related pun. He exists in loving memory of my own ginger cat who after 20 years as the star of this book sadly passed away, which he promised me he'd never do. You can't trust a cat.



Cramming Sam: Samantha thinks statistics is a boring waste of time. She wants to pass her exam and forget that she ever had to know anything about normal distributions. She appears and gives you a summary of the key points that you need to know. If, like Samantha, you're cramming for an exam, she will tell you the essential information to save you having to trawl through hundreds of pages of my drivel.



Jane Superbrain: Jane is the cleverest person in the universe. She has acquired a vast statistical knowledge, but no one knows how. She is an enigma, an outcast, and a mystery. Brian has a massive crush on her. Jane appears to tell you advanced things that are a bit tangential to the main text. Can Brian win his way into her heart? You'll have to read and find out.

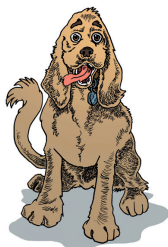


Just Another Statistics Professor: For every professor you meet there's always another one who thinks they know better. You might think that in a field like statistics there'd be little room for disagreement; after all, maths is fact, isn't it? Sadly not. The history of statistics is littered with petty intellectual battles that rage on to this day. Therefore, whenever you are drinking from a warm cup of understanding during afternoon tea, Just Another Statistics Professor will pop up with her own take on the situation.

HOW TO USE THIS BOOK



Labcoat Leni: Leni is a budding young scientist and he's fascinated by real research. He says, 'Andy, I like an example about using an eel as a cure for constipation as much as the next guy, but all of your data are made up. We need some real examples, buddy!' Leni walked the globe, a lone data warrior in a thankless quest for real data. When you see Leni you know that you will get some real data, from a real research study, to analyse. Keep it real.



Misconception Mutt: The Misconception Mutt follows his owner to statistics lectures and finds himself learning about stats. Sometimes he gets things wrong, though, and when he does, something very strange happens. A ginger cat materializes out of nowhere and corrects him.



Oliver Twisted: With apologies to Charles Dickens, Oliver, like the more famous fictional London urchin, asks 'Please, Sir, can I have some more?' Unlike Master Twist though, Master Twisted always wants more statistics information. Who wouldn't? Let us not be the ones to disappoint a young, dirty, slightly smelly boy who dines on gruel. When Oliver appears he's telling you that there is additional information on the companion website. (It took a long time to write, so someone please actually read it.)



Smart Alex: Alex was aptly named because she's, like, super smart. She likes teaching people, and her hobby is posing people questions so that she can explain the answers to them. Alex appears at the end of each chapter to pose you some questions. Her answers are on the companion website.

What is on the companion websites?

Here is what to find on my website, and what is provided on the publisher's site.

My exciting website

The data files for the book and the solutions to the various tasks and pedagogic features in this book are at www.discoverjasp.com, which is a website that I wrote and maintain. This website contains:

- Self-test solutions: Where the self-test is not answered within the chapter, the solution is on this website.
- Smart Alex answers: Each chapter ends with a set of tasks for you to test your newly acquired expertise. The website contains detailed answers. Will I ever stop writing?
- Data sets: We use a lot of bizarre but hopefully interesting data sets throughout this book, all of which can be downloaded so that you can practise what you learn.
- Labcoat Leni solutions: There are full answers to the Labcoat Leni tasks.
- Oliver Twisted's pot of gruel: Oliver Twisted will draw your attention to the hundreds of pages of more information that we have put online so that (1) the planet suffers a little less, and (2) you won't die when the book falls from your bookshelf onto your head.
- Cyberworms of knowledge: I have used nanotechnology to create cyberworms that crawl down your broadband, wifi or 5G, pop out of a port on your computer, tablet, or phone, and fly through space into your brain. They rearrange your neurons so that you understand statistics. You don't believe me? You'll never know for sure unless you visit my website ...

Sage's less exciting website for lecturers and professors

In addition, the publishers have an official companion website that contains a bunch of useful stuff for lecturers and professors, most of which I haven't had any part in. Their site will also link to my resources above, so you can use the official site as a one-stop shop. To enter Sage's world of delights, go to <https://study.sagepub.com/fieldJASP1e>. Once you're there, Sage will flash up subliminal messages that make you use more of their books, but you'll also find resources for use with students in the classroom and for assessment.

- **Testbank:** There is a comprehensive testbank of multiple-choice questions for instructors to use for substantive assessment. It comes in a lovely file that you can upload into your online teaching system (with answers separated out). This enables you to assign questions for exams and assignments. Students can see feedback on correct/incorrect answers, including pointers to areas in the book where the right answer can be found.
- **Chapter based multiple-choice questions:** Organized to mirror your journey through the chapters, these allow you to test students' formative understanding week by week and to see whether you are wasting your life, and theirs, trying to explain statistics. This should give them the confidence to waltz into the examination. If everyone fails said exam, please don't sue me.
- **PowerPoint decks:** I can't come and teach your classes for you (although you can watch my lectures on YouTube). Instead, I like to imagine a *Rocky*-style montage of lecturers training to become a crack team of highly skilled and superintelligent pan-dimensional beings poised to meet any teaching-based challenge. To assist in your mission to spread the joy of statistics I have provided PowerPoint decks for each chapter. If you see something weird on the slides that upsets you, then remember that's probably my fault.

Happy reading, and don't get distracted by social media.

A BRIEF MATHS OVERVIEW

There are good websites that can help you if any of the maths in this book confuses you. Use a search engine to find something that suits you.

Working with expressions and equations

Here are three important things to remember about working with equations:

- 1 Two negatives make a positive: Although in life two wrongs don't make a right, in mathematics they do. When we multiply a negative number by another negative number, the result is a positive number. For example, $(-2) \times (-4) = 8$.
- 2 A negative number multiplied by a positive one makes a negative number: If you multiply a positive number by a negative number then the result is another negative number. For example, $2 \times (-4) = -8$, or $(-2) \times 6 = -12$.
- 3 BODMAS and PEMDAS: These two acronyms are different ways of remembering the order in which mathematical operations are performed. BODMAS stands for Brackets, Order, Division, Multiplication, Addition, and Subtraction; whereas PEMDAS stems from Parentheses, Exponents, Multiplication, Division, Addition, and Subtraction. Having two widely used acronyms is confusing (especially because multiplication and division are the opposite way around), but they mean the same thing:
 - Brackets/Parentheses: When solving any expression or equation you deal with anything in brackets/parentheses first.
 - Order/Exponents: Having dealt with anything in brackets, you next deal with any order terms/exponents. These refer to power terms such as squares. Four squared, or 4^2 , used to be called 4 raised to the order of 2, hence the word 'order' in BODMAS. These days the term 'exponents' is more common (so by all means use BEDMAS as your acronym if you find that easier).
 - Division and Multiplication: The next things to evaluate are any division or multiplication terms. The order in which you handle them is from the left to the right of the expression/equation. That's why BODMAS and PEMDAS can list them the opposite way around, because they are considered at the same time (so, BOMDAS or PEDMAS would work as acronyms too).
 - Addition and Subtraction: Finally, deal with any addition or subtraction. Again, go from left to right doing any addition or subtraction in the order that you meet the terms. (So, BODMSA would work as an acronym too but it's hard to say.)

Let's look at an example of BODMAS/PEMDAS in action: what would be the result of $1 + 3 \times 5^2$? The answer is 76 (not 100 as some of you might have thought). There are no brackets, so the first thing is to deal with the order/exponent: 5^2 is 25, so the equation becomes $1 + 3 \times 25$. Moving from left to

right, there is no division, so we do the multiplication: 3×25 , which gives us 75. Again, going from left to right we look for addition and subtraction terms. There are no subtractions so the first thing we come across is the addition: $1 + 75$, which gives us 76 and the expression is solved. If I'd written the expression as $(1 + 3) \times 5^2$, then the answer would have been 100 because we deal with the brackets first: $(1 + 3) = 4$, so the expression becomes 4×5^2 . We then deal with the order/exponent (5^2 is 25), which results in $4 \times 25 = 100$.

Logarithms

The logarithm is the inverse function to exponentiation, which means that it reverses exponentiation. In particular, the natural logarithm is the logarithm of a number using something called Euler's number ($e \approx 2.718282$) as its base. So, what is the natural logarithm in simple terms? It is the number of times that you need to multiply e by itself to get that number. For example, the natural logarithm of 54.6 is approximately 4, because you need to multiply e by itself 4 times to get 54.6:

$$e^4 = 2.718282^4 \approx 54.6$$

Therefore,

$$\ln(54.6) \approx 4$$

In general, if x and y are two values,

$$y = \ln(x) \text{ is equivalent to } x \approx e^y$$

For example,

$$4 \approx \ln(54.6) \text{ is equivalent to } 54.6 \approx e^4$$

There are two rules relevant to this book that apply to logarithms. First, when you add the natural logarithms of two values, the result is the natural logarithm of their products (i.e., the values multiplied). In general,

$$\ln(x) + \ln(y) = \ln(xy)$$

Second, when you subtract the natural logarithms of two values, the result is the natural logarithm of their ratio (i.e., the first value divided by the second). In general,

$$\ln(x) - \ln(y) = \ln\left(\frac{x}{y}\right)$$

Matrices

We don't use matrices much in this book, but they do crop up. A **matrix** is a grid of numbers arranged in columns and rows. A matrix can have many columns and rows, and we specify its dimensions using numbers. In general people talk of an $i \times j$ matrix in which i is the number of rows and j is the number of columns. For example, a 2×3 matrix is a matrix with two rows and three columns, and a 5×4 matrix is one with five rows and four columns.

A BRIEF MATHS OVERVIEW

$$\begin{bmatrix} 2 & 5 & 6 \\ 3 & 5 & 8 \end{bmatrix}$$

2 × 3 matrix

$$\begin{bmatrix} 3 & 21 & 14 & 20 \\ 6 & 21 & 3 & 11 \\ 19 & 8 & 9 & 20 \\ 3 & 15 & 3 & 11 \\ 23 & 1 & 14 & 11 \end{bmatrix}$$

5 × 4 matrix

The values within a matrix are *components* or *elements* and the rows and columns are *vectors*. A *square matrix* has an equal number of columns and rows. When using square matrices we sometimes refer to the diagonal components (i.e., the values that lie on the diagonal line from the top left component to the bottom right component) and the off-diagonal ones (the values that do not lie on the diagonal). In the square matrix below, the diagonal components are 3, 21, 9 and 11 (the highlighted values) and the off-diagonal components are the other values. An **identity matrix** is a square matrix in which the diagonal elements are 1 and the off-diagonal elements are 0. Hopefully, the concept of a matrix is less scary than you thought it might be: it is not some magical mathematical entity, merely a way of representing data – like a spreadsheet.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Identity matrix

Diagonal elements

3	21	14	20
6	21	3	11
19	8	9	20
3	15	3	11

Square matrix

Off-diagonal elements



THE JASP ENVIRONMENT

- 4.1** What will this chapter tell me? 152
- 4.2** What is JASP? 153
- 4.3** Versions of JASP 154
- 4.4** Getting started 154
- 4.5** The data editor 156
- 4.6** Entering data into JASP 157
- 4.7** JASP modules 169
- 4.8** The input window 170
- 4.9** The output window 171

- 4.10** Saving and exporting 173
- 4.11** A few useful options 173
- 4.12** Jane and Brian's story 174
- 4.13** What next? 175
- 4.14** Key terms that I've discovered 176
 - Smart Alex's Tasks 176

Letters A to D indicate increasing difficulty

4.1 What will this chapter tell me? A

Aged 7, we started a school project about ‘the future’ in which I wrote about robots (Figure 4.1). Astute child that I was, I noticed several important differences between robots and humans: they have lots of wires in them, you can control them and they will do whatever you want and – perhaps most important – if you get angry with them you can press their self-destruct button. I’m ashamed to say that I have contemplated a world where humans have self-destruct buttons rather more than the 7-year-old me pondered on the morality of blowing up a robot every time you got out of bed on the wrong side. I’ve also considered how nice it would be if I could control people, especially if I could control them into being a bit more tolerant and kind, and by ‘a bit’ I mean ‘quite a lot’. Both current and 7-year-old me could see the use of robots to make the world a better place, it’s just that for 7-year-old me the world being a better place consisted of being delivered an endless supply of tea and sweets.

Sadly the humanoid robotic tea and sweet delivery system has yet to materialize in the many decades since school. (Unless you are one of my two children, who live with robots called ‘mum’ and ‘dad’ who are under their control and deliver endless supplies of things that they want.) What has materialized is the statistical equivalent: JASP. Like a robot, it has wires in it (well, the computer that runs it), it will do anything you want it to (not the tea and

sweets thing, but ...), and when you get angry with it you can press its self-destruct button. Very rarely, it presses its own self-destruct button just for kicks and giggles. This chapter introduces you to your new robotic statistics slave: JASP. Over subsequent chapters, JASP will help us to build a utopian understanding of statistics, or to press our self-destruct buttons. Time will tell.

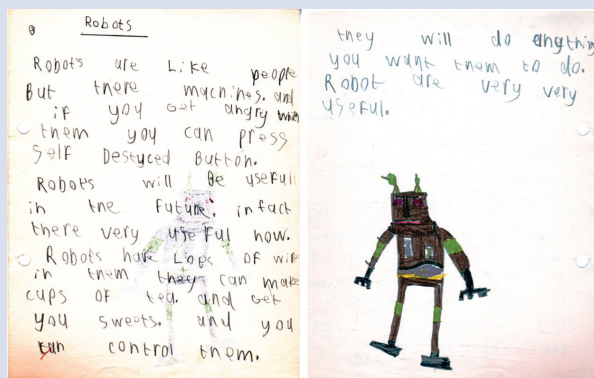


Figure 4.1 'Is he alive or dead? Has he thoughts within his head?'

4.2 What is JASP?

JASP is a statistical software program that is the brainchild of a crack team of statistics addicts, or a statistics team of crack addicts, I can never remember which.¹ Since 2012, they have wandered the halls of the University of Amsterdam on a mission to provide an attractive open-source (i.e., free) alternative to proprietary (i.e., paid, commercial) statistics software. While there are some fantastic programming languages out there (e.g., Python, R, Julia) that allow their users the flexibility to conduct any analysis they can dream of, having a graphical user interface is a lot more user-friendly and accessible. With JASP, you can execute standard statistical tests with ease, and without fear of making a programming error. As you will experience firsthand, this allows us to focus on the ideas and the interpretation ('Why on earth do I need to do this particular test?', 'What is this plot trying to tell me?') rather than the execution – in JASP the question of *how* you do a particular analysis generally has a simple answer.

As you start tinkering with JASP, you will discover that it offers a ton of useful features. For instance, you can copy and paste formatted tables² into Word (or even LaTeX), you can annotate your results, there are help files you can consult when you get lost, and you can edit many of the figures and save them in PDF format (for instance). In general, the JASP team has paid a lot of attention to making the figures look clean and pretty. JASP is also available in many languages, including Spanish and Chinese.

Two design principles set JASP apart from many other statistical software programs. First, JASP dynamically updates results. This means that JASP changes its output as soon as you change any of the input options. This makes it easier to know what input options correspond to what output. For instance, as soon as you tick the box *mean*, the output table adds the mean; tick the box *median*, and the output table adds the median; untick the box *mean* and the mean is removed from the output table. Second, JASP uses progressive disclosure. Instead of flooding users with information, the initial output of JASP is purposefully simple, presenting only the most basic output. Users can then request additional information by selecting additional options – this makes it easier to see the forest and not be distracted by the trees. More generally, JASP will try to shield the user from information they don't need. Complicated analysis options are tucked away under separate tabs with explicit warnings on them (like *Advanced Options*). Also, JASP has over 25 (!) different 'modules' whose functionality will become available only when you activate them. For the analyses covered in this book, you don't need to select any of the modules. But you will undoubtedly sleep better at night knowing that there are more statistics available in JASP than we can cover here. In particular, we won't cover any of the Bayesian philosophy.³ This is ironic because most members of the JASP team adopt this philosophy, and you would have thought that they would have jumped at the opportunity to indoctrinate the new generation.

Something that you will notice immediately is that JASP is free – you can just download it from the website (www.jasp-stats.org) and use it, for any purpose you like, 'no strings attached'. For complete transparency, the JASP source code is publicly available online.⁴ When you use JASP to learn statistics, you will learn to work with a software program that you can still use even after you

¹ Visit <https://jasp-stats.org/team/> to see these lovely people.

² Formatted in the style sanctioned by the American Psychological Association.

³ Despite the fact that only Bayesian procedures are *coherent*, in the sense that they always yield conclusions that are internally consistent. A free textbook on Bayesian inference is available at www.bayesianspectacles.org/ (Wagenmakers & Matzke, 2023).

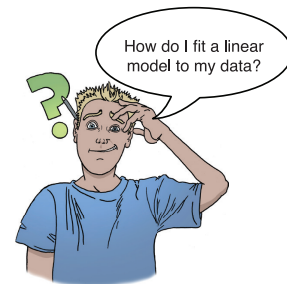
⁴ Visit <https://github.com/jasp-stats/jasp-desktop> to take a look. JASP is built using a mix of programming languages: it uses the R statistical programming language (R Core Team, 2023) for the underlying computations, QML for shaping the user interface, and C++/JavaScript for all things in-between.

graduate. Unfortunately, the fact that JASP is free to use does not mean it is free to develop and maintain. To continue the development of JASP, there is the ‘JASP Community’ that individual universities can join.⁵

Finally, we should indicate our background and biases. This book has existed in many guises. Initially it was *Discovering Statistics Using SPSS* (Field, 2024), but versions exist for the software R (Field, 2026) and SAS (Field & Miles, 2010). The version you hold in your hands exists because the JASP team members Johnny van Doorn and Eric-Jan ‘EJ’ Wagenmakers helped translate the book to JASP. When it comes to JASP, Johnny and EJ cannot pretend to be objective, but don’t worry, I will reign them in when they get too excited. We all agree that JASP is amazing, at least I think that’s what EJ told me to write if I want to get my passport back.


4.3 Versions of JASP

This book is based primarily on JASP version 0.19. New versions of JASP typically come out twice a year, and each new version brings improvements. This book covers the core of the functionality in JASP and focuses on tools that have been in the software for a long time. Consequently, improvements made in new versions of JASP are unlikely to impact the contents of this book (at most, the layout will slightly change). With a bit of common sense, you can get by with a book that doesn’t explicitly cover the latest version (or the version you’re using). So, although this edition was written using version 0.19, it will happily cater to future versions.



4.4 Getting started

JASP mainly uses two modes: the **data editor mode** (this is where you view and edit your data) and the **analysis mode** (this is where you conduct your analyses). When JASP loads, the start-up window in Figure 4.2 appears. The ribbon bar at the top contains all sorts of spooky-sounding analyses, but luckily for you they are greyed out and we can access them only after we have loaded or created some data. JASP can load several types of files, the two most important ones being *.csv* and *.jasp*.⁶ The former, *.csv* (‘comma-separated values’), is a lightweight file format that contains only the raw data. The latter, *.jasp*, contains the original data, the analysis results, and the analysis settings that created the results. This file format is ideal for sharing your research: by opening the file in JASP, your friends, family and professor will be able to see your results and the settings that were used to obtain them. Throughout this book, I will be presenting you with *.jasp* files that contain just the data, but know that *.jasp* files with all the output are also available in the JASP data library and the companion website (at www.discoverjasp.com/), in case you get stuck following the analysis steps on your own.

The top left of Figure 4.2 shows a hamburger button () and clicking it reveals various options. The top option (*Open*) leads to a new list of options all aimed at opening a data file:

- *Recent Files*: Shows the last five files that were opened.
- *Computer*: Load files from somewhere on your computer.

⁵ If you like JASP, consider sending your professor the request form (<https://jasp-stats.org/community-request-form/>).

⁶ The full list of files that JASP can currently read is: *.csv*, *.txt*, *.tsv*, *.ods*, *.dta*, *.zsav*, *.por*, *.sas7bdat*, *.sas7bcat*, *.xpt*, *.xlsx* and the *.jasp* format.

THE JASP ENVIRONMENT

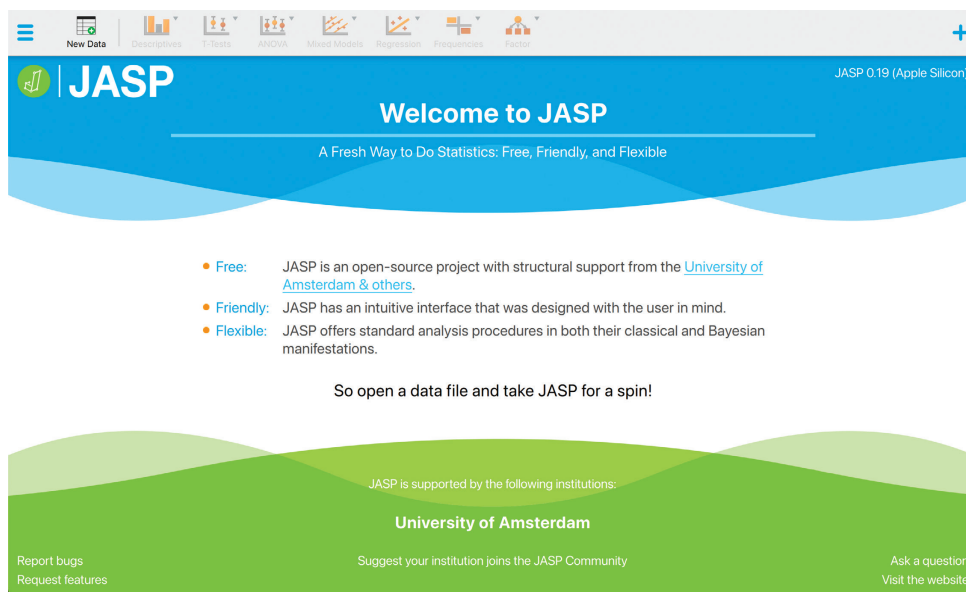




Figure 4.2 The JASP start-up window

- **OSF:** Load files from a repository on the Open Science Framework (<https://osf.io/>), which is a popular platform for hosting research materials such as data, results, analysis scripts and .jasp files. Good research starts with transparency, and posting a .jasp file on the OSF (where it can also be previewed in the browser) is an excellent start.
- **Database:** Load data from an SQL database. You probably won't need this, but it's there.
- **Data Library:** Load data from the JASP data library, which is a collection of data sets (including all data sets mentioned in this book!) for educative purposes. An online version is available at <https://jasp-stats.github.io/jasp-data-library/>, and an accompanying book at <https://edu.nl/bhjj6>

Try loading the *Sleep* data from the data library by going to **≡ > Open > Data Library > 1. Descriptives > Sleep** (see Figure 4.3).⁷ This opens the .csv file for the Sleep data and immediately shows you the data you just requested. Alternatively, you can click  to open the .jasp file from the data library. Doing so opens a premade analysis, including its input menu and annotated output, but we're getting ahead of ourselves. We'll be returning to these menus in the next chapter. As you can see, opening a data file enables all the analyses in the top ribbon bar, and allows you to edit the data by clicking the *Edit Data* icon on the ribbon in the top left corner.

Instead of loading an existing data file, you can also choose to create a data file from scratch by clicking on  (*New Data*) in the JASP start-up window. Regardless of whether you choose to create new data from scratch or edit the Sleep data, the end result is the same: you have now entered the JASP data editor.

⁷ When referring to selecting items in a menu I will use the menu item names (in orange) connected by arrows to indicate moving down items or through submenus. For example, if I were to say that you should select the *Data Library* option in the *Open* menu, you will see *Open > Data Library*.

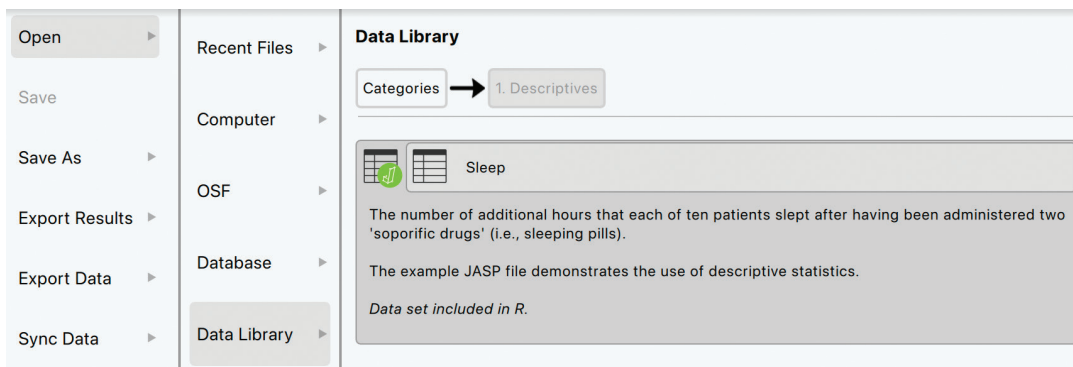


Figure 4.3 Opening the Sleep data from the JASP data library

4.5 The data editor A

Unsurprisingly, the data editor is where you enter and edit data (Figure 4.4 if you loaded the Sleep data – if you went straight to this mode, the cells will be empty). As I am sure you're aware, you can navigate menus by using your mouse/trackpad to move the on-screen arrow to the menu you want and pressing (*clicking*) the left mouse button once. The click will reveal a list of menu items in a list, which again you can click using the mouse. In JASP if a menu item is followed by a ▼ then clicking on it will reveal another list of options (a *submenu*) to the right of that menu item; if it doesn't then clicking on it will immediately do a thing, such as returning to *analysis mode*, or undoing an operation.

	extra	group	ID
1	0.7	1	1
2	-1.6	1	2
3	-0.2	1	3
4	-1.2	1	4
5	-0.1	1	5

Figure 4.4 The JASP data editor for the Sleep data

Let's look at some features of the data editor. The **data view** is made up of lots of *cells*, which are boxes in which data values can be placed. Each cell has a row (horizontal) and a column (vertical), which can be selected by clicking on the row number or column header (e.g., **group** in Figure 4.4). When there are variables, the column header shows their name and variable type (see Section 4.6.2). An individual cell can be selected by clicking on it. When a cell, row or column is active it becomes highlighted. You can move around the data editor, from cell to cell, using the arrow keys ← ↑ ↓ → (on the right of the keyboard) or by clicking on the cell that you wish to activate. To enter a number into the data editor move to the cell in which you want to place the data value, type the value, then press the appropriate arrow button for the direction in which you wish to move. So, to enter a row of data, move to the far left of the row, type the first value and then press → (this process inputs the value and moves you into the next cell on the right).

THE JASP ENVIRONMENT

Next, let's look at the items in the top ribbon bar (similar options are available when you right-click a row, column, or cell):



Analyses



Synchronisation



Insert



Remove



Undo



Redo

Return to *Analyses*, where the top ribbon bar features the analyses. In *Analyses* mode, you still see the data, but will not be able to edit its values.

If a data file was loaded, synchronization can be enabled so that the file can also be edited in other software (such as Excel). When the file is changed elsewhere, JASP will immediately recognize these changes and update the data that are shown. If data are from the data library or a .jasp file, it will first generate a .csv file on your computer.

Insert a new row or column. The first four options will insert an empty row or column. The bottom four options allow you to create a column using the column constructor, or R code/syntax (see Section 4.6.2.4).

Remove the selected column, row, or cell contents. If a row is selected, and *Delete column* is selected, everything will be deleted.

Undo an operation, such as creating or deleting a column or value, because at some point we all execute some operation and immediately regret it.

Redo an operation, such as creating or deleting a column or value

4.6 Entering data into JASP

We will now explore the specifics of data handling in JASP.

4.6.1 Data formats

There are two common data entry formats, which are sometimes referred to as **wide format data** and **long format data**. In the wide format *each row represents data from one entity and each column represents a variable*. There is no distinction between predictor (independent) and outcome (dependent) variables: both appear in a separate column. The key point is that each row represents one entity's data (be that entity a human, mouse, tulip, business, or water sample) and any information about that entity should be entered across the data editor. Contrast this with long format, in which scores on an outcome variable appear in a single column and rows represent a combination of the attributes of those scores. In long format data, scores from a single entity can appear over multiple rows, where each row represents a combination of the attributes of the score (the entity from which the score came, to which level of an independent variable the score belongs, the time point at which the score was recorded, etc.). The distinction between wide and long is relevant only when we have more than a single observation per entity (as in Chapters 9 and 14), and where relevant we will use the wide format.

Let's look at an example. Imagine you were interested in how perceptions of pain created by hot and cold stimuli were influenced by whether or not you swore while in contact with the

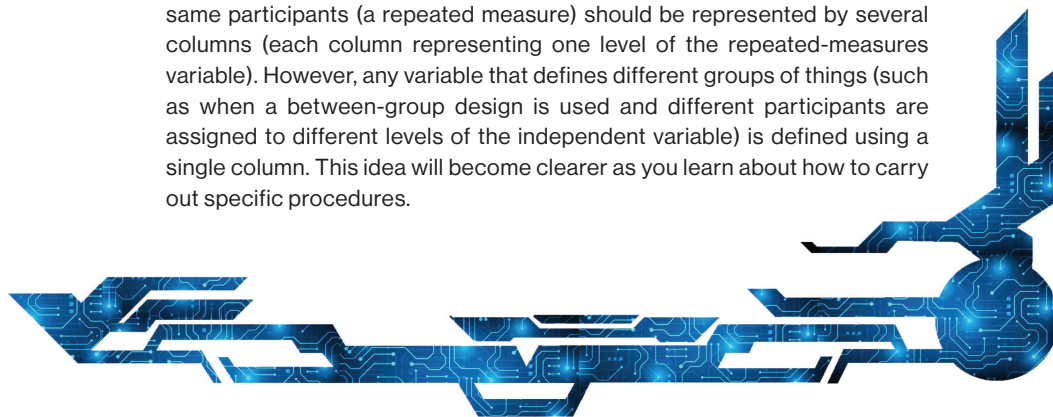
stimulus (Stephens et al., 2009). You time how many seconds each of six people can keep their hand in a bucket of ice-cold water. You then time for how many seconds they can hold a hot potato. Half the participants are encouraged to shout profanities during the experiences. Figure 4.5 shows the data for our six fictional participants. The first person who was encouraged to swear held her hand in the ice water for 86 s and held the hot potato for 67 s. In wide format every participant's information is stored in a single row. There is a column representing the group to which they were assigned. This column contains what's known as a grouping variable, nominal variable, coding variable or, in experimental designs, an independent measure: an entity can belong to either the group that could swear or the group that was forbidden, but not both (see Section 4.6.2.2). The two timings make up a repeated measure because all participants were timed while in contact with a hot and with a cold stimulus. In wide format repeated measures are spread across columns in JASP. So, for this example, in wide format the data for each participant occupies a single row, with their group membership, time with their hand in ice water, and time holding a hot potato spread across three columns (see JASP Tip 4.1).



JASP Tip 4.1 Wide format data entry

When using the wide format, there is a simple rule: data from different entities go in different rows of the data editor, whereas data from the same entities go in different columns of the data editor. As such, each person (or mollusc, goat, organization, or whatever you have measured) is represented in a different row. Data within each person (or mollusc, etc.) go in different columns. So, if you've prodded your mollusc, or human, several times with a pencil and measured how much it, or she, twitches as an outcome, then each prod will be represented by a column.

In experimental research this means that variables measured with the same participants (a repeated measure) should be represented by several columns (each column representing one level of the repeated-measures variable). However, any variable that defines different groups of things (such as when a between-group design is used and different participants are assigned to different levels of the independent variable) is defined using a single column. This idea will become clearer as you learn about how to carry out specific procedures.



THE JASP ENVIRONMENT

Contrast this format with long format. In long format, every *observation* occupies a row in the data editor and columns are used to code information about that observation (e.g., the participant from which it came, whether they could swear, and whether the score relates to the ice water or the potato). In this format, participants' scores are spread across multiple rows (in this specific example each participant occupies two rows, one containing their score for the ice water task and the other for the hot potato task).

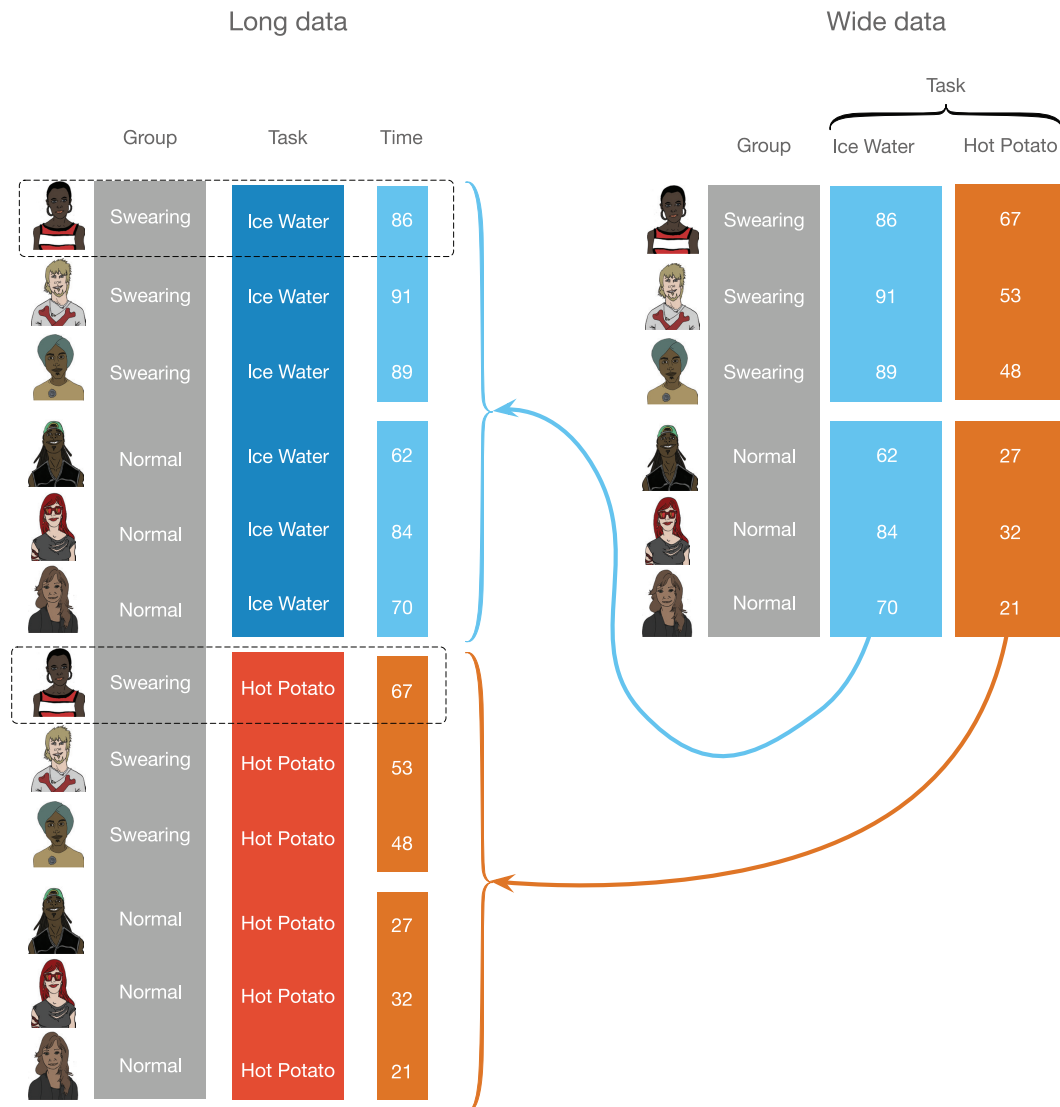


Figure 4.5 Wide and long data formats

4.6.2 Variable types A

As we saw in Section 1.5.2, there are several types of measurements. In JASP, there are three types of variables you can specify:



Nominal

Nominal variables, also called grouping or categorical variables, represent different groups or categories of data. The groups of data represented by nominal variables could be levels of a treatment variable in an experiment (an experimental group or a control group), different naturally occurring groups (ethnic groups, gender identity, socioeconomic categories, marital status, etc.). There is no ordering to these categories.



Ordinal

Ordinal variables are a type of categorical variable that have an ordered relationship among their categories. Unlike nominal variables, the categories of ordinal variables can be ranked or arranged in a meaningful sequence, but the intervals between the categories are not necessarily equal. Examples of ordinal variables include levels of satisfaction (e.g., very unsatisfied, unsatisfied, neutral, satisfied, very satisfied), educational attainment (e.g., high school, associate's degree, bachelor's degree, master's degree, doctorate), and rating scales (e.g., 1 to 5 stars).



Scale

Scale variables, also known as continuous or interval variables, represent quantities with a meaningful order and equal intervals between values. They can take on any value within a range and include measures like temperature, height, weight, and income. These variables allow for a wide range of statistical analyses due to their precise and quantifiable nature.


In JASP, each of these variables has a value and a label. The *value* is what is being passed to the statistics genie⁸ inside JASP who does the computations, while the *label* is used to dress up the plots and tables in the output. While nominal and ordinal variables can have non-numeric values, scale variables need to have numeric values to be useful. For the labels, this is a lot more relaxed, and any variable can have any type of label.

Let's put this into practice and use the data editor to create some variables. We're going to input some data about the rock band Metallica. The complete data are in Table 4.1.

Table 4.1 Some Metallica-data


Name	Instrument	Current member	Headbanging intensity	Songs	Net worth (\$ million)	Net worth per song (\$ million)
Lars Ulrich	Drums	Yes	Light	123	350	2.85
James Hetfield	Guitar	Yes	Heavy	124	300	2.44
Kirk Hammett	Guitar	Yes	Light	60	200	3.33
Rob Trujillo	Bass	Yes	Moderate	19	40	2.11
Jason Newsted	Bass	No	Heavy	3	60	20.00
Cliff Burton	Bass	No	Heavy	11	1	0.09
Dave Mustaine	Guitar	No		6	14	2.33




4.6.2.1 Creating nominal variables



The first variable in Table 4.1 is the name of the band member. This variable is a *nominal variable* because it consists of names. To create this variable in the data editor (make sure you start with a clean slate by starting a fresh instance of JASP and clicking on ):

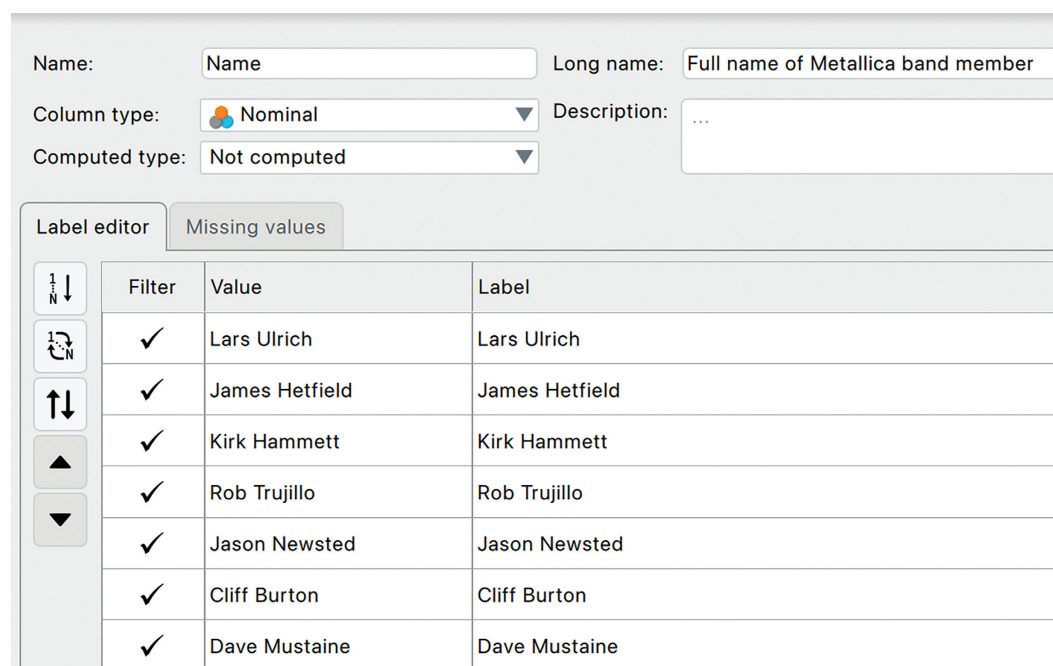
⁸ The genie is in fact the statistical computing language R, which powers all analyses in JASP.

THE JASP ENVIRONMENT

- 1 Click in the first white cell in the column labelled *Column 1*.
- 2 Type the first name to input: 'Lars Ulrich'.
- 3 Move from this cell to the next using the arrow keys or the  key on the keyboard (you can also just click in the next cell, but this is a very slow way of doing it).
- 4 Repeat steps 2 and 3 for each of the remaining band members.

Well done, you've just created your first variable. Notice that once you've typed the first observation, JASP creates default settings for the variable (such as assuming it's nominal, as you can see from the icon in the column name). 'Column 1' is not such an informative variable name (see JASP Tip 4.2) so we'll rename it by double-clicking on the name to open the **Variable settings** menu (see Figure 4.6). Here, you can change 'Column 1' in the *Name* box into something more informative, such as **Name**. You can use the *Long name* box to write something more elaborate, such as 'Full name of Metallica band member'. This name is not used in the JASP output but informs your peers and future you (!) what the variable means. In the *Description* box you can be even more elaborate and provide a full description of the immense euphoria you are experiencing while following these instructions. Both the long name and description will appear when you hover over the column names with your mouse. In the drop-down menu labelled *Column type*, you can change the variable type to , , or  in case you are not happy with the current setting.

The *Label editor* allows you to change the value and/or label of your observations. While not very useful here, you could specify a numeric representation for each of the band members by changing their *Value*. We will return to this option when we tackle variables where this is more relevant. The *Filter* column lets you specify which categories you would like to include in your analyses. Clicking on the  turns it into a  and will exclude that category. So if you feel like excluding James Hetfield from your analyses, here's your chance.



The screenshot shows the 'Variable settings' dialog for a variable named 'Name'. The 'Name' field contains 'Name' and the 'Long name' field contains 'Full name of Metallica band member'. The 'Column type' is set to 'Nominal' (indicated by a nominal icon) and the 'Computed type' is 'Not computed'. The 'Description' field is empty. Below these fields is the 'Label editor' tab, which contains a table with columns 'Filter', 'Value', and 'Label'. The table lists seven band members, all with checkmarks in the 'Filter' column and their names in the 'Value' and 'Label' columns.








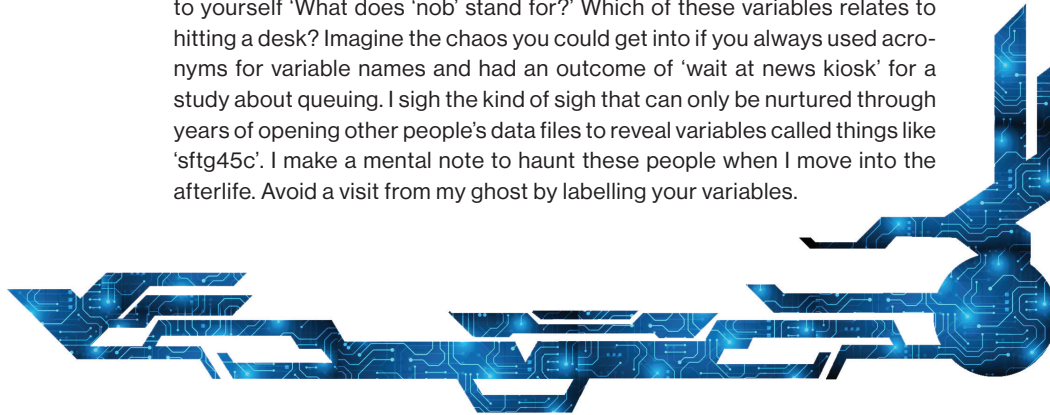
Filter	Value	Label
	Lars Ulrich	Lars Ulrich
	James Hetfield	James Hetfield
	Kirk Hammett	Kirk Hammett
	Rob Trujillo	Rob Trujillo
	Jason Newsted	Jason Newsted
	Cliff Burton	Cliff Burton
	Dave Mustaine	Dave Mustaine

Figure 4.6 Variable settings menu for Name



JASP Tip 4.2 Naming variables

'Surely it's a waste of my time to type in long names for my variables when I've already given them a short one?', I hear you ask. I can understand why it would seem so, but as you go through university or your career accumulating data files, you will be grateful that you did. Imagine you had a variable called 'number of times I wanted to bang the desk with my face during Andy Field's statistics lecture'; then you might have named the column in JASP 'nob' (short for number of bangs). You thought you were smart coming up with such a succinct label. If you don't add a more detailed label, JASP uses this variable name in all the output from an analysis. Fast forward a few months when you need to look at your data and output again. You look at the 300 columns all labelled things like 'nob', 'pom', 'p', 'lad', 'sit' and 'ssoass' and think to yourself 'What does 'nob' stand for?' Which of these variables relates to hitting a desk? Imagine the chaos you could get into if you always used acronyms for variable names and had an outcome of 'wait at news kiosk' for a study about queuing. I sigh the kind of sigh that can only be nurtured through years of opening other people's data files to reveal variables called things like 'sftg45c'. I make a mental note to haunt these people when I move into the afterlife. Avoid a visit from my ghost by labelling your variables.




You can practice the creation of nominal variables by adding two more variables: **Instrument** and **Current member**, in the same way that you created the **Name** variable. In experiments that use an independent design, nominal variables represent predictor (independent) variables that have been measured between groups (i.e., different entities were assigned to different groups). We do not, generally, use this kind of coding variable for experimental designs where the independent variable was manipulated using repeated measures (i.e., participants take part in all experimental conditions). For repeated-measures designs we typically use different columns to represent different experimental conditions (i.e., the wide data format).

Think back to our swearing and pain experiment. This *was* an independent design because we had two groups representing the two levels of our independent variable: one group could swear during the pain tasks, the other could not. We might assign the experimental group (swearing) a value of 1 and the control group (no swearing) a value of 0. To input these data you would create a variable (which you might call 'group') and type the value 1 for any participants in the experimental group, and 0 for


THE JASP ENVIRONMENT

any participant in the control group. These codes tell JASP that the cases that have been assigned the value 1 should be treated as belonging to the same group, and likewise for the cases assigned the value 0. The values you use are arbitrary because the numbers themselves won't be analysed, so although people typically use 0, 1, 2, 3, etc., if you're a particularly arbitrary person feel free to code one group as 616 and another as 11 and so on.



SELF TEST

Using what you have learned, try specifying the values for the variable **Current member**, for yes = 1 and no = 0.



4.6.2.2 Creating ordinal variables A

We have an *ordinal variable* in our data that codes the headbanging intensity of each of the members during their live shows. While a little subjective, it seems like relevant information to keep track of when characterizing a rock band.⁹ There are various ways to operationalize the headbanging intensity, and here I have chosen a three-point scale: light, moderate, heavy. While these values are also categorical (i.e., not numeric), there is an inherent ordering to them. As such, we need to make sure that this ordering is encoded properly when we input the data. We start in the same

Name:

Column type:

Ordinal

Computed type:

Not computed

Label editor

Missing values

1 ↓

↺

↕

▲

▼

Filter	Value	Label
✓	1	Light
✓	2	Moderate
✓	3	Heavy

Figure 4.7 Specifying the values for an ordinal variable

⁹ These ratings are from an online forum thread assessing each member's headbanging ferocity, they are not my own judgements. Incidentally, shaking your head about is almost certainly not good for your brain.

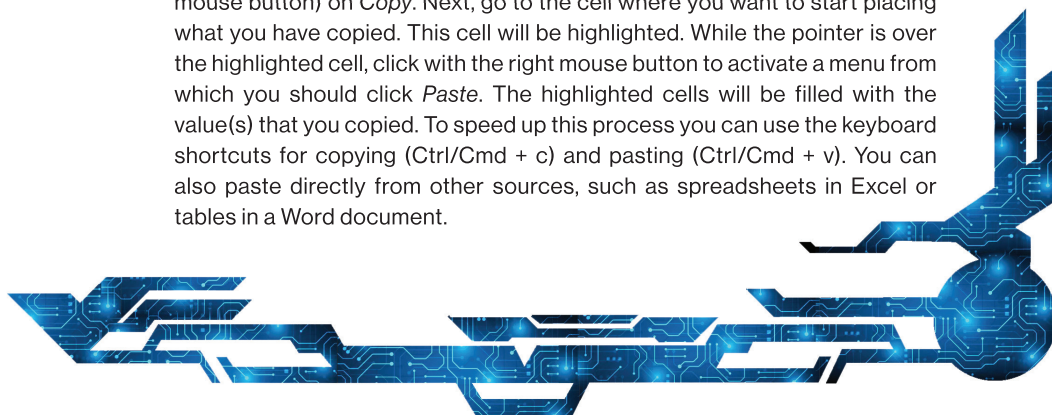
way as we did before, by initializing the variable. This can be done by specifying the first observation, but we can also start by first naming the variable and setting its type. To do so, click on the header of the first empty column (the fourth one, if you have been following along). In the *Variable settings* menu, specify the name of the new variable (e.g., **Headbanging intensity**), and set the *Column type* to *Ordinal*. Next, the observations can be specified in the same way we have done before. The last value is missing, since nobody assessed Dave's headbanging, so you can just leave that cell empty. We discuss missing values in Section 4.6.3.

Now we need to make sure that the ordering of the observations is encoded properly by taking a look at the *Variable settings* menu again. If you entered the observations as they appear in Table 4.1, the current ordering is Light, Heavy, Moderate, since JASP just takes the order in which each observation was entered, and Heavy was entered before Moderate. To make sure that Heavy is treated as the highest category, we can specify numeric *Values* for each of the categories.



JASP Tip 4.3 Copying and pasting into the data editor and variable viewer

Often (especially with nominal variables) you need to enter the same value lots of times into the data editor. Similarly, in the variable view you might have a series of variables that all have the same value labels (e.g., variables representing questions on a questionnaire might all have value labels of 0 = never, 1 = sometimes, 2 = always to represent responses to those questions). Rather than typing the same number or word lots of times, or entering the same value labels multiple times, you can use the copy and paste functions to speed things up. Select the cell containing the information that you want to copy (whether that is a number or text in the data editor, or a set of value labels or another characteristic within the variable view) and click with the right mouse button to activate a menu within which you can click (with the left mouse button) on *Copy*. Next, go to the cell where you want to start placing what you have copied. This cell will be highlighted. While the pointer is over the highlighted cell, click with the right mouse button to activate a menu from which you should click *Paste*. The highlighted cells will be filled with the value(s) that you copied. To speed up this process you can use the keyboard shortcuts for copying (Ctrl/Cmd + c) and pasting (Ctrl/Cmd + v). You can also paste directly from other sources, such as spreadsheets in Excel or tables in a Word document.



THE JASP ENVIRONMENT

This means filling in Value = 1 for Light, Value = 3 for Heavy, and Value = 2 for Medium. Additionally, we could then sort the categories by their value by pressing the sort button (↕) located to the left of the label box. Sorting the categories influences the order in which they are shown in plots and tables. You can also manually reorder the categories by using the up and down arrows. Pressing the reverse label button (↕) will reverse the order of the labels (including their values) as they are shown. The finished coding window should look like in Figure 4.7.

The reverse score button (↕) will reverse the values associated with the labels, so that Heavy will be assigned the value 1, and Light the value 3. This can be convenient when you analyse a questionnaire and some items are reverse-coded.

4.6.2.3 Creating numeric variables A

Our next variable is **Songs**, which is a count of the number of songs that the person has contributed to writing. This variable is numeric. Go to the next empty column (the fifth column, if you have been following along) in the data editor, type 123 in the first cell of the column, and press the ☐ key on the keyboard. JASP will now recognize that you are trying to specify a **numeric variable** and will create a new numeric variable called 'Column 5'. Our variable is numeric so we don't need to change the variable type, but it is a good idea to change the name of the variable by double-clicking on the column name to open the *Variable settings* menu. Here you can again change the variable name to something more informative, such as **Songs**. Let's continue our good habit of naming variables and move to the box labelled *Long name* and type 'Number of song writing credits'. Now you can also fill out the remaining values (or copy and paste them directly from Table 4.1).

The next variable in the data is the net worth of each band member. This variable is a currency variable, and is recorded in US dollars. It is also likely to be highly inaccurate because the data were gathered from www.celebritynetworth.com (in February 2023), a site that more than likely doesn't know the actual net worth of anyone. However, Lars has blocked my number so I can't get more accurate estimates until the restraining order is lifted (perhaps I should stop calling him a light headbanger?). You can follow the same process as we did for **Songs**, but now call the variable **Net worth**.




4.6.2.4 Computing a variable B

Instead of creating variables by manual input, you can use JASP's *compute columns* functionality. This is useful when you want to create a new variable based on one or more existing variables. In this case, we can use it for computing a new variable **Net worth per song** by using the existing variables **Songs** and **Net worth**. When you click on an empty column, a fresh *Variable settings* menu will open, ready for some input. You start by writing the name (**Net worth per song**) in the *Name* box, and then select one of the two options from the drop-down menu next to *Computed type*:

- *Computed with R code*: Those lucky enough to be blessed with R skills can write any R code that computes a new variable.
- *Computed with drag-and-drop*: Apply various operations using a graphical user interface.

When any of these options are selected, a new tab called *Computed column definition* will open where you can define the new column. Figure 4.8 shows what this looks like for the drag-and-drop. On the

left-hand side, there is a list with all the variables in your data set. On the top, there is a series of operations that can be applied, such as addition and division (hover over them to see what each does, in case it does not look familiar). On the right-hand side, various functions are available that can be applied to a variable, such as converting to z-scores, taking the mean, or log-transforming it. To obtain the net worth per song, we need to divide **Net worth** by **Songs**. First click **Net worth**, which places it in the big box, then click the division sign (\div), which will automatically use **Net worth** as the numerator. Then click **Songs**, which will place it as the denominator. To finalize the computation, click on the big button *Compute column* and your new column will be filled with freshly divided values. If at any point you get stuck and want to reset the window, you can double-click the bin () in the lower right corner. You can also drag individual items to the bin to remove them from the screen. When a column is computed, JASP adds an ' f_x ' to the column name to remind you that this column was computed. You can always return to the compute settings by double-clicking the column name.

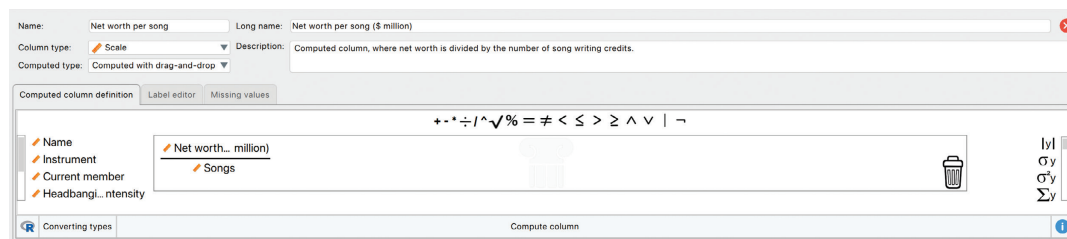



Figure 4.8 The drag-and-drop interface for computing a new variable

As a bonus, you can click the  button in the bottom left corner, which will then show you the underlying R code of the computation you just did. This is handy for sharing your operations, and can even teach you a little bit of R. You can even test whether that R code works, by copying the R code, then selecting *Computed with R code* in the *Computed type* box, pasting the code, and clicking *Compute column* – this should lead to the same values as before. Also my congratulations are now in order, because you might have just run your very first R code!

The possibilities with the **compute columns** functionality are vast, especially when using the R mode. We return to the compute columns functionality when we discuss variable transformations for more robust inference (Chapter 6).

4.6.3 Missing values

Although we strive to collect complete sets of data, often scores are missing. Missing data can occur for a variety of reasons: in long questionnaires participants accidentally (or, depending on how paranoid you're feeling, deliberately to irritate you) miss out questions; in experimental procedures mechanical faults can lead to a score not being recorded; and in research on delicate topics (e.g., sexual behaviour) participants may exercise their right not to answer a question. However, just because we have missed out on some data for a participant, we don't have to ignore the data we do have (although it creates statistical difficulties). The simplest way to record a missing score is to leave the cell in the data editor empty (which is what we have done for **Headbanging intensity**), but it can be helpful to tell JASP explicitly that a score is missing. By default, JASP has various values that it automatically recognizes as missing: 'NA', 'NaN', 'nan', and '.'. When any of these values are specified in a cell, JASP knows that there is missingness. You also have the freedom to specify custom values to represent a missing data point, by going to the *Missing values* tab in *Variable settings* and checking the



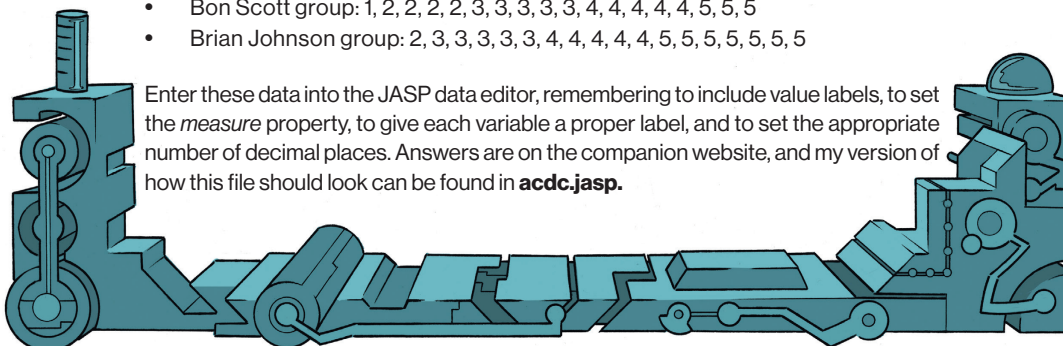
Labcoat Leni's Real Research 4.1 Gonna be a rock 'n' roll singer A

Oxoby, R. J. (2008). *Economic Enquiry*, 47(3), 598–602.

AC/DC are one of the best-selling hard rock bands in history, with around 100 million certified sales, and an estimated 200 million actual sales. In 1980 their original singer Bon Scott died. He was replaced by Brian Johnson who has been their singer ever since.¹⁰ Debate rages with unerring frequency within the rock music press over who is the better frontman. The conventional wisdom is that Bon Scott was better, although personally, and I seem to be somewhat in the minority here, I prefer Brian Johnson. Anyway, Robert Oxoby in a playful paper decided to put this argument to bed once and for all (Oxoby, 2008).

Using a task from experimental economics called the ultimatum game, individuals are assigned the role of either proposer or responder and paired randomly. Proposers are allocated \$10 from which they have to make a financial offer to the responder (e.g., \$2). The responder can accept or reject this offer. If the offer is rejected neither party gets any money, but if the offer is accepted the responder keeps the offered amount (\$2), and the proposer keeps the original amount minus what they offered (\$8). For half of the participants the song 'It's a long way to the top' sung by Bon Scott was playing in the background, for the remainder 'Shoot to thrill' sung by Brian Johnson was playing. Oxoby measured the offers made by proposers, and the minimum offers that responders accepted (called the minimum acceptable offer). He reasoned that people would accept lower offers and propose higher offers when listening to something they like (because of the 'feel-good factor' the music creates). Therefore, by comparing the value of offers made and the minimum acceptable offers in the two groups he could see whether people have more of a feel-good factor when listening to Bon or Brian. The offers made (in \$) are¹¹ as follows (there were 18 people per group):

- Bon Scott group: 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5
- Brian Johnson group: 2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5



Enter these data into the JASP data editor, remembering to include value labels, to set the *measure* property, to give each variable a proper label, and to set the appropriate number of decimal places. Answers are on the companion website, and my version of how this file should look can be found in **acdc.jasp**.

¹⁰ Well, until all that weird stuff with W. Axl Rose in 2016, which I'm trying to pretend didn't happen.

¹¹ These data are estimated from Figures 1 and 2 in the paper because I couldn't get hold of the author to get the original data files.

box *Use custom values* (Figure 4.9). In the box on the right, you write the value you want, and click **+**. For obvious reasons, it is important to choose a value that cannot naturally occur in the data. For example, if we use the value 9 to code missing values and several participants genuinely scored 9, then JASP will wrongly treat those scores as missing. You need an ‘impossible’ value, so people usually pick a score greater than the maximum possible score on the measure. For example, in an experiment in which attitudes are measured on a 100-point scale (so scores vary from 1 to 100) a reasonable code for missing values might be 101, 999 or, my personal favourite, 666 (because missing values *are* the devil’s work).

JASP allows you to specify multiple values to represent missing data. The reason why you might choose to have several numbers to represent missing values is that you can assign a different meaning to each discrete value. For example, you could have the number 101 representing a response of ‘not applicable’, a code of 102 representing a ‘don’t know’ response, and a code of 999 meaning that the participant failed to give any response. JASP treats these values in the same way (it ignores them), but different codes can be helpful to remind you of why a particular score is missing.

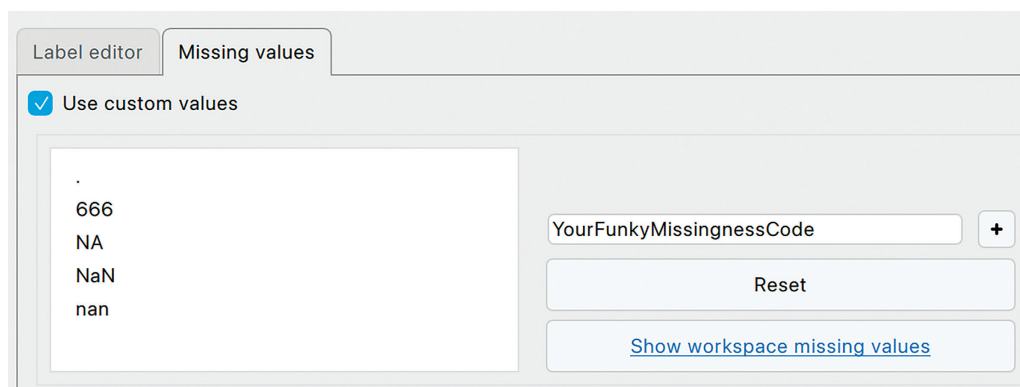


Figure 4.9 Defining missing values


4.6.4 Filtering data B

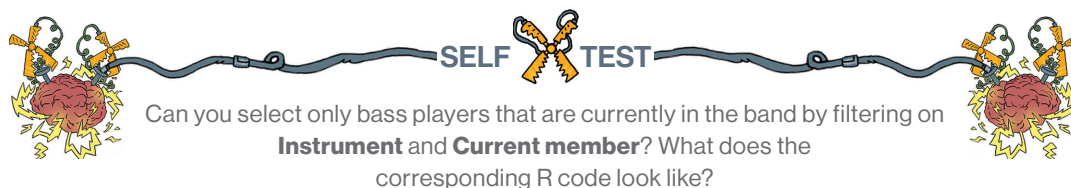
Sometimes you don’t want to use all the data you have collected and would rather analyse only a subgroup of your sample. For instance, maybe you want to look at the net worth of only the guitar players in Metallica or only analyse those members who have written over 10 songs. In JASP, you can apply various types of **filters**, such that your analyses will only use a subset of your data. One way to filter data based on a nominal or ordinal variable is by going to its *Variable settings* (see Figure 4.6) and adjusting the checkmarks in the *Filter* column. To only analyse the guitar players, make sure that the Guitar category is the only one that has a ✓ in the *Variable settings* of **Instrument**. You can see the consequences of your filter in the data, because JASP will grey out any rows that are being filtered out. JASP will also add a **▼** to the column name to indicate which variable is responsible for the filtering.

A more versatile method of filtering is available by clicking the **▼** button that is shown in the top left corner of the data window (either in data editing mode or analysis mode). This opens another drag-and-drop window, just like in compute columns. Instead of generating a new variable, however, here we can pass along various criteria that we would like to filter on. For instance, if we want to select only those members who have more than 10 song writing credits, we first select the **Songs** variable from the list on the left. At the top of the screen there are various logical operators that can be used to specify selection criteria. The one we need here is the ‘greater than’ symbol (>); when selected, it gets placed behind **Songs**, and three little dots appear right after it, which means we can click on them to type a value (or drag another variable there, if we want to compare one variable to another). In this case, type the number 10 to complete the criterion. Now you can press the big button at the

THE JASP ENVIRONMENT

bottom, *Apply pass-through filter*, and you'll see that only those band members with more than 10 songs are selected and that **Songs** has received a **T** next to it. To remove the filter you double-click on the bin (🗑).

Filtering can also be done using R code by selecting the **R** button. When you do this when some filters are already applied, JASP will show the corresponding R code in the top field. Clicking on the  will remove those filters, so you can start from scratch. The R code here is useful for transparency and reproducibility, because it makes it easy to communicate to others how your filter was specified.



4.7 JASP modules

By default, JASP contains the following *modules* (i.e., analyses) in the top ribbon bar:



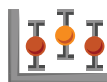
Descriptives

We'll use this for conducting descriptive statistics (mean, mode, median, etc.), frequencies and general data exploration, including various nifty visualizations (Chapter 5).



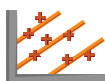
T-Tests

We'll use this menu for *t*-tests (independent and paired; Chapter 9) for comparing means between two groups.



ANOVA

This menu is for comparing means between two or more groups, typically experimental designs in which you have manipulated a predictor variable using different cases (independent design), the same cases (repeated measures design) or a combination of these (mixed designs), covered in Chapters 11–14.



Mixed Models

This menu features something called multilevel linear models, which is an advanced modelling technique from which you'll be spared in this book (but who knows what a second edition might bring ...).



Regression

Here measures of correlation hang out, including bivariate correlations such as Pearson's *r*, Spearman's rho (*ρ*) and Kendall's tau (*τ*) and partial correlations (see Chapter 7). Also included are linear regression (Chapter 8) and logistic regression (Chapter 17).



Frequencies

For exploring frequency data and performing tests such as chi-square and Fisher's exact test (Chapter 16)



Factor

You'll find factor analysis here, which is not covered in this book.

The fun does not stop here, however. As mentioned earlier, when you click on the **+** in the top right corner, there is a huge list of extra modules that you can add to the top bar by checking the box in front of it. Each of these modules contain additional functionality that some people might really appreciate, but that would clutter the JASP interface for everybody else. Later in this book (Chapter 10), we will look at the *Process* module, so you can already practice enabling and disabling this module.

4.8 The input window

Once your data are in JASP, you'll want to analyse them. Besides the data window (which gives you the spreadsheet), there are two additional windows in JASP: the analysis **input window** and the analysis **output window**. You can resize the various panels in JASP by dragging the grey columns where you see the three dots (⋮), and collapse or expand a panel by clicking the arrow buttons (▶ or ◀). When one of the analyses in the top ribbon bar is activated, the input window will open where you can select all sorts of options for that analysis. Figure 4.10 shows what the input window looks like for the Descriptives module, which can provide you with some descriptive statistics (e.g., mean and standard deviation) and plots (e.g., histogram and Q-Q plot – see Chapters 5 and 6). The next chapter dives into this module properly, but let's already review some general features of the input window.

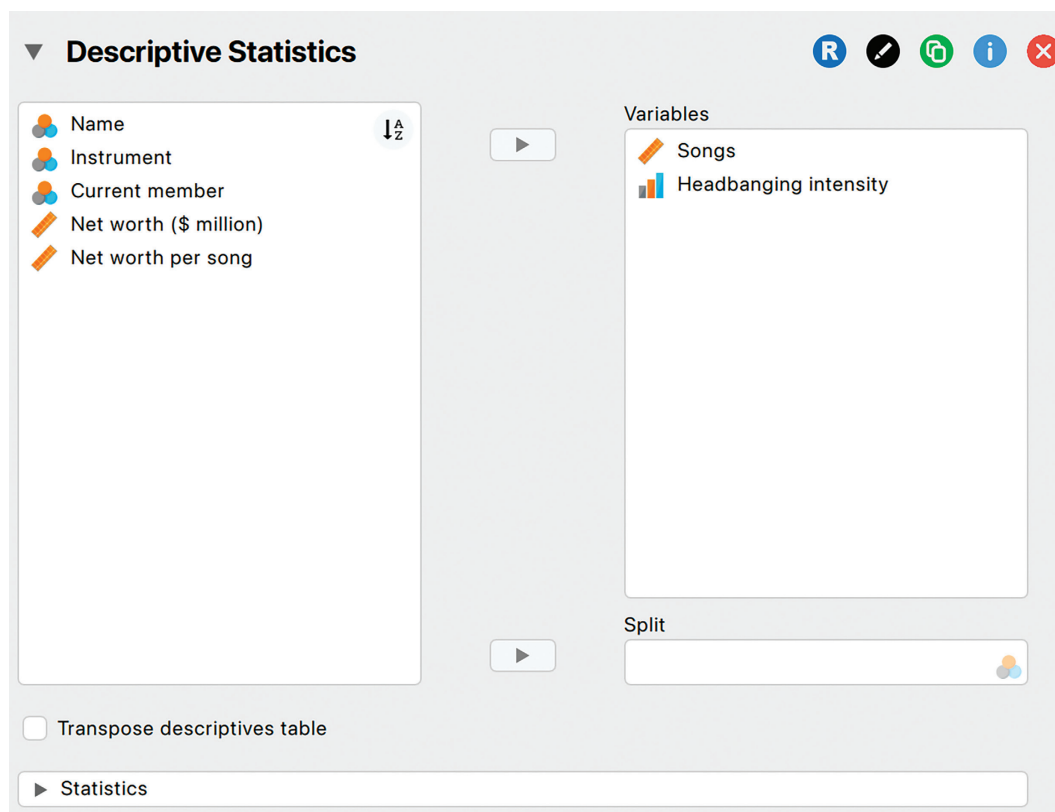










Figure 4.10 Input window for the Descriptives module

Whenever you select an analysis from the top ribbon bar, a new *analysis menu* will appear in the input window that can be collapsed/expanded using the ▶/▼ buttons next to the analysis' name. There are various icons at the top of each analysis menu:

THE JASP ENVIRONMENT

-  Show/hide the R code that can be used to recreate this analysis using code mode (see Section 4.9.1).
-  Edit the analysis header name as it appears in the input and output window, which is handy if you have many analyses in a single file.
-  Duplicate the analysis. This is handy when you want to execute the same analysis, and explore the effect of changing some of the input settings. Note that such explorations quickly lead to additional researcher degrees of freedom and should *not* be used to change the settings until you get the results you want (see Section 3.3.2).
-  Access the analysis helpfiles, which describe the analysis options and all the output elements.
-  Remove the analysis.

Most of the analysis menus will have their options divided over several *tabs*, or *submenus*, based on their purpose. For instance, the Descriptives module has tabs for numeric output, basic plots, and for customizable (i.e., more advanced) plots. Each tab can also be collapsed/expanded using the ►/▼ buttons next to the tab's name.

Most JASP analyses feature a set of *assignment boxes* because they need you to tell them onto which variables to unleash their statistical magic. Often, there is a box on the left with the available variables and a box on the right where those variables can end up (i.e., the *input box*). In Figure 4.10, we selected **Songs** and **Headbanging intensity**, so JASP will compute some descriptives for both of those variables. To select variables for the analysis, you drag a variable to the *Variables* box on the right or click ► while you have the relevant variable(s) selected. Some analyses allow only certain types of variables to be used as input. When this is the case, the input box shows the icons of the allowed variable types , , or . When you try to specify a variable whose type does not match the allowed type, JASP will try its best to convert the variable to the allowed type for that analysis.


4.9 The output window

When the input is opened by selecting an analysis, the output window will also open and will be updated in real time, as the analysis options are specified. The output window is a separate panel that displays the output of any procedures in JASP: tables of results, plots, error messages and pretty much everything you could want, except for photos of your cat. I won't go too much into detail here, because that is what the rest of this book is for, but I will highlight some useful features of the output window.

When there is any output in the output window, there will usually be a drop-down menu available next to its name, indicated by ▼ when you hover over it. The plots and tables from the output window can be copied directly from there, straight to a text document (e.g., .odt or .docx) or presentation document (e.g., .odp or .pptx). JASP's tables are already in APA format, so they are already publication ready if you happen to be on a course (or want to submit your work to a journal) that uses that format! Or, because they are table objects, you can further tweak them in your text document.

The drop-down menu for the plots includes the option to save the plot to a separate file, or to edit the plot with JASP's plot editor, which we will cover in more detail in the next chapter. For tables and titles, you can select to *Add Note* to annotate the output. In the text field that opens when you select that option, you can do all sorts of things, such as embed images and videos, write a formula, or just give a plain description of what can be seen in the output. An example is given in Figure 4.11, which shows distribution plots (covered in Chapter 5) together with annotations. Unfortunately, the hyperlink to the cute cat video does not work in this figure, but I'm sure you can find such videos by yourself (take a cat video break now, I'll wait).

Distribution Plots

We can have fancy $LaTeX$ formulas in here, [link cute cat video's](#), or insert a drumkit . Below are two distribution plots outlining the members of Metallica. On the left, we see the various instruments being played and their frequencies, and on the right we see how many members are still active, and how many left the band.

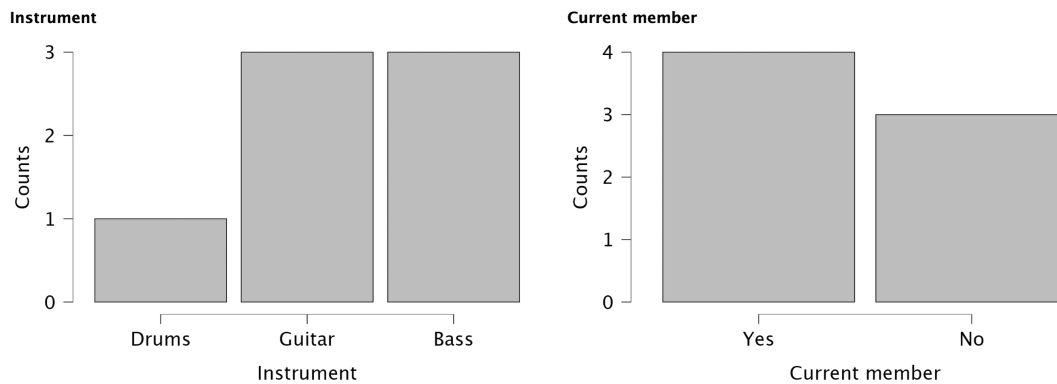






Figure 4.11 Example of annotated output




4.9.1 R integration



For most people, ‘doing stuff’ involves navigating through the modules of JASP and selecting the analysis they want to carry out using a graphical user interface. This book assumes that’s what you’ll do. While clicking on stuff to get results is pretty straightforward, JASP also features a steadily expanding **code mode** which takes advantage of its underlying statistical programming language R – you can learn more about this language in Field (2026). When you use code you’re typing and/or viewing instructions for JASP to follow instead of generating those instructions by clicking on the screen. Most beginners ignore code, but using it can help you communicate exactly how you arrived at your results, which is an important part of engaging in open science practices (Section 3.6). Sharing a .jasp file will also show others how you have arrived at your results, but it requires others to load the file into JASP and look through the options that were selected. There are two ways in which you can access the underlying R code:

- *In the input window:* When you click on the  button at the top of the analysis menu, a window will open below, where you see the R code. Here, you can copy the code and share it with someone else or tweak the code to update the options (note how the graphical options also change when you adjust the code).
- *In the output window:* At the top of the output, click on *Results* ▼ to access a drop-down menu where you can *Show R code*. This will display, for each analysis in the output, the underlying R code. You can also choose to always display the R code by going to  > *Preferences* > *Results*.



When you include the R code in the output, your peers can immediately see which options were used. They can also copy the code to run it themselves in the input window, or in JASP’s R console () , which can be enabled in the same way as other modules: click the  in the top right corner and select *R console* from the list that appears to add it to the top ribbon bar.

4.10 Saving and exporting

Most of you should be familiar with how to save files. Like most software, JASP has a save button and you can use  > *Save* or  > *Save as ...* or Ctrl + S (⌘ + S on Mac OS). If the file hasn't been saved previously then initiating a save will open the *Save As* dialog box. You use this dialog box as you would in any other software: type a name in the space next to where it says *Save As*. Once a file has been previously saved, it can be saved again (updated) by clicking on  > *Save* or pressing Ctrl + S (⌘ + S on Mac OS). Doing this will create a .jasp file, which contains the data, any output that was generated, and the settings that were used to create that output.

You can also choose to export only the data to a .csv file by going to  > *Export Data*. A .csv file is plain text, which means that the file type can be opened by virtually any data analysis software you can think of (including Excel, OpenOffice, Numbers, R, SAS, and Systat). If you want to share your JASP output with other people who don't have JASP installed, you can export the output into a format that they can open on their computer. By going to  > *Export Results*, you can generate an .html or .pdf file that can be opened in any web browser and is easily hosted online, for easy sharing of (annotated) results. Another way of sharing your results with anyone is by hosting the .jasp file on the OSF, where it can also be previewed directly in a web browser.


4.11 A few useful options

- JASP offers additional explanations whenever you see the  button. Clicking it will open a window that explains the settings that are listed in the current menu. This button can be found in all analyses and will tell you about all the input options and the statistical output that the analyses bombard you with, but also in the various other places, such as the compute columns or filtering menu.
- You can set default options for JASP by selecting  > *Preferences*. Here you can specify (among other things):
 - *Data*: You can tell JASP when to assume a variable is numeric rather than ordinal (from a certain number of unique values onward); you can also choose how missing data are displayed.¹²
 - *Results*: You can ask to display exact *p*-values, tweak how results are displayed, and set whether R code (Section 4.9.1) should be included in the output.
 - *Interface*: Here you can enable the almighty dark theme, and change the language and font.
- JASP features *keyboard shortcuts* for accessing items, which are revealed by pressing *Alt* (or ⌘ on Mac OS) on the keyboard. So, to access the hamburger menu, you would press *Alt*, which reveals that the menu is accessed by 'O', so then pressing 'O' will open the menu.

¹² JASP developer Joris uses the kiss emoji, , to make JASP a bit more cheerful.



JASP Tip 4.4 Funny numbers

If you enable *Use exponent notation* in  **Preferences** > **Results**, JASP will report numbers with the letter 'E' placed in the mix just to confuse you. For example, you might see a value such as $9.612\text{E}-02$. Many students find this notation confusing. It means 9.612×10^{-2} , which might be a more familiar notation or could be even more confusing. Think of E-02 as meaning 'move the decimal place 2 places to the left', so $9.612\text{E}-02$ becomes 0.09612. If the notation reads $9.612\text{E}-01$, then that would be 0.9612, and $9.612\text{E}-03$ would be 0.009612. Conversely, E+02 (notice the minus sign has changed) means 'move the decimal place 2 places to the right', so, $9.612\text{E}+02$ becomes 961.2. Although initially confusing, this notation becomes tremendously helpful when dealing with numbers that are either very large or very small.

4.12 Jane and Brian's story

Brian had been stung by Jane's comment. He was many things, but he didn't think he had his head up his own backside. He retreated from Jane to get on with his single life. He listened to music, met his friends, and played *Uncharted 4*. Truthfully, he mainly played *Uncharted 4*. The more he played the more he thought of Jane, and the more he thought of Jane, the more convinced he became that she'd be the sort of person who was into video games. When he next saw her he tried to start a conversation about games, but it went nowhere. She said computers were good only for analysing data. The seed was sewn, and Brian went about researching statistics packages. There were a lot of them. Too many. After hours on Google, he decided that one called JASP looked the easiest to learn. He would learn it, and it would give him something to talk about with Jane. Over the following week he read books, blogs, watched tutorials on YouTube, bugged his lecturers, and practised his new skills. He was ready to chew the statistical software fat with Jane.

He searched around campus for her: the library, numerous cafés, the quadrangle – she was nowhere. Finally, he found her in the obvious place: one of the computer rooms at the back of campus called the Euphoria cluster. Jane was studying numbers on the screen, but it didn't look like JASP. 'What the hell ...', Brian thought to himself as he sat next to her and asked ...

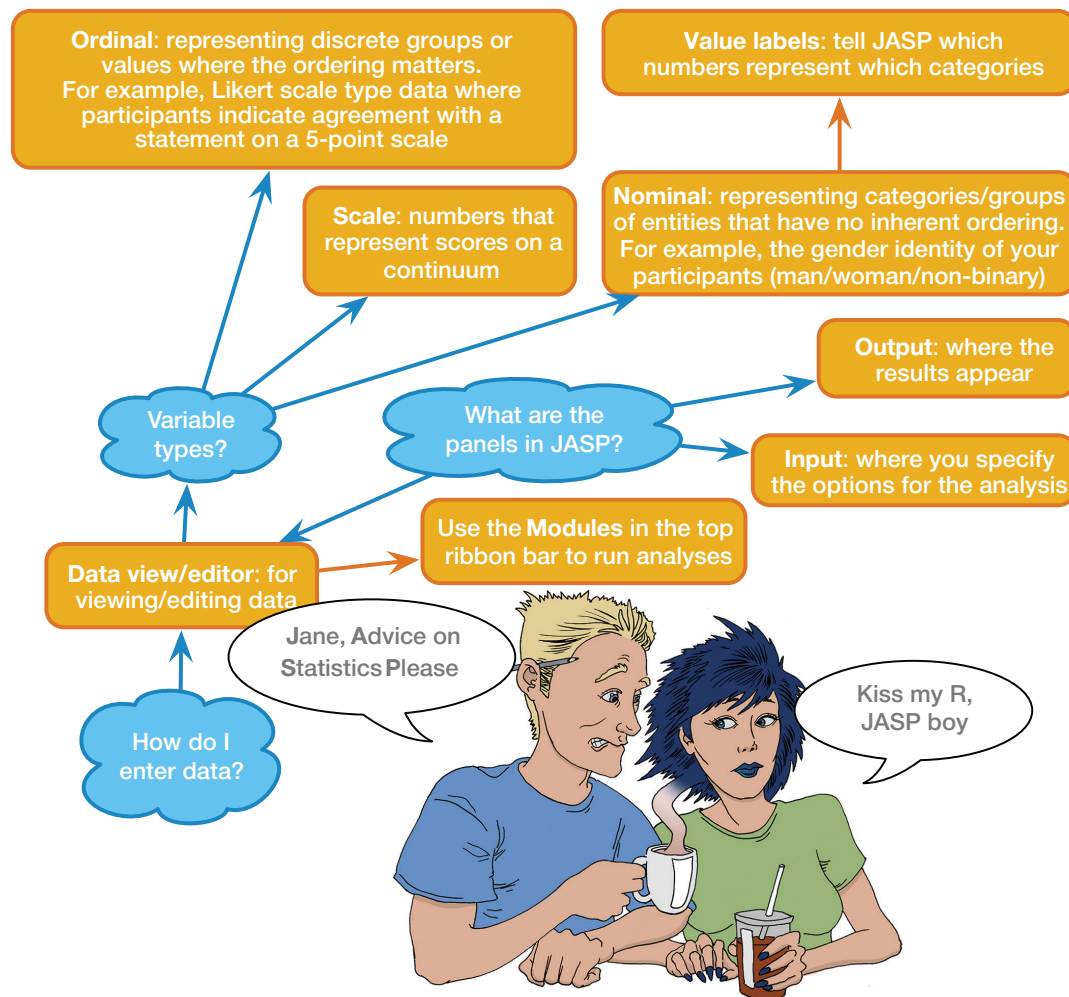


Figure 4.12 What Brian learnt from this chapter

4.13 What next? A

At the start of this chapter we discovered that I yearned for a world full of obedient robots to carry out my wishes. If these robots had existed their first instruction would have been to zap the school's daily milk delivery with their obliteratedronic laser fingers. Let me explain. During the time I was growing up in England, a politician had decided that all school children had to drink a small bottle of milk at the start of the day. The government supplied the milk, I think, for free, but most free things come at some kind of price. The price of free milk turned out to be a lifetime of unpleasant milk-related memories. The milk was usually delivered early in the morning and then left in the hottest place someone could find until we innocent children hopped and skipped into the playground oblivious to the gastric turmoil that awaited. We were greeted with one of these bottles of warm milk and a very small straw. We were then forced to drink it through grimacing faces. The straw was a blessing because it filtered out the lumps formed in the gently curdling milk. Politicians take note: if you want children to enjoy school, don't force-feed them warm, lumpy milk.



Smart Alex's Tasks

- Task 1:** Smart Alex's first task for this chapter is to save the data that you've entered in this chapter. Save it somewhere on the hard drive of your computer or the cloud. Give it a sensible title and save it somewhere easy to find (perhaps create a folder called 'My


- 

- Task 4:** Thinking back to Labcoat Leni's Real Research 4.1, Oxoby also measured the minimum acceptable offer; these MAOs (in dollars) are below (again, they are


THE JASP ENVIRONMENT

approximations based on the graphs in the paper). Enter these data into the JASP data editor and save this file as **acdc.jasp**. 


- Bon Scott group: 2, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5
- Brian Johnson group: 0, 1, 2, 2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 1

- **Task 5:** According to some highly unscientific research done by a UK department store chain and reported in *Marie Claire* magazine (<https://tinyurl.com/mcsgh>), shopping is good for you. They found that the average woman spends 150 minutes and walks 2.6 miles when she shops, burning off around 385 calories. In contrast, men spend only about 50 minutes shopping, covering 1.5 miles. This was based on strapping a pedometer on a mere 10 participants. Although I don't have the actual data, some simulated data based on these means are below. Enter these data into JASP and save them as **shopping.jasp**. 


Male		Female	
Distance	Time	Distance	Time
0.16	15	1.40	22
0.40	30	1.81	140
1.36	37	1.96	160
1.99	65	3.02	183
3.61	103	4.82	245

- **Task 6:** I wondered whether a fish or cat made a better pet. I found some people who had either fish or cats as pets and measured their life satisfaction and how much they like animals. Enter these data into JASP and save as **pets.jasp**. 

Fish		Cat	
Animal liking	Life satisfaction	Animal liking	Life satisfaction
69	47	16	52
25	6	65	66
31	47	39	65
29	33	35	61
12	13	19	60
49	56	53	68
25	42	27	37
35	51	44	72
51	42		
40	46		
23	27		
37	48		

- **Task 7:** One of my favourite activities, especially when trying to do brain-melting things like writing statistics books, is drinking tea. I am English, after all. Fortunately, tea improves your cognitive function – well, it does in older Chinese people at any rate (Feng et al., 2010). I may not be Chinese and I'm not *that* old, but I nevertheless enjoy the idea that tea might help me think. Here are some data based on Feng et al.'s study that measured the number of cups of tea drunk and cognitive functioning in 15 people. Enter these data into JASP and save the file as **tea_15.jasp**. 

Cups of tea	Cognitive functioning
2	60
4	47
3	31
4	62
2	44
3	41
5	49
5	56
2	45
5	56
1	57
3	40
3	54
4	34
1	46

- **Task 8:** Statistics and maths anxiety are common and affect people's performance on maths and stats assignments; women in particular can lack confidence in mathematics (Field, 2010, 2014). Zhang et al. (2013) did an intriguing study in which students completed a maths test and some put their own name on the test booklet, while others were given a booklet that already had either a male or female name on. Participants in the latter two conditions were told that they would use this other person's name for the purpose of the test. Women who completed the test using a different name performed better than those who completed the test using their own name. (There were no such effects for men.) The data below are a random subsample of Zhang et al.'s data. Enter them into JASP and save the file as **zhang_sample.jasp**. 

Male			Female		
Female fake name	Male fake name	Own name	Female fake name	Male fake name	Own name
33	69	75	53	31	70
22	60	33	47	63	57
46	82	83	87	34	33
53	78	42	41	40	83
14	38	10	62	22	86
27	63	44	67	17	65
64	46	27	57	60	64
62	27			47	37
75	61			57	80
50	29				

- **Task 9:** What is a nominal variable? 
- **Task 10:** What is the difference between wide and long format data? 

Solutions at www.discoverjasp.com

