

Rosetta Stats

The Rosetta Stats Authors

2023-03-05 20:03:10

Contents

	14
I Basics	17
1 Datasets	19
1.1 pp15: Party Panel 2015	19
2 Software Basics	23
2.1 File and variable name conventions	24
2.2 jamovi	24
2.3 R	28
2.4 SPSS	36
3 Loading data	39
3.1 jamovi	39
3.2 R	41
3.3 SPSS	44
4 Aggregating data: mean	47
4.1 Intro	47
4.2 Input: jamovi	48
4.3 Input: R	51
4.4 Input: SPSS	52
4.5 Output	53

5	Aggregating data: sum	55
5.1	Intro	55
5.2	Input: jamovi	55
5.3	Input: R	57
5.4	Input: SPSS	59
5.5	Output	60
6	Recoding data: invert	61
6.1	Intro	61
6.2	Input: jamovi	62
6.3	Input: R	69
6.4	Input: SPSS	71
6.5	Output	72
7	Standardizing	73
7.1	Intro	73
7.2	Input: jamovi	73
7.3	Input: R	73
7.4	Input: SPSS	74
7.5	Output	75
8	Changing Variable Type	77
8.1	Intro	77
8.2	Input: jamovi	78
8.3	Input: R	78
8.4	Input: SPSS	79
8.5	Output: jamovi	82
8.6	Output: R	82
8.7	Output: SPSS	82
8.8	Read more	82

<i>CONTENTS</i>	5
II Univariate analyses	83
9 Bar chart	85
9.1 Intro	85
9.2 Input: jamovi	85
9.3 Input: R	85
9.4 Input: SPSS	86
9.5 Output: jamovi	90
9.6 Output: R	90
9.7 Output: SPSS	90
9.8 Read more	90
10 Box plot	91
10.1 Intro	91
10.2 Input: jamovi	91
10.3 Input: R	93
10.4 Input: SPSS	93
10.5 Output: jamovi	95
10.6 Output: R	97
10.7 Output: SPSS	97
10.8 Read more	98
11 Descriptives	99
11.1 Intro	99
11.2 Input: jamovi	99
11.3 Input: R	99
11.4 Input: SPSS	102
11.5 Output: jamovi	104
11.6 Output: R	104
11.7 Output: SPSS	107
11.8 Read more	107

12	Frequencies	109
12.1	Intro	109
12.2	jamovi	109
12.3	R	110
12.4	SPSS	111
12.5	Read more	112
13	Histogram	113
13.1	Intro	113
13.2	Input: jamovi	113
13.3	Input: R	113
13.4	Input: SPSS GUI	114
13.5	Input: SPSS Syntax	114
13.6	Output: jamovi	117
13.7	Output: R	117
13.8	Output: SPSS	117
13.9	Read more	117
III	Psychometrics	119
14	Coefficient Alpha	121
14.1	Intro	121
14.2	Input: jamovi	121
14.3	Input: R	124
14.4	Input: SPSS	125
14.5	Output: jamovi	125
14.6	Output: R	125
14.7	Output: SPSS	127

15 Exploratory Factor Analysis	129
15.1 Intro	129
15.2 Input: jamovi	129
15.3 Input: R	131
15.4 Input: SPSS	133
15.5 Output: jamovi	135
15.6 Output: R	135
15.7 Output: SPSS	137
15.8 Read more	137
16 Omega	141
16.1 Intro	141
16.2 Input: jamovi	141
16.3 Input: R	144
16.4 Input: SPSS	146
16.5 Output: jamovi	146
16.6 Output: R	146
16.7 Output: SPSS	147
17 Reliability analysis	149
17.1 Intro	149
17.2 Input: jamovi	149
17.3 Input: R	152
17.4 Input: SPSS	153
17.5 Output: jamovi	154
17.6 Output: R	155
17.7 Output: SPSS	159
IV Bivariate analyses	163
18 Anova	165
18.1 jamovi	165

18.2 R	165
18.3 SPSS	165
19 Chi squared	167
19.1 jamovi	167
19.2 R	167
19.3 SPSS	167
20 Cohen's d	169
20.1 jamovi	169
20.2 R	169
20.3 SPSS	169
21 Correlation	171
21.1 Intro	171
21.2 Input: jamovi	171
21.3 Input: R	173
21.4 Input: SPSS	174
21.5 Output: jamovi	175
21.6 Output: R	175
21.7 Output: SPSS	176
21.8 Read more	176
22 Scattermatrix	179
22.1 Intro	179
22.2 Input: jamovi	179
22.3 Input: R	181
22.4 Input: SPSS	183
22.5 Output: jamovi	184
22.6 Output: R	184
22.7 Output: SPSS	187
22.8 Read more	187

<i>CONTENTS</i>	9
23 Scatterplot	189
23.1 jamovi	189
23.2 R	189
23.3 SPSS	189
24 Simple Regression Analysis	191
24.1 Intro	191
24.2 Input: jamovi	191
24.3 Input: R	192
24.4 Input: SPSS	196
24.5 Output: jamovi	197
24.6 Output: R	198
V Multivariate analyses	201
25 Multiple Regression Analysis	203
25.1 Intro	203
25.2 Input: jamovi	204
25.3 Input: R	207
25.4 Input: SPSS	208
25.5 Output: jamovi	209
25.6 Output: R	209
25.7 Output: SPSS	210
26 Logistic Regression Analysis	213
26.1 Intro	213
26.2 Input: jamovi	213
26.3 Input: R	214
26.4 Input: SPSS	217
26.5 Output: jamovi	218
26.6 Output: R	218
26.7 Output: SPSS	220

27 Moderated mediation	223
27.1 jamovi	223
27.2 R	226
27.3 SPSS	227
28 Multilevel analysis	229
28.1 jamovi	229
28.2 R	231
28.3 SPSS	233
VI Sample Size Calculations	235
29 Cohen's d: accuracy	237
29.1 Intro	237
29.2 Input: jamovi	240
29.3 Input: R	240
29.4 Input: SPSS	240
29.5 Output: jamovi	240
29.6 Output: R	240
29.7 Output: SPSS	240
VII Appendices and references	243
30 Authors	245
30.1 Roles	245
30.2 People	246
31 Contribution Guidelines	249
31.1 How can I contribute?	249
31.2 Some rules	250

<i>CONTENTS</i>	11
32 Style Guide	251
32.1 Filenames	251
32.2 Symbols and statistical characters	252
32.3 Chapter and section identifiers	252
32.4 Cross-references	252
32.5 Citations and references	253
32.6 Figures	253
32.7 Software-specific guidelines	255
33 Empty Chapter Template	257
33.1 Intro	257
33.2 Input: jamovi	257
33.3 Input: R	257
33.4 Input: SPSS	258
33.5 Output: jamovi	262
33.6 Output: R	262
33.7 Output: SPSS	262
33.8 Read more	262
34 Software and technologies	265
34.1 Bookdown	265
34.2 Zotero	265
34.3 Git	265
34.4 GitLab	265
34.5 Greenshot	266
34.6 Sizer	266
35 Code of Conduct	267
35.1 Our Pledge	267
35.2 Our Standards	267
35.3 Enforcement Responsibilities	268
35.4 Scope	268

35.5 Enforcement	268
35.6 Enforcement Guidelines	269
35.7 Attribution	270
36 References	271

<https://rosettastats.com>

Rosetta Stats

A statistics chrestomathy



The Rosetta Stats Authors



Rosetta Stats is a statistics chrestomathy. It illustrates how common analyses can be conducted in a variety of statistical software packages. Rosetta Stats is available at <https://rosettastats.com>.¹

Note that this is not a textbook about statistics. It doesn't explain which analysis is appropriate when, nor provide the relevant background or specify each analysis' assumptions. A number of excellent Open Access statistics textbooks exist already, such as Learning Statistics with R (<https://learningstatisticswithr.com>) or the jamovi version of this book (<https://www.learnstatswithjamovi.com>).

The authors of this book are less important than its content. Therefore, they are listed at the end, in Appendix 30.

¹Note that the website that was the original Rosetta Stats is still available at <https://psytext.github.io/rosettastats>.

Part I

Basics

Chapter 1

Datasets

The examples in this book use a number of freely available datasets. These datasets are described here.

All datasets are available in the `rosetta` R package as well as in the `rosetta` jamovi module, as well as directly from the associated GitLab repository at <https://gitlab.com/r-packages/rosetta> (in the `data` directory). The direct links to download each dataset will be included with each dataset’s description. Chapter 3 explains how to load datasets.

1.1 pp15: Party Panel 2015

The Party Panel is an annual semi-panel study that maps the determinants of a nightlife-related risk behavior. The aim is to enable selection of (sub-)determinants¹ to target in behavior change interventions. In 2015, the target behaviors were related to using a high dose of MDMA. MDMA, or 3,4-methylenedioxymethamphetamine, is the active ingredient in tablets sold as ecstasy. In the Netherlands, the dose of MDMA per tablet increased radically from the nineties, when around 90% of the ecstasy pills contained less than 100 mg of MDMA, to the 2010s, when less than 10% of the ecstasy pills contained less than 100 mg of MDMA. Given this rise, there were concerns that the typical ecstasy user had also changed. Prevention professionals were worried that where in the nineties and early 00s, ecstasy users mostly used to feel the euphoric, loving effects of ecstasy, the ecstasy users of the 2010s instead just wanted to get very intoxicated, preferring higher doses over the recommended dosing guidelines of 1-1.5mg of MDMA per kilogram of body weight.

¹In behavior change research, a determinant is a psychological construct that is theoretically postulated to cause behavior, such as “attitude” or “self-efficacy”. A sub-determinant is a determinant that is sufficiently specific to verbalize or visualize, such as “if I exercise every day, I will feel happier” or “if it is raining, I have a hard time pushing myself to go for a run”.

Based on these concerns, a determinant study was designed to measure the determinants of using a high dose of MDMA. The `pp15` dataset is a subset of this dataset. Most variables relate to the sub-determinants of the three Reasoned Action Approach variables: attitude, perceived norms, and perceived behavioral control. In addition, the direct measures of those three determinants are included, as well as intention to use a high dose, what participants considered their preferred dose, and what they considered a high dose. In addition, gender, age, work situation, education level, and bodyweight were also measured.

The dataset contains the following variables, with these labels (in English):

column	variable
gender	Gender
age	Age (in years)
hasJob	Employment status
jobHours	Hours of employment
currentEducation	If in education, study type
currentEducation_cat	If finished education, study type
prevEducation	Categorized current education type
prevEducation_cat	Categorized previous education type
xtcUseDoseHigh	MDMA dose (in mg) that is considered a high dose
xtcUseDosePref	Preferred MDMA dose (in mg)
xtcUsePillHigh	Number of pills that is considered a high dose
xtcUsePillPref	Preferred number of pills
weight_other	Body weight (in kg)
highDose_intention	RAA intention: planning to
highDose_attitude	RAA intention: want to
highDose_perceivedNorm	RAA intention: expect to
highDose_pbc	Intention frequency: number of times
highDose_IntentionRAA_intention	Intention frequency: multiplier
highDose_IntentionRAA_want	Attitude (direct): bad-good
highDose_IntentionRAA_expectation	Attitude (direct): unpleasant-pleasant
highDose_IntentionFrq_freq.0	Attitude (direct): dumb-smart
highDose_IntentionFrq_freq.1	Attitude (direct): unhealthy-healthy
highDose_AttGeneral_good	Attitude (direct): boring-exciting
highDose_AttGeneral_prettig	Injunctive norm (direct): people important to me
highDose_AttGeneral_slim	Injunctive norm (direct): people whose opinion I value
highDose_AttGeneral_gezond	Descriptive norm (direct): people I respect and admire
highDose_AttGeneral_spannend	Descriptive norm (direct): people like me
highDose_NormGeneral_in1	Perceived behavioral control (direct): would not/would be ab
highDose_NormGeneral_in2	Perceived behavioral control (direct): easy-hard
highDose_NormGeneral_dn1	Perceived behavioral control (direct): no/complete control
highDose_NormGeneral_dn2	Perceived behavioral control (direct): no/many external facto
highDose_PBCgeneral_ifwanted	Perceived behavioral control (direct): under others' control/u
highDose_PBCgeneral_easy	Perceived reasons for using a high dose of MDMA (open ques
highDose_PBCgeneral_control	Perceived reasons for *not* using a high dose of MDMA (ope
highDose_PBCgeneral_externalFactors	Expectation (attitudinal belief): shorter/longer trip
highDose_PBCgeneral_notOnlyMe	Expectation (attitudinal belief): less/more intense trip
highDose_OpenWhy	Expectation (attitudinal belief): less/more wasted
highDose_OpenWhyNot	Expectation (attitudinal belief): less/more energy
highDose_AttBeliefs_long	Expectation (attitudinal belief): less/more euphoric
highDose_AttBeliefs_intensity	Expectation (attitudinal belief): less/more self-insights
highDose_AttBeliefs_intoxicated	Expectation (attitudinal belief): less/more connected to other
highDose_AttBeliefs_energy	Expectation (attitudinal belief): easier/harder to make contac
highDose_AttBeliefs_euphoria	Expectation (attitudinal belief): feel less/more like sex
highDose_AttBeliefs_insight	Expectation (attitudinal belief): forget problems slower/faste
highDose_AttBeliefs_connection	Expectation (attitudinal belief): less/more socially isolated
highDose_AttBeliefs_contact	Expectation (attitudinal belief): can probe my boundaries wo
highDose_AttBeliefs_sex	Expectation (attitudinal belief): music sounds worse/better
highDose_AttBeliefs_coping	Expectation (attitudinal belief): less/more hallucinations
highDose_AttBeliefs_isolated	Expectation (attitudinal belief): time passes slower/faster
highDose_AttBeliefs_boundaries	Expectation (attitudinal belief): remember less/more
highDose_AttBeliefs_music	Expectation (attitudinal belief): less/more healthy
highDose_AttBeliefs_hallucinate	Expectation (attitudinal belief): experience worse/better
highDose_AttBeliefs_timeAwareness	Expectation (attitudinal belief): worry less/more physical sid

To directly download this dataset, use <https://gitlab.com/r-packages/rosetta/raw/prod/data/pp15.csv?inline=false>.

Chapter 2

Software Basics

This chapter introduces the software packages included in Rosetta Stats. In addition to providing a brief introduction, some required basics will be discussed. Some terms that you will encounter in this chapter are the following:

Console A console is an interface for sending commands and receiving responses. Until *graphical user interfaces* such as introduced by Windows and macOS became commonplace, they were pretty much how people interacted with personal computers. However, they remain ubiquitous. Consoles use plain text for communication. Users type commands, which are sent to an interpreter and processed further, and the results are again presented to the user using text printed to the console.

FLOSS FLOSS stands for Free/Libre Open Source Software. This is software that is available to everybody at no cost, that can freely be copied and distributed, and with publicly available source code, that is also available for others to view and edit. The benefits of FLOSS over proprietary software (i.e. software owned by a corporation or individual) include that errors can easily be spotted and corrected, that the software can be extended by anybody who wants to, and that the software can be downloaded and used by anybody, which removes funding as a barrier.

Graphical User Interface A graphical user interface (GUI) is an interface for sending commands and receiving responses. Unlike *consoles*, GUIs do not rely on plain text but instead use graphics. Most users will be familiar with GUIs: the operating systems of smartphones and personal computers all use GUIs.

GUI See Graphical User Interface.

Open Science Open Science is a collection of principles and practices that aim to enhance transparency, accessibility, and diversity both within science and of scientific products. These principles include full disclosure (i.e. making all materials, stimuli, datasets, analysis scripts, and output

public), preregistration, preprinting, choosing open standards and FLOSS over proprietary standards and software, and making articles, books and chapters open access. In short, Open Science is doing one's best to avoid making choices that introduce any kind of barrier that is not absolutely necessary, so that scientific products and participation in science are not gatekept.

2.1 File and variable name conventions

When working with data and analysis scripts, it is important to strive for platform- and software-independence. This results in a number of naming conventions that you should follow:

- Only use characters in filename and variable names that are platform- and software independent
- As much as possible, use the English language when naming files and variables (English is both the most widely understood language in the world and the de facto language of science)
- Make sure filenames and variable names are as much as possible self-explanatory: always prioritize clarity of brevity.

The characters that are safe to use in filenames are lower and upper case latin characters (**a-z** and **A-Z**), arabic digits (**0-9**), underscores (**_**), dashes (**-**), and periods (**.**; although periods are also used to distinguish file extensions from the filename itself and are therefore discouraged).

The characters that are safe to use in variable names are lower and upper case latin characters (**a-z** and **A-Z**), arabic digits (**0-9**), underscores (**_**), and periods (**.**).

This means that the set of characters that are always safe to use are (**a-z** and **A-Z**), arabic digits (**0-9**), and underscores (**_**).

If you want to indicate word boundaries, you can use underscores (**_**) or camel-Case, where the letter following a (omitted) space is capitalized.

A useful convention is to use camelCase to clearly distinguish words that are related, and reserve underscores for parts of variable names. For example: `preTest_selfEfficacy_item1`.

2.2 jamovi

The jamovi¹ project was founded to develop a free and open statistical platform which is intuitive to use, and can provide the latest developments in statistical

¹Yes, the preferred spelling is without a capital.

methodology. At the core of the jamovi philosophy, is that scientific software should be “community driven”, where anyone can develop and publish analyses, and make them available to a wide audience. It is available for Windows, macOS, Linux and ChromeOS from <https://jamovi.org>.

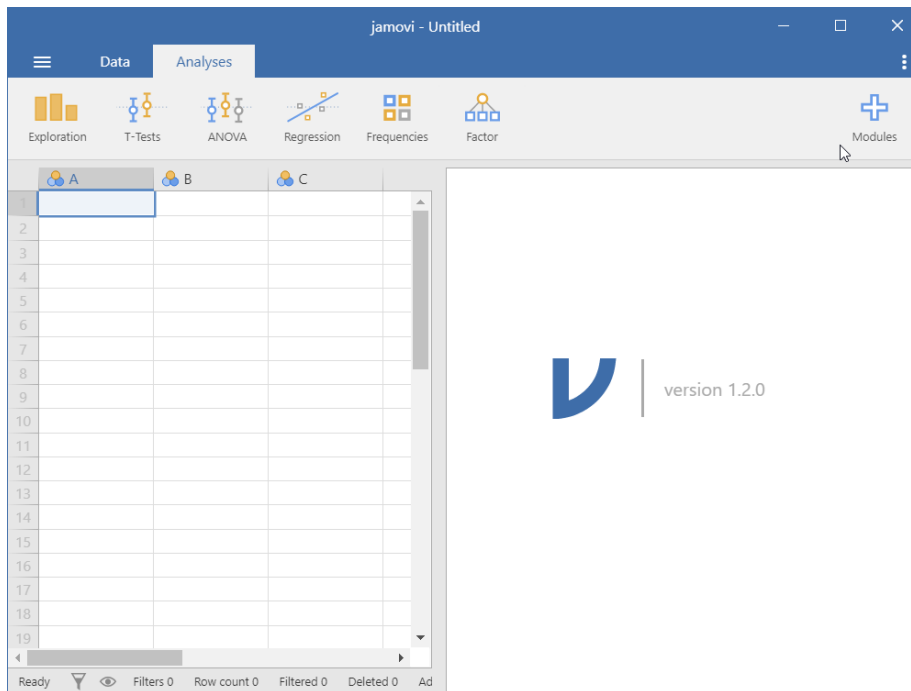


Figure 2.1: A fresh install of jamovi.

A nice feature of jamovi is that when you save a project, it stores the data, analyses, and output in the same file. This makes both collaboration and supervision much more straightforward. Another nice feature is that consistent with the jamovi philosophy, everybody can contribute modules. This means that users can install additional functionality from the jamovi library. It also means that the jamovi ecosystem will slowly keep growing over time.

If you just installed and opened jamovi, you will see something similar to what’s shown in Figure 2.1. This book is accompanied by a jamovi module that contains some of the analyses we refer to as well as the datasets we listed in Chapter 1.

2.2.1 Installing jamovi modules

To install a new module in jamovi, click the button with blue “plus” labelled “Modules” in the top-right corner. This will open the menu shown in Figure 2.2.

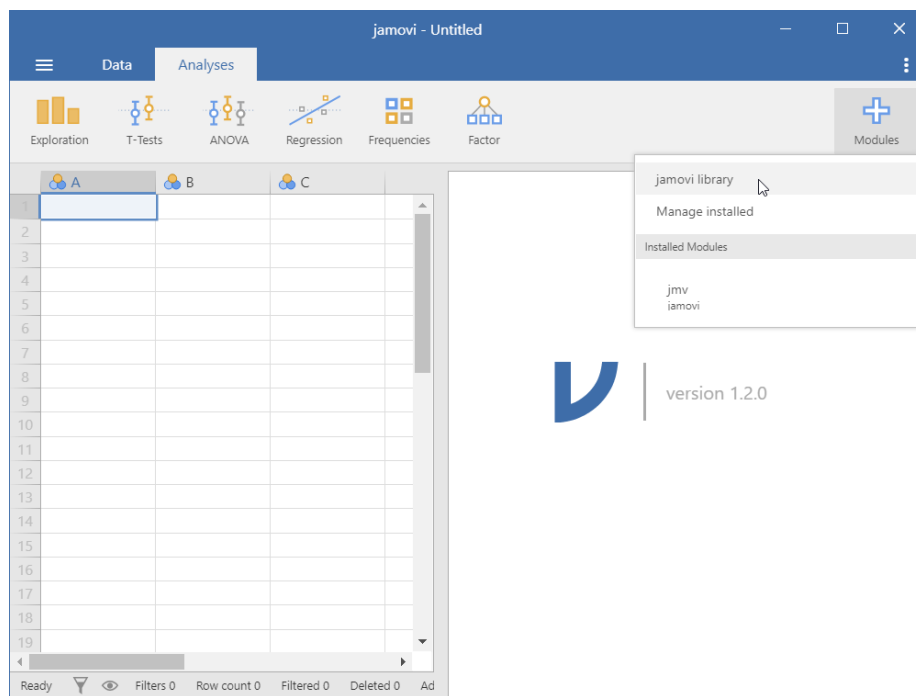


Figure 2.2: The modules menu in jamovi.

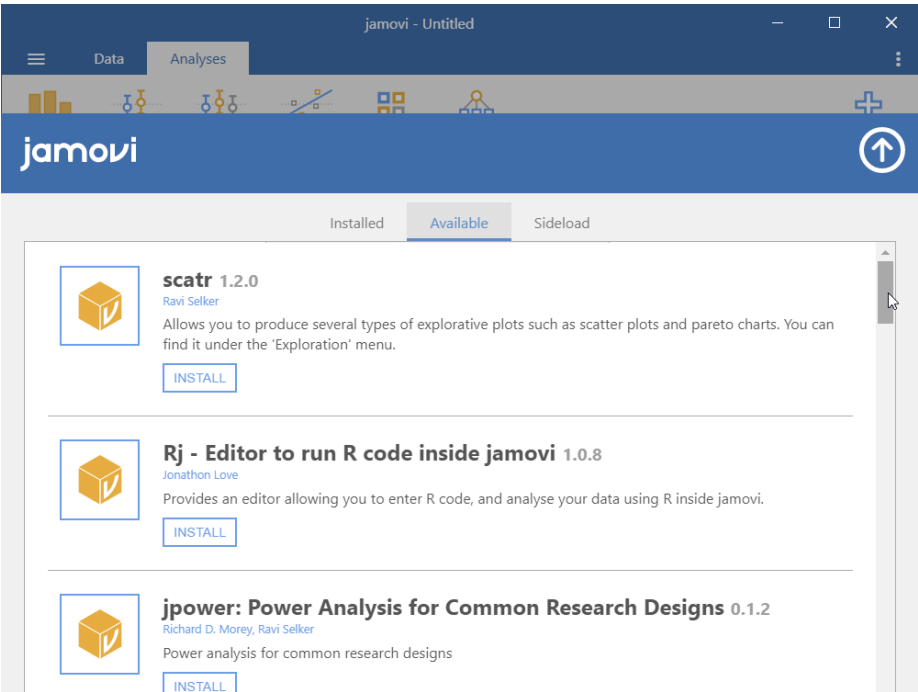


Figure 2.3: The jamovi library.

From this menu, select “jamovi library”, which will open the jamovi library with an overview of all available modules as shown in Figure 2.3. Scroll down to the “rosetta” module, which is called “Parallel Use of Statistical Packages in Teaching” and click the corresponding “Install” button to start the installation. Once the module is installed, it appears in the menu bar as shown in Figure 2.4.

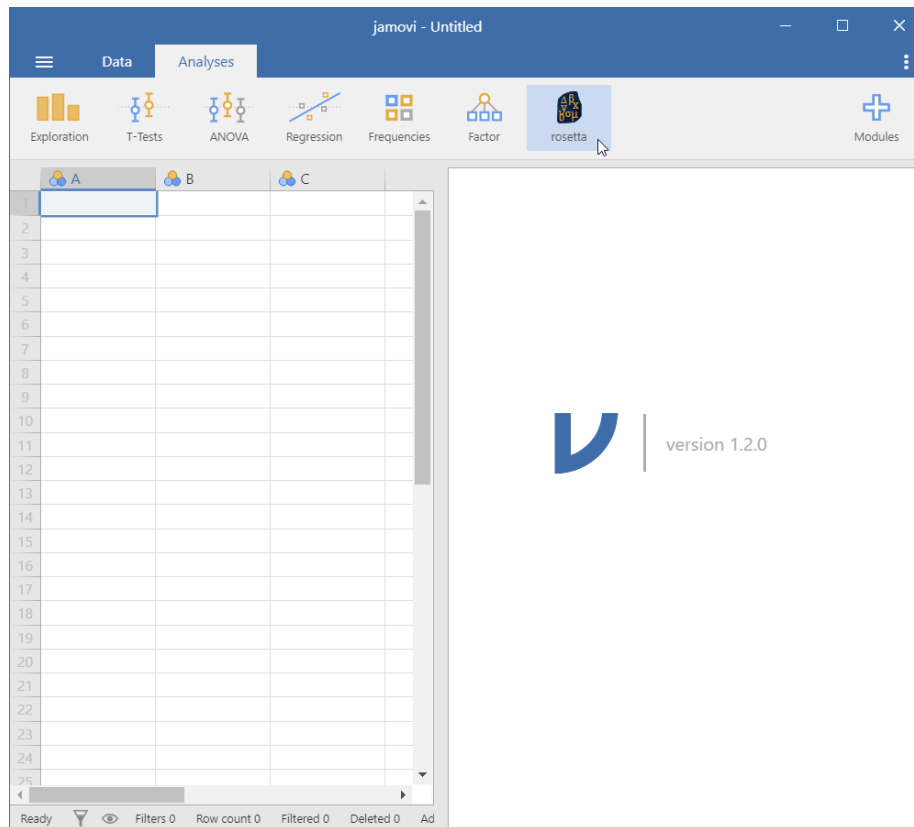


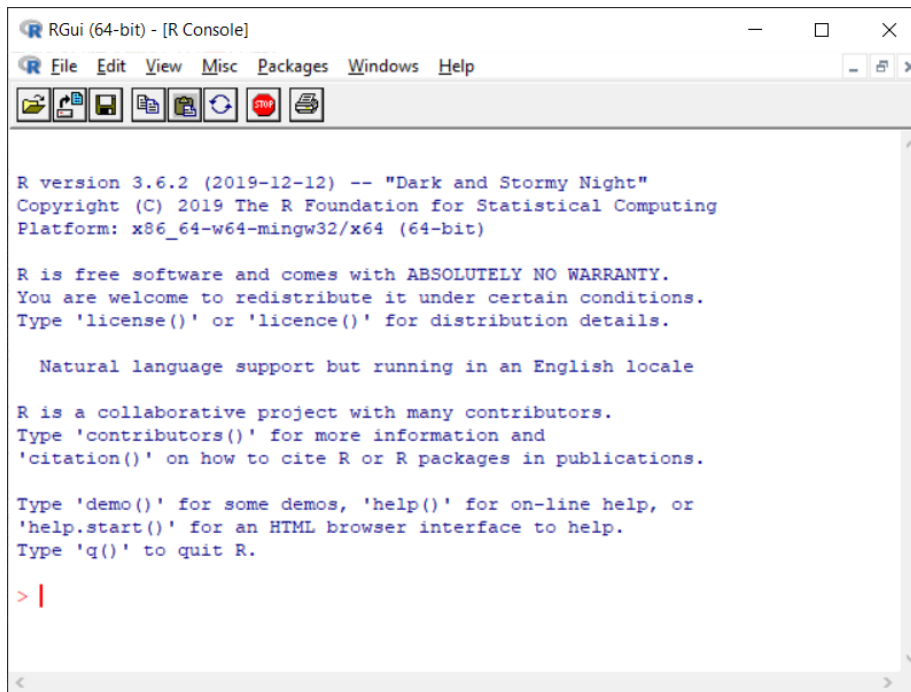
Figure 2.4: The jamovi library.

Installing the “rosetta” module will make the example datasets used in this book available. Chapter 3 explains how to load these datasets (or other datasets) into jamovi.

2.3 R

R is a very powerful and extensible FLOSS for statistical analyses. Because it is readily extensible through contributed packages, it has grown into a tool that can help you with pretty much anything: there are packages for multi-level analysis, structural equation modeling, but also for visualising geographi-

cal maps, rendering three-dimensional objects, doing text mining, working with big data, directly interacting with databases, advanced data visualisation, qualitative analysis, and the list goes on.



```
RGui (64-bit) - [R Console]
File Edit View Misc Packages Windows Help
R version 3.6.2 (2019-12-12) -- "Dark and Stormy Night"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

Figure 2.5: R.

R can be downloaded from <https://cloud.r-project.org/>. When you download R, you mostly get the statistical package itself. Most of this is the R language, which you can't see. Instead, as a user, you see a rudimentary console interface (see Figure 2.5). There are a lot of ways to make working with R more fun and efficient, such as the popular FLOSS package RStudio.

2.3.1 R Studio

A very popular environment is RStudio. RStudio is FLOSS and can be downloaded from <https://rstudio.com/products/rstudio/> (choose RStudio Desktop). See Figure 2.6 for an illustration of how RStudio looks.

RStudio is an interface that uses four tabbed panes to facilitate interactions with R. The top-left pane contains the analysis script, or multiple scripts, you are working on. The bottom-left pane contains the R console. The top-right pane contains an overview of all datasets and other objects you have loaded, as well as a history of the commands you provided to R. Finally, the bottom-right

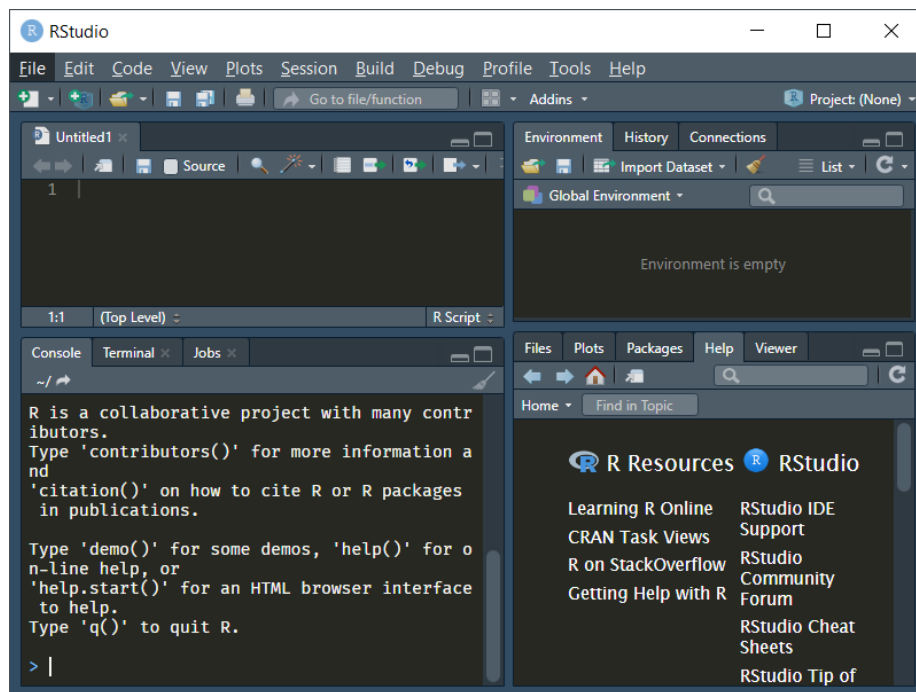


Figure 2.6: RStudio. Note that you can choose how RStudio looks by installing different themes: this theme can be downloaded from <https://gitlab.com/snippets/1846452/raw?inline=false>.

pane provides easy access to an overview of all files in your project, the plots you produced, the R packages you have installed and loaded, and the online manual of R and your packages.

2.3.2 Packages

R comes installed with many basic functions, commonly referred to as Base R. Researchers create organized collections of R functions, known as packages, to facilitate the use of various statistical methodologies and analyses. These packages are free and produce useful output for users. Anyone can create an R package. It is common to cite the packages you use in the references section of your research.

Packages typically live in two places: the Comprehensive R Archive Network, CRAN for short, which performs some quality control on packages. By default, R only looks in the CRAN repositories when you tell it to install a package. Many package developers also offer packages in Git repositories. These are often less polished than CRAN packages, but also more cutting edge. We will explain how to install “non-CRAN packages” below.

One package that you may want to install if you use this book is the `rosetta` package. We created this to facilitate parallel use of, and transition between, different statistical packages. It contains a number of functions designed to behave similarly to their counterparts in other software packages. This means we strive to use similar names for the functions and arguments, similar default settings, and similar output. We will use this package as an example in the following sections.

2.3.2.1 Installing CRAN packages

To install a package, you can use the following command:

```
install.packages('rosetta');
```

Replace “`rosetta`” with the name of the package you want to install. The name of the package is case sensitive and must go in single (') or double (") quotes. Once the package is installed, you will never need to install it again (although you may be asked to update it).²

If you use RStudio, you can also install packages using the Packages tab in the bottom-right pane of RStudio by clicking on “Packages” and “Install” and typing the name of the package you want to install (see Figure 2.7).

²Unless you update to a new version of R, e.g. from 4.0 to 4.1, or of course if you get a new computer.

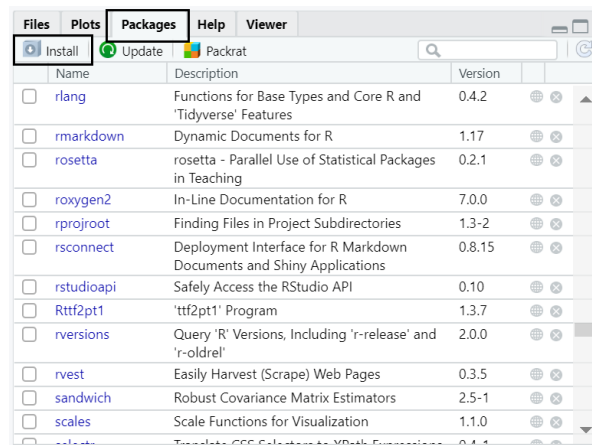


Figure 2.7: How to install package using the RStudio GUI.

2.3.2.2 Installing non-CRAN packages

If you want to install a package that is not on CRAN, or a newer version of a package than the version on CRAN, you can install packages directly from Git repositories. This requires the package `remotes`. Once installed, you can use the functions `remotes::install_gitlab()` and `remotes::install_github()` to install packages from repositories hosted by GitLab and GitHub, respectively.

To do this, first find the URL to the repository (this will usually have the form of “`https://gitlab.com/xxx/yyy`” or “`https://github.com/xxx/yyy`”), and remember the bit after the domain name (so “`xxx/yyy`”). The pass this as an argument to the `remotes::install_gitlab()` or `remotes::install_github()` function.

For example, to install the current development version of `rosetta` (which has more features but is usually less stable), use the following two commands (you can skip the first one if you already installed the `remotes` package):

```
install.packages('remotes');
remotes::install_gitlab('r-packages/rosetta');
```

2.3.2.3 Calling functions from packages

Once a package is installed, you can run all functions in the package by prepending the function name with “`package::`”. For example, if you installed the `rosetta` package, you can run all functions in the Rosetta Stats book that start with `rosetta::`. Note that if a function call starts with `somethingelse::`, you

need to install the package `somethingelse`. For example, advanced data visualisations require the `ggplot2` package, and functions from `ggplot` all start with `ggplot2::`. The `rosetta` package also uses the `ggplot2` package in the background, so you will not have to install that separately.

If you try to use a function but you forget to tell R from which package it should get the function, R will give you an error. For example, if we try to use the `varView()` function from the `rosetta` package to show the variable view for the built-in `Orange` dataframe like this:

```
varView(Orange);
```

R will throw the following error:

```
Error: object 'varView' not found
```

This is because we forgot to specify that we want to use the `varView` function from the `rosetta` package (and not from another package) by using the `rosetta::` prefix, so R can't find the object (the `varView()` function): it doesn't know where to look.³ If do we include the package name, R happily produces the variable view:

```
rosetta::varView(Orange);
```

```
## <p>
## Variable view for 'Orange':</p>
## <style>p,th,td{font-family:sans-serif}td{padding:3px;vertical-align:top;}tr:nth-child(even){ba
##
## <table>
## <thead>
## <tr>
## <th style="text-align:left;"> </th>
## <th style="text-align:right;"> index </th>
## <th style="text-align:left;"> values </th>
## <th style="text-align:left;"> level </th>
## <th style="text-align:left;"> valids </th>
## <th style="text-align:left;"> NAs </th>
## <th style="text-align:left;"> class </th>
## </tr>
```

³Formally, the `rosetta` package isn't on R's so-called "search path". Note that it's also possible to add a package to R's search path: see the next section.

```

## </thead>
## <tbody>
## <tr>
## <td style="text-align:left;"> Tree </td>
## <td style="text-align:right;"> 1 </td>
## <td style="text-align:left;"> 3 (1), 1 (2), 5 (3), 2 (4) & 4 (5) </td>
## <td style="text-align:left;"> ordinal </td>
## <td style="text-align:left;"> 35 </td>
## <td style="text-align:left;"> 0 </td>
## <td style="text-align:left;"> ordered & factor </td>
## </tr>
## <tr>
## <td style="text-align:left;"> age </td>
## <td style="text-align:right;"> 2 </td>
## <td style="text-align:left;"> 7 unique values ranging from 118 to 1582. </td>
## <td style="text-align:left;"> continuous </td>
## <td style="text-align:left;"> 35 </td>
## <td style="text-align:left;"> 0 </td>
## <td style="text-align:left;"> numeric </td>
## </tr>
## <tr>
## <td style="text-align:left;"> circumference </td>
## <td style="text-align:right;"> 3 </td>
## <td style="text-align:left;"> 30 unique values ranging from 30 to 214. </td>
## <td style="text-align:left;"> continuous </td>
## <td style="text-align:left;"> 35 </td>
## <td style="text-align:left;"> 0 </td>
## <td style="text-align:left;"> numeric </td>
## </tr>
## </tbody>
## </table>

```

2.3.2.4 Loading packages in memory

Sometimes you may want to tell R to always search for objects (such as functions) in a certain package. In that case, you can attach the package to R's "search path" for the duration of your session. You can conceptualize as "loading" the package. Although you only need to install a package once, if you want to load it, you will have to repeat that command every time you open a new R session.

You can load a package (i.e. attach it to R's search path) with this command:

```
library("rosetta")
```

If you use RStudio to work with R, you can load packages by opening the Packages tab in the bottom-right pane and clicking on the package you want to load (see Figure 2.8). If the package does not appear in your list of packages, it is because it has not been installed (see Section ??).

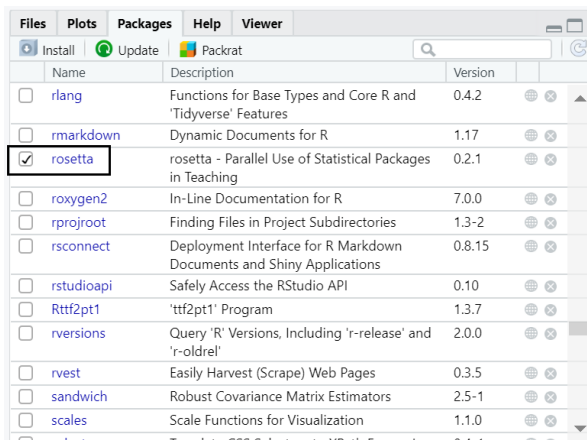


Figure 2.8: How to load a package using RStudio.

Note that it pays off to make a habit out of always explicitly specifying the package name using the `::` operator. Because R does not enforce unique function names across packages, many packages have functions with the same names (such as `recode()` in the `Hmisc`, `car`, and `rosetta` packages and `filter` in both base R and `dplyr`). If you load multiple packages, and use such a function, it is not always intuitive which version of a function you will “get”.

Because of this, debugging is much more efficient if you explicitly specify the packages along with the functions in every function call. Since when one works with R (or other statistical packages), most time isn’t spent writing code, but *debugging* code, this quickly pays off despite the slight inefficiency of having to always also type the package names.

A second reason to explicitly specify the packages when calling functions is that it makes R code more portable. When you rely on loaded packages, if you copy fragments of R code to another script, the code will throw errors until you remember to also load those functions at the top of your code. Identifying which packages you need to load to be able to run a fragment of code is impossible unless you happen to know by heart which functions come from which packages.

A third closely related reason is that explicitly specifying packages in function calls makes your code easier to read for others. They don’t have to scroll up and down to try and figure out which functions exactly you’re using.

Finally, a fourth reason applies to more advanced users: if you develop R packages, it is also best practise to always call functions using the `::` operator. Therefore, it's best to automate this from the get-go.

If you want to check for the existence of a package at the top of your code, there is a function called `checkPkgs()` in the `ufs` package.

2.4 SPSS

SPSS is a popular proprietary data analytic software owned by IBM. SPSS offers both a GUI and Syntax option for analysis, although people predominantly use SPSS for the simplicity of its GUI. SPSS comes with a set number of analytic options in its GUI and they cannot be modified by users.

As of January 2020, SPSS starts at \$99/month per user per month. Universities and businesses may have existing SPSS contracts in place that provide access to the software for their employees, faculty, staff, or students. Student discounts are also available for students who do not otherwise have access.

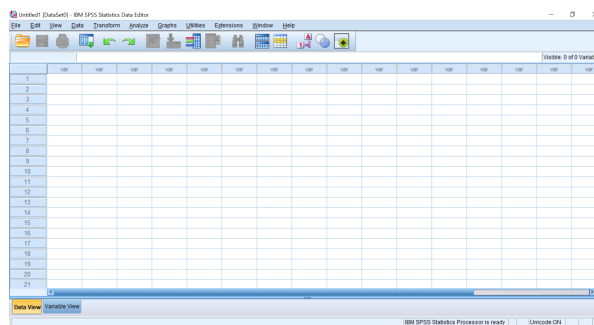


Figure 2.9: A fresh install of SPSS version 26

As a proprietary software, SPSS does not have any packages or modules that can be created or added by users. This makes it straight forward and simple, but much more limited than a FLOSS. This tutorial uses SPSS version 26.

2.4.1 The active dataset

In SPSS, each dataset will open in a separate window. Only one dataset can be interacted with at any one time. The user therefore manually has to switch between datasets.

When using the **GUI**, a dataset can be activated by clicking its window. As such, it is important to make sure you are operating in the dataset you intend to use.

To specify which dataset should be activated using **SPSS Syntax**, use the command `DATASET ACTIVATE` followed by the dataset name. For example, to activate a dataset named `dat`, one can use:

```
DATASET ACTIVATE dat.
```

Note that if you only have one dataset opened, that dataset is active, and so you will not have to activate it.

Chapter 3

Loading data

This chapter explains how to load data. For each statistical package, it has two sections. The first section is about the general case: loading data from a file. The second section is about loading one of the example datasets that accompany this book.

3.1 jamovi

The first steps to open a dataset are the same, regardless of whether you want to open a Rosetta Stats example dataset or a different file. First, open the main jamovi menu by clicking the hamburger menu in the top-left corner, just left of the text “Data” (see Figure 3.1).

This will open the menu shown in Figure 3.2. In that menu, click the text “Open”. This will expand two options in this menu: “This PC” and “Data Library”. Which you choose depends on whether you want to open a file from your PC, or whether you want to open a Rosetta Stats example dataset.

3.1.1 Loading data from a file

To open a file on your PC, click “This PC” in jamovi’s “Open” menu.

3.1.2 Loading a Rosetta Stats example dataset

To open a Rosetta Stats example dataset, click “Data Library” in jamovi’s “Open” menu (see Figure 3.2). By default, the jamovi data library already comes with a number of datasets, but installed modules can add more datasets.

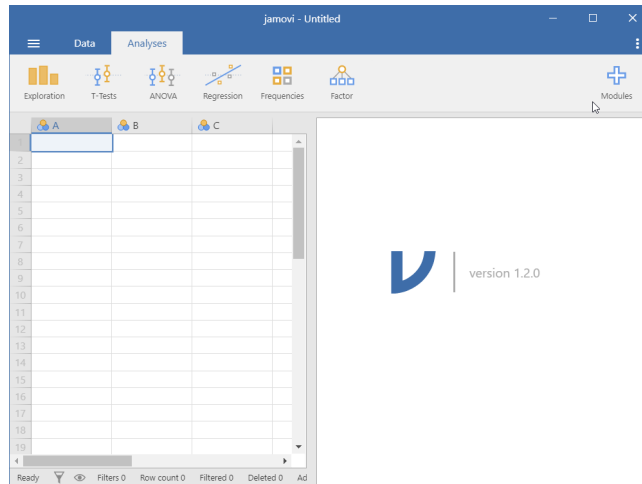


Figure 3.1: The jamovi GUI, with the hamburger menu in the top-left corner of the window.

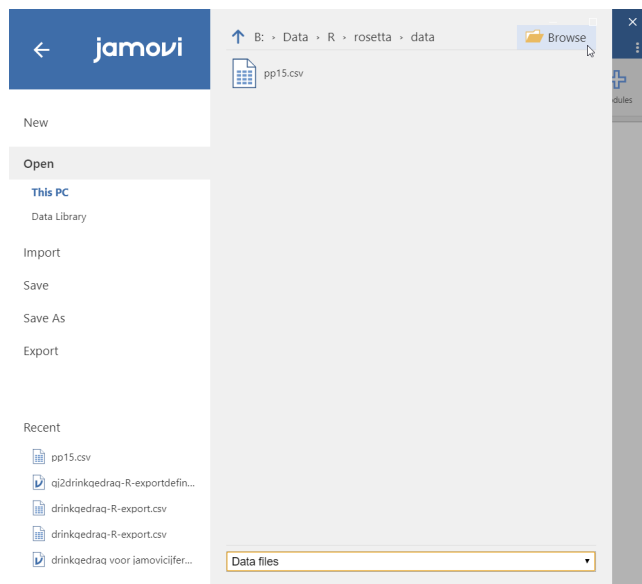


Figure 3.2: The opened hamburger menu in jamovi, with the "Open" menu section expanded.

If you installed the “rosetta” jamovi module, you should see the corresponding folder in jamovi’s data library (see Figure [/?\(fig:software-basics-jamovi-opened-data-library\)](#)). If you do not see this folder, install the “rosetta” module as explained in section 2.2.1.

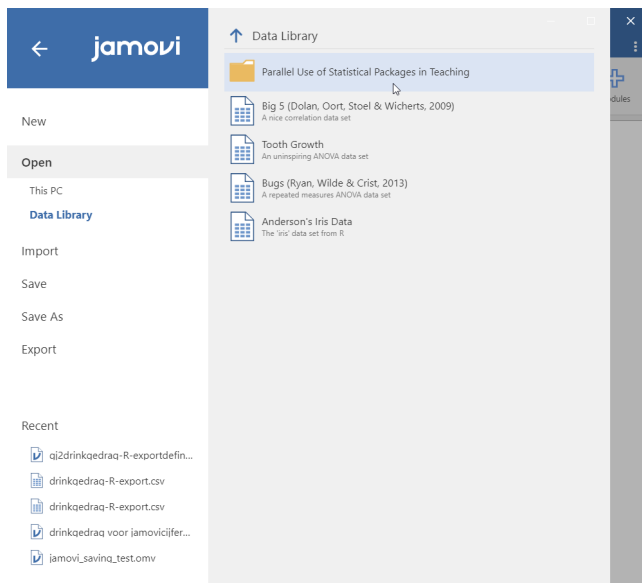


Figure 3.3: A fresh install of jamovi.

Open the Rosetta data library by clicking the corresponding folder. This will show the Rosetta Stats example datasets, as shown in Figure [??](#). Click the dataset you wish to open, and jamovi will open it.

3.2 R

3.2.1 Rosetta

The `rosetta` package has a function called `getData()` that will open a dialog where you can select your datafile. You have to specify under what name to store the data frame using an arrow:

```
dat <- getData();
```

There is a convenience function called `getDat()` that doesn’t require you to specify the name, but always stores it as `dat`:

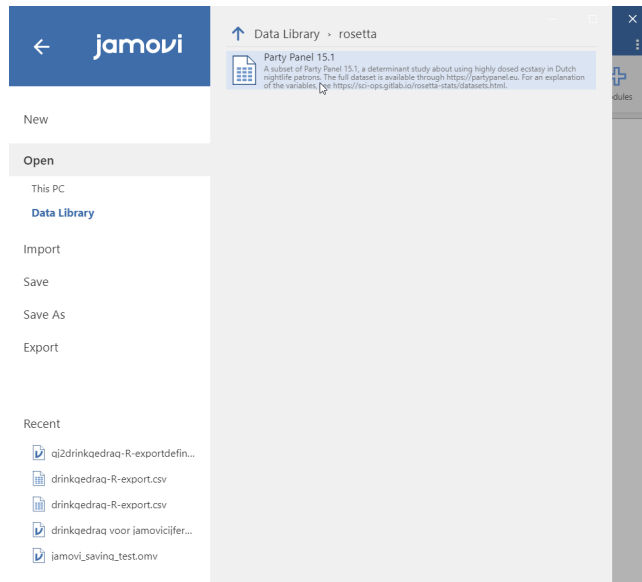


Figure 3.4: A fresh install of jamovi.

```
getDat();
```

3.2.2 RStudio

If you use RStudio, there is a very simple way to load data: you can use the “Import Dataset” menu from the top right menu in RStudio. (see Figure 3.5). From there, select the type of data you want to import and follow the point and click menus.

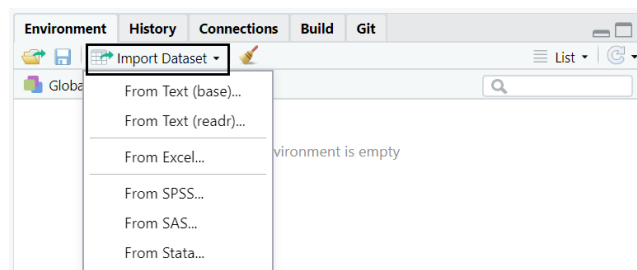


Figure 3.5: How to import data using GUI in R.

The dataset will import into R with the name of the saved file. This can be relatively long and so tedious to keep typing, so it is common to rename these

datasets to a shorter name, such as “dat”. To do this, you can copy the dataset to another name with the arrow.

```
dat <- longdatasetname;
```

To confirm the dataset was renamed correctly, you can view the variables stored in it:

```
names(dat);
```

You can also view the data in a spreadsheet format. If you use RStudio, it will appear as a new tab next to your R script. You can toggle back and forth between the two.

```
View(dat);
```

3.2.3 Loading a Rosetta Stats example dataset

To see a list of the Rosetta Stats example datasets, use the command `data`, specifying the package name as named argument `package`:

```
data(package="rosetta");
```

To load one of the datasets, assign it to a name of your choice, specifying both the package name and the dataset name, separated by two colons. In the examples in this book, we will always use `dat` as the name. For example, to load the dataset `pp15`, use:

```
dat <- rosetta::pp15;
```

Note that to use variables from the dataset, you don't have to store it locally; you can directly specify variables in analyses using `rosetta::pp15` as dataframe name (a “dataframe” is the name for a loaded dataset in R).

3.3 SPSS

3.3.1 Loading an SPSS dataset

To open an SPSS dataset in SPSS, select “File”, “Open”, “Data” from the menu bar (see Figure 3.6). Navigate to the dataset you wish to open and select it.

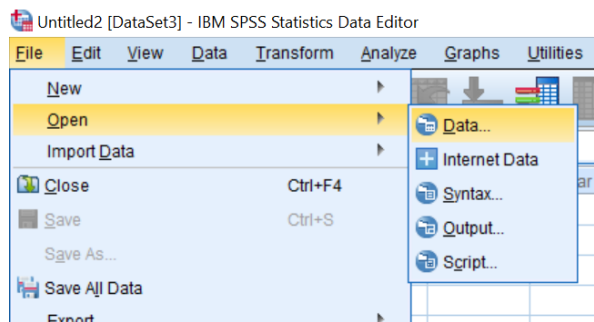


Figure 3.6: How to open a dataset in SPSS.

3.3.2 Importing a non-SPSS dataset

To import a different type of dataset into SPSS, select “File”, “Import Data” from the menu bar and select the data type (see Figure 3.7). Navigate through the menu options for the data type to open the dataset.

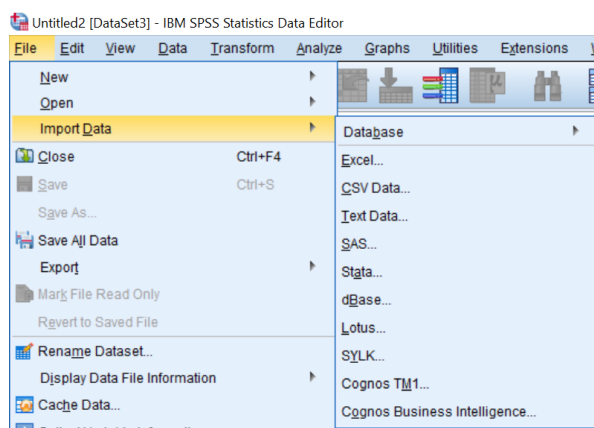


Figure 3.7: How to import a dataset in SPSS.

3.3.3 Loading a Rosetta Stats example dataset

The pp15 dataset is saved as a text file. To import a text file into SPSS, select “File”, “Import Data”, “Text Data” (see Figure 3.7). Navigate to the pp15 dataset, select it, and click “Open” (see Figure 3.8).

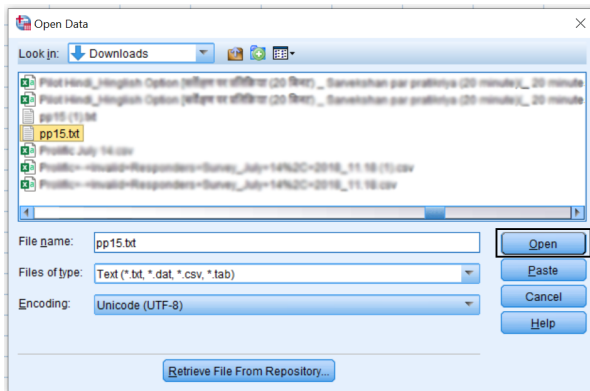


Figure 3.8: Importing the Rosetta pp15 dataset in SPSS.

There are 6 steps to click through to load the dataset. We do not have a predefined format for our text file, so we will press “Next” on Step 1 without modifying anything. At Step 2, there are several things to check. Look at the snapshot of the data at the bottom of (Figure 3.9) in the black box and note that the data are delimited by a semi-colon. Therefore, we will select “Delimited” under “How are your variables arranged?” (1). Also note in the data snapshot that there are variable names at the top of the file. Therefore, will select “Yes” under “Are variable names included at the top of your file?” (2). Finally, note that this dataset comes from the Netherlands, where it is common to use a comma as a decimal. Thus, we will select “Comma” under “What is the decimal symbol?” (3).

We do not need to change any of the defaults in Step 3 and can just press “Next”. We saw in the data snapshot at the bottom of Figure 3.9 that the data were delimited by a semicolon, so we will make sure to *only* select “Semicolon” under “Which delimiters appear between variables?” in Step 4. There is nothing to modify in Step 5 or 6, so we can simply press “Next” and “Finish”. The pp15 dataset will now load in SPSS.

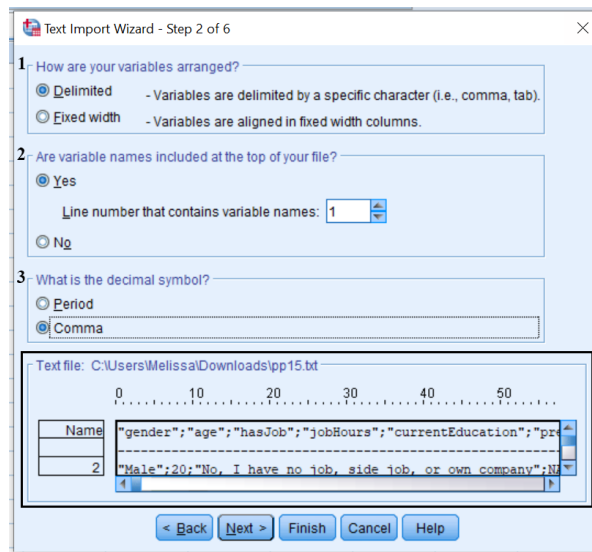


Figure 3.9: Step 2 of the Data Import Wizard.

Chapter 4

Aggregating data: mean

4.1 Intro

A common task is aggregating multiple variables (columns in a dataset) into one new variable (column). For example, you may want to compute the average score on the items of a questionnaire.

Note that when creating new variable names, it is important to follow the convention for variable names (see section [??software-basics-file-and-variable-name-conventions](#))).

4.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

4.1.2 Variable(s)

From this dataset, this example uses variables `highDose_AttGeneral_good`, `highDose_AttGeneral_prettig`, `highDose_AttGeneral_slim`, `highDose_AttGeneral_gezond` & `highDose_AttGeneral_spannend`.

We will aggregate these into the variable `highDose_attitude` (note that this variable already exists in the dataset, and that existing variable is also the mean of those five variables).

4.2 Input: jamovi

In the “Data” tab, click the “Compute” button as shown in Figure 4.1.

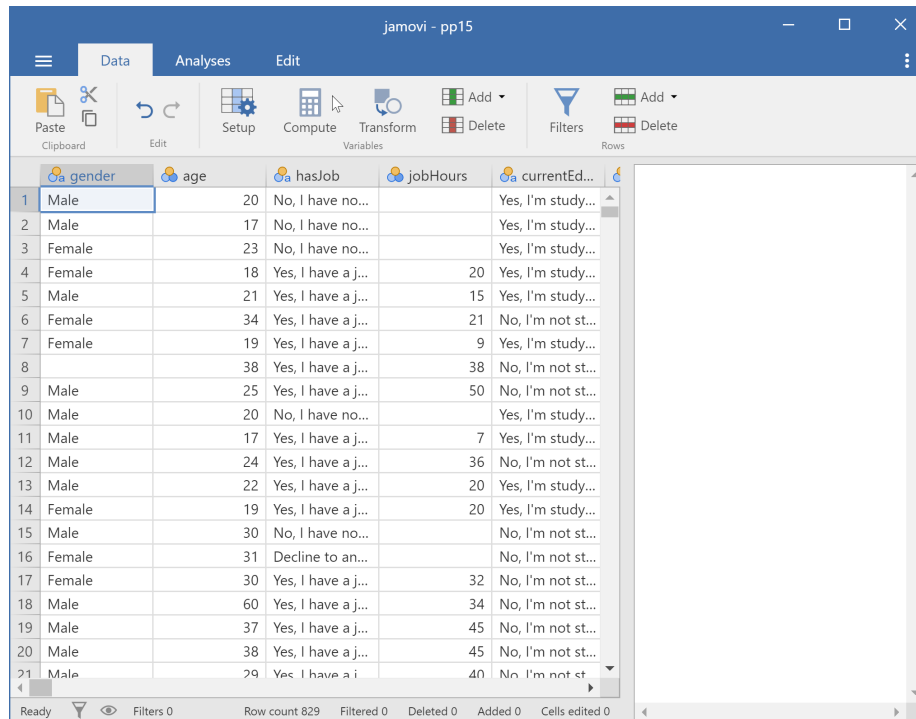


Figure 4.1: Aggregating in jamovi: opening Compute menu

Type in the new variable name in the text field at the top, labelled “COMPUTED VARIABLE”. Then click the function button, marked f_x , select the MEAN function from the box labelled “Functions”, and double click all variables for which you want the mean in the box labelled “Variables”, while typing a comma in between each variable name as shown in Figure 4.1.

Alternatively, you can type the function name and list of variables directly without using the function (f_x) dialog as shown in Figure 4.2.

If you want to allow missing values, you can specify the `ignore_missing=1` argument. In that case, you would type:

```
MEAN(highDose_AttGeneral_good, highDose_AttGeneral_prettig,
      highDose_AttGeneral_slim, highDose_AttGeneral_gezond,
      highDose_AttGeneral_spannend, ignore_missing=1)
```

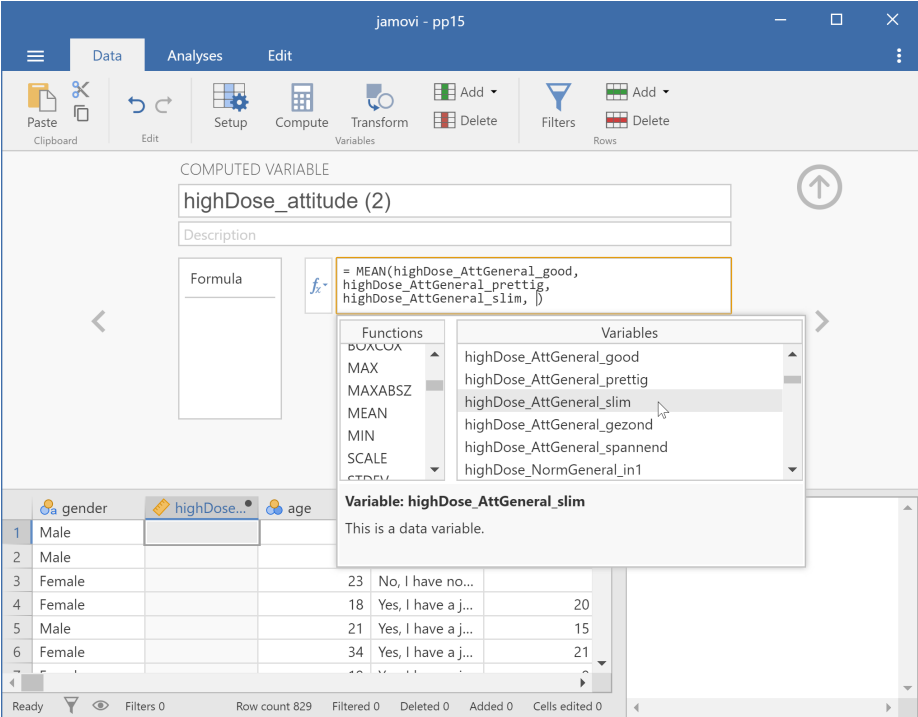



Figure 4.2: Aggregating in jamovi: using the function menu to specify a computation

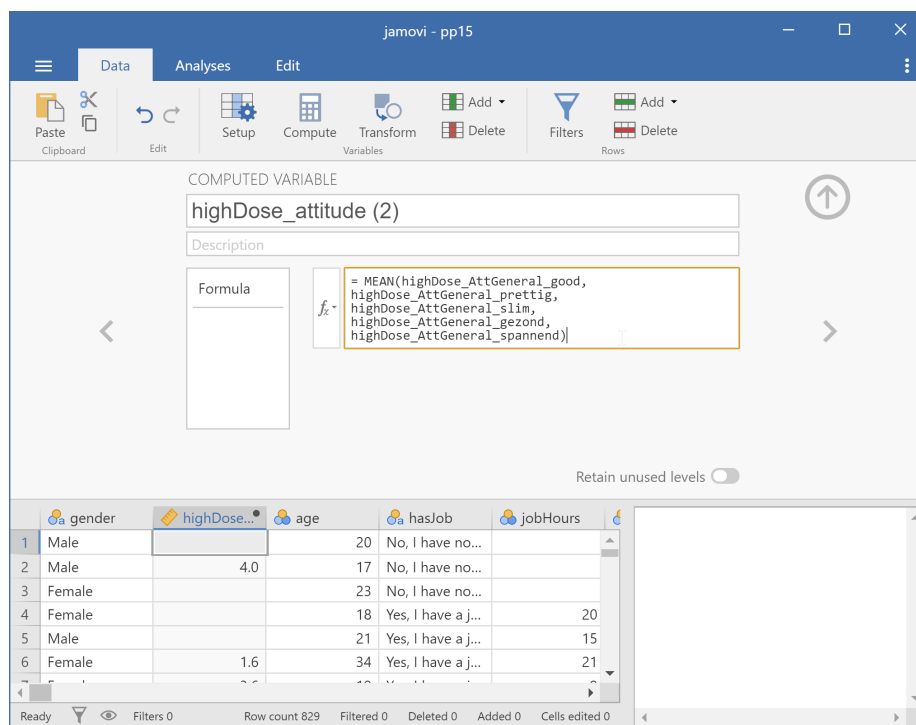


Figure 4.3: Aggregating in jamovi: directly typing in a computation

It is as yet not possible to indicate the number of valid values that is required; either no missings are allowed at all, or any number of missing values is accepted.

4.3 Input: R

In R, there are roughly three approaches. Many analyses can be done with base R without installing additional packages. The `rosetta` package accompanies this book and aims to provide output similar to jamovi and SPSS with simple commands. Finally, the tidyverse is a popular collection of packages that try to work together consistently but implement a different underlying logic than base R (and so, the `rosetta` package).

4.3.1 R: base R

```
dat$highdose_attitude <-  
  rowMeans(  
    dat[,  
      c(  
        'highDose_AttGeneral_good',  
        'highDose_AttGeneral_prettig',  
        'highDose_AttGeneral_slim',  
        'highDose_AttGeneral_gezond',  
        'highDose_AttGeneral_spannend'  
      )  
    ]  
  );
```

4.3.2 R: Rosetta

```
dat$highdose_attitude <-  
  rosetta::means(  
    data = dat,  
    'highDose_AttGeneral_good',  
    'highDose_AttGeneral_prettig',  
    'highDose_AttGeneral_slim',  
    'highDose_AttGeneral_gezond',  
    'highDose_AttGeneral_spannend'  
  );
```

To indicate that a certain number of values must be valid (i.e. “non-missing”), the argument `requiredValidValues` can be passed. For example, to require four valid values (instead of requiring only one valid value, the default), use:

```
dat$highdose_attitude <-  
  rosetta::means(  
    data = dat,  
    'highDose_AttGeneral_good',  
    'highDose_AttGeneral_prettig',  
    'highDose_AttGeneral_slim',  
    'highDose_AttGeneral_gezond',  
    'highDose_AttGeneral_spannend',  
    requiredValidValues = 4  
  );
```

4.4 Input: SPSS

For SPSS, there are two approaches: using the Graphical User Interface (GUI) or specify an analysis script, which in SPSS are called “syntax”.

4.4.1 SPSS: GUI

First activate the `dat` dataset (see 2.4.1).

4.4.2 SPSS: Syntax

```
COMPUTE highdose_attitude =  
  MEAN(  
    highDose_AttGeneral_good,  
    highDose_AttGeneral_prettig,  
    highDose_AttGeneral_slim,  
    highDose_AttGeneral_gezond,  
    highDose_AttGeneral_spannend  
  ).
```

To indicate that a certain number of values must be valid (i.e. “non-missing”), the command `MEAN` can be appended with a period and the number of required valid values. For example, to required four valid values, use:

```
COMPUTE highdose_attitude =
```



Figure 4.4: A screenshot placeholder

```
MEAN.4(  
  highDose_AttGeneral_good,  
  highDose_AttGeneral_prettig,  
  highDose_AttGeneral_slim,  
  highDose_AttGeneral_gezond,  
  highDose_AttGeneral_spannend  
).
```

4.5 Output

Aggregating variables is not an analysis, and as such, does not produce output. You can inspect the newly created variable to ensure it has been created properly.

Chapter 5

Aggregating data: sum

5.1 Intro

A common task is aggregating multiple variables (columns in a dataset) into one new variable (column). For example, you may want to compute the sum of the items of a questionnaire.

Note that when creating new variable names, it is important to follow the convention for variable names (see section ??software-basics-file-and-variable-name-conventions)).

5.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

5.1.2 Variable(s)

From this dataset, this example uses variables `highDose_AttGeneral_good`, `highDose_AttGeneral_prettig`, `highDose_AttGeneral_slim`, `highDose_AttGeneral_gezond` & `highDose_AttGeneral_spannend`.

We will aggregate these into the variable `highDose_attitude_sum`.

5.2 Input: jamovi

In the “Data” tab, click the “Compute” button as shown in Figure 5.1.

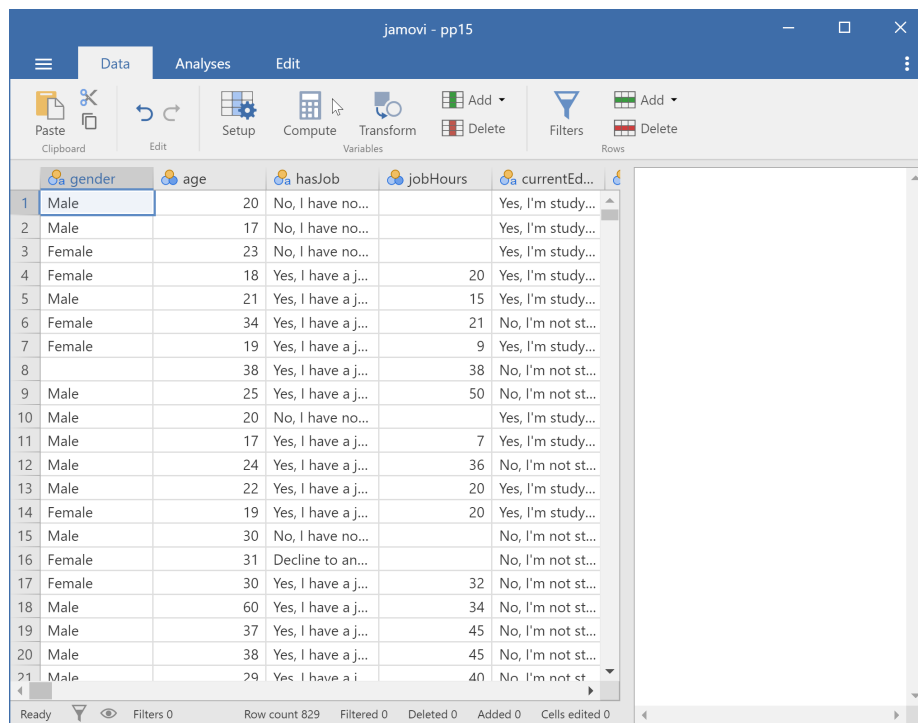


Figure 5.1: Aggregating in jamovi: opening Compute menu

Type in the new variable name in the text field at the top, labelled “COMPUTED VARIABLE”. Then click the function button, marked f_x , select the SUM function from the box labelled “Functions”, and double click all variables for which you want the sum in the box labelled “Variables”, while typing a comma in between each variable name as shown in Figure 5.2.

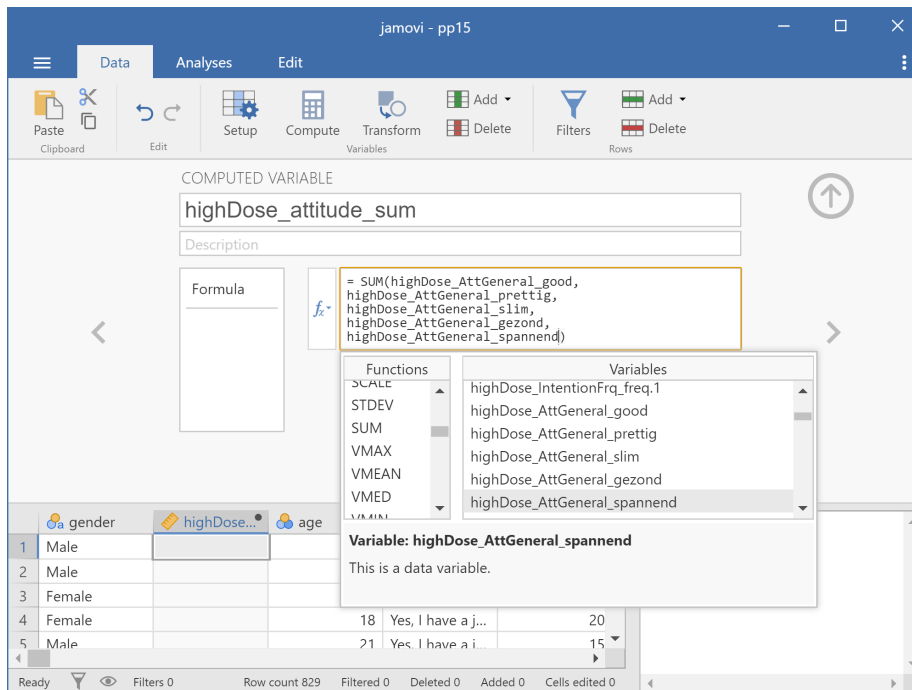


Figure 5.2: Aggregating in jamovi: using the function menu to specify a computation

Alternatively, you can type the function name and list of variables directly without using the function (f_x) dialog as shown in Figure 5.3.

5.3 Input: R

In R, there are roughly three approaches. Many analyses can be done with base R without installing additional packages. The `rosetta` package accompanies this book and aims to provide output similar to jamovi and SPSS with simple commands. Finally, the tidyverse is a popular collection of packages that try to work together consistently but implement a different underlying logic than base R (and so, the `rosetta` package).

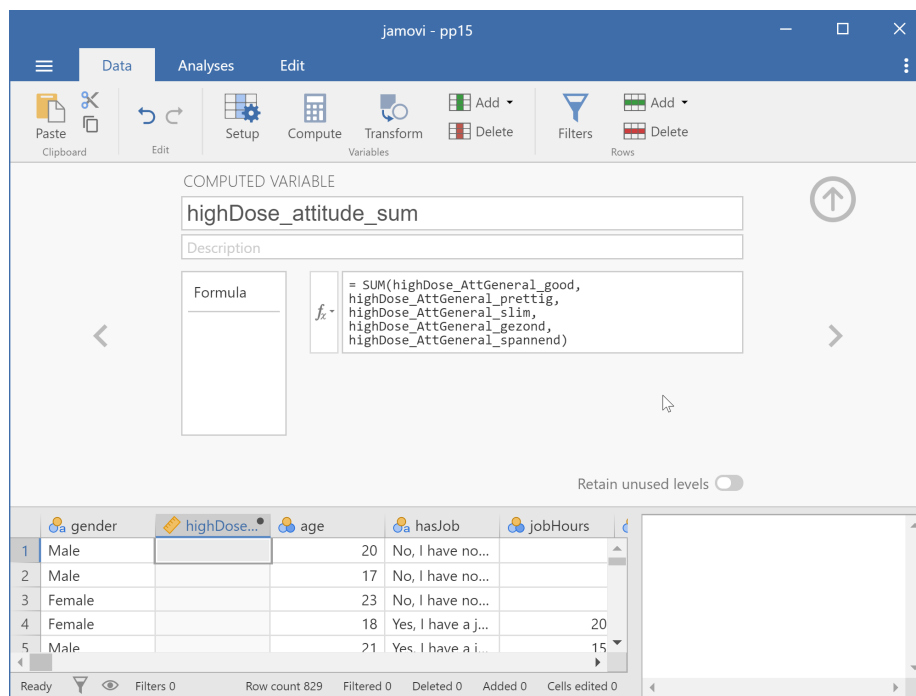


Figure 5.3: Aggregating in jamovi: directly typing in a computation

5.3.1 R: base R

```
dat$highdose_attitude <-  
  colSums(  
    dat[  
      ,  
      c(  
        'highDose_AttGeneral_good',  
        'highDose_AttGeneral_prettig',  
        'highDose_AttGeneral_slim',  
        'highDose_AttGeneral_gezond',  
        'highDose_AttGeneral_spannend'  
      )  
    ]  
  );
```

5.3.2 R: rosetta

```
dat$highdose_attitude <-  
  rosetta::sums(  
    data = dat,  
    'highDose_AttGeneral_good',  
    'highDose_AttGeneral_prettig',  
    'highDose_AttGeneral_slim',  
    'highDose_AttGeneral_gezond',  
    'highDose_AttGeneral_spannend'  
  );
```

5.4 Input: SPSS

For SPSS, there are two approaches: using the Graphical User Interface (GUI) or specify an analysis script, which in SPSS are called “syntax”.

5.4.1 SPSS: GUI

First activate the `dat` dataset (see 2.4.1).



Figure 5.4: A screenshot placeholder

5.4.2 SPSS: Syntax

```
COMPUTE highdose_attitude =  
  SUM(  
    highDose_AttGeneral_good,  
    highDose_AttGeneral_prettig,  
    highDose_AttGeneral_slim,  
    highDose_AttGeneral_gezond,  
    highDose_AttGeneral_spannend  
  ).
```

5.5 Output

Aggregating variables is not an analysis, and as such, does not produce output. You can inspect the newly created variable to ensure it has been created properly.

Chapter 6

Recoding data: invert

6.1 Intro

A common recoding task is inverting an item. For example, if scores used to be on a 1-5 scale, after recoding, 1 has become 5, 2 became 4, 4 became 2, and 5 became 1.

There are generally two ways of achieving this:

- subtracting all values from the maximum value minus one (so to recode values on a 1-5 scale, subtract all values from 6 to obtain the recoded values)
- specifying, for each value, the new value you want

In this chapter, both methods will be listed where appropriate. Note that when creating new variable names, it is important to follow the convention for variable names (see section [??software-basics-file-and-variable-name-conventions](#))).

6.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

6.1.2 Variable(s)

From this dataset, this example uses variable `highDose_IntentionRAA_intention`, a variable with these data:

1	2	3	4	5	6	7
64	37	31	53	44	47	32

We want to invert these, such that after recoding, in the new variable 64 participants have value 7, 37 participants have value 6, 31 participants have value 5, 53 participants have value 4, 44 participants have value 3, 47 participants have value 2, and 32 participants have value 1.

6.2 Input: jamovi

6.2.1 Subtraction

To subtract all values from the maximum value (in this case, 8), use the following steps.

In the “Data” tab, click the “Compute” button as shown in Figure 6.1.

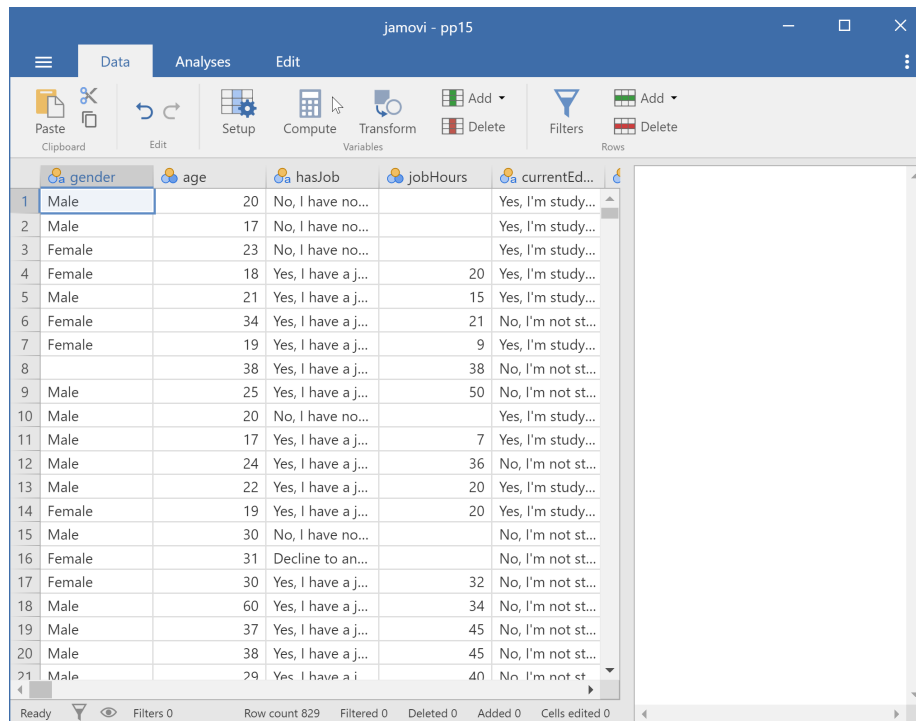


Figure 6.1: Recoding in jamovi: opening Compute menu

The, type in the new variable name in the top-most text field labelled “COMPUTED VARIABLE”, and in the bottom-right text field, specify the value from which you want to subtract all variable values (in this case,

8), a minus (the dash), and the source variable name (in this example, `highDose_IntentionRAA_intention`), as shown in Figure 6.2.

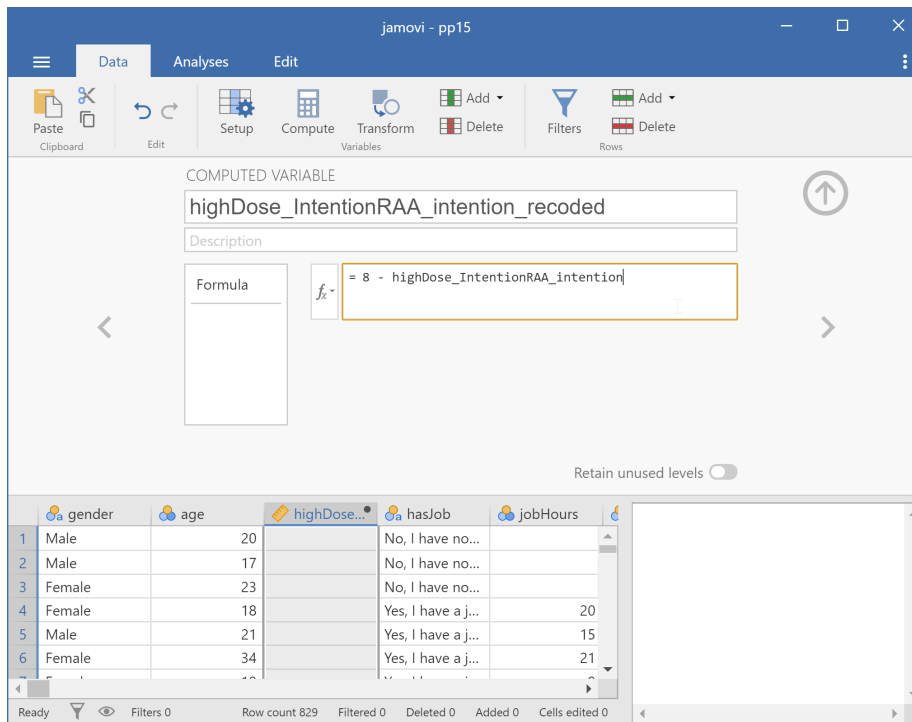


Figure 6.2: Recoding in jamovi: specifying Compute function

6.2.2 Manual specification

To manually specify each value you want, use the following steps.

In the “Data” tab, click the “Transform” button as shown in Figure 6.3.

In the dropdown labelled “Source variable”, select the variable to transform as shown in Figure 6.4.

In the text field labelled “TRANSFORMED VARIABLE” at the top, specify the new variable name as shown in Figure 6.5.

In the dropdown labelled “using transform”, select “Create New Transform...” as shown in Figure 6.6.

In the top-most text field, just below the text “TRANSFORM”, you can name this transformation in case you plan to do you can easily re-use it for other variables. Then, click the “Add recode condition” to add a new recoding condition as shown in Figure 6.7.

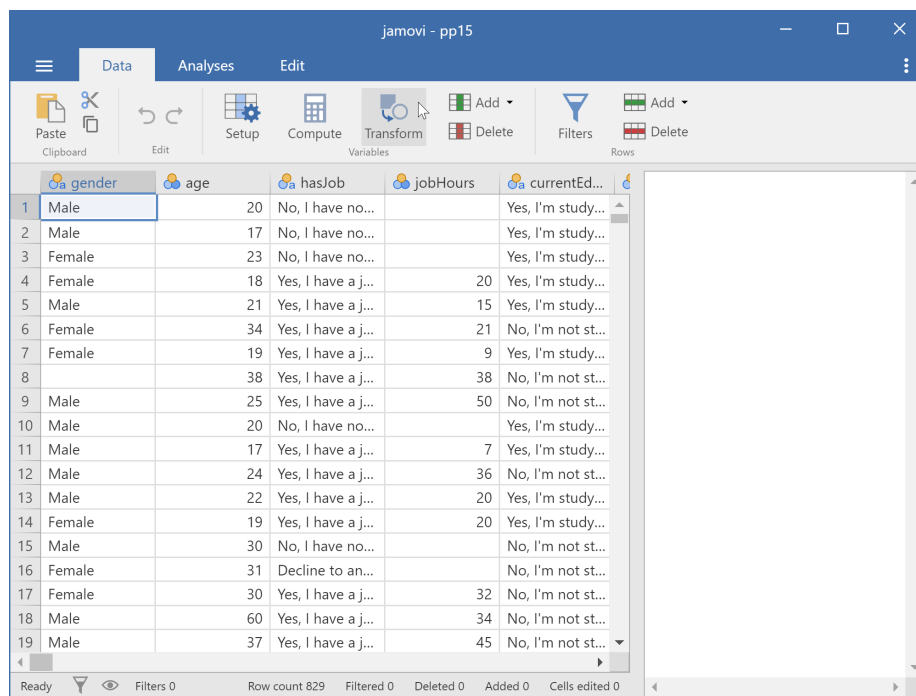


Figure 6.3: Recoding in jamovi: opening Transform menu

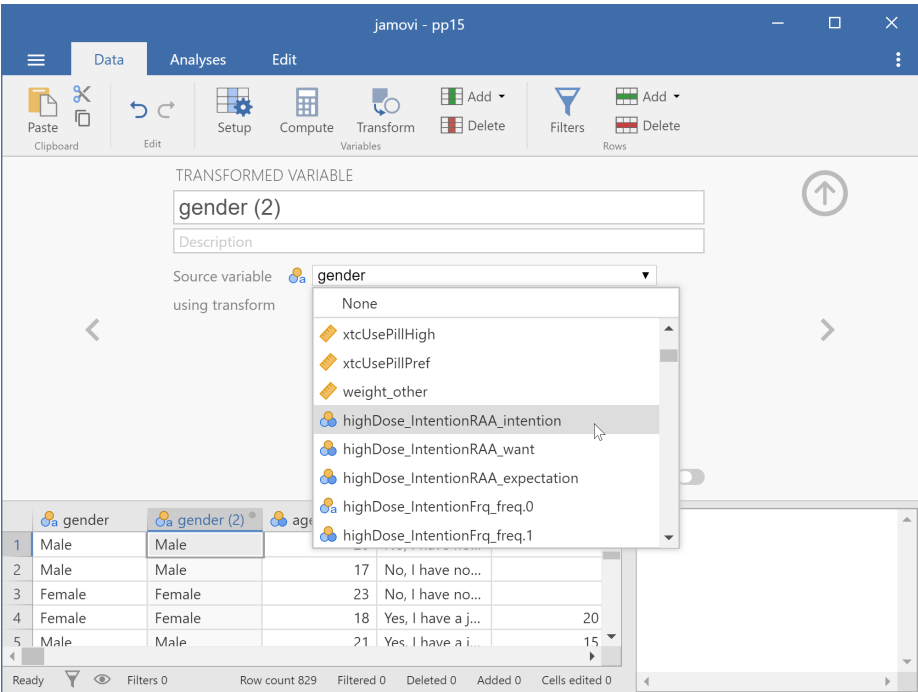


Figure 6.4: Recoding in jamovi: selecting the variable to transform

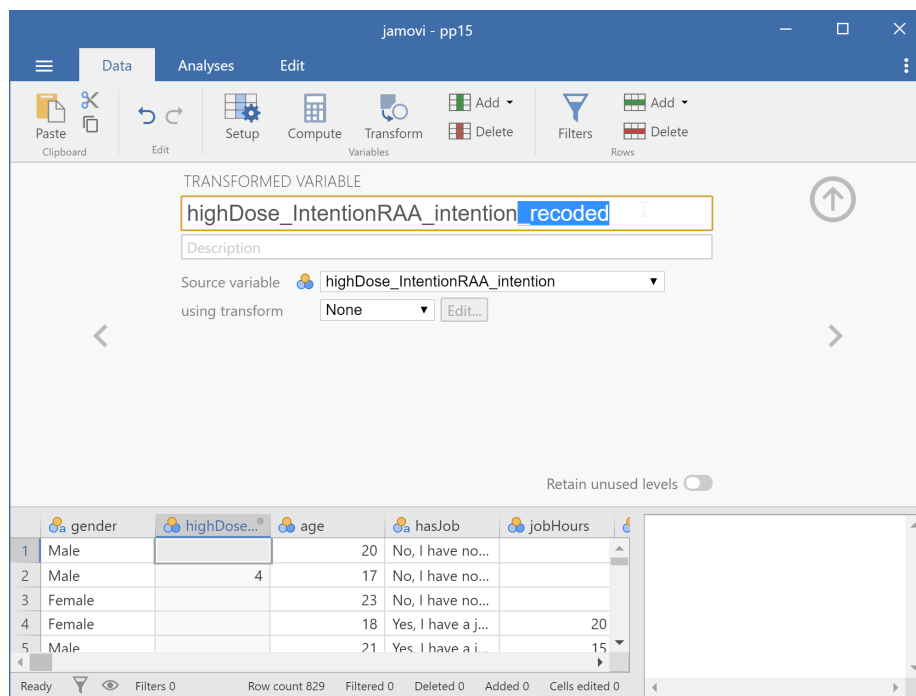


Figure 6.5: Recoding in jamovi: specifying the new variable name

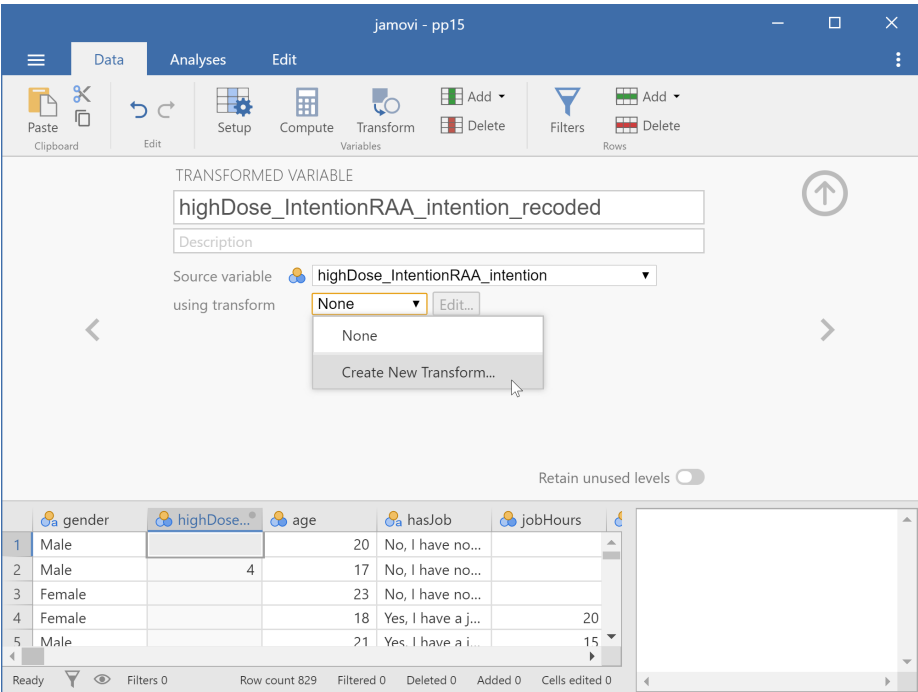


Figure 6.6: Recoding in jamovi: creating a new transformation

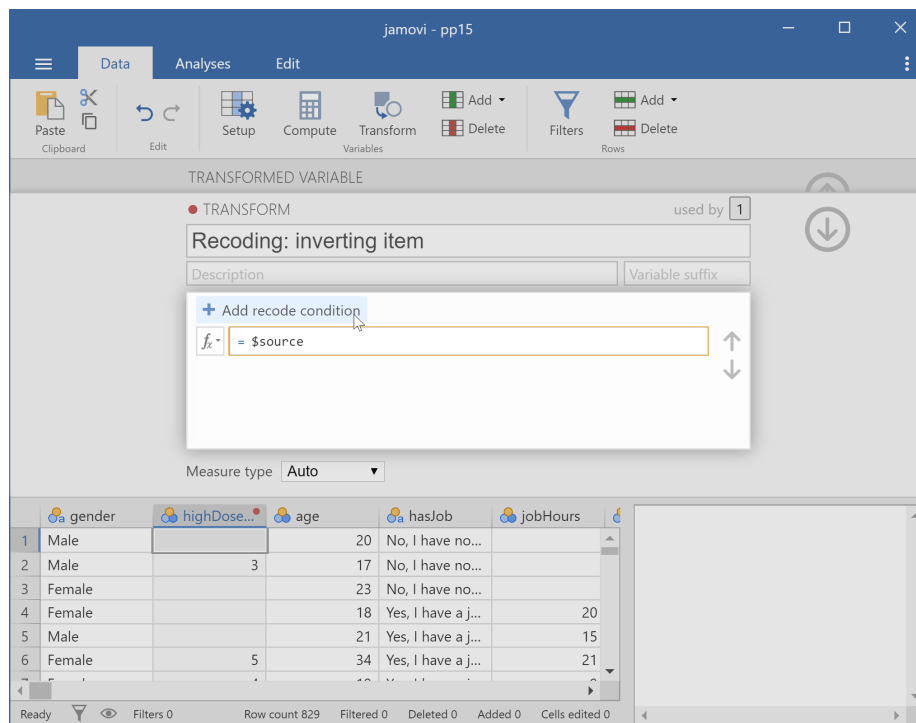


Figure 6.7: Recoding in jamovi: naming a transformation

Keep repeating this as many times as you have values that you want to recode, so that you get a list of recode conditions as shown in Figure 6.8.

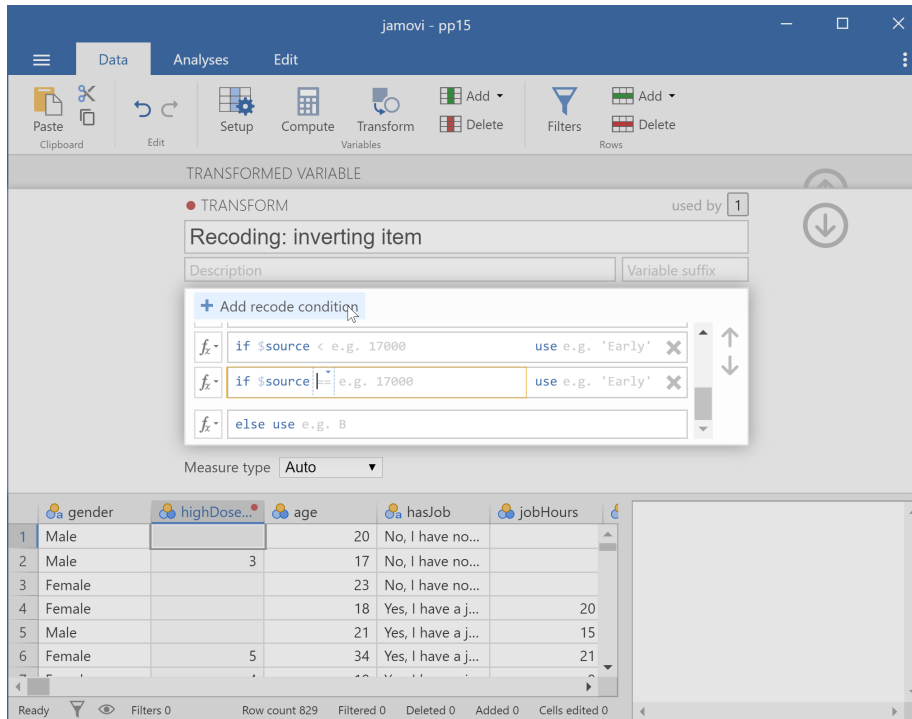


Figure 6.8: Recoding in jamovi: adding recode conditions

Then, for each recoding condition, specify the condition between “\$source” and “use”, and specify the value you want to assign if that condition is true after “use” as shown in Figure 6.9.

On the last row, after “else use”, you can specify which value to assign in rows where the source variable does not match any of the conditions. Because jamovi immediately shows every change in the data spreadsheet you see at the bottom, it is easy to play around until you get it right.

6.3 Input: R

6.3.1 Subtraction

To subtract all values from the maximum value (in this case, 8), you can use the following command (note that this requires the example dataset to be stored under name `dat`, see section 3):

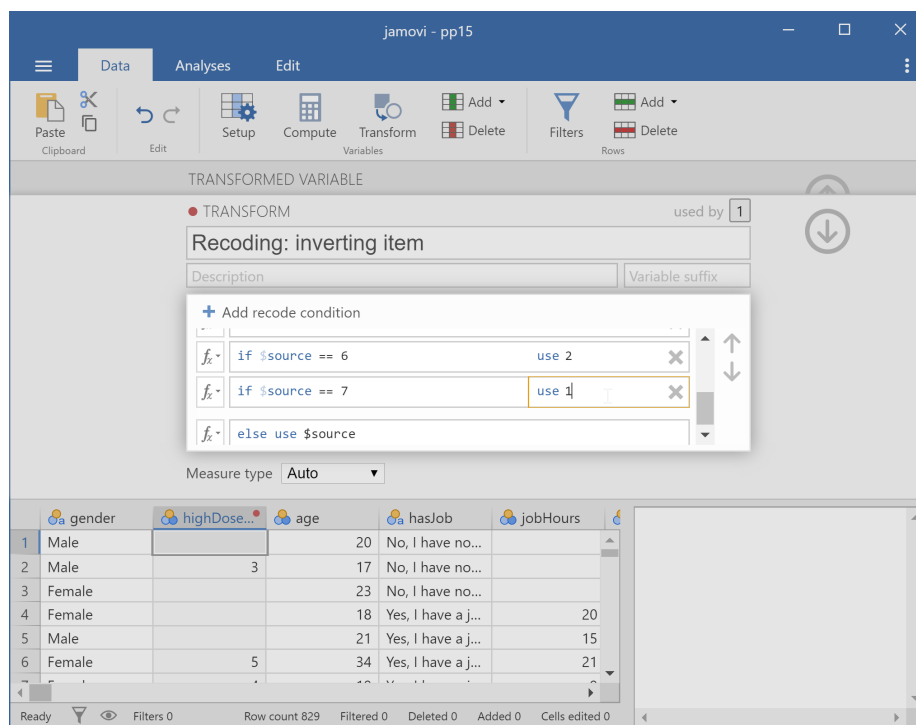


Figure 6.9: Recoding in jamovi: specifying the recodes

```
dat$highDose_IntentionRAA_intention_recoded <-  
  8 - dat$highDose_IntentionRAA_intention;
```

6.3.2 Manual specification

To manually specify each value you want, you can use the following command (this requires the `rosetta` package to be installed, see section 2.3.2, and the example dataset to be stored under name `dat`, see section 3):

```
dat$highDose_IntentionRAA_intention_recoded <-  
  rosetta::recode(  
    dat$highDose_IntentionRAA_intention,  
    "1=7; 2=6; 3=5; 4=4; 5=3; 6=2; 7=1"  
  );
```

6.4 Input: SPSS

6.4.1 SPSS: GUI

First activate the `dat` dataset (see 2.4.1).

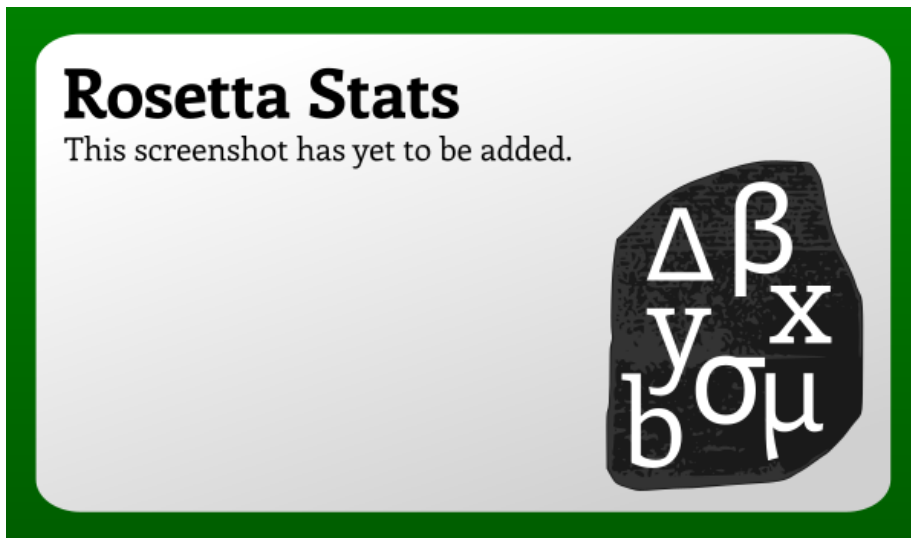


Figure 6.10: A screenshot placeholder

6.4.2 SPSS: Syntax

6.4.2.1 Subtraction

To subtract all values from the maximum value (in this case, 8), you can use the following command (this requires the `dat` dataset to be the active dataset, see 2.4.1):

```
COMPUTE highDose_IntentionRAA_intention_recoded =  
      8 - highDose_IntentionRAA_intention.
```

6.4.2.2 Manual specification

To manually specify each value you want, you can use the following command:

```
RECODE highDose_IntentionRAA_intention  
      (1=7) (2=6) (3=5) (4=4) (5=3) (6=2) (7=1)  
      INTO highDose_IntentionRAA_intention_recoded.
```

6.5 Output

Recoding a variable is not an analysis, and as such, does not produce output. You can inspect the newly created variable to ensure it has been created properly.

Chapter 7

Standardizing

7.1 Intro

Standardizing a variable means subtracting its mean from every data point in the data series, and dividing the resulting numbers by the variable's standard deviation. The result is a variable with a mean of 0 and a standard deviation of 1.

7.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

7.1.2 Variable(s)

From this dataset, this example uses variable `xtcUsePillHigh`.

7.2 Input: jamovi

7.3 Input: R

This stores the standardized values in a variable called `xtcUsePillHigh_standardized`:



Figure 7.1: A screenshot placeholder

```
dat$xtcUsePillHigh_standardized <-  
  scale(dat$xtcUsePillHigh);
```

In R is also easy to center a variable around its mean (i.e. omit the division by the standard deviation from the standardization procedure). The following command stores the centered values in a variable called `xtcUsePillHigh_centered`:

```
dat$xtcUsePillHigh_centered <-  
  scale(dat$xtcUsePillHigh, scale = FALSE);
```

7.4 Input: SPSS

This command orders descriptives, but the `/SAVE` subcommand also saves the standardized values. These are then given the original variable name prepended by Z, so in this case, `ZxtcUsePillHigh`:

```
DESCRIPTIVES VARIABLES = xtcUsePillHigh  
  /SAVE.
```

7.5 Output

Recoding a variable is not an analysis, and as such, does not produce output. You can inspect the newly created variable to ensure it has been created properly.

Chapter 8

Changing Variable Type

8.1 Intro

The data series that represent measurements or variables generally have meta-data specifying their ‘data type’, ‘measure type’, or ‘class’. These types entail certain restrictions regarding what can be stored in each data point. For example, data series where the data points have the ‘text’ or ‘string’ data type have no restrictions (and as a consequence, cannot be used to perform computations); ‘numeric’ or ‘scale’ data series can only contain numbers (and allow computations); ‘factor’ data series contain categorical data; and ‘datetime’ or ‘date’ dataseries contain dates and times. This chapter explains how to change the type of a column in a dataset.

8.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

8.1.2 Variable(s)

Here, explain which variable(s) people should open to copy the examples. For example:

From this dataset, this example uses the following variables:

- As a numeric variable: `weight_other`
- As a binary variable: `hadJob_bi`

- As a variable stored as numeric but that may also be considered categorical: `highDose_IntentionRAA_intention`

8.2 Input: jamovi

In jamovi, there's a very general data type that's called 'nominal'. That doesn't mean that it represents a measurement at the 'nominal level of measurement'; instead, it is a sort of general-purpose data type.

To change a variable's data type in jamovi, click the 'Data' tab at the top and select the column of the relevant variable. Then, you can specify the data type using two dropdown boxes: one labelled 'Measure type', which determines how the variable is treated in jamovi, and one labelled 'Data type' which relates to the way the variable is stored in the dataset. By setting 'Measure type' to nominal, as shown in Figure 8.1, a variable can be used in most analyses.

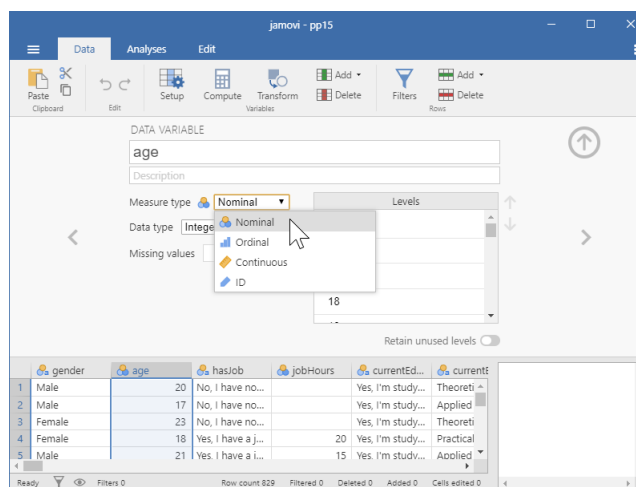


Figure 8.1: A screenshot placeholder

8.3 Input: R

In R, columns in a data frame can have the following frequently used data types:

- `logical`, which can only have `TRUE` and `FALSE` as values;
- `numeric`, which can only have numbers as values;
- `character`, which can handle literal text strings;
- `factor`, which can handle categorical data.

To convert a column from one data type to another, the corresponding `as.X` can be used, where `X` is the new data type (so `as.logical()`, `as.numeric()`, `as.character()`, and `as.factor()`).

8.4 Input: SPSS

For SPSS, there are two approaches: using the Graphical User Interface (GUI) or specify an analysis script, which in SPSS are called “syntax”.

8.4.1 SPSS: GUI



Figure 8.2: A screenshot placeholder



Figure 8.3: A screenshot placeholder



Figure 8.4: A screenshot placeholder



Figure 8.5: A screenshot placeholder



Figure 8.6: A screenshot placeholder

8.4.2 SPSS: Syntax

8.5 Output: jamovi

8.6 Output: R

8.7 Output: SPSS

8.8 Read more

Here, you can list one or more sources with background reading, for example:

If you would like more background on this topic, you can read more in these sources:

- Learning Statistics with R (Navarro, 2018): section XXXXXXXXXXXX, page XXXXXXXXXXXX (or follow this link for the Bookdown version)

Part II

Univariate analyses

Chapter 9

Bar chart

9.1 Intro

Bar charts are normally used to inspect the distribution of categorical (dichotomous, nominal or ordinal) variables.

9.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

9.1.2 Variable(s)

From this dataset, this example uses variable `currentEducation`.

9.2 Input: jamovi

9.3 Input: R

Use the following command (this requires the `rosetta` package to be installed, see section 2.3.2, and the example dataset to be stored under name `dat`, see section 3):



Figure 9.1: A screenshot placeholder

```
rosetta::ggBarChart(dat$currentEducation);
```

For more advanced stuff, visit the Rosetta Stats Pro: SPSS to R book.

9.4 Input: SPSS

9.4.1 SPSS: GUI

First activate the pp15 dataset by clicking on it (see 2.4.1).

Open the “Graphs” menu and select the “Chart Builder” option as shown in Figure ??.

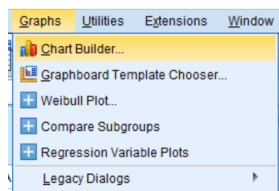


Figure 9.2: Opening the “Chart Builder” option in SPSS.

In the “Chart Builder” dialog, select “Bar” from “Gallery” and drag “Simple Bar” to the “Chart Preview” as shown in Figure 9.3.

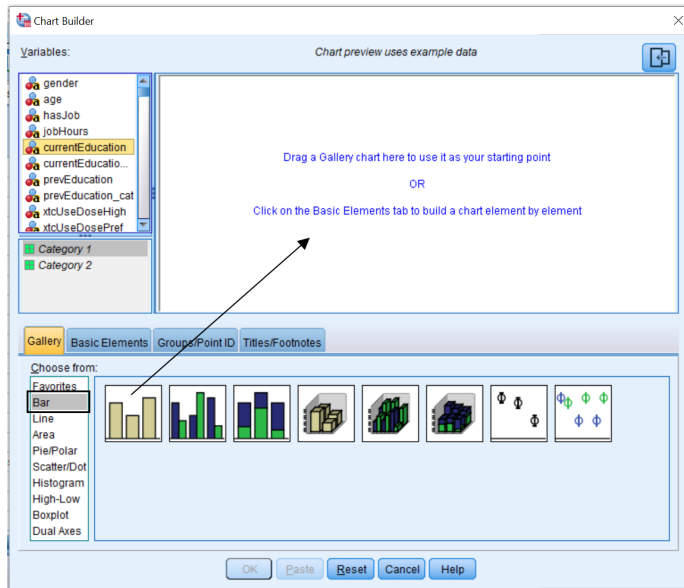


Figure 9.3: Selecting Bar Chart from the Chart Builder in SPSS

Under the “Variables” menu, select “currentEducation” and drag it onto the “X-Axis?” option in “Chart Preview” as shown in Figure 9.4. Note that there are options to modify the aesthetics of the graph on the right side of the screen under “Element Properties”, “Chart Appearance”, and “Options”.

9.4.2 SPSS: Syntax

Use the following command (this requires the `dat` dataset to be the active dataset, see 2.4.1):

THIS COMMAND HAS YET TO BE ADDED

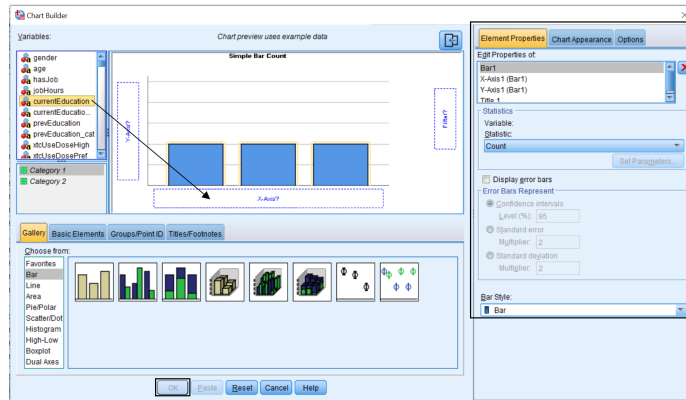


Figure 9.4: Adding the variable to the bar chart in SPSS.



Figure 9.5: A screenshot placeholder

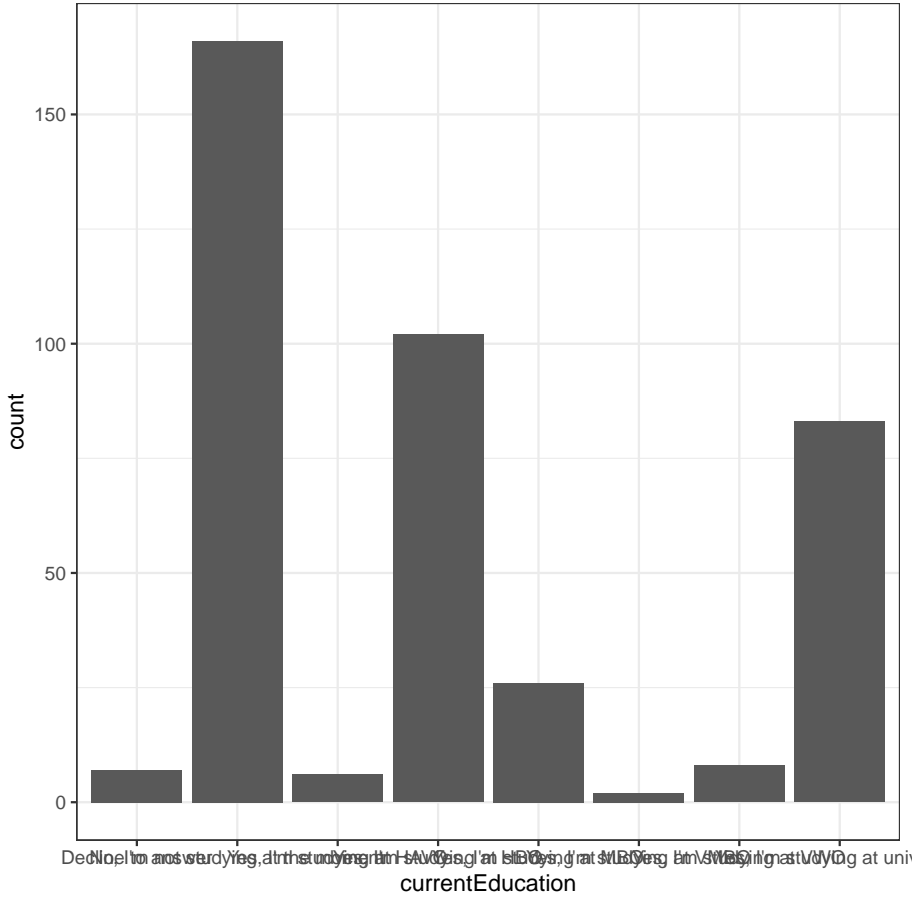


Figure 9.6: The produced box plot in R.

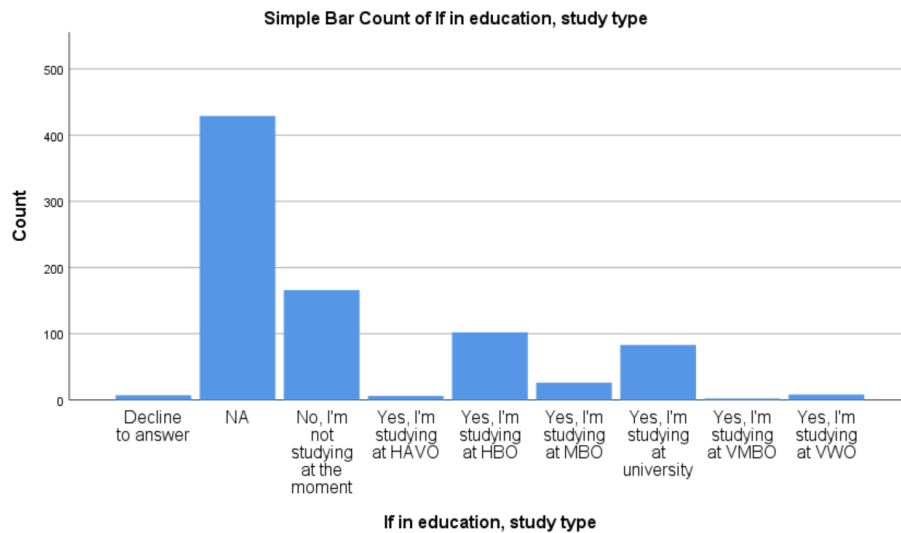


Figure 9.7: The box plot produced in SPSS.

9.5 Output: jamovi

9.6 Output: R

9.7 Output: SPSS

9.8 Read more

If you would like more background on this topic, you can read more in these sources:

- Discovering Statistics with SPSS (closed access):
- Learning Statistics with R (Navarro, 2018): section XXXXXXXXXXXX, page XXXXXXXXXXXX (or follow this link for the Bookdown version)

Chapter 10

Box plot

10.1 Intro

Boxplots are normally used to inspect the distribution of continuous (cardinal, e.g. interval or ratio-level) variables.

10.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

10.1.2 Variable(s)

From this dataset, this example uses variable `xtcUsePillHigh`.

10.2 Input: jamovi

Open the “Exploration” menu and select the “Descriptives” option as shown in Figure 10.1.

In the “Descriptives” dialog that appears, expand the “Plots” section at the bottom and check the checkbox labelled “Box plot” as shown in Figure 10.2.

Select the variable(s) for which you want to produce a box plot by dragging those variables from the list at the top left to the box marked “Variables” just to the right of that list. The box plot will be automatically produced in the output pane on the right-hand side.

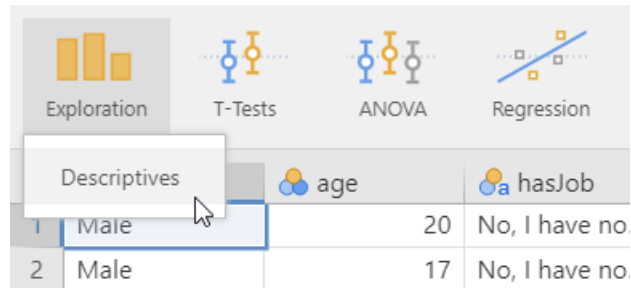


Figure 10.1: Opening the “Exploration” menu in jamovi.

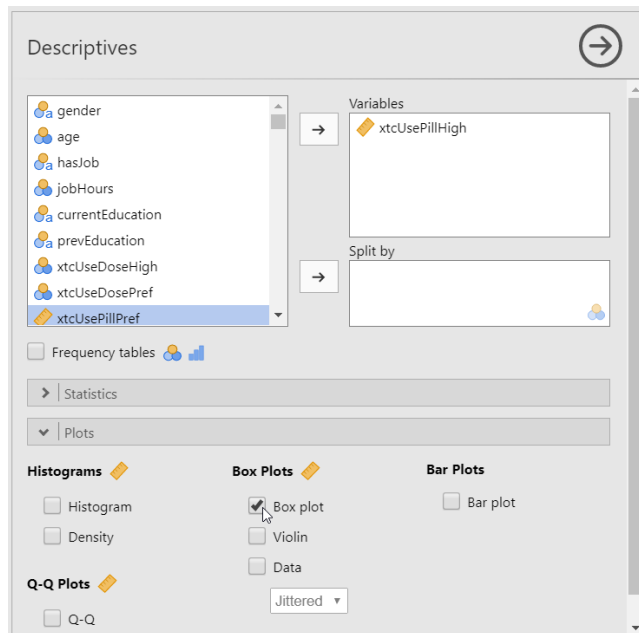


Figure 10.2: Specifying the options in the jamovi “Descriptives” dialog.

10.3 Input: R

10.3.1 R: rosetta

This requires the `rosetta` package to be installed, see section 2.3.2, and the example dataset to be stored under name `dat`, see section 3.

```
rosetta::ggBoxplot(dat$xtcUsePillHigh);
```

10.3.2 R: base

This requires the example dataset to be stored under name `dat`, see section 3.

```
boxplot(dat$xtcUsePillHigh);
```

10.4 Input: SPSS

10.4.1 SPSS: GUI

First activate the `dat` dataset by clicking on it (see 2.4.1).

Open the “Graphs” menu and select the “Chart Builder” option as shown in Figure 10.3.

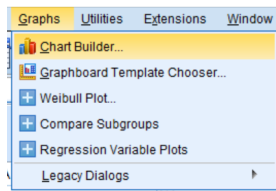


Figure 10.3: Opening the “Chart Builder” option in SPSS.

In the “Chart Builder” dialog, select “Boxplot” from “Gallery” and drag “1-D Boxplot” to the “Chart Preview” as shown in Figure 10.4.

Under the “Variables” menu, select “`xtcUsePillHigh`” and drag it onto the “X-Axis?” option in “Chart Preview” as shown in Figure 10.5. The variable you put on the vertical x-axis *must* be labeled as “scale” type. Note that there are options to modify the aesthetics of the graph on the right side of the screen under “Element Properties”, “Chart Appearance”, and “Options”.

Press OK and the graph will appear in the SPSS output window.

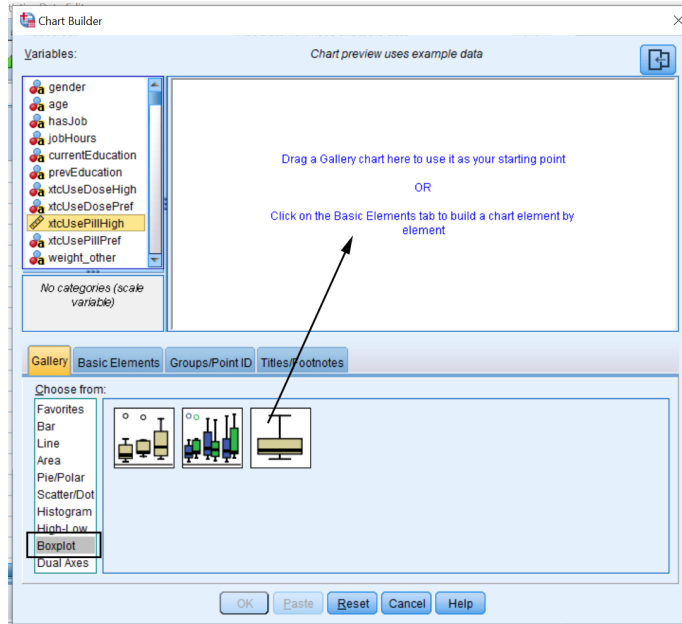


Figure 10.4: Selecting Box Plot from the Chart Builder in SPSS.

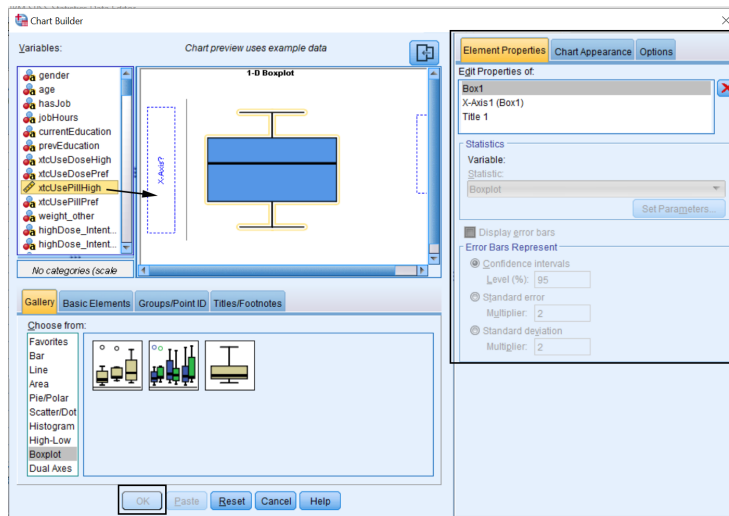


Figure 10.5: Adding the variable to the box plot in SPSS.

10.4.2 SPSS: Syntax

Use the following command (this requires the `dat` dataset to be the active dataset, see 2.4.1):

```
GRAPH /BOXPLOT = xtcUsePillHigh.
```

10.5 Output: jamovi

xtcUsePillHigh

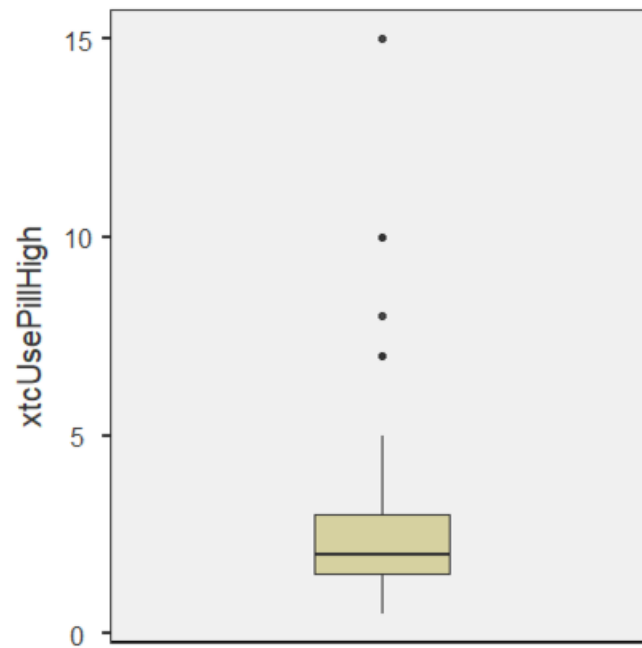


Figure 10.6: The produced box plot in jamovi.

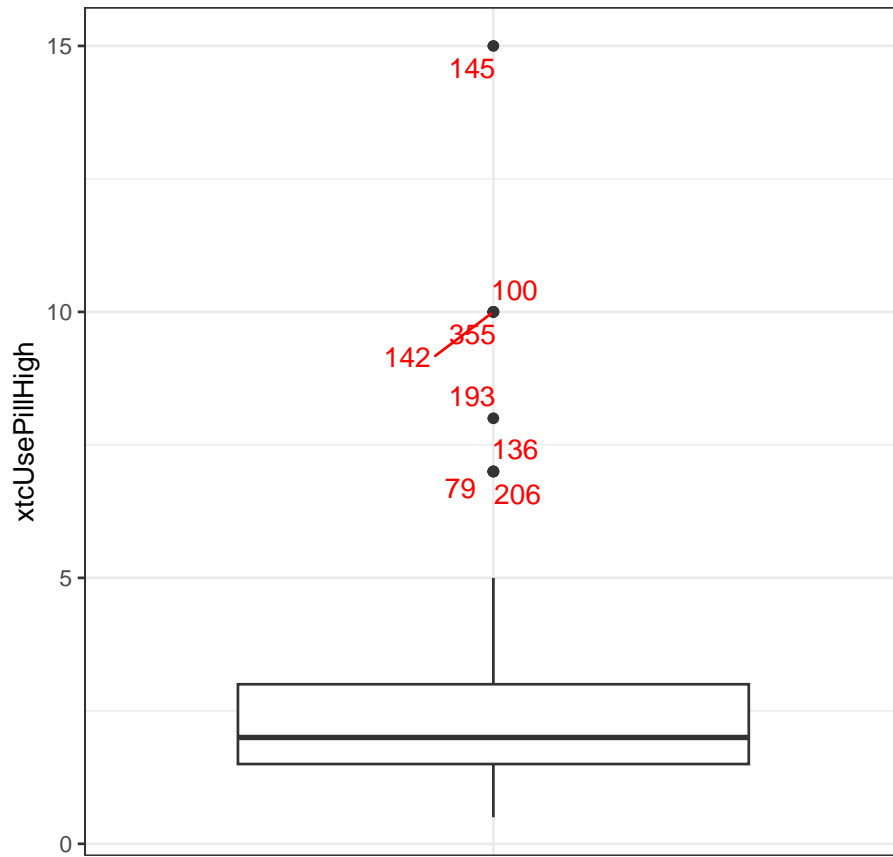


Figure 10.7: The produced box plot using the rosetta package in R.

10.6 Output: R

10.6.1 rosetta

10.6.2 Base R

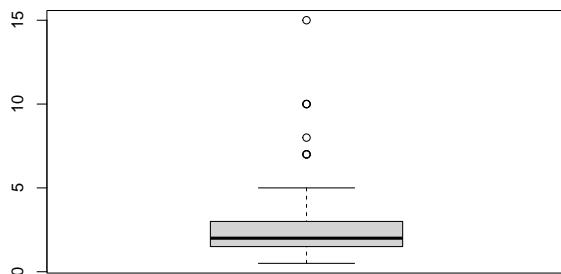


Figure 10.8: The produced box plot in Base R.

10.7 Output: SPSS

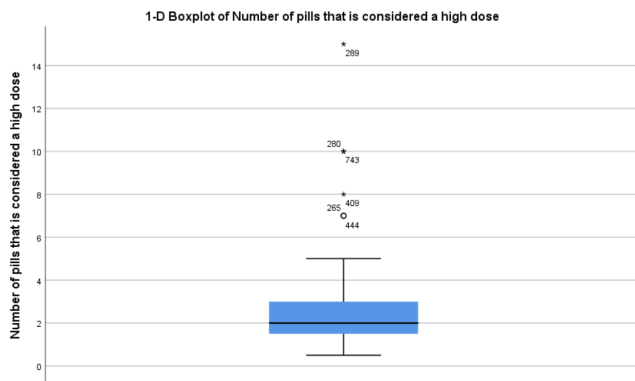


Figure 10.9: The boxplot produced in SPSS.

10.8 Read more

If you would like more background on this topic, you can read more in these sources:

- [Discovering Statistics with SPSS \(closed access\)](#):
- [Learning Statistics with R \(Navarro, 2018\): section 6.5, page 175 \(or follow this link for the Bookdown version\)](#)

Chapter 11

Descriptives

11.1 Intro

The term ‘descriptives’ is typically used to refer to a set of measures summarizing a distribution, such as central tendency and spread measures.

11.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

11.1.2 Variable(s)

From this dataset, this example uses variables `xtcUseDoseHigh`, `highDose_intention`, `highDose_attitude` (all three numeric variables) and `hasJob_bi` (a dichotomous factor, i.e. a categorical variable).

11.2 Input: jamovi

11.3 Input: R

11.3.1 R: base

Use the following command:

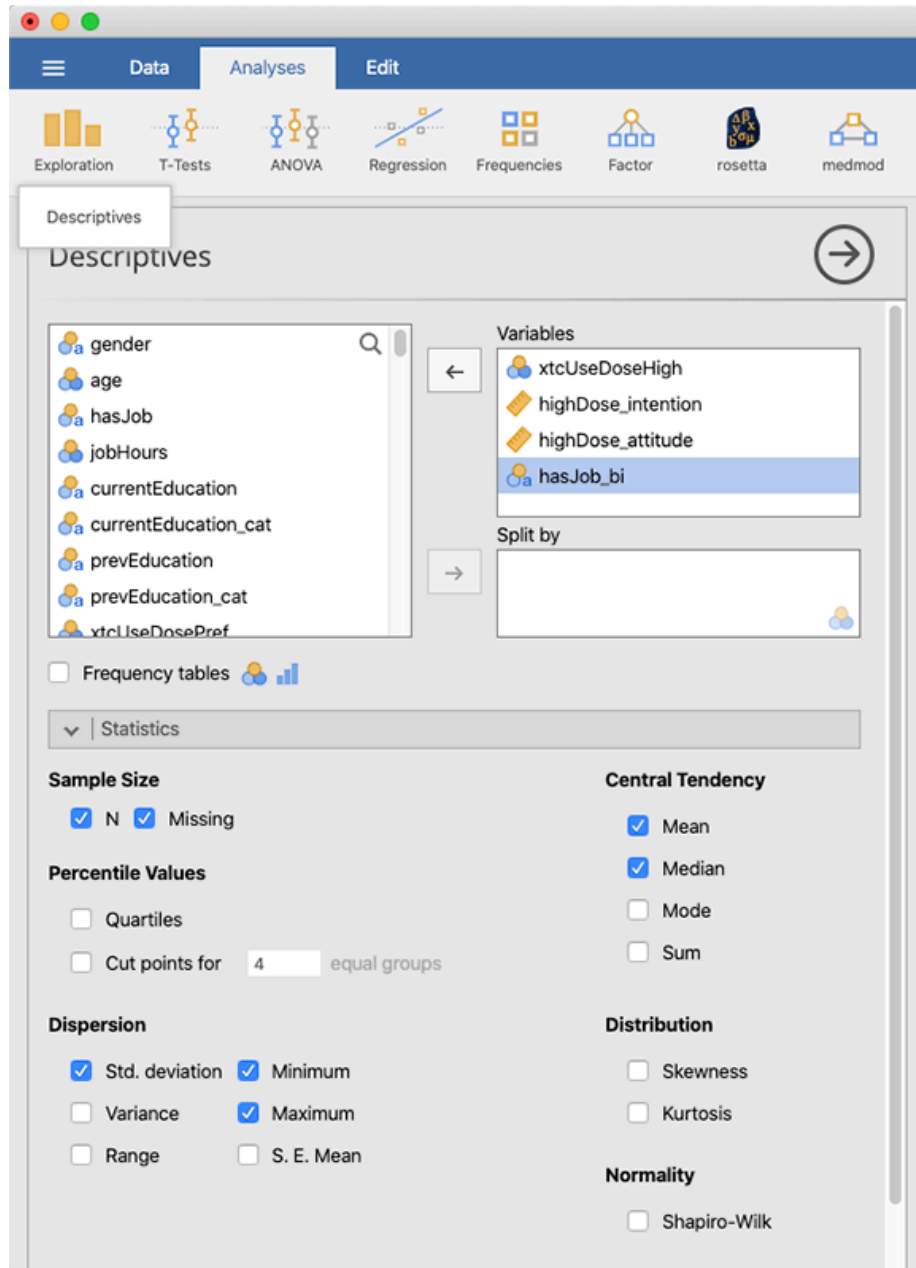


Figure 11.1: Opening the “Exploration” menu in jamovi.

```
summary(  
  dat[,  
    c(  
      "xtcUseDoseHigh",  
      "highDose_intention",  
      "highDose_attitude",  
      "hasJob_bi"  
    )  
  ]  
);
```

11.3.2 R: Rosetta

Use the following command:

```
rosetta::descr(  
  dat,  
  items = c(  
    "xtcUseDoseHigh",  
    "highDose_intention",  
    "highDose_attitude",  
    "hasJob_bi"  
  ),  
  histogram = TRUE,  
  boxplot = TRUE  
);
```

When ordering descriptives for a single variable (and in version 0.3.2 of the `rosetta` package, also for multiple variables), you can finetune which descriptives you get by passing `TRUE` or `FALSE` for arguments `mean`, `meanCI`, `median`, `mode`, `var`, `sd`, `se`, `min`, `max`, `q1`, `q3`, `IQR`, `skewness`, `kurtosis`, `dip`, `totalN`, `missingN`, and `validN`.

In addition, you can order histograms (or bar charts, for factors) and box plots (for numeric variables only) by passing `histogram=TRUE` and `boxplot=TRUE`, respectively.

If you omit the `items` argument, you get descriptives for all variables in the dataframe; and if you don't pass a dataframe, but a single variable, you will just get the descriptives for that variable.

11.4 Input: SPSS

11.4.1 SPSS: GUI

First activate the pp15 dataset by clicking on it (see 2.4.1).

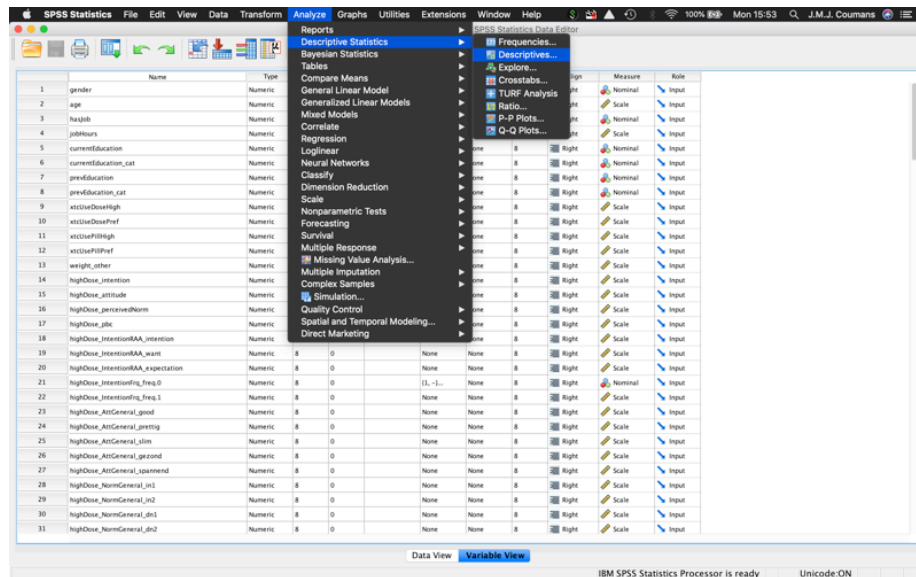


Figure 11.2: Opening the “descriptives menu” in SPSS

Then select the variables of interest.

11.4.2 SPSS: Syntax

Use the following command (this requires the `dat` dataset to be the active dataset, see 2.4.1):

```
DESCRIPTIVES VARIABLES=xtcUseDoseHigh highDose_intention highDose_attitude hasJob_bi
/STATISTICS=MEAN STDDEV MIN MAX.
```

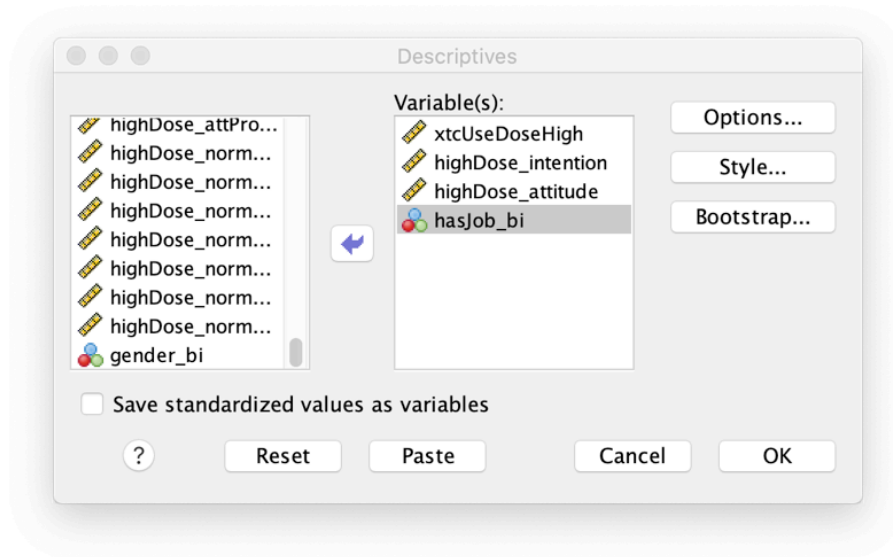


Figure 11.3: Selection of variables of interest

Descriptives

Descriptives	xtcUseDoseHigh	highDose_intention	highDose_attitude	hasJob_bi
N	287	398	304	383
Missing	542	431	525	446
Mean	208	2.37	3.64	
Median	180	2.00	3.80	
Standard deviation	130	1.47	1.08	
Minimum	25	0.500	1.00	
Maximum	880	15.0	6.40	

Figure 11.4: The produced descriptives output in jamovi

11.5 Output: jamovi

11.6 Output: R

11.6.1 R: rosetta

```
## Warning: ggrepel: 11 unlabeled data points (too many overlaps). Consider  
## increasing max.overlaps
```

11.6.1.1 Descriptives for variables in data frame pp15

```
mean  
meanCI  
median  
mode  
var  
sd  
min  
max  
skewness  
kurtosis  
dip  
totalN  
missingN  
validN  
xtcUseDoseHigh  
208.35  
[193.2; 223.5]  
180.0  
200  
17000.69  
130.39  
25.0
```


880.0

1.84

4.74

0.06

829

542

287

highDose_intention

2.37

[2.22; 2.51]

2.0

2

2.17

1.47

0.5

15.0

3.14

18.41

0.09

829

431

398

highDose_attitude

3.64

[3.52; 3.76]

3.8

4

1.16

1.08

1.0

6.4

-0.43

-0.05

0.04

829

525

304

11.6.1.1.1 Frequencies for hasJob_bi Frequencies

Perc.Total

Perc.Valid

Cumulative

No

81

9.8

21.1

21.1

Yes

302

36.4

78.9

100.0

Total valid

383

46.2

100.0

NA (missing)

446

53.8

Total

829

100.0

11.6.1.1.2 Histograms for variables in data frame pp15

```
## Warning: The dot-dot notation (`.scaled.`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(scaled)` instead.
## i The deprecated feature was likely used in the ufs package.
## Please report the issue at <https://gitlab.com/r-packages/ufs/-/issues>.
```

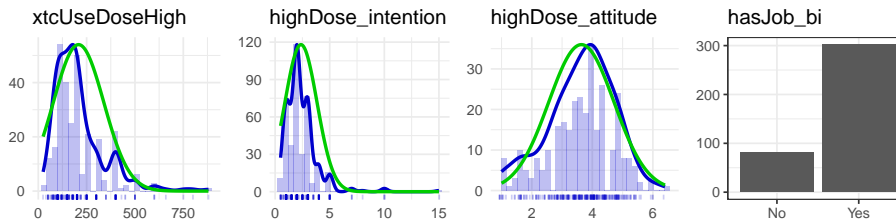


Figure 11.5: Histograms for variables in data frame pp15

11.6.1.1.3 Boxplots for variables in data frame pp15

```
## Warning: ggrepel: 11 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

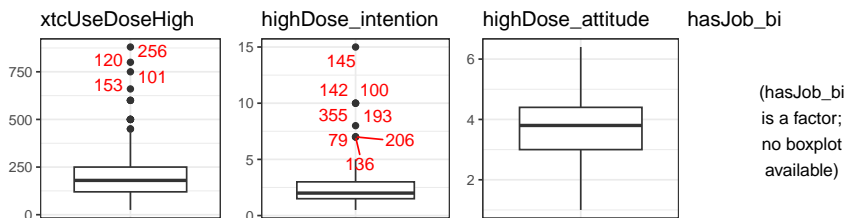


Figure 11.6: Boxplots for variables in data frame pp15

11.7 Output: SPSS

11.8 Read more

If you would like more background on this topic, you can read more in these sources:

- Discovering Statistics with SPSS (closed access):
- Learning Statistics with R (Navarro, 2018): section XXXXXXXXXXXX, page XXXXXXXXXXXX (or follow this link for the Bookdown version)

Descriptive Statistics

	N	Minimum	Maximum	Mean	Std. Deviation
xtcUseDoseHigh	287	25	880	208.35	130.387
highDose_intention	398	.50	15.00	2.3675	1.47257
highDose_attitude	304	1.00	6.40	3.6375	1.07809
hasJob_bi	383	1	2	1.79	.409
Valid N (listwise)	10				

Figure 11.7: The descriptives produced in SPSS.

Chapter 12

Frequencies

12.1 Intro

Frequency tables are normally used to inspect the distribution of categorical (dichotomous, nominal or ordinal) variables.

12.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

12.1.2 Variable(s)

From this dataset, this example uses variable `currentEducation_cat`.

12.2 jamovi

In the **analyses** tab there is a symbol with a barchart, called ‘Exploration’. Go to this menu and select **Descriptives**. Drag the variables ypu want to describe to the variables window and check the little box **Frequency tables** to obtain a frequency table.

12.3 R

There are **many** packages that can be used to create a frequency table. We have only presented two examples. Other packages include (but are not limited to):

- `summarytools`
- `Deducer`
- `janitor`
- `questionr`
- `sjmisc`
 - If you read an SPSS dataset into R, consider using the “`freq`” command from the “`sjmisc`” package. It presents both values and value labels (similar to SPSS output).

Note: To use the following commands, it is necessary to install and load the packages first (see section 2.3.2). The example dataset is stored under the name `dat` (see section 3).

12.3.1 rosetta package

Use the following command (this requires the `rosetta` package to be installed, see section 2.3.2, and the example dataset to be stored under name `dat`, see section 3):

```
rosetta::freq(dat$currentEducation_cat);
```

To also order a barchart, use:

```
rosetta::freq(dat$currentEducation_cat, plot=TRUE);
```

To order frequencies for multiple variables simultaneously, use:

```
rosetta::frequencies(dat$currentEducation_cat,  
                    dat$prevEducation_cat);
```

12.3.2 `descr` and `kableExtra` packages

The `descr` package is used to run the “descriptive statistics” for the variable (in this case, a frequency table). By default, the `freq` command in the `descr` package will also create a basic bar graph. The `kableExtra` package can be combined with many packages to create aesthetically pleasing tables.

```
kableExtra::kable_styling(
  knitr::kable(as.data.frame(descr::freq(dat$currentEducation_cat)),
    booktabs=T, digits=2));
```

In words:

1. From the `descr` package, use the `frequencies` command for the `currentEducation` variable from the dataset: `(descr::freq(dat$currentEducation_cat))`.
2. To make the aesthetically pleasing output, we are going to create a kable from the `kableExtra` package. Kables require dataframes, so we need to turn this frequency output into a dataframe: `as.data.frame()`.
3. Now, let’s call the `kable` function from the `knitr` package: `knitr::kable()`.
4. And add some stylistic elements, such as (what does `booktabs=T` actually do?) `booktabs=T` and changing the number of decimal places to 2 digits `digits=2`.
5. Lastly, let’s add `kablestyling` to make the kable aesthetically pleasing: `kableExtra::kable_styling()`.

12.4 SPSS

Use the following command (this requires the `dat` dataset to be the active dataset, see 2.4.1):

```
FREQ VARIABLES=currentEducation_cat.
```

To also order a barchart, use:

```
FREQ VARIABLES=currentEducation_cat
  /BARCHART FREQ.
```

To order frequencies for multiple variables simultaneously, use:

```
FREQ VARIABLES=currentEducation_cat prevEducation_cat.
```

12.5 Read more

If you would like more background on this topic, you can read more in these sources:

- Discovering Statistics with SPSS (closed access):
- Learning Statistics with R (Navarro, 2018): section XXXXXXXXXXXX, page XXXXXXXXXXXX (or follow this link for the Bookdown version)

Chapter 13

Histogram

13.1 Intro

Histograms are normally used to inspect the distribution of continuous (cardinal, e.g. interval or ratio-level) variables.

13.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

13.1.2 Variable(s)

From this dataset, this example uses variable `xtcUsePillHigh`.

13.2 Input: jamovi

13.3 Input: R

Use the following command (this requires the `rosetta` package to be installed, see section 2.3.2, and the example dataset to be stored under name `dat`, see section 3):



Figure 13.1: A screenshot placeholder

```
hist(dat$xtcUsePillHigh);
```

13.4 Input: SPSS GUI

First activate the `dat` dataset (see 2.4.1).

13.5 Input: SPSS Syntax

Use the following command (this requires the `dat` dataset to be the active dataset, see 2.4.1):

```
GRAPH /HISTOGRAM(NORMAL) = xtcUsePillHigh.
```



Figure 13.2: A screenshot placeholder



Figure 13.3: A screenshot placeholder

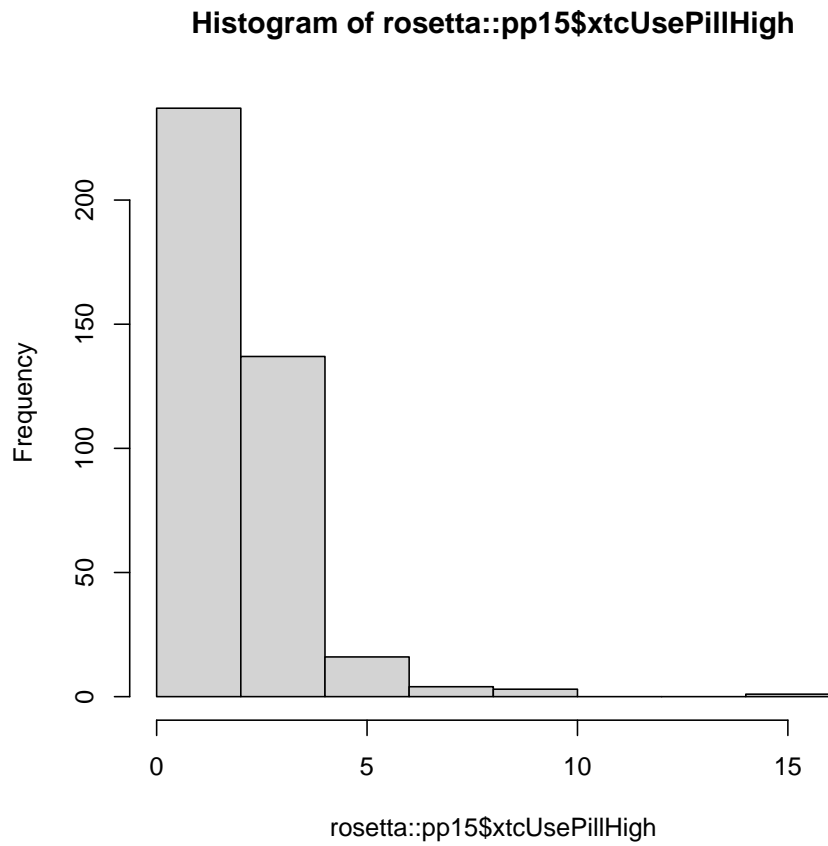


Figure 13.4: The produced box plot in R.

13.6 Output: jamovi

13.7 Output: R

13.8 Output: SPSS



Figure 13.5: A screenshot placeholder

13.9 Read more

If you would like more background on this topic, you can read more in these sources:

- Discovering Statistics with SPSS (closed access):
- Learning Statistics with R (Navarro, 2018): section XXXXXXXXXXXX, page XXXXXXXXXXXX (or follow this link for the Bookdown version)

Part III

Psychometrics

Chapter 14

Coefficient Alpha

14.1 Intro

Coefficient Alpha [also known as Cronbach’s Alpha, but Cronbach disliked that association; Cronbach and Shavelson (2004)] is considered a measure of internal consistency and can, if a set of severe assumptions is met, estimate the reliability of a set of items.

14.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

14.1.2 Variable(s)

From this dataset, this example uses variables `highDose_AttGeneral_good`, `highDose_AttGeneral_prettig`, `highDose_AttGeneral_slim`, `highDose_AttGeneral_gezond` & `highDose_AttGeneral_spannend`.

14.2 Input: jamovi

In the “Analyses” tab, click the “Factor” button and from the menu that appears, select “Reliability Analysis” as shown in Figure 14.1.

In the box at the left, select all variables you want to include in this analysis and move them to the box labelled “Items” using the button labelled with the rightward-pointing arrow as shown in Figure 14.2.

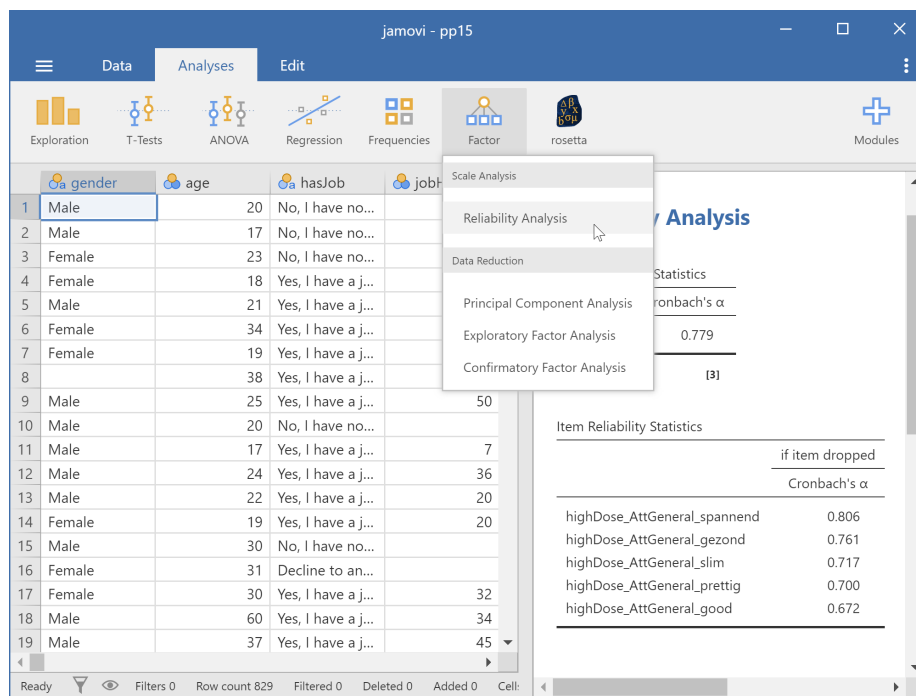


Figure 14.1: Opening the reliability analysis menu in jamovi

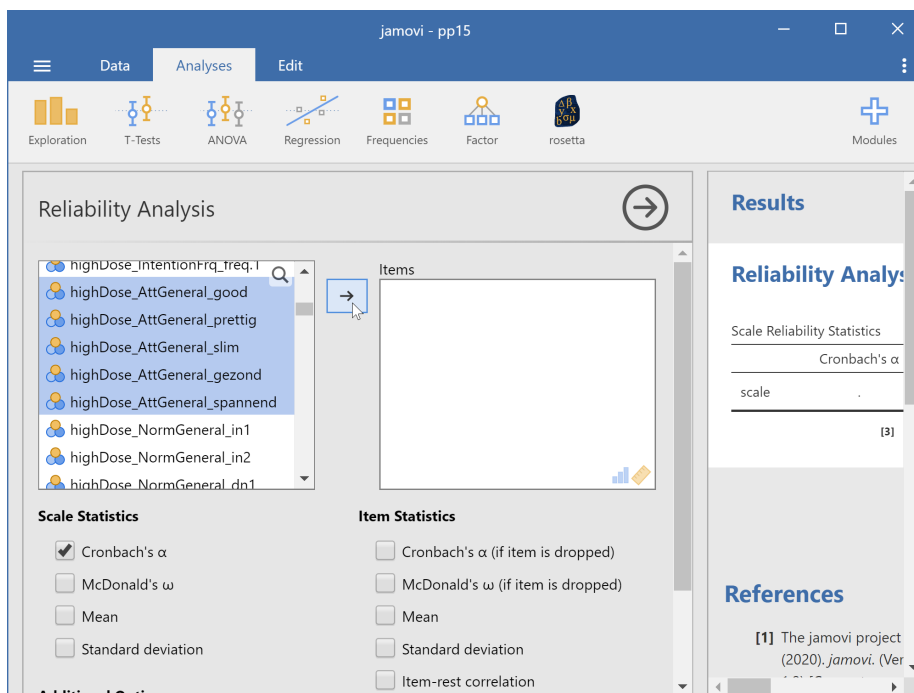


Figure 14.2: Adding the items to analyse in jamovi

Because Coefficient Alpha (called Cronbach's Alpha in the jamovi interface) has already been checked (see the left-most column at the bottom labelled "Scale Statistics"), you will immediately see Coefficient Alpha for the selected items appear in the table in jamovi's Results pane on the right-hand side. You can now also order additional statistics, such as the value Coefficient Alpha would have if you were to omit each item, as shown in Figure 14.3.

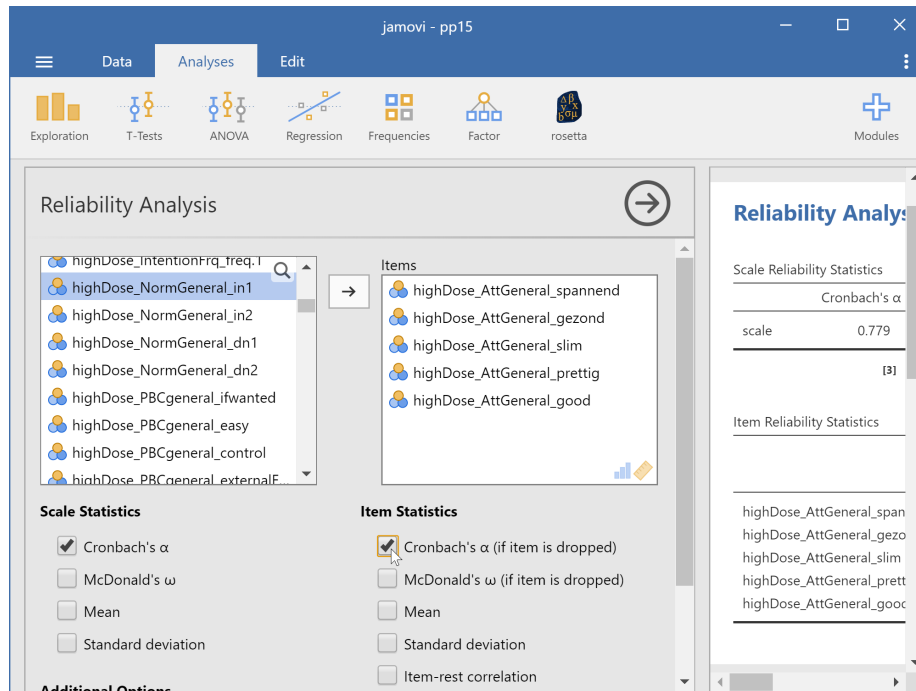


Figure 14.3: Ordering “Alpha if item dropped” in jamovi

14.3 Input: R

14.3.1 R: rosetta

```
rosetta::reliability(
  data = dat,
  items = c(
    "highDose_AttGeneral_good",
    "highDose_AttGeneral_prettig",
    "highDose_AttGeneral_slim",
    "highDose_AttGeneral_gezond",
    "highDose_AttGeneral_spannend"
```

```
)
);
```

14.4 Input: SPSS

To compute Coefficient Alpha in SPSS, use the following command:

```
RELIABILITY
/VARIABLES =
    highDose_AttGeneral_good
    highDose_AttGeneral_prettig
    highDose_AttGeneral_slim
    highDose_AttGeneral_gezond
    highDose_AttGeneral_spannend
/MODEL =
    ALPHA.
```

To order additional information, such as descriptive statistics, inter-item correlations, and other scale statistics, you can specify additional options:

```
RELIABILITY
/VARIABLES =
    highDose_AttGeneral_good
    highDose_AttGeneral_prettig
    highDose_AttGeneral_slim
    highDose_AttGeneral_gezond
    highDose_AttGeneral_spannend
/MODEL = ALPHA
/STATISTICS = DESCRIPTIVE SCALE CORR
/SUMMARY = TOTAL.
```

14.5 Output: jamovi

14.6 Output: R

14.6.1 R: rosetta

14.6.1.1 Reliability analysis

14.6.1.1.1 Scale structure

Reliability Analysis

Scale Reliability Statistics

	Cronbach's α
scale	0.779

[3]

Item Reliability Statistics

	if item dropped
	Cronbach's α
highDose_AttGeneral_spannend	0.806
highDose_AttGeneral_gezond	0.761
highDose_AttGeneral_slim	0.717
highDose_AttGeneral_prettig	0.700
highDose_AttGeneral_good	0.672

Figure 14.4: The output of a reliability analysis where Coefficient Alpha is computed in jamovi

14.6.1.1.1.1 Scale structure Information about this scale

Dataframe:	res\$data
Items:	highDose_AttGeneral_good, highDose_AttGeneral_prettig, highL
Observations:	303
Positive correlations:	10
Number of correlations:	10
Percentage positive correlations:	100

Estimates assuming interval level

Omega (total):	0.79
Omega (hierarchical):	0.80
Revelle's Omega (total):	0.79
Greatest Lower Bound (GLB):	0.86
Coefficient H:	0.85
Coefficient Alpha:	0.78

Note: the normal point estimate and confidence interval for omega are based on the procedure suggested by Dunn, Baguley & Brunsten (2013) using the MBESS function `ci.reliability`, whereas the psych package point estimate was suggested in Revelle & Zinbarg (2008). See the help ('?ufs::scaleStructure') for more information.

14.7 Output: SPSS

The SPSS output still has to be added.

Chapter 15

Exploratory Factor Analysis

15.1 Intro

Exploratory Factor Analysis (EFA) is a data reduction method that can be useful to identify what in psychology are called latent constructs.

15.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

15.1.2 Variable(s)

From this dataset, this example uses variables `highDose_AttBeliefs_long`, `highDose_AttBeliefs_intensity`, `highDose_AttBeliefs_intoxicated`, `highDose_AttBeliefs_energy`, `highDose_AttBeliefs_euphoria`, `highDose_AttBeliefs_insight`, `highDose_AttBeliefs_connection`, `highDose_AttBeliefs_contact` & `highDose_AttBeliefs_sex`.

15.2 Input: jamovi

In the “Analyses” tab, click the “Factor” button and from the menu that appears, select “Exploratory Factor Analysis” as shown in Figure 15.1.

In the box at the left, select the items and move them to the box labelled “Variables” using the button labelled with the rightward-pointing arrow as shown in Figure 15.1.

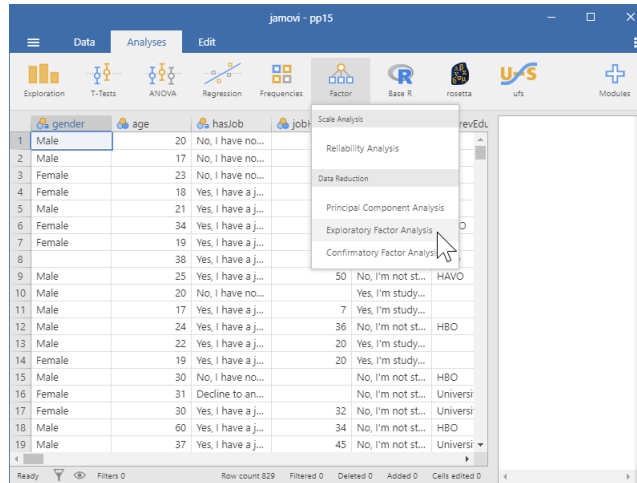


Figure 15.1: Opening the exploratory factor analysis menu in jamovi

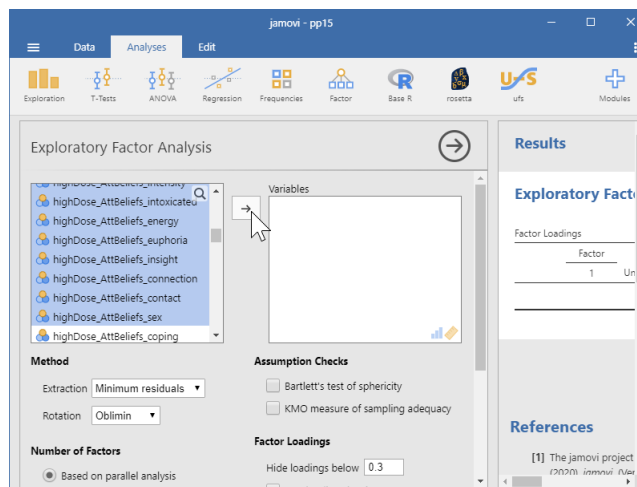


Figure 15.2: Selecting the variables for the exploratory factor analysis in jamovi

The factor analysis is then immediately executed and shown in the right-hand Results panel with a number of default settings.

If you scroll down, you can specify your analysis and change these defaults. For example, we can indicate that we want to select factors with an eigenvalue over 1 and that we want to see the factor summary, the correlation between factors, the model fit measures, and the scree plot, as shown in Figure 15.3.

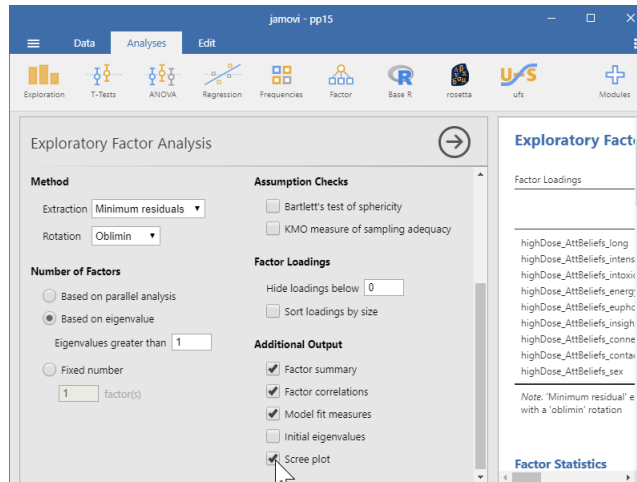


Figure 15.3: Selecting the variables for the exploratory factor analysis in jamovi

15.3 Input: R

15.3.1 R: rosetta

In R, using the `rosetta` package, you can use the following command:

```
rosetta::factorAnalysis(  
  data = dat,  
  items = c(  
    'highDose_AttBeliefs_long',  
    'highDose_AttBeliefs_intensity',  
    'highDose_AttBeliefs_intoxicated',  
    'highDose_AttBeliefs_energy',  
    'highDose_AttBeliefs_euphoria',  
    'highDose_AttBeliefs_insight',  
    'highDose_AttBeliefs_connection',
```

```

    'highDose_AttBeliefs_contact',
    'highDose_AttBeliefs_sex'
  ),
  nfactors = "eigen"
);

```

Note that this function forces you to specify how many factor you want to extract with the `nfactors` argument. You can also specify “`eigen`” to use the Kaiser criterion, in which case you can specify the minimum eigen value with the `kaiser` argument (set to 1 by default).

To order additional information, such as a factor summary, the correlations between the factors, a scree plot, and the residuals, and to specify pretty item labels, you can specify additional options:

```

rosetta::factorAnalysis(
  data = dat,
  items = c(
    'highDose_AttBeliefs_long',
    'highDose_AttBeliefs_intensity',
    'highDose_AttBeliefs_intoxicated',
    'highDose_AttBeliefs_energy',
    'highDose_AttBeliefs_euphoria',
    'highDose_AttBeliefs_insight',
    'highDose_AttBeliefs_connection',
    'highDose_AttBeliefs_contact',
    'highDose_AttBeliefs_sex'
  ),
  itemLabels = c(
    'Expectation that a high dose results in a longer trip',
    'Expectation that a high dose results in a more intense trip',
    'Expectation that a high dose makes you more intoxicated',
    'Expectation that a high dose provides more energy',
    'Expectation that a high dose produces more euphoria',
    'Expectation that a high dose yields more insight',
    'Expectation that a high dose strengthens your connection with others',
    'Expectation that a high dose facilitates making contact with others',
    'Expectation that a high dose improves sex'
  ),
  nfactors = "eigen",
  summary = TRUE,
  correlations = TRUE,
  scree = TRUE,

```

```
residuals = TRUE  
);
```

15.4 Input: SPSS

15.4.1 SPSS: GUI



Figure 15.4: A screenshot placeholder

15.4.2 SPSS: Syntax

In SPSS, the `FACTOR` command is used. Important arguments are `/VARIABLES` to specify the items, `/CRITERIA` to specify how many factors to extract (or how to decide that; e.g. `MINEIGEN(1)` to select factor with an eigenvalue over 1, or `FACTORS(2)` to extract 2 factors), `/EXTRACTION` to specify the factor extraction method (e.g. `ULS` for ordinary least squares, `PAF` for principal axis factoring, and `ML` for maximum likelihood), and `/ROTATION` to set the rotation (e.g. `NOROTATE` for no rotation, `VARIMAX` for an orthogonal rotation, and `OBLIMIN` for an oblique oblimin rotation. Don't forget the period at the end (`.`), the command terminator.

```
FACTOR
/VARIABLES
  highDose_AttBeliefs_long
  highDose_AttBeliefs_intensity
  highDose_AttBeliefs_intoxicated
  highDose_AttBeliefs_energy
  highDose_AttBeliefs_euphoria
  highDose_AttBeliefs_insight
  highDose_AttBeliefs_connection
  highDose_AttBeliefs_contact
  highDose_AttBeliefs_sex
/CRITERIA =
  MINEIGEN(1)
/EXTRACTION =
  ULS
/ROTATION =
  OBLIMIN
.
```

To request specific results, the /PRINT and /PLOT arguments can be used, for example:

```
FACTOR
/VARIABLES
  highDose_AttBeliefs_long
  highDose_AttBeliefs_intensity
  highDose_AttBeliefs_intoxicated
  highDose_AttBeliefs_energy
  highDose_AttBeliefs_euphoria
  highDose_AttBeliefs_insight
  highDose_AttBeliefs_connection
  highDose_AttBeliefs_contact
  highDose_AttBeliefs_sex
/CRITERIA =
  FACTORS(1)
/PRINT =
  INITIAL
  EXTRACTION
  UNIVARIATE
  CORRELATION
  REPR
/PLOT =
  EIGEN
```

```

/EXTRACTION =
  ULS
/ROTATION =
  OBLIMIN
.

```

15.5 Output: jamovi

Factor Statistics

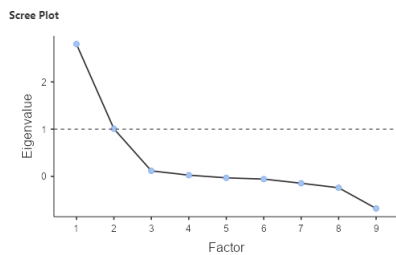
Summary			
Factor	SS Loadings	% of Variance	Cumulative %
1	2.77	30.8	30.8
2	1.62	18.0	48.8

Correlation Matrix		
	1	2
1	—	0.107
2		—

Model Fit

Model Fit Measures					Model Test		
RMSEA	RMSEA 90% CI		TLI	BIC	χ^2	df	p
	Lower	Upper					
0.0649	0.0312	0.0973	0.949	-65.7	36.1	19	0.010

Eigenvalues



15.6 Output: R

15.6.1 R: rosetta

15.6.1.1 Exploratory Factor Analysis (EFA)

Factor loadings

Factor 1

Table 15.1: Input parameters

Extraction method:	Minimum Residuals
Rotation:	Oblimin rotation
Sample size:	213

Factor 2

Uniqueness

Expectation that a high dose results in a longer trip

0.17

0.40

0.80

Expectation that a high dose results in a more intense trip

0.08

0.77

0.39

Expectation that a high dose makes you more intoxicated

-0.07

0.89

0.22

Expectation that a high dose provides more energy

0.37

0.14

0.83

Expectation that a high dose produces more euphoria

0.71

0.17

0.44

Expectation that a high dose yields more insight

0.69

-0.07

0.53

Table 15.2: Factor summary

	SS Loadings	% of Variance	Cumulative %
Factor 1	2.77	31	31
Factor 2	1.62	18	49

Table 15.3: Factor correlations

	Factor 1	Factor 2
Factor 1	1.00	0.11
Factor 2	0.11	1.00

Expectation that a high dose strengthens your connection with others

0.86

-0.01

0.26

Expectation that a high dose facilitates making contact with others

0.84

-0.05

0.30

Expectation that a high dose improves sex

0.41

-0.12

0.83

15.7 Output: SPSS

15.8 Read more

- The SPSS FACTOR manual section

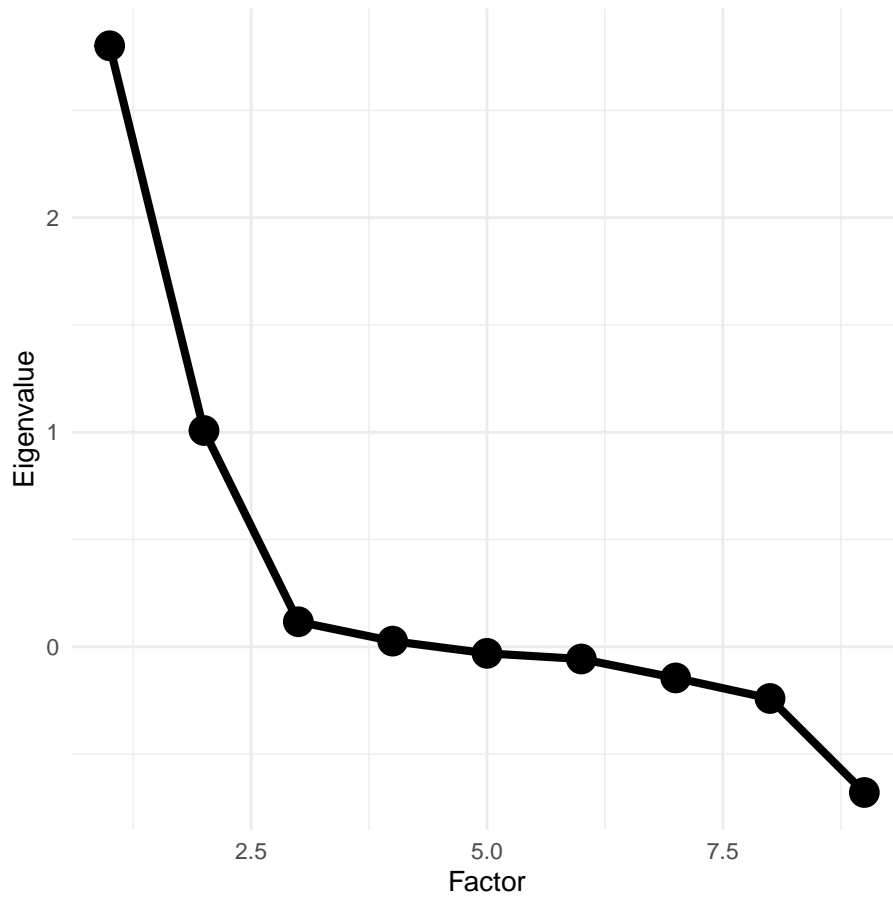


Figure 15.5: Scree plot of the eigen values

Table 15.4: Residuals

	1	2	3	4
1: Expectation that a high dose results in a longer trip	0.80	-0.01	-0.01	0.00
2: Expectation that a high dose results in a more intense trip	-0.01	0.39	0.01	-0.00
3: Expectation that a high dose makes you more intoxicated	-0.01	0.01	0.22	0.00
4: Expectation that a high dose provides more energy	0.05	-0.05	0.02	0.80
5: Expectation that a high dose produces more euphoria	0.02	0.02	-0.03	0.00
6: Expectation that a high dose yields more insight	0.05	-0.06	0.02	0.00
7: Expectation that a high dose strengthens your connection with others	-0.01	0.02	-0.01	-0.00
8: Expectation that a high dose facilitates making contact with others	-0.06	0.03	0.01	0.00
9: Expectation that a high dose improves sex	-0.03	0.03	-0.01	-0.00

Factor Analysis

Communalities		
	Initial	Extraction
highDose_AttBeliefs_iong	.188	.201
highDose_AttBeliefs_intensity	.530	.613
highDose_AttBeliefs_intoxicated	.520	.781
highDose_AttBeliefs_energy	.167	.165
highDose_AttBeliefs_euphoria	.480	.557
highDose_AttBeliefs_insight	.419	.468
highDose_AttBeliefs_connection	.673	.741
highDose_AttBeliefs_contact	.645	.696
highDose_AttBeliefs_sex	.164	.170

Extraction Method: Unweighted Least Squares.

Total Variance Explained

Factor	Initial Eigenvalues			Extraction Sums of Squared Loadings			Rotation Sums of Squared Loadings ^a
	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %	
1	3.282	36.464	36.464	2.844	31.600	31.600	2.783
2	1.922	21.353	57.817	1.547	17.191	48.790	1.740
3	.876	9.738	67.555				
4	.754	8.375	75.930				
5	.741	8.233	84.164				
6	.500	5.554	89.717				
7	.443	4.927	94.645				
8	.274	3.042	97.687				
9	.208	2.313	100.000				

Extraction Method: Unweighted Least Squares.

a. When factors are correlated, sums of squared loadings cannot be added to obtain a total variance.

Factor Matrix^a

	Factor	
	1	2
highDose_AttBeliefs_iong	.285	.346
highDose_AttBeliefs_intensity	.301	.723
highDose_AttBeliefs_intoxicated	.187	.864
highDose_AttBeliefs_energy	.401	.067
highDose_AttBeliefs_euphoria	.746	.023
highDose_AttBeliefs_insight	.655	-.198
highDose_AttBeliefs_connection	.841	-.181
highDose_AttBeliefs_contact	.808	-.209
highDose_AttBeliefs_sex	.362	-.196

Extraction Method: Unweighted Least Squares.

a. 2 factors extracted. 4 iterations required.

Figure 15.6: The output of the factor analysis in SPSS part 1

Pattern Matrix^a

	Factor	
	1	2
highDose_AttBeliefs_lon g	.132	.405
highDose_AttBeliefs_inte nsity	-.001	.783
highDose_AttBeliefs_into xicated	-.163	.898
highDose_AttBeliefs_ene rgy	.350	.154
highDose_AttBeliefs_eup horia	.690	.187
highDose_AttBeliefs_insi ght	.691	-.052
highDose_AttBeliefs_con nection	.860	.006
highDose_AttBeliefs_con tact	.839	-.029
highDose_AttBeliefs_sex	.416	-.115

Extraction Method: Unweighted Least Squares.
Rotation Method: Oblimin with Kaiser Normalization.

a. Rotation converged in 4 iterations.

Structure Matrix

	Factor	
	1	2
highDose_AttBeliefs_lon g	.203	.429
highDose_AttBeliefs_inte nsity	.137	.783
highDose_AttBeliefs_into xicated	-.005	.869
highDose_AttBeliefs_ene rgy	.377	.216
highDose_AttBeliefs_eup horia	.723	.309
highDose_AttBeliefs_insi ght	.682	.070
highDose_AttBeliefs_con nection	.861	.157
highDose_AttBeliefs_con tact	.834	.119
highDose_AttBeliefs_sex	.396	-.042

Extraction Method: Unweighted Least Squares.
Rotation Method: Oblimin with Kaiser Normalization.

Factor Correlation Matrix

Factor	1	2
1	1.000	.176
2	.176	1.000

Extraction Method:
Unweighted Least Squares.
Rotation Method: Oblimin
with Kaiser Normalization.

Figure 15.7: The output of the factor analysis in SPSS part 2

Chapter 16

Omega

16.1 Intro

Omega is considered a measure of internal consistency and can, if a number of assumptions are met, estimate the reliability of a set of items.

16.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

16.1.2 Variable(s)

From this dataset, this example uses variables `highDose_AttGeneral_good`, `highDose_AttGeneral_prettig`, `highDose_AttGeneral_slim`, `highDose_AttGeneral_gezond` & `highDose_AttGeneral_spannend`.

16.2 Input: jamovi

In the “Analyses” tab, click the “Factor” button and from the menu that appear, select “Reliability Analysis” as shown in Figure 16.1.

In the box at the left, select all variables you want to include in this analysis and move them to the box labelled “Items” using the button labelled with the rightward-pointing arrow as shown in Figure 16.2.

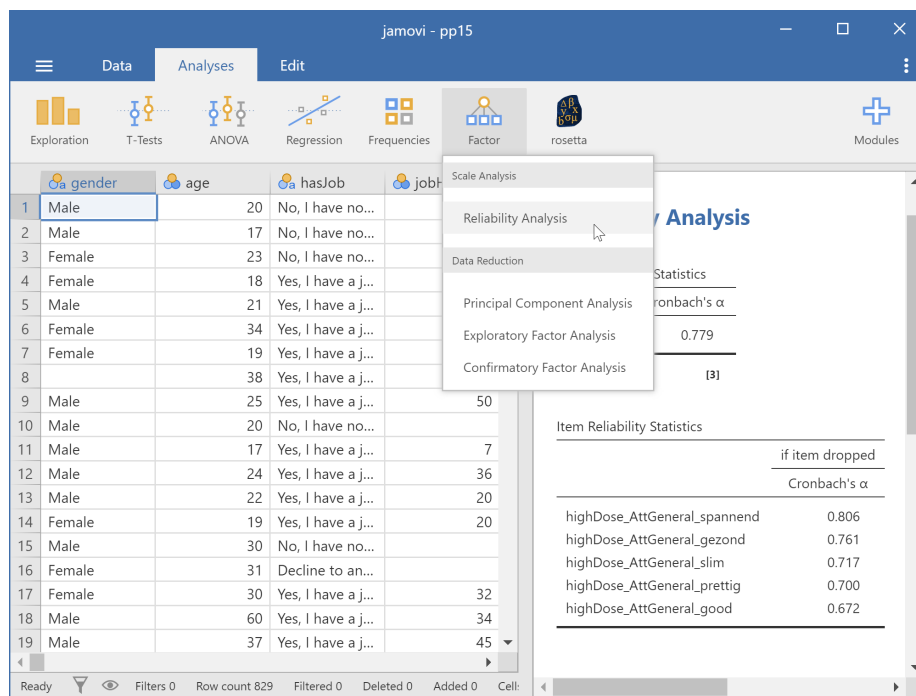


Figure 16.1: Opening the reliability analysis menu in jamovi

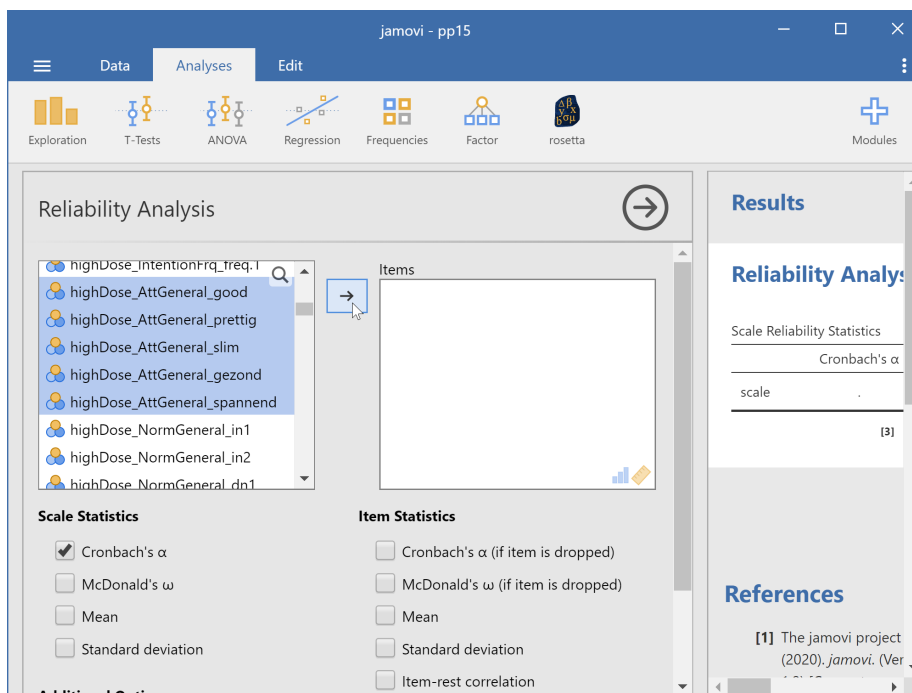


Figure 16.2: Adding the items to analyse in jamovi

Select the checkbox labelled “Omega” in the left-most column at the bottom labelled “Scale Statistics”, as shown in Figure 16.3.

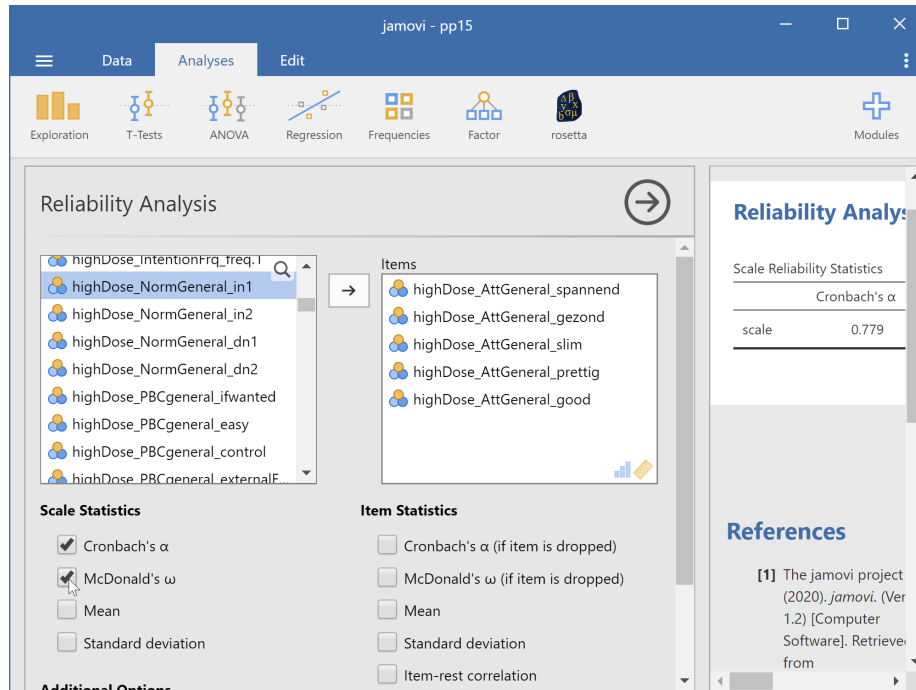


Figure 16.3: Ordering “Selecting Omega in jamovi

You can now also order additional statistics, such as the value Omega would have if you were to omit each item, as shown in Figure 16.4.

16.3 Input: R

```
rosetta::reliability(
  data = dat,
  items = c(
    "highDose_AttGeneral_good",
    "highDose_AttGeneral_prettig",
    "highDose_AttGeneral_slim",
    "highDose_AttGeneral_gezond",
    "highDose_AttGeneral_spannend"
  )
);
```

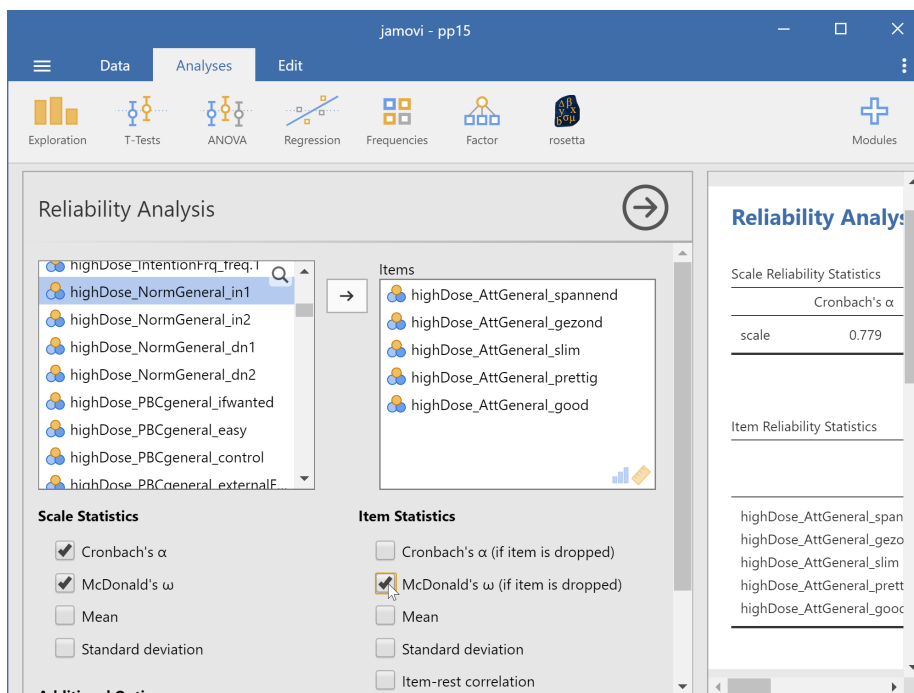



Figure 16.4: Ordering “Omega if item dropped” in jamovi

16.4 Input: SPSS

SPSS is unable to compute Omega.

16.5 Output: jamovi

Reliability Analysis

Scale Reliability Statistics		
	Cronbach's α	McDonald's ω
scale	0.779	0.794

[3]

Item Reliability Statistics	
	if item dropped
	McDonald's ω
highDose_AttGeneral_spannend	0.824
highDose_AttGeneral_gezond	0.776
highDose_AttGeneral_slim	0.734
highDose_AttGeneral_prettig	0.746
highDose_AttGeneral_good	0.718

Figure 16.5: The output of a reliability analysis where Omega is computed in jamovi

16.6 Output: R

16.6.1 R: rosetta

16.6.1.1 Reliability analysis

16.6.1.1.1 Scale structure

16.6.1.1.1.1 Scale structure Information about this scale

Dataframe:	res\$data
Items:	highDose_AttGeneral_good, highDose_AttGeneral_prettig, highDose_AttGer
Observations:	303
Positive correlations:	10
Number of correlations:	10
Percentage positive correlations:	100

Estimates assuming interval level

Omega (total):	0.79
Omega (hierarchical):	0.80
Revelle's Omega (total):	0.79
Greatest Lower Bound (GLB):	0.87
Coefficient H:	0.85
Coefficient Alpha:	0.78

Note: the normal point estimate and confidence interval for omega are based on the procedure suggested by Dunn, Baguley & Brunsten (2013) using the MBESS function `ci.reliability`, whereas the psych package point estimate was suggested in Revelle & Zinbarg (2008). See the help (`?ufs::scaleStructure`) for more information.

16.7 Output: SPSS

SPSS is unable to compute Omega.

Chapter 17

Reliability analysis

17.1 Intro

Many statistical packages combine multiple specific statistics in one command or interface element often called “reliability analysis”. This chapter describes how to conduct that analysis.

17.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

17.1.2 Variable(s)

From this dataset, this example uses variables `highDose_AttGeneral_good`, `highDose_AttGeneral_prettig`, `highDose_AttGeneral_slim`, `highDose_AttGeneral_gezond` & `highDose_AttGeneral_spannend`.

17.2 Input: jamovi

In the “Analyses” tab, click the “Factor” button and from the menu that appear, select “Reliability Analysis” as shown in Figure 17.1.

In the box at the left, select all variables you want to include in this analysis and move them to the box labelled “Items” using the button labelled with the rightward-pointing arrow as shown in Figure 17.2.

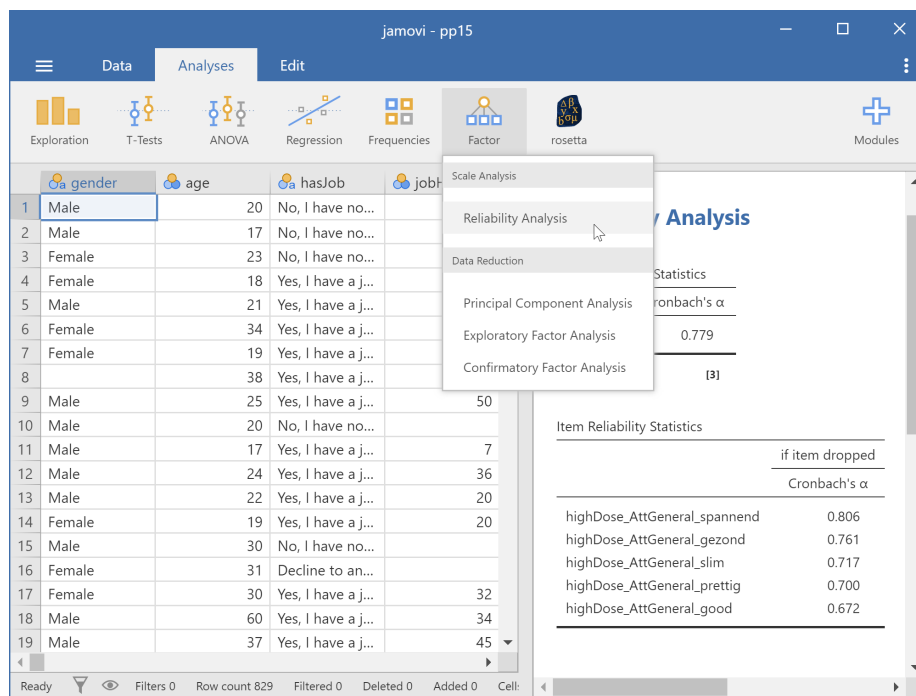


Figure 17.1: Opening the reliability analysis menu in jamovi

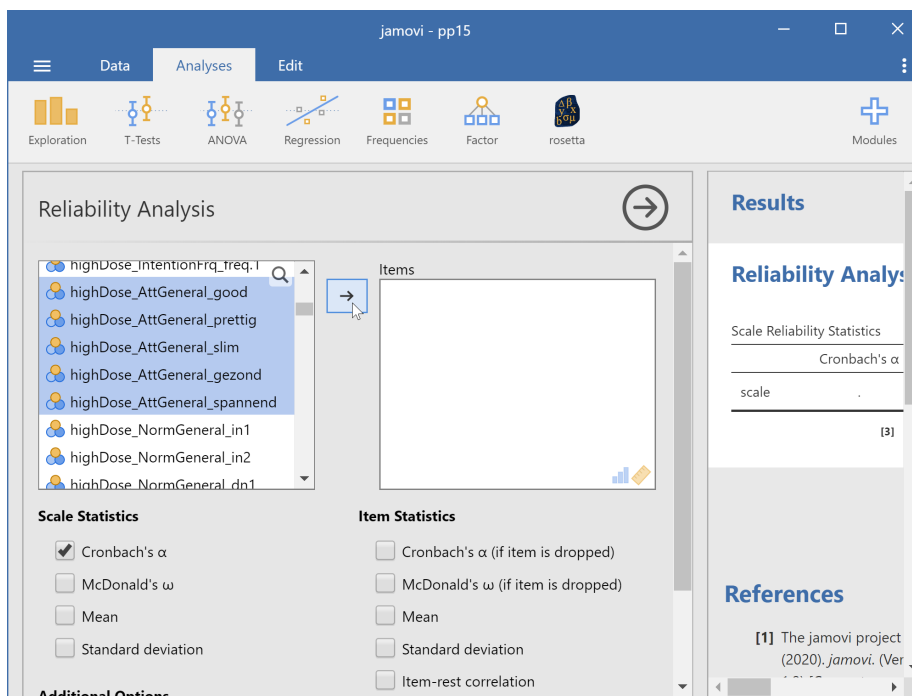


Figure 17.2: Adding the items to analyse in jamovi

You can now select which options you want by checking more checkboxes and indicating other settings in the left-hand panel. You will immediately see the results in the right-hand panel update as shown in Figure 17.3.

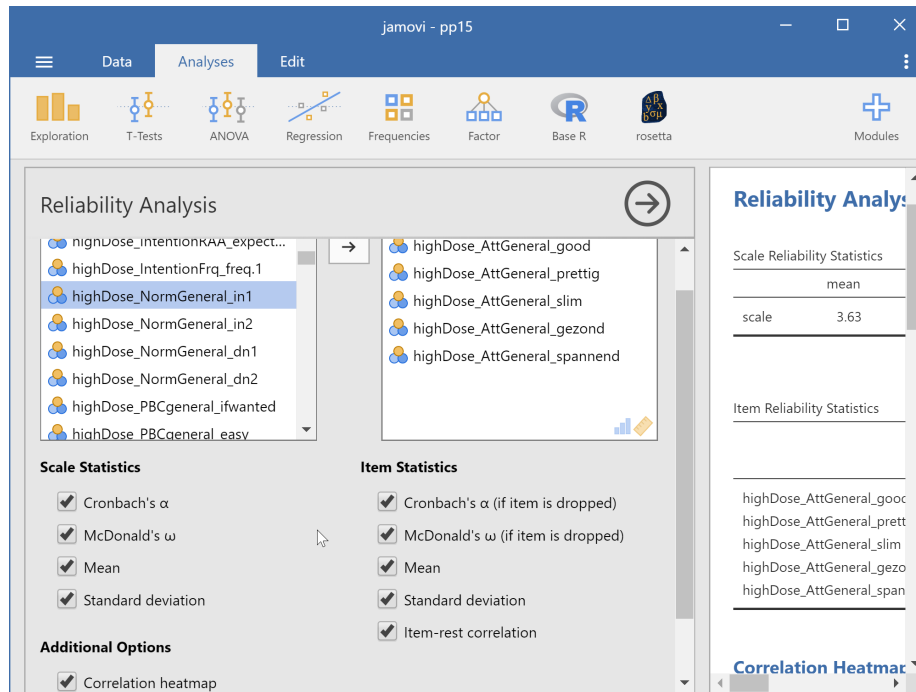


Figure 17.3: Adding the items to analyse in jamovi

17.3 Input: R

17.3.1 R: rosetta

In R, using the `rosetta` package, you can use the following command:

```
rosetta::reliability(
  data = dat,
  items = c(
    "highDose_AttGeneral_good"
    "highDose_AttGeneral_prettig"
    "highDose_AttGeneral_slim"
    "highDose_AttGeneral_gezond"
    "highDose_AttGeneral_spannend"
```



```
)
);
```

To order additional information, such as descriptive statistics, inter-item correlations, and other scale statistics, you can specify additional options. You can also specify item labels to print instead of the variable names:

```
rosetta::reliability(
  data = dat,
  items = c(
    "highDose_AttGeneral_good",
    "highDose_AttGeneral_prettig",
    "highDose_AttGeneral_slim",
    "highDose_AttGeneral_gezond",
    "highDose_AttGeneral_spannend"
  ),
  itemLabels = c(
    "Attitude: good",
    "Attitude: pleasant",
    "Attitude: smart",
    "Attitude: healthy",
    "Attitude: exciting"
  ),
  descriptives = TRUE,
  itemLevel = TRUE,
  scatterMatrix = TRUE,
  itemOmittedCorsWithRest = TRUE,
  alphaOmittedCIs = TRUE
);
```

17.4 Input: SPSS

In SPSS, you can use the following command:

```
RELIABILITY
/VARIABLES =
  highDose_AttGeneral_good
  highDose_AttGeneral_prettig
  highDose_AttGeneral_slim
  highDose_AttGeneral_gezond
  highDose_AttGeneral_spannend
/MODEL = ALPHA.
```

To order additional information, such as descriptive statistics, inter-item correlations, and other scale statistics, you can specify additional options:

```
RELIABILITY
/VARIABLES =
  highDose_AttGeneral_good
  highDose_AttGeneral_prettig
  highDose_AttGeneral_slim
  highDose_AttGeneral_gezond
  highDose_AttGeneral_spannend
/MODEL = ALPHA
/STATISTICS = DESCRIPTIVE SCALE CORR
/SUMMARY = TOTAL.
```

17.5 Output: jamovi

Reliability Analysis

Scale Reliability Statistics

	mean	sd	Cronbach's α	McDonald's ω
scale	3.63	1.07	0.779	0.794

[3]

Item Reliability Statistics

	mean	sd	item-rest correlation	if item dropped	
				Cronbach's α	McDonald's ω
highDose_AttGeneral_good	3.69	1.71	0.722	0.672	0.718
highDose_AttGeneral_prettig	4.15	1.84	0.665	0.700	0.746
highDose_AttGeneral_slim	2.94	1.23	0.639	0.717	0.734
highDose_AttGeneral_gezond	2.51	1.13	0.487	0.761	0.776
highDose_AttGeneral_spannend	4.87	1.31	0.317	0.806	0.824

Correlation Heatmap

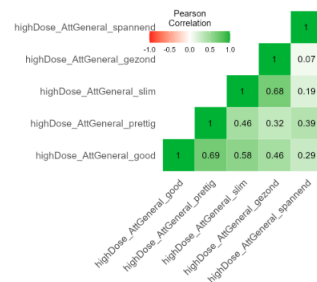


Figure 17.4: The output of a reliability analysis in jamovi

17.6 Output: R

17.6.1 Reliability analysis

17.6.1.1 Scale structure

17.6.1.1.1 Scale structure

17.6.1.1.1.1 Information about this scale

Dataframe:	res\$data
Items:	highDose_AttGeneral_good, I
Observations:	303
Positive correlations:	10
Number of correlations:	10
Percentage positive correlations:	100

17.6.1.1.1.2 Estimates assuming interval level

Omega (total):	0.79
Omega (hierarchical):	0.80
Revelle's Omega (total):	0.79
Greatest Lower Bound (GLB):	0.87
Coefficient H:	0.85
Coefficient Alpha:	0.78

Note: the normal point estimate and confidence interval for omega are based on the procedure suggested by Dunn, Baguley & Brunsten (2013) using the MBESS function `ci.reliability`, whereas the psych package point estimate was suggested in Revelle & Zinbarg (2008). See the help (`?ufs::scaleStructure`) for more information.

17.6.1.2 Scale descriptives

Mean, 95% CI lower bound:

Mean, pointestimate:

Mean, 95% CI upper bound:

SD, 95% CI lower bound:

SD, pointestimate:

SD, 95% CI upper bound:

scale

3.51

3.63

3.75

0.99

1.07

1.16

17.6.1.3 Item-level descriptives

Mean, 95% CI lower bound:

Mean, pointestimate:

Mean, 95% CI upper bound:

SD, 95% CI lower bound:

SD, pointestimate:

SD, 95% CI upper bound:

Attitude: good

3.49

3.69

3.88

1.58

1.71

1.85

Attitude: pleasant

3.94

4.15

4.35

1.71

1.84

2.00

Attitude: smart

2.80

2.94

3.08

1.14

1.23

1.34

Attitude: healthy

2.38

2.51

2.64

1.04

1.13

1.22

Attitude: exciting

4.72

4.87

5.02

1.21

1.31

1.42

17.6.1.4 Correlations of items with scale

Item-rest correlations:95% CI lower bound

Item-rest correlations:point estimate

Item-rest correlations:95% CI upper bound

Attitude: good

0.66

0.72

0.77

Attitude: pleasant

0.60

0.66

0.72

Attitude: smart

0.57

0.64

0.70

Attitude: healthy

0.40

0.49

0.57

Attitude: exciting

0.21

0.32

0.42

17.6.1.5 Internal consistency estimates with items omitted

Coefficient Alpha:95% CI lower bound

Coefficient Alpha:point estimate

Coefficient Alpha:95% CI upper bound

Omega (from psych):point estimate

Attitude: good

0.61

0.67

0.73

0.72

Attitude: pleasant

0.64

0.70

0.75

0.75

Attitude: smart

0.66

0.72

0.76

0.73

Attitude: healthy

0.72

0.76

0.80

0.78

Attitude: exciting

0.77

0.81

0.84

0.82

17.6.1.6 Scatter matrix

17.7 Output: SPSS

Scatter matrix with 95% confidence intervals

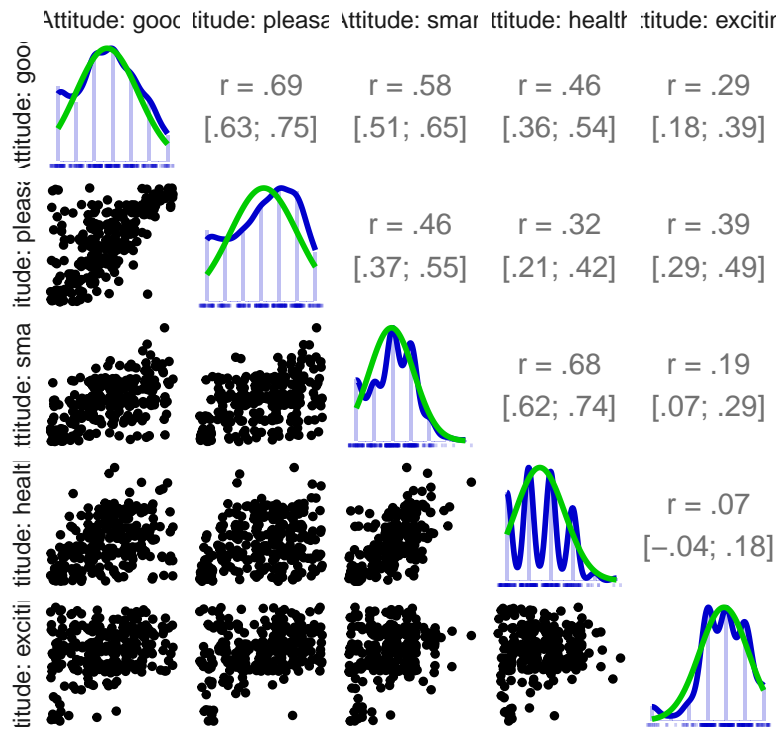


Figure 17.5: Scatter matrix of all items.

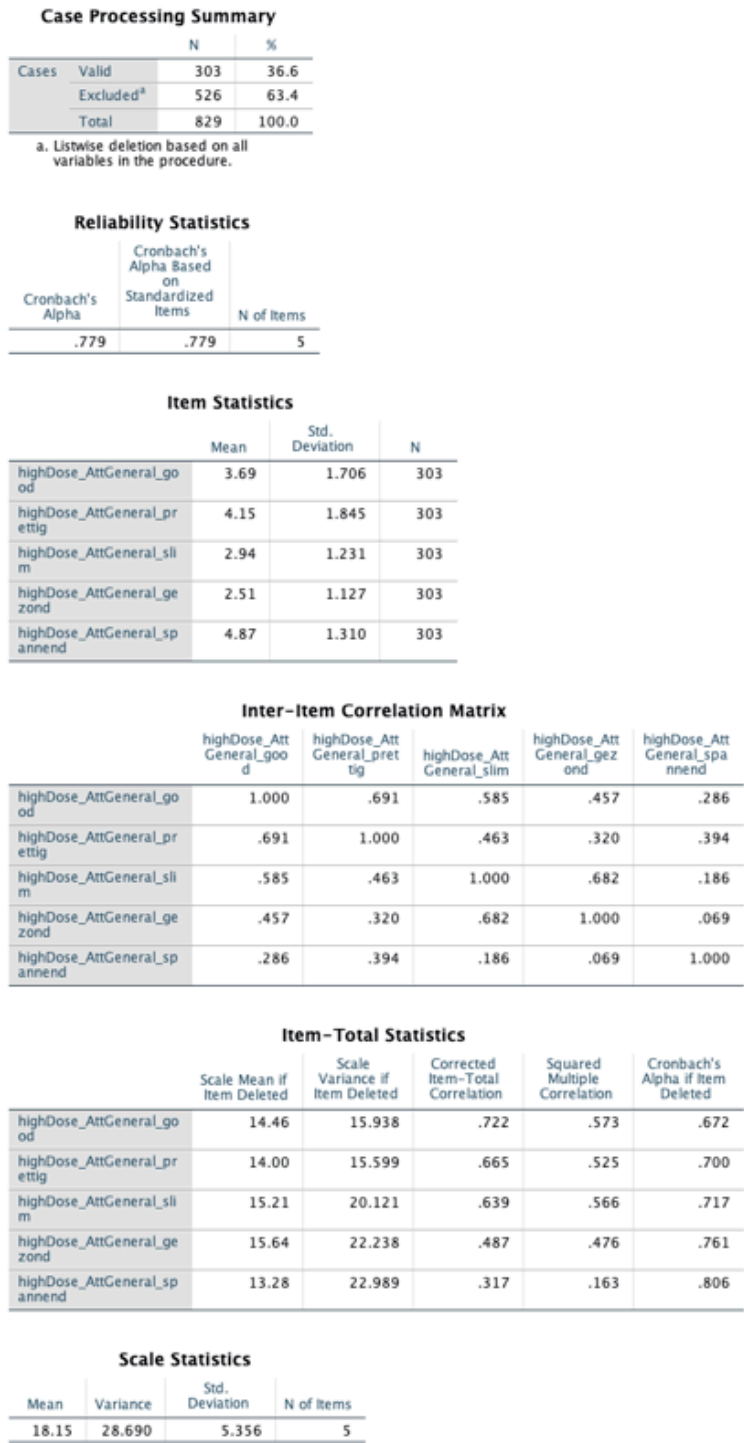


Figure 17.6: The output of a reliability analysis in SPSS

Part IV

Bivariate analyses

Chapter 18

Anova

18.1 jamovi

The jamovi section has not yet been added.

18.2 R

The R section has not yet been added.

18.3 SPSS

The SPSS section has not yet been added.

Chapter 19

Chi squared

19.1 jamovi

The jamovi section has not yet been added.

19.2 R

The R section has not yet been added.

19.3 SPSS

The SPSS section has not yet been added.

Chapter 20

Cohen's d

20.1 jamovi

The jamovi section has not yet been added.

20.2 R

The R section has not yet been added.

20.3 SPSS

The SPSS section has not yet been added.

Chapter 21

Correlation

21.1 Intro

A correlation coefficient is an estimation of the degree to which two variables have a linear association, and is the square root of their mutual proportions of explained variance.

21.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

21.1.2 Variable(s)

From this dataset, this example uses variables `highDose_AttDesirable_long`, `highDose_AttDesirable_intens`, `highDose_AttDesirable_intoxicated`, `highDose_AttDesirable_energy`, and `highDose_AttDesirable_euphoria`; these are a number of expressions of which effects people prefer when using MDMA (see Chapter 1).

21.2 Input: jamovi

In jamovi, use the ‘Regression’ menu, choose ‘Correlation matrix’, and select the variables you want to include.

You can check the checkbox for confidence intervals to order confidence intervals.

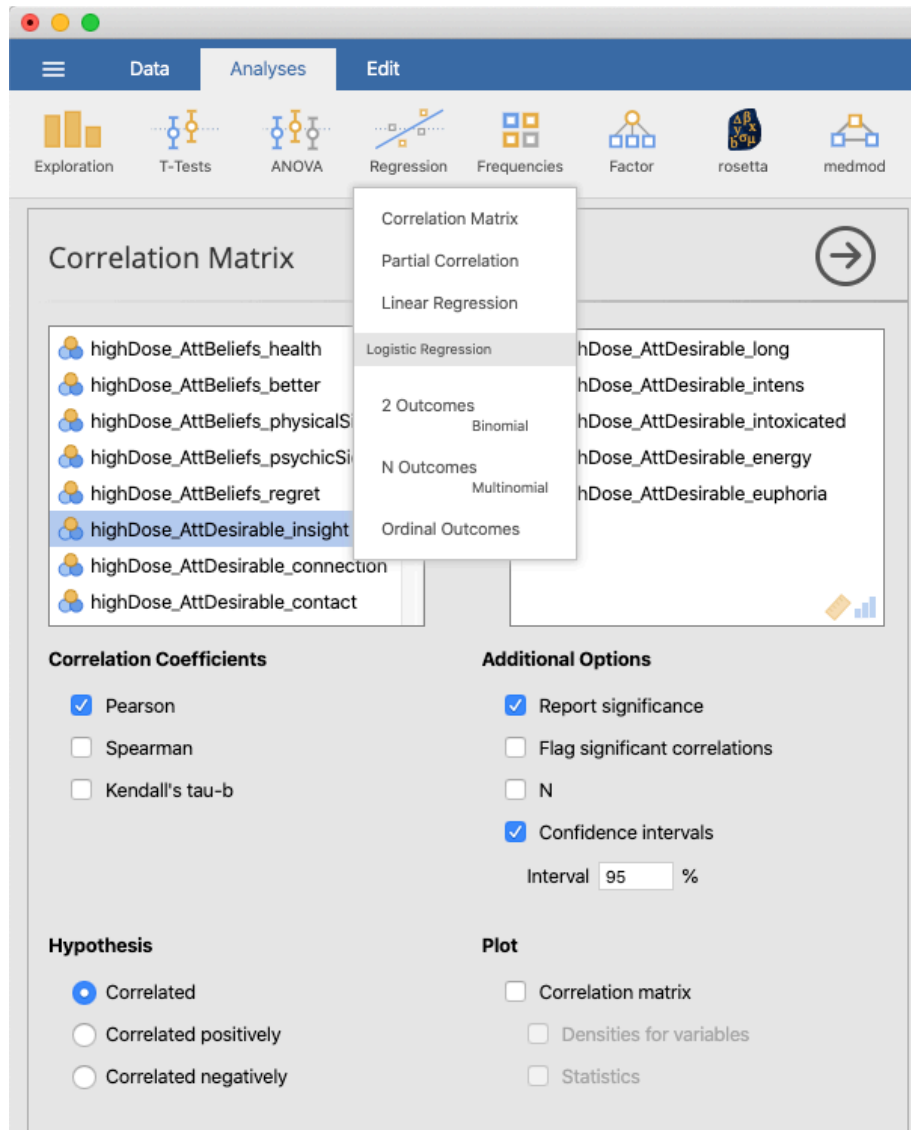


Figure 21.1: Opening the “Regression” menu in jamovi

21.3 Input: R

Many analyses can be done with base R without installing additional packages. The `rosetta` package accompanies this book and aims to provide output similar to jamovi and SPSS with simple commands.

21.3.1 R: base R

A basic correlation matrix can be produced with `cor()`, passing argument `use="complete.obs"` if there are missing values in the dataset (otherwise, missing values result in a correlation estimate that it also missing).

```
cor(  
  dat[  
    ,  
    c(  
      'highDose_AttDesirable_long',  
      'highDose_AttDesirable_intens',  
      'highDose_AttDesirable_intoxicated',  
      'highDose_AttDesirable_energy',  
      'highDose_AttDesirable_euphoria'  
    )  
  ],  
  use = "complete.obs"  
);
```

To obtain confidence intervals for a correlation, `cor.test()` can be used. However, this function only works for one correlation.

```
cor.test(  
  dat$highDose_AttDesirable_long,  
  dat$highDose_AttDesirable_intens  
);
```

21.3.2 R: rosetta (ufs)

A correlation matrix function has not yet been made available in the `rosetta` package, but it *is* available in the `ufs` package that comes installed with `rosetta`. Therefore, if you have `rosetta` installed, you can use the following command.

```
ufs::associationMatrix(  
  dat,  
  x = c(  
    'highDose_AttDesirable_long',  
    'highDose_AttDesirable_intens',  
    'highDose_AttDesirable_intoxicated',  
    'highDose_AttDesirable_energy',  
    'highDose_AttDesirable_euphoria'  
  )  
);
```

This function provides the confidence intervals (the confidence level, by default 95%, can be set with argument `conf.level`) as well as the point estimates and associated p -values. The p -values are corrected for multiple testing (using the false detection rate approach by default; this can be set using the `correction` argument; for example, pass `correction="none"` to not correct the p -values), and sample sizes are printed as well if they differ for each comparison (and omitted if they are the same for all correlation coefficients).

21.4 Input: SPSS

For SPSS, there are two approaches: using the Graphical User Interface (GUI) or specify an analysis script, which in SPSS are called “syntax”.

21.4.1 SPSS: GUI

Click the “Analyze” menu, then select the “Correlate” submenu, and then select “Bivariate”. Then specify the variables you’re interested in.

21.4.2 SPSS: Syntax

```
CORRELATIONS  
  /VARIABLES =  
    highDose_AttDesirable_long  
    highDose_AttDesirable_intens  
    highDose_AttDesirable_intoxicated  
    highDose_AttDesirable_energy  
    highDose_AttDesirable_euphoria  
  .
```

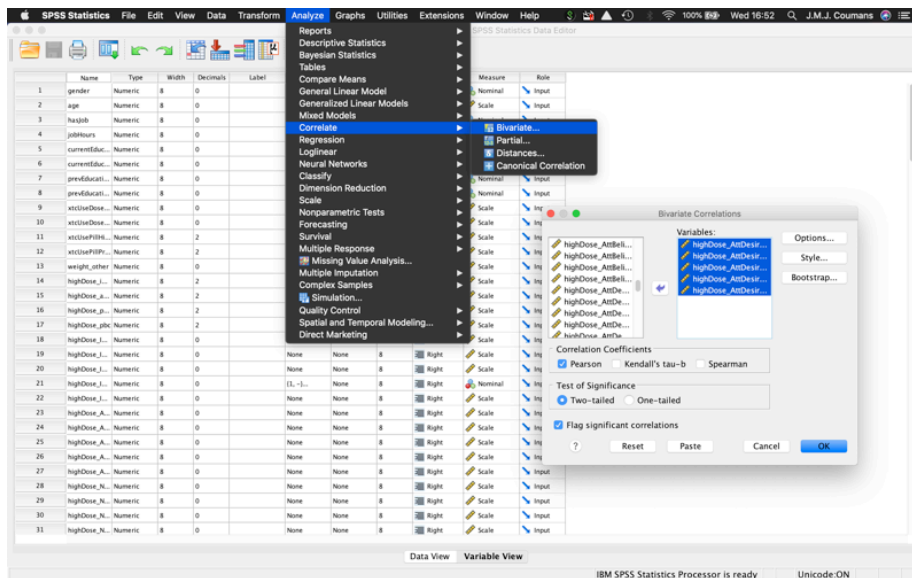


Figure 21.2: Opening the “bivariate” submenu in SPSS

21.5 Output: jamovi

21.6 Output: R

21.6.1 R: base

A correlation matrix (note: the variable names have been manually shortened, and the resulting correlations have been rounded to four decimal places, to make this example fit in the book):

```

      long intens intoxi energy euphor
long  1.0000 0.5724 0.3737 0.3885 0.4663
intens 0.5724 1.0000 0.5843 0.3476 0.3441
intoxi 0.3737 0.5843 1.0000 0.3519 0.1474
energy 0.3885 0.3476 0.3519 1.0000 0.4772
euphor 0.4663 0.3441 0.1474 0.4772 1.0000

```

The results of `cor.test()` including the confidence interval:

```
Pearson's product-moment correlation
```

Correlation Matrix

Correlation Matrix		highDose_AttDesirable_long	highDose_AttDesirable_intens	highDose_AttDesirable_intoxicated	highDose_AttDesirable_energy	highDose_AttDesirable_euphoria
highDose_AttDesirable_long	Pearson's r	--				
	p-value	--				
	95% CI Upper	--				
	95% CI Lower	--				
highDose_AttDesirable_intens	Pearson's r	0.572	--			
	p-value	<.001	--			
	95% CI Upper	0.657	--			
	95% CI Lower	0.474	--			
highDose_AttDesirable_intoxicated	Pearson's r	0.374	0.584	--		
	p-value	<.001	<.001	--		
	95% CI Upper	0.485	0.667	--		
	95% CI Lower	0.251	0.488	--		
highDose_AttDesirable_energy	Pearson's r	0.389	0.348	0.352	--	
	p-value	<.001	<.001	<.001	--	
	95% CI Upper	0.498	0.461	0.465	--	
	95% CI Lower	0.267	0.223	0.227	--	
highDose_AttDesirable_euphoria	Pearson's r	0.466	0.344	0.147	0.477	--
	p-value	<.001	<.001	0.033	<.001	--
	95% CI Upper	0.566	0.458	0.277	0.575	--
	95% CI Lower	0.353	0.219	0.012	0.365	--

Figure 21.3: The produced correlation matrix in jamovi

```
data: dat$highDose_AttDesirable_long and dat$highDose_AttDesirable_intens
t = 10.068, df = 208, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.4737307 0.6568901
sample estimates:
      cor
0.5724077
```

21.6.2 R: rosetta (ufs)

Note: the variable names in the first column have been adjusted to make the table fit, and make the labels consistent with those in Chapter 22.

	1.	2.	3.
Prefer long effects			
Prefer intense effects	r=[0.47; 0.66], r=0.57, p<.001		
Prefer more intoxication	r=[0.25; 0.48], r=0.37, p<.001	r=[0.49; 0.67], r=0.58, p<.001	
Prefer more energy	r=[0.27; 0.5], r=0.39, p<.001	r=[0.22; 0.46], r=0.35, p<.001	r=[0.23; 0.57], r=0.40, p<.001
Prefer more euphoria	r=[0.35; 0.57], r=0.47, p<.001	r=[0.22; 0.46], r=0.34, p<.001	r=[0.01; 0.66], r=0.34, p<.001

21.7 Output: SPSS

21.8 Read more

If you would like more background on this topic, you can read more in these sources:

		Correlations				
		highDose_Att Desirable_lo ng	highDose_Att Desirable_int ens	highDose_Att Desirable_int oxicated	highDose_Att Desirable_en ergy	highDose_Att Desirable_eu phoria
highDose_AttDesirable_lo ng	Pearson Correlation	1	.572**	.374**	.389**	.466**
	Sig. (2-tailed)		.000	.000	.000	.000
	N	210	210	210	210	210
highDose_AttDesirable_int ens	Pearson Correlation	.572**	1	.584**	.348**	.344**
	Sig. (2-tailed)	.000		.000	.000	.000
	N	210	210	210	210	210
highDose_AttDesirable_int oxicated	Pearson Correlation	.374**	.584**	1	.352**	.147*
	Sig. (2-tailed)	.000	.000		.000	.033
	N	210	210	210	210	210
highDose_AttDesirable_en ergy	Pearson Correlation	.389**	.348**	.352**	1	.477**
	Sig. (2-tailed)	.000	.000	.000		.000
	N	210	210	210	210	210
highDose_AttDesirable_eu phoria	Pearson Correlation	.466**	.344**	.147*	.477**	1
	Sig. (2-tailed)	.000	.000	.033	.000	
	N	210	210	210	210	210

** . Correlation is significant at the 0.01 level (2-tailed).

* . Correlation is significant at the 0.05 level (2-tailed).

Figure 21.4: The output produced in SPSS

- More options for creating scattermatrices in R are available here: <http://www.sthda.com/english/wiki/scatter-plot-matrices-r-base-graphs>

Chapter 22

Scattermatrix

22.1 Intro

A scattermatrix combines multiple scatterplots to allow quick visual inspection of bivariate associations between variables.

22.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

22.1.2 Variable(s)

From this dataset, this example uses variables `highDose_AttDesirable_long`, `highDose_AttDesirable_intens`, `highDose_AttDesirable_intoxicated`, `highDose_AttDesirable_energy`, and `highDose_AttDesirable_euphoria`; these are a number of expressions of which effects people prefer when using MDMA (see Chapter 1).

22.2 Input: jamovi

In jamovi, use the ‘Regression’ menu, choose ‘Correlation matrix’, and select the variables you want to include. Then, check the checkbox at ‘Correlation Matrix’.

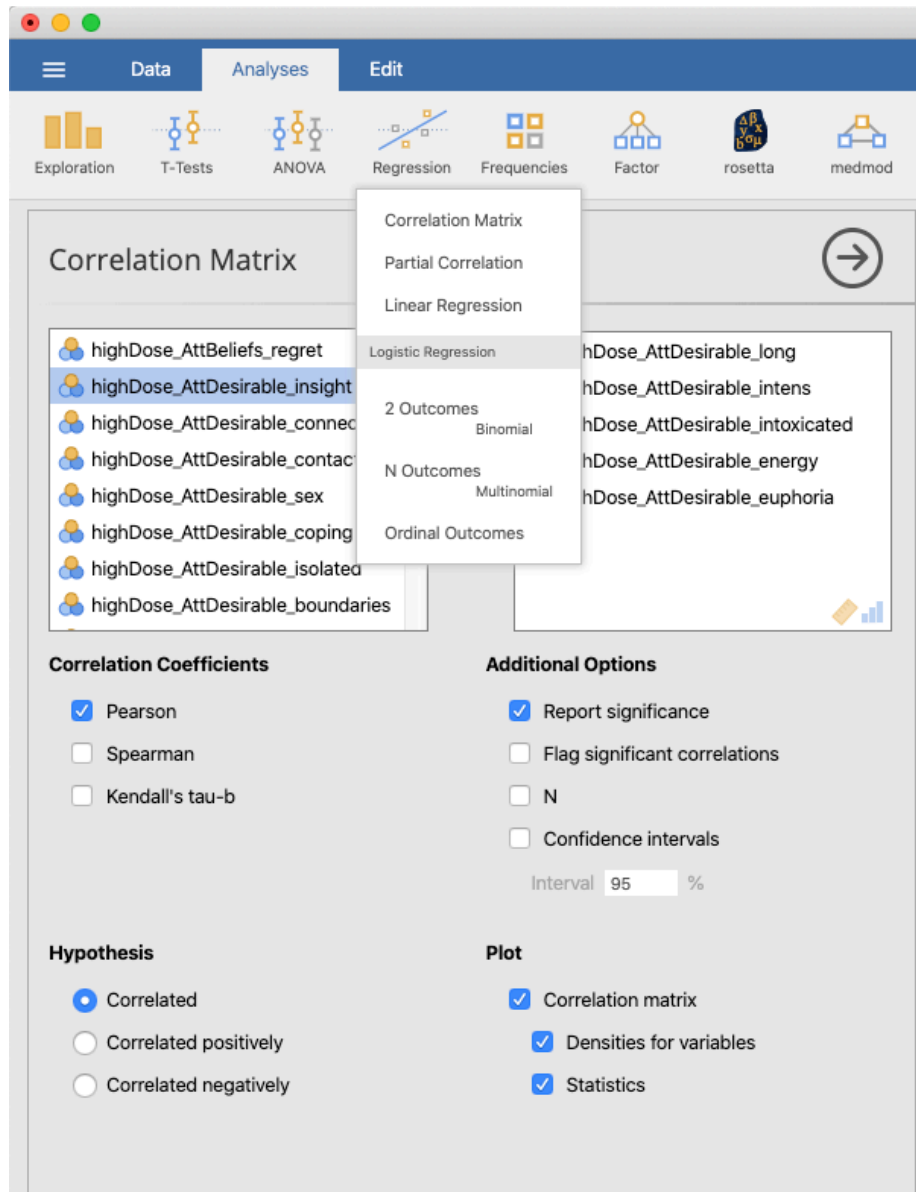


Figure 22.1: Opening the “Regression” menu in jamovi

You can also check the checkboxes at ‘Densities for variables’ and ‘Statistics’ to also include the densities on the diagonal and the point estimates in the upper triangle.

22.3 Input: R

Many analyses can be done with base R without installing additional packages. The `rosetta` package accompanies this book and aims to provide output similar to jamovi and SPSS with simple commands.

22.3.1 R: base R

A basic scattermatrix can be produced with `pairs()`, specifying the used symbol by passing a number as argument `pch` (the default is, an open circle; see this R manual page for the full list).

```
pairs(  
  dat[  
    ,  
    c(  
      'highDose_AttDesirable_long',  
      'highDose_AttDesirable_intens',  
      'highDose_AttDesirable_intoxicated',  
      'highDose_AttDesirable_energy',  
      'highDose_AttDesirable_euphoria'  
    )  
  ],  
  pch = 19  
);
```

Often, points end up overlapping, like in this case. In that situation, the data can be ‘jittered’. In this example, we will do this manually, first, and then use the jittered data in the call to `pairs()` (as opposed to integrating both commands into one, more efficient, but more complex statement).

```
jitteredDat <-  
  sapply(  
    dat[  
      ,  
      c(  
        'highDose_AttDesirable_long',  
        'highDose_AttDesirable_intens',  
        'highDose_AttDesirable_intoxicated',  
        'highDose_AttDesirable_energy',  
        'highDose_AttDesirable_euphoria'  
      )  
    ],  
    function(x) jitter(x, amount = 0.1))
```

```
      'highDose_AttDesirable_long',
      'highDose_AttDesirable_intens',
      'highDose_AttDesirable_intoxicated',
      'highDose_AttDesirable_energy',
      'highDose_AttDesirable_euphoria'
    )
  ],
  jitter,
  amount = .5
);

pairs(
  jitteredDat,
  pch = 19
);
```

22.3.2 R: rosetta (ufs)

A scatter matrix function has not yet been made available in the `rosetta` package, but it *is* available in the `ufs` package that comes installed with `rosetta`. Therefore, if you have `rosetta` installed, you can use the following command.

```
ufs::scatterMatrix(
  dat,
  items = c(
    'highDose_AttDesirable_long',
    'highDose_AttDesirable_intens',
    'highDose_AttDesirable_intoxicated',
    'highDose_AttDesirable_energy',
    'highDose_AttDesirable_euphoria'
  )
);
```

This function always has jittering turned on, and always produces correlation confidence intervals and point estimates. The `itemLabels` argument can optionally be used to specify “pretty” item labels.



Figure 22.2: A screenshot placeholder

22.3.3 R: tidyverse

22.4 Input: SPSS

For SPSS, there are two approaches: using the Graphical User Interface (GUI) or specify an analysis script, which in SPSS are called “syntax”.

22.4.1 SPSS: GUI

Click the “Graphs” menu; then select the “Legacy Dialogs” submenu, and then select “Scatter/Dot”. Select the “Matrix Scatter” option, and then specify the variables you’re interested in.

Alternatively, Open SPSS’ Chart Builder. Then, in the ‘Chart Types’ section, select the ‘Scatterplot matrix’ chart type. Then you can select the variables you want to include.

22.4.2 SPSS: Syntax

```
GRAPH  
  /SCATTERPLOT(MATRIX)=
```

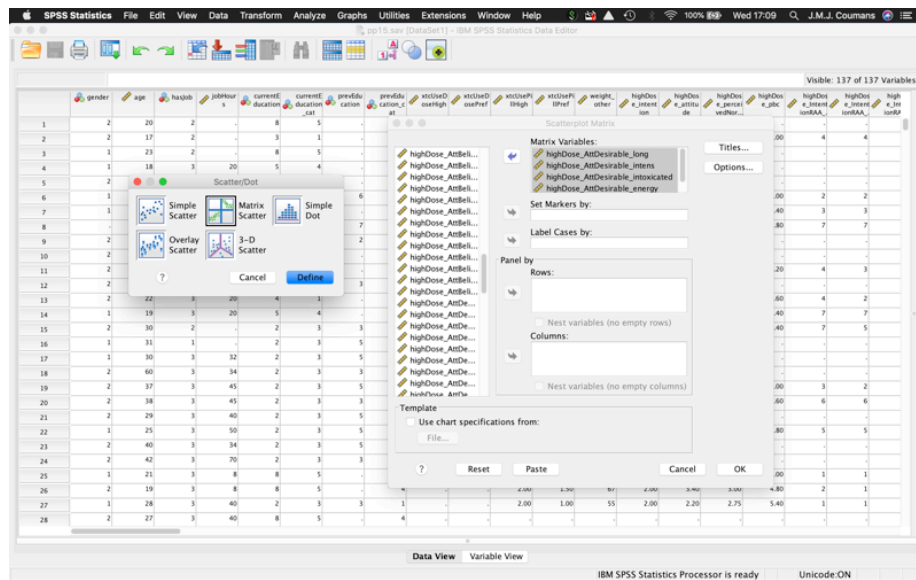


Figure 22.3: Opening the scattermatrix menu

```

highDose_AttDesirable_long
highDose_AttDesirable_intens
highDose_AttDesirable_intoxicated
highDose_AttDesirable_energy
highDose_AttDesirable_euphoria
WITH
highDose_AttDesirable_long
highDose_AttDesirable_intens
highDose_AttDesirable_intoxicated
highDose_AttDesirable_energy
highDose_AttDesirable_euphoria

```

22.5 Output: jamovi

22.6 Output: R

22.6.1 R: base

Without jittering:

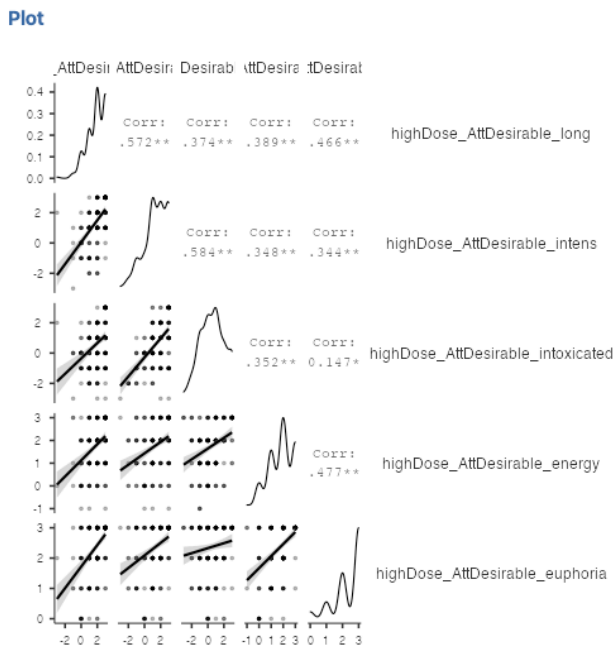
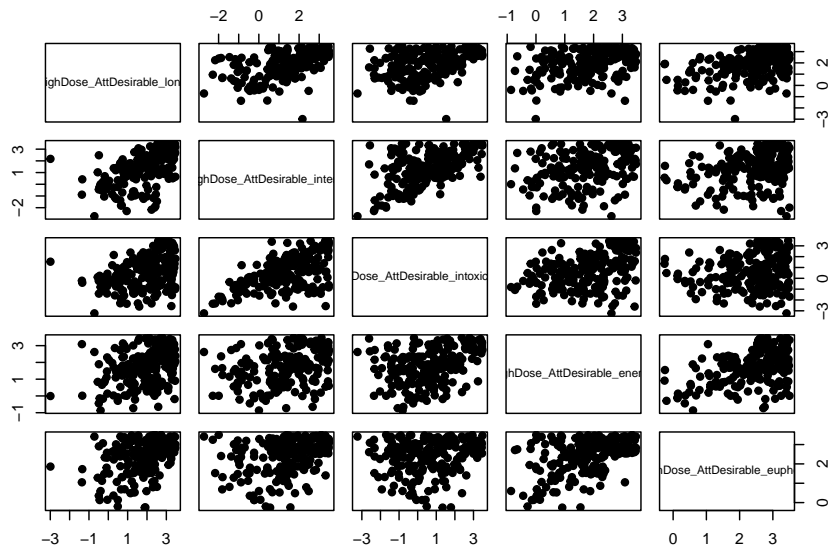
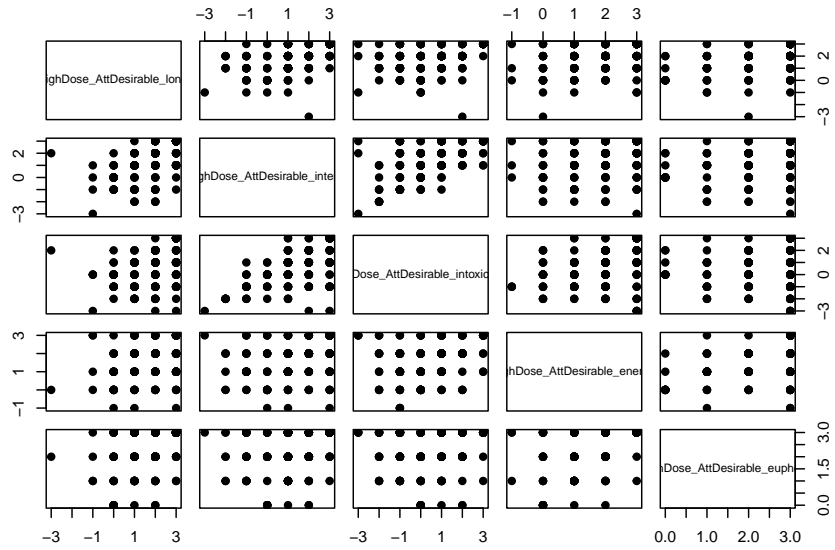


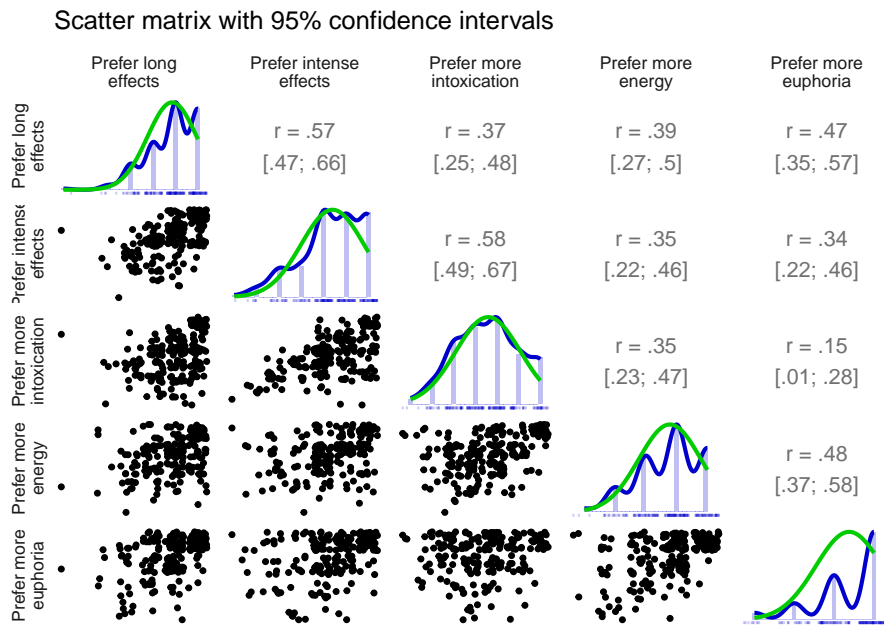
Figure 22.4: The produced scattermatrix in jamovi.



With jittering:

22.6.2 R: rosetta (ufs)

In this example, the `itemLabels` argument was used to specify variable labels.



22.7 Output: SPSS

22.8 Read more

If you would like more background on this topic, you can read more in these sources:

- More options for creating scattermatrices in R are available here: <http://www.sthda.com/english/wiki/scatter-plot-matrices-r-base-graphs>

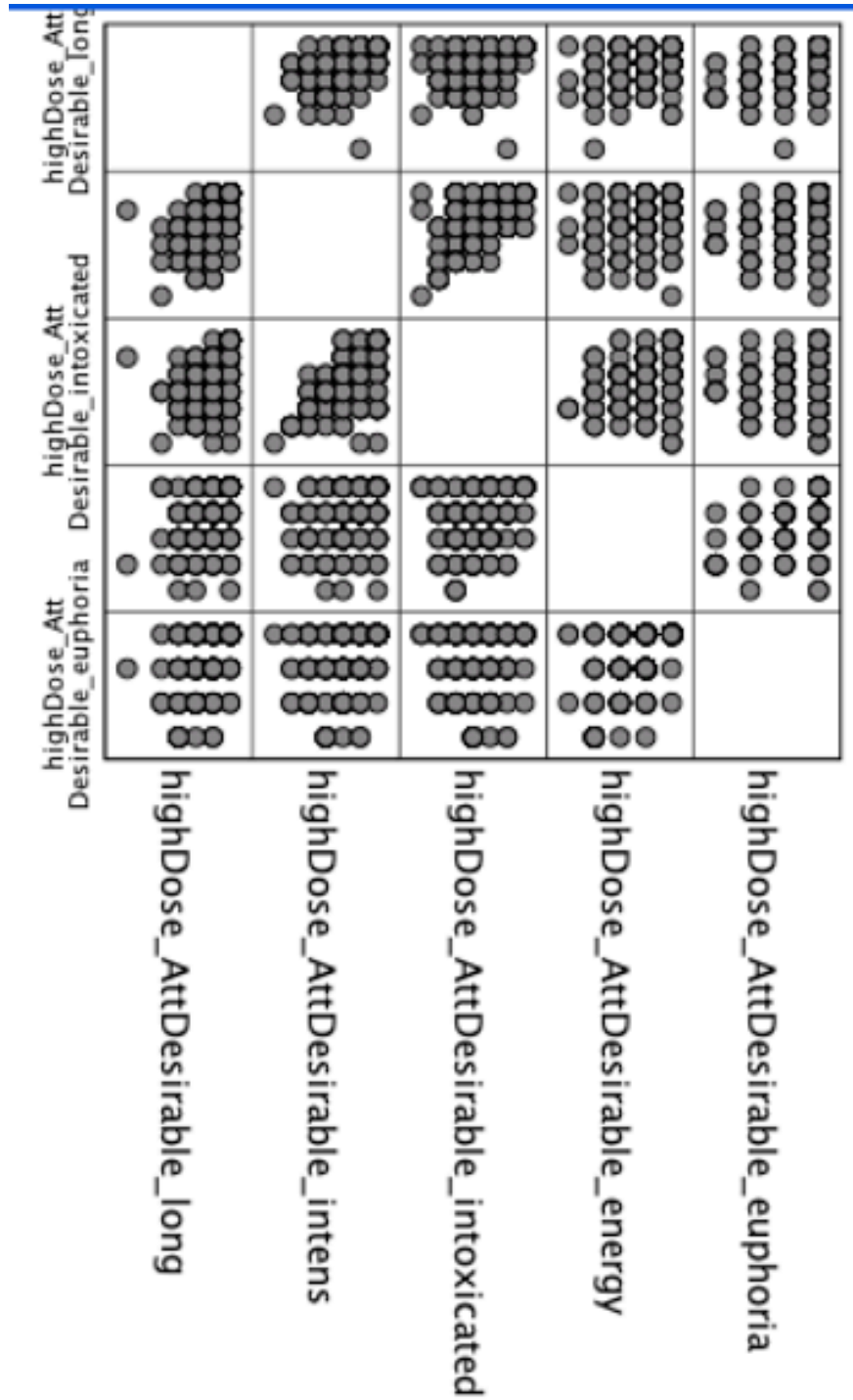


Figure 22.5: The scattermatrix produced in SPSS

Chapter 23

Scatterplot

23.1 jamovi

The jamovi section has not yet been added.

23.2 R

The R section has not yet been added.

23.3 SPSS

The SPSS section has not yet been added.

```
GRAPH
  /SCATTERPLOT(BIVAR) =
    x WITH y
  /MISSING =
    LISTWISE.
```


Chapter 24

Simple Regression Analysis

24.1 Intro

Univariate regression analysis (with one predictor) can be useful to get an estimate of the increase in the dependent variable as a function of the predictor variable. The approach differs for continuous or categorical predictors, and both will be shown.

24.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

24.1.2 Variable(s)

From this dataset, this example uses variables `xtcUseDosePref` as dependent variable (the MDMA dose a participant prefers), `highDose_attitude` as a continuous predictor (the attitude towards using a high dose of MDMA), and `hasJob` as a categorical predictor.

24.2 Input: jamovi

In the “Analyses” tab, click the “Regression” button and from the menu that appears, select “Linear Regression” as shown in Figure 24.1.

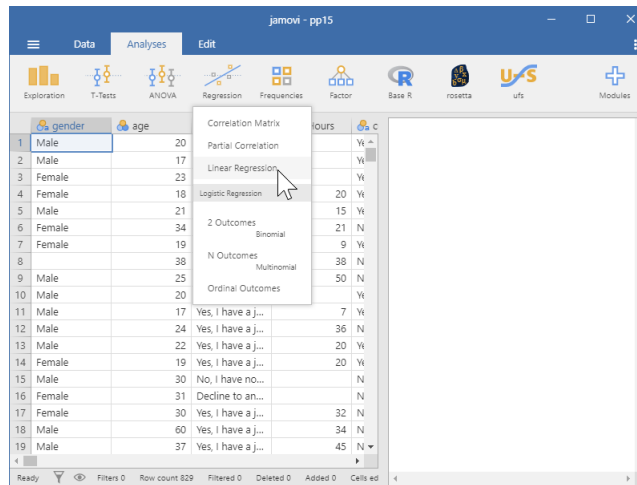


Figure 24.1: Opening the linear regression menu in jamovi

In the box at the left, select the dependent variable and move it to the box labelled “Dependent variable” using the button labelled with the rightward-pointing arrow as shown in Figure 24.2.

Then, select the predictor variable and move it to the box labelled “Covariates” if it is a continuous variable, and to the box labelled “Factors” if it is a categorical variable, as shown in Figure 24.3.

The results will immediately be shown in the right-hand “Results” panel. You can scroll down to specify additional analyses, for example to order more details about the coefficient by opening the “Model Coefficients” section as shown in Figure 24.4.

For example, to request the confidence interval for the slope coefficient and the standardized (scaled) coefficient, check the corresponding check boxes as shown in Figure 24.5.

24.3 Input: R

In R, there are roughly three approaches. Many analyses can be done with base R without installing additional packages. The `rosetta` package accompanies this book and aims to provide output similar to jamovi and SPSS with simple commands. Finally, the tidyverse is a popular collection of packages that try to work together consistently but implement a different underlying logic than base R (and so, the `rosetta` package).

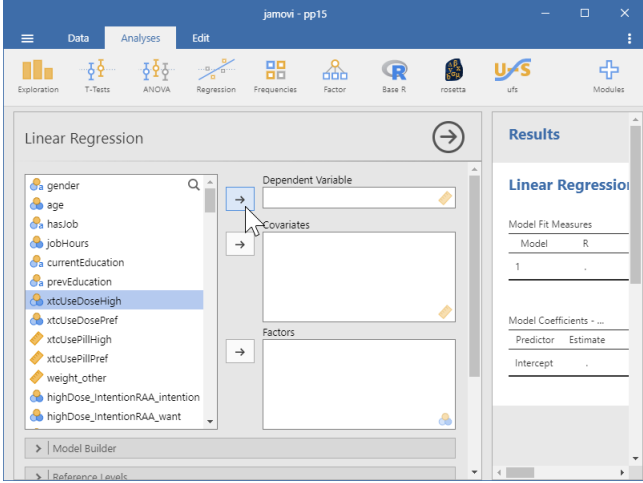


Figure 24.2: Selecting the dependent variable for the regression analysis in jamovi

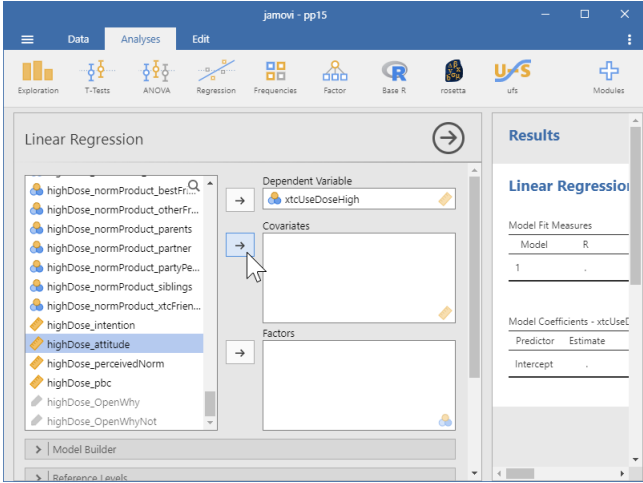


Figure 24.3: Selecting the predictor for the regression analysis in jamovi

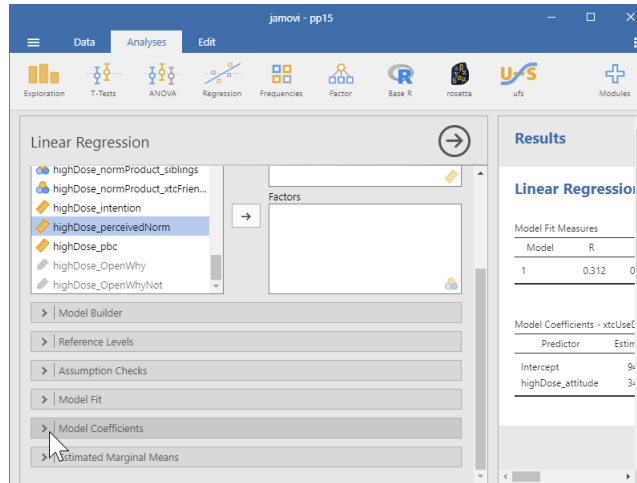


Figure 24.4: Opening the Model Coefficients section in jamovi

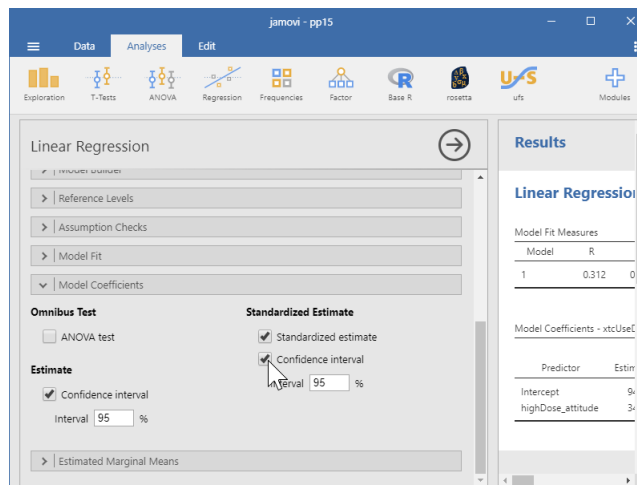


Figure 24.5: Selecting the predictor for the regression analysis in jamovi

24.3.1 R: base R

In base R, the `lm` (linear model) function can be combined with the `summary` function to show the most important results. With a continuous predictor, the code is as follows.

```
result <-  
  lm(  
    xtcUseDosePref ~ highDose_attitude,  
    data=dat  
  );  
summary(  
  result  
);
```

With a categorical predictor, the code is essentially the same, since R automatically treats variables that are factors as categorical, and numeric vectors as continuous variables.

```
result <-  
  lm(  
    xtcUseDosePref ~ hasJob,  
    data=dat  
  );  
summary(  
  result  
);
```

24.3.2 R: rosetta

In the `rosetta` package, the `regr` function wraps base R's `lm` function to present output similar to that provided by other statistics programs in one command.

```
rosetta::regr(  
  xtcUseDosePref ~ highDose_attitude,  
  data=dat  
);
```

Like `lm` in base R, the command is the same for a continuous predictor as it is for a categorical predictor (i.e. by replacing `highDose_attitude` with `hasJob`).

A scatter plot can be requested using argument `plot=TRUE`):

```
rosetta::regr(  
  xtcUseDosePref ~ highDose_attitude,  
  data=dat,  
  plot=TRUE  
);
```

24.4 Input: SPSS

For SPSS, there are two approaches: using the Graphical User Interface (GUI) or specify an analysis script, which in SPSS are called “syntax”.

24.4.1 SPSS: GUI



Figure 24.6: A screenshot placeholder

24.4.2 SPSS: Syntax

In SPSS, the `REGRESSION` command is used (don’t forget the period at the end (`.`), the command terminator):

```

REGRESSION
  /DEPENDENT
    xtcUseDosePref
  /METHOD
    ENTER highDose_attitude
  /STATISTICS
    COEF
    CI(95)
    R
    ANOVA
.

```

With a categorical predictor the code is essentially the same, since SPSS “sees” whether the predictor is continuous or categorical.

```

REGRESSION
  /DEPENDENT
    xtcUseDosePref
  /METHOD
    ENTER gender
  /STATISTICS
    COEF
    CI(95)
    R
    ANOVA
.

```

24.5 Output: jamovi

Linear Regression

Model Fit Measures		
Model	R	R ²
1	0.312	0.0972

Model Coefficients - xtcUseDoseHigh						
Predictor	Estimate	SE	95% Confidence Interval		t	p
			Lower	Upper		
Intercept	94.9	34.13	27.4	162.4	2.78	0.006
highDose_attitude	34.8	9.19	16.6	53.0	3.78	<.001

Figure 24.7: Univariate regression analysis output in jamovi

Formula: xtcUseDosePref ~ highDose_attitude
 Sample size: 135
 Multiple R-squared: [.06; .28] (point estimate = 0.15, adjusted = 0.15)
 Test for significance: (of full model) $F[1, 133] = 24.16, p < .001$

	95% conf. int.	estimate	se	t	p
(Intercept)	[7.16; 105.63]	56.39	24.89	2.27	.025
highDose_attitude	[19.7; 46.22]	32.96	6.70	4.92	<.001

^a These are unstandardized beta values, called 'B' in SPSS.

24.6 Output: R

24.6.1 R: base R

Call:

```
lm(formula = xtcUseDosePref ~ highDose_attitude, data = pp15)
```

Residuals:

```
      Min       1Q   Median       3Q      Max
-144.58  -58.56  -11.85   41.44  348.60
```

Coefficients:

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    56.394    24.890   2.266  0.0251 *
highDose_attitude 32.956     6.704   4.916 2.56e-06 ***
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 88.8 on 133 degrees of freedom
(694 observations deleted due to missingness)
```

```
Multiple R-squared:  0.1538,    Adjusted R-squared:  0.1474
```

```
F-statistic: 24.16 on 1 and 133 DF,  p-value: 2.559e-06
```

24.6.2 R: rosetta

24.6.2.1 Regression analysis

Summary

Raw regression coefficients

Scaled regression coefficients

	95% conf. int.	estimate	se	t	p
(Intercept)	[-0.05; 0.25]	0.10	0.08	1.32	.188
highDose_attitude	[0.21; 0.5]	0.36	0.07	4.92	<.001

^a These are standardized beta values, called 'Beta' in SPSS.

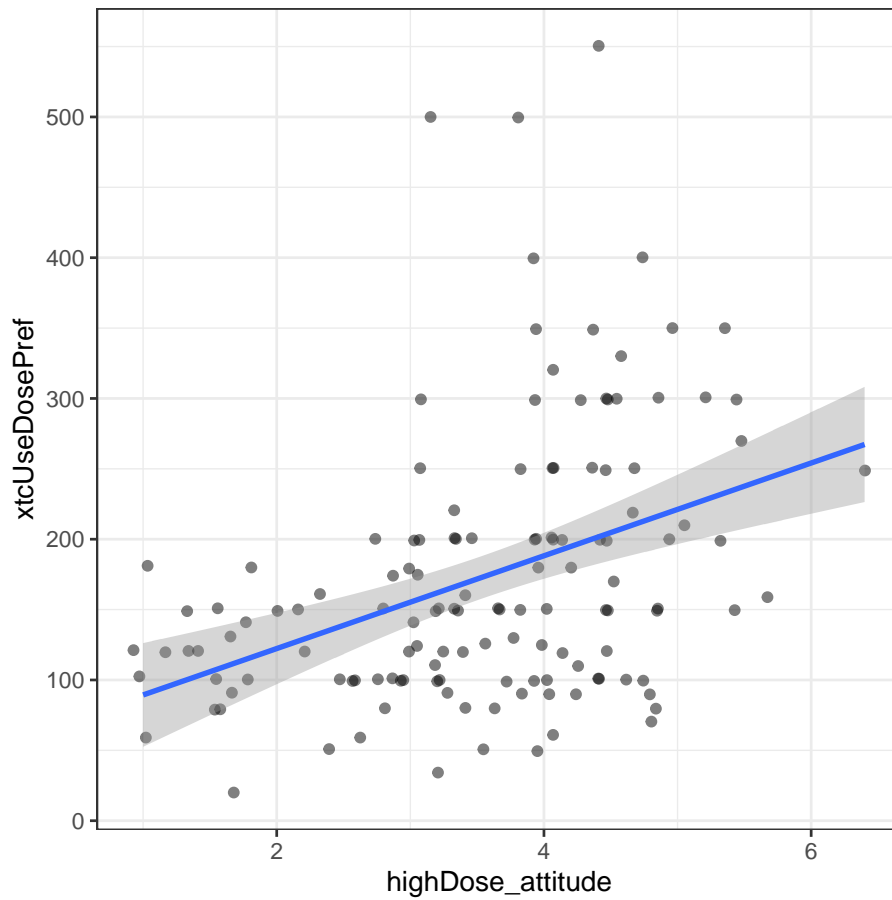


Figure 24.8: Scatterplot with regression line

Part V

Multivariate analyses

Chapter 25

Multiple Regression Analysis

25.1 Intro

Multivariate regression analysis can be useful to obtain a model to predict the dependent variable as a function of two or more predictor variables and estimate what proportion of the variance of that dependent variable can be understood using the predictor variables. The approach differs for continuous or categorical predictors, and both will be shown.

25.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

25.1.2 Variable(s)

From this dataset, this example uses variables `xtcUseDosePref` as dependent variable (the MDMA dose a participant prefers), `highDose_attitude`, `highDose_perceivedNorm`, and `highDose_pbc` as continuous predictors (the attitude, perceived norms, and perceived behavior control with respect to using a high dose of MDMA), and `hasJob_bi` as a categorical predictor (whether a participant has a job or not (e.g. is a student or unemployed)).

25.2 Input: jamovi

In the “Analyses” tab, click the “Regression” button and from the menu that appears, select “Linear Regression” as shown in Figure 24.1.

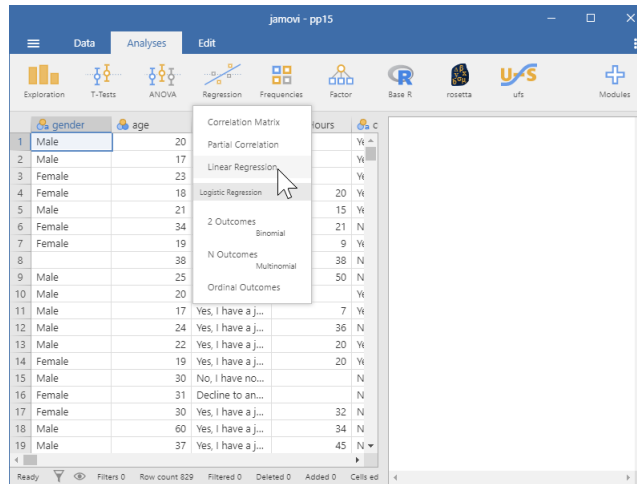


Figure 25.1: Opening the linear regression menu in jamovi

In the box at the left, select the dependent variable and move it to the box labelled “Dependent variable” using the button labelled with the rightward-pointing arrow as shown in Figure 24.2.

In the box at the left, select the continuous predictors and move them to the box labelled “Covariates” using the button labelled with the rightward-pointing arrow as shown in Figure 25.3.

Categorical predictors are moved to the box labelled “Factors” instead as shown in Figure 25.4.

The results will immediately be shown in the right-hand “Results” panel. You can scroll down to specify additional analyses, for example to order more details about the coefficients by opening the “Model Coefficients” section as shown in Figure 25.5.

For example, to request the confidence interval for the coefficient and the standardized (scaled) coefficients, check the corresponding check boxes as shown in Figure 25.6.

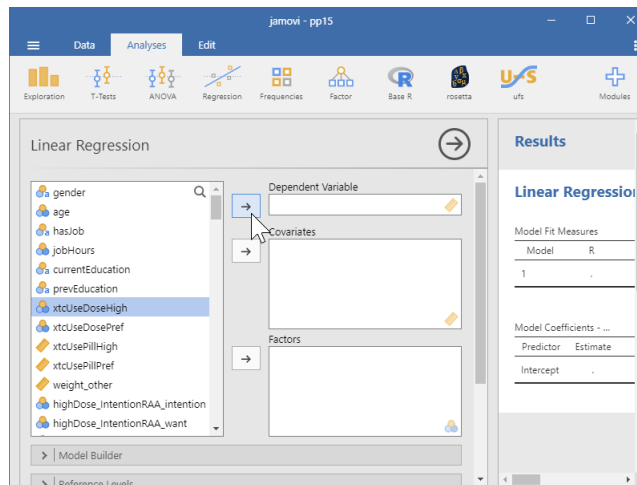


Figure 25.2: Selecting the dependent variable for the regression analysis in jamovi

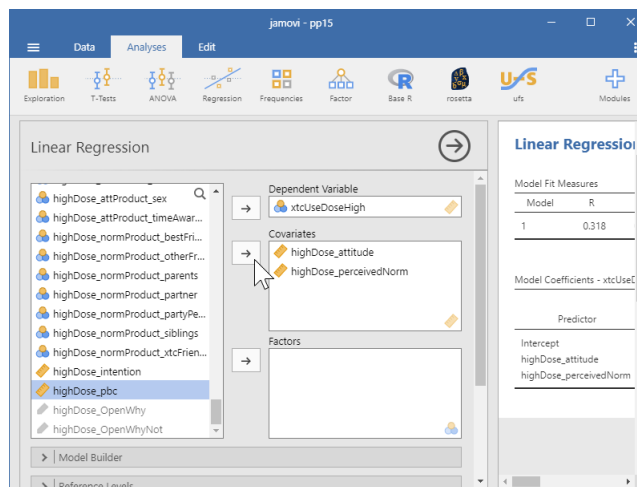


Figure 25.3: Selecting the continuous predictors for the regression analysis in jamovi

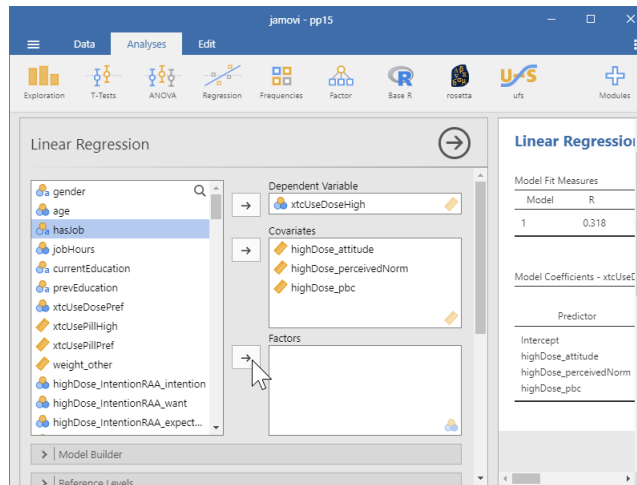


Figure 25.4: Selecting the categorical predictors for the regression analysis in jamovi

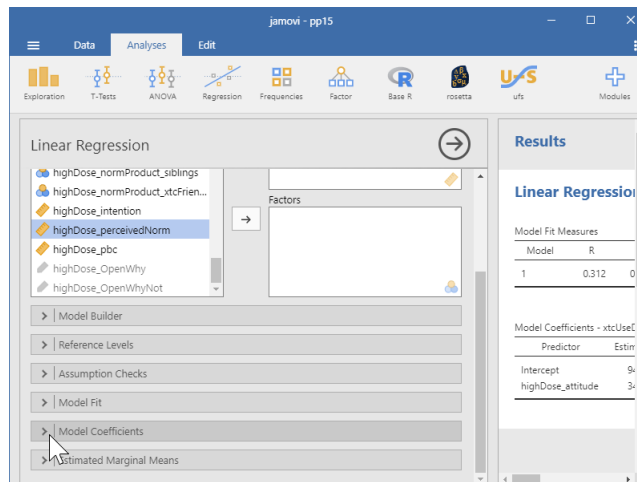


Figure 25.5: Opening the Model Coefficients section in jamovi

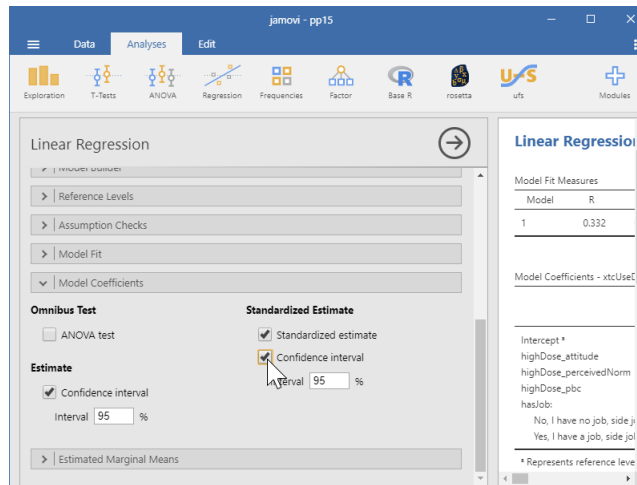


Figure 25.6: Selecting the predictor for the regression analysis in jamovi

25.3 Input: R

25.3.1 R: base R

In base R, the `lm` (linear model) function can be combined with the `summary` function to show the most important results. With a continuous predictor, the code is as follows. R automatically treats variables that are factors as categorical, and numeric vectors as continuous variables.

```
result <-
  lm(
    xtcUseDosePref ~
      highDose_attitude +
      highDose_perceivedNorm +
      highDose_pbc +
      hasJob_bi,
    data=dat
  );
summary(
  result
);
```

25.3.2 R: rosetta

In the `rosetta` package, the `regr` function wraps base R's `lm` function to present output similar to that provided by other statistics programs in one command.

```
rosetta::regr(  
  xtcUseDosePref ~  
    highDose_attitude +  
    highDose_perceivedNorm +  
    highDose_pbc +  
    hasJob_bi,  
  data=dat  
);
```

Like `lm` in base R, the command is the same for a continuous predictor as it is for a categorical predictor. Additional output can be requested using arguments `collinearity=TRUE`, and when there are two predictors and an interaction term, `plot=TRUE`):

```
rosetta::regr(  
  xtcUseDosePref ~  
    highDose_attitude +  
    hasJob_bi +  
    highDose_attitude:hasJob_bi,  
  data=dat,  
  collinearity=TRUE,  
  plot=TRUE  
);
```

25.4 Input: SPSS

In SPSS, the `REGRESSION` command is used (don't forget the period at the end (`.`), the command terminator):

```
REGRESSION  
  /STATISTICS COEFF OUTS CI(95) R ANOVA  
  /DEPENDENT xtcUseDosePref  
  /METHOD=ENTER highDose_attitude highDose_perceivedNorm highDose_pbc hasJob_bi.
```


Linear Regression

Model Fit Measures		
Model	R	R ²
1	0.332	0.110

Model Coefficients - xtcUseDoseHigh						
Predictor	Estimate	SE	95% Confidence Interval		t	p
			Lower	Upper		
Intercept *	-25.95	122.7	-270.05	218.1	-0.212	0.833
highDose_attitude	36.54	14.6	7.58	65.5	2.511	0.014
highDose_perceivedNorm	-5.66	12.5	-30.49	19.2	-0.454	0.651
highDose_pbc	14.30	20.6	-26.64	55.2	0.695	0.489
hasJob:						
No, I have no job, side job, or own company – Decline to answer	60.37	67.6	-74.25	195.0	0.892	0.375
Yes, I have a job, side job, or own company – Decline to answer	45.48	63.2	-80.39	171.3	0.719	0.474

* Represents reference level

Figure 25.7: Multivariate regression analysis output in jamovi

25.5 Output: jamovi

25.6 Output: R

25.6.1 R: base R

Call:

```
lm(formula = xtcUseDosePref ~ highDose_attitude + highDose_perceivedNorm +
    highDose_pbc + hasJob_bi, data = pp15)
```

Residuals:

```
      Min       1Q   Median       3Q      Max
-136.90  -48.76   -8.61   37.55  341.33
```

Coefficients:

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    36.011    80.152   0.449  0.65449
highDose_attitude  31.424    10.210   3.078  0.00289 **
highDose_perceivedNorm  6.244     8.613   0.725  0.47072
highDose_pbc     1.928    14.495   0.133  0.89456
hasJob_biYes    -3.310    21.400  -0.155  0.87750
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 82.25 on 77 degrees of freedom
(747 observations deleted due to missingness)
```

```
Multiple R-squared:  0.2058,    Adjusted R-squared:  0.1646
```

```
F-statistic: 4.988 on 4 and 77 DF,  p-value: 0.001252
```

Formula: `xtcUseDosePref ~ highDose_attitude + hasJob_bi + highDose`
Sample size: 82
Multiple R-squared: [.07; .37] (point estimate = 0.2, adjusted = 0.17)
Test for significance: (of full model) $F[3, 78] = 6.69, p < .001$

	95% conf. int.	estimate	se	t	p
(Intercept)	[-41.36; 216.05]	87.34	64.65	1.35	.181
highDose_attitude	[-9.46; 60.68]	25.61	17.62	1.45	.150
hasJob_biYes	[-196.13; 93.09]	-51.52	72.64	-0.71	.480
highDose_attitude:hasJob_biYes	[-26.45; 52.66]	13.11	19.87	0.66	.511

^a These are unstandardized beta values, called 'B' in SPSS.

25.6.2 R: rosetta

```
## there are higher-order terms (interactions) in this model
## consider setting type = 'predictor'; see ?vif
## there are higher-order terms (interactions) in this model
## consider setting type = 'predictor'; see ?vif
```

25.6.2.1 Regression analysis

Summary

Raw regression coefficients

Scaled regression coefficients

Collinearity diagnostics

25.7 Output: SPSS

	95% conf. int.	estimate	se	t	p
(Intercept)	[-0.22; 0.51]	0.14	0.18	0.78	.436
highDose_attitude	[-0.1; 0.65]	0.28	0.19	1.45	.150
hasJob_biYes	[-0.46; 0.38]	-0.04	0.21	-0.18	.857
highDose_attitude:hasJob_biYes	[-0.29; 0.57]	0.14	0.21	0.66	.511

^a These are standardized beta values, called 'Beta' in SPSS.

	VIF	Tolerance
highDose_attitude	4.68	0.21
hasJob_bi	11.93	0.08
highDose_attitude:hasJob_bi	15.13	0.07

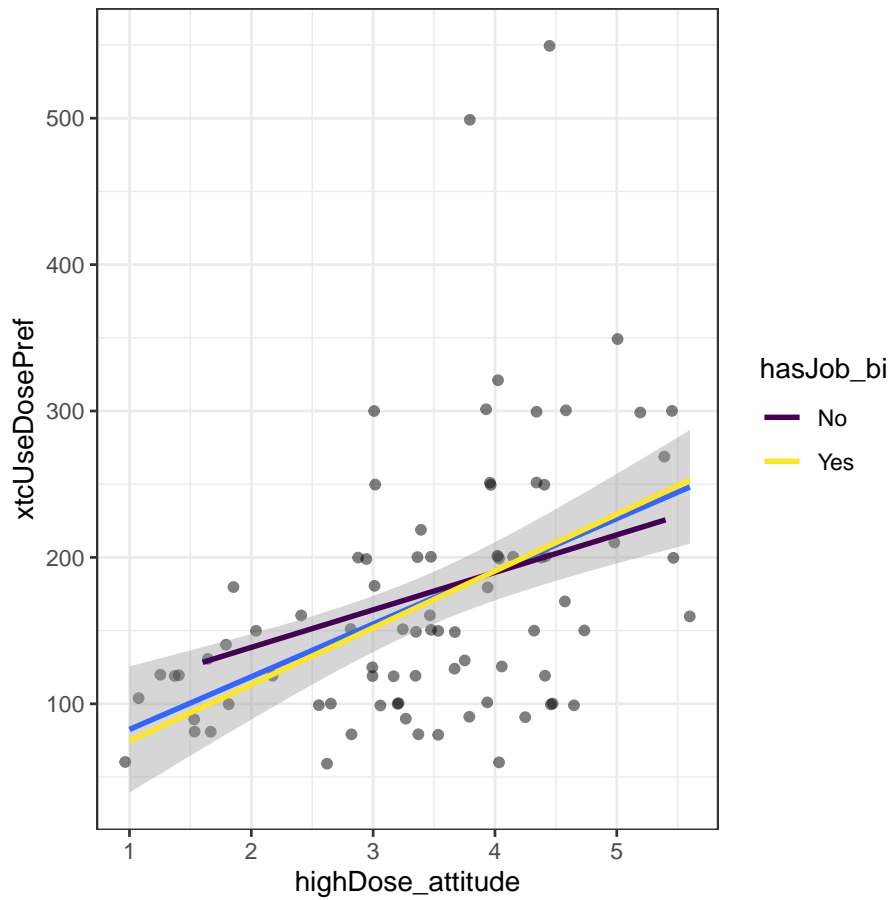


Figure 25.8: Scatterplot with regression line

➔ **Regression**

Variables Entered/Removed^a

Model	Variables Entered	Variables Removed	Method
1	hasjob_bi, highDose_pbc, highDose_perceivedNorm, highDose_attitude ^b	.	Enter

a. Dependent Variable: xtcUseDosePref

b. All requested variables entered.

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.454 ^a	.206	.165	82.246

a. Predictors: (Constant), hasjob_bi, highDose_pbc, highDose_perceivedNorm, highDose_attitude

ANOVA^a

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	134975.589	4	33743.897	4.988	.001 ^b
	Residual	520860.313	77	6764.420		
	Total	655835.902	81			

a. Dependent Variable: xtcUseDosePref

b. Predictors: (Constant), hasjob_bi, highDose_pbc, highDose_perceivedNorm, highDose_attitude

Coefficients^a

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.	95.0% Confidence Interval for B	
		B	Std. Error	Beta			Lower Bound	Upper Bound
1	(Constant)	39.321	87.721		.448	.655	-135.355	213.996
	highDose_attitude	31.424	10.210	.390	3.078	.003	11.094	51.755
	highDose_perceivedNorm	6.244	8.613	.091	.725	.471	-10.908	23.395
	highDose_pbc	1.928	14.495	.014	.133	.895	-26.936	30.791
	hasjob_bi	-3.310	21.400	-.016	-.155	.877	-45.923	39.303

a. Dependent Variable: xtcUseDosePref

Figure 25.9: The output of the regression analysis in SPSS

Chapter 26

Logistic Regression Analysis

26.1 Intro

Logistic regression analysis can be useful to obtain a model to predict a dichotomous dependent variable (also known as a binary or logical variable) as a function of one or more predictor variables and estimate how well that dependent variable can be predicted using the predictor variables.

26.1.1 Example dataset

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

26.1.2 Variable(s)

From this dataset, this example uses variables `gender_bi` as dependent variable (the participant’s gender) and `xtcUseDosePref` (the MDMA dose a participant prefers in milligrams) and `weight_other` (a participant’s weight in kilograms) as predictors.

26.2 Input: jamovi

In the “Analyses” tab, click the “Regression” button and from the menu that appears, in the “Logistic Regression” section, select “2 Outcomes (Binomial)”

as shown in Figure 26.1.

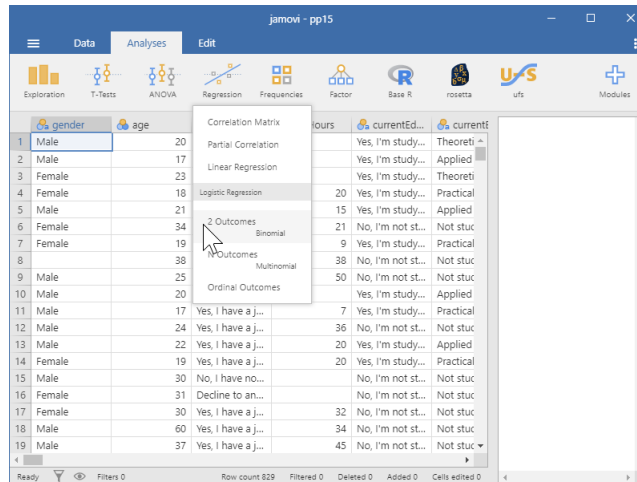


Figure 26.1: Opening the linear regression menu in jamovi

In the box at the left, select the dependent variable and move it to the box labelled “Dependent variable” using the button labelled with the rightward-pointing arrow as shown in Figure 26.2.

In the box at the left, select any continuous predictors and move them to the box labelled “Covariates” using the button labelled with the rightward-pointing arrow as shown in Figure 26.3. If there are categorical predictors, move them to the box labelled “Factors”.

The results will immediately be shown in the right-hand “Results” panel. You can scroll down to specify additional analyses, for example, to request more details about the regression coefficients you can open the “Model Coefficients” section as shown in Figure 26.4.

For example, to order the odds ratios and their confidence intervals, check the corresponding check boxes as shown in Figure 26.5.

26.3 Input: R

26.3.1 R: base R

In base R, the `glm` (generalized linear model) function can be called with argument `family = binomial(link = "logit")` to perform a logistic regression. The `summary` function can then be used to show the most important results.

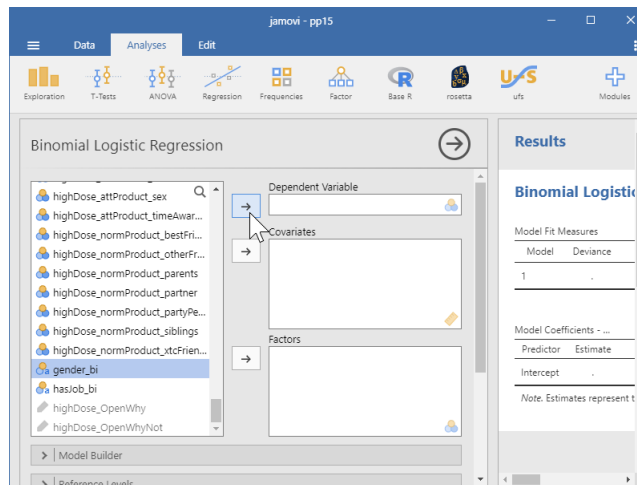


Figure 26.2: Selecting the dependent variable for the regression analysis in jamovi

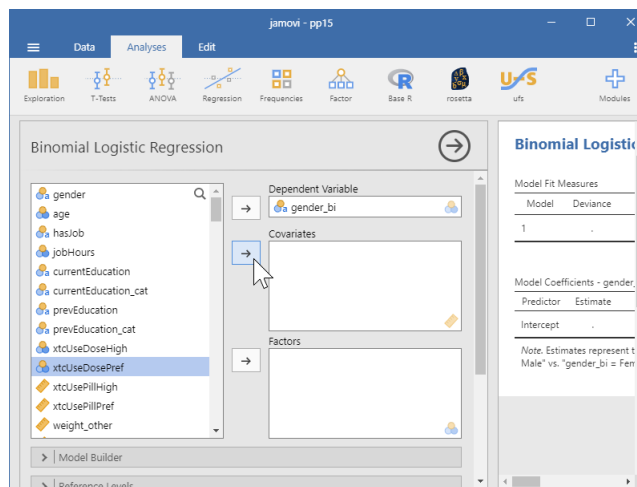


Figure 26.3: Selecting the continuous predictors for the regression analysis in jamovi

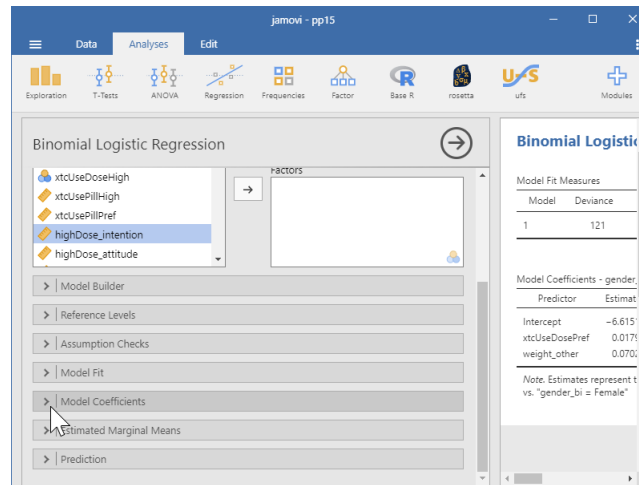


Figure 26.4: Selecting the categorical predictors for the regression analysis in jamovi

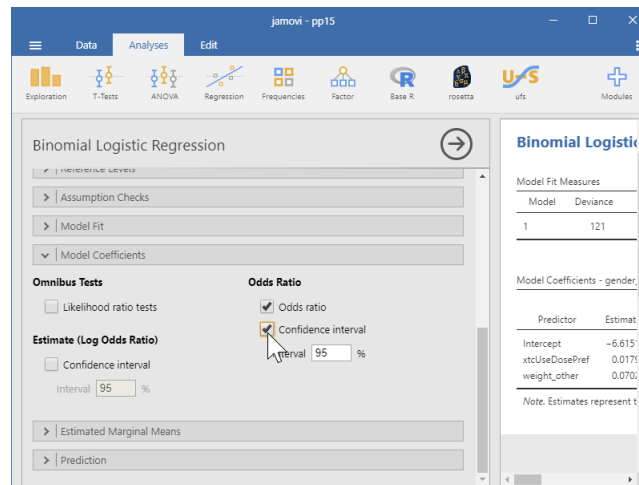


Figure 26.5: Selecting the predictor for the regression analysis in jamovi


```

result <-
  glm(
    gender_bi ~
      xtcUseDosePref +
      weight_other,
    data = dat,
    family = binomial(link = "logit")
  );
summary(result);

```

26.3.2 R: rosetta

In the `rosetta` package, the `logRegr` function wraps base R's `glm` function to present output similar to that provided by other statistics programs in one command.

```

rosetta::logRegr(
  gender_bi ~
    xtcUseDosePref +
    weight_other,
  data=dat
);

```

With multiple predictors, if `collinearity=TRUE`, collinearity diagnostics are ordered. If there is only one predictor, you can request a visualisation of the raw data, the (binned) means, and the logistic curve by using `plot=TRUE`.

```

rosetta::logRegr(
  gender_bi ~
    xtcUseDosePref,
  data = dat,
  plot = TRUE
);

```

26.4 Input: SPSS

In SPSS, the `REGRESSION` command is used (don't forget the period at the end `(.)`, the command terminator):

```
LOGISTIC REGRESSION VARIABLES gender_bi
  /METHOD=ENTER xtcUseDosePref weight_other.
.
```

26.5 Output: jamovi

Binomial Logistic Regression

Model Fit Measures

Model	Deviance	AIC	R ² _{NLSE}
1	121	127	0.254

Model Coefficients - gender_bi

Predictor	Estimate	SE	Z	p	Odds ratio	95% Confidence Interval	
						Lower	Upper
Intercept	-6.6151	1.80592	-3.66	< .001	0.00134	3.89e-5	0.0462
xtcUseDosePref	0.0179	0.00490	3.65	< .001	1.01803	1.01	1.0279
weight_other	0.0702	0.02462	2.85	0.004	1.07268	1.02	1.1257

Note: Estimates represent the log odds of "gender_bi = Male" vs. "gender_bi = Female"

Figure 26.6: Logistic regression analysis output in jamovi

26.6 Output: R

26.6.1 R: base R

Call:

```
glm(formula = gender_bi ~ xtcUseDosePref + weight_other, family = binomial(link = "logit",
  data = pp15)
```

Deviance Residuals:

```
      Min       1Q   Median       3Q      Max
-2.5760  -0.7852   0.3836   0.7113   1.6096
```

Coefficients:

```
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -6.615057   1.805922  -3.663 0.000249 ***
xtcUseDosePref  0.017874   0.004902   3.646 0.000266 ***
weight_other   0.070162   0.024616   2.850 0.004369 **
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 162.31  on 134  degrees of freedom
```

```

      Formula:  gender_bi ~ xtcUseDosePref + weight_other
    Sample size: 135
      Predicting: Male
Cox & Snell R-squared: .26
  Nagelkerke R-squared: .38
Test for significance: (of full model)  ChiSq[2] = 41.29, p < .001

```

Male	
Female	39
Male	96

Residual deviance: 121.02 on 132 degrees of freedom
 (694 observations deleted due to missingness)
 AIC: 127.02

Number of Fisher Scoring iterations: 6

26.6.2 R: rosetta

```

## Warning in rosetta::logRegr(gender_bi ~ xtcUseDosePref + weight_other, data =
## pp15, : You requested a plot, but for now plots are only available for logistic
## regression analyses with one predictor.

```

26.6.3 Logistic regression analysis {#rosettaLogRegr-AxSgmzUK3X}}

Summary

Predictions by the null model (71.11% correct)

Predictions by the tested model (80% correct)

Regression coefficients

Regression coefficients as odds ratios

Collinearity diagnostics

	0	1
0	21	18
1	9	87

	95% conf. int.	estimate	se	z	p
(Intercept)	[-10.48; -3.36]	-6.62	1.81	-3.66	<.001
xtcUseDosePref	[0.01; 0.03]	0.02	0.00	3.65	<.001
weight_other	[0.03; 0.12]	0.07	0.02	2.85	.004

^a These are log odds values, called 'B' in SPSS.

	95% conf. int.	Odds Ratio point estimate
(Intercept)	[0; 0.03]	0.00
xtcUseDosePref	[1.01; 1.03]	1.02
weight_other	[1.03; 1.13]	1.07

^a These are odds ratios, called 'Exp(B)' in SPSS.

26.7 Output: SPSS

→ Logistic Regression

Case Processing Summary

Unweighted Cases ^a		N	Percent
Selected Cases	Included in Analysis	135	16.3
	Missing Cases	694	83.7
	Total	829	100.0
Unselected Cases		0	.0
Total		829	100.0

a. If weight is in effect, see classification table for the total number of cases.

Dependent Variable Encoding

Original Value	Internal Value
Female	0
Male	1

Block 0: Beginning Block

Classification Table^{a,b}

	Observed		Predicted		Percentage Correct
			gender_bi Female	Male	
Step 0	gender_bi	Female	0	39	.0
		Male	0	96	100.0
Overall Percentage					71.1

a. Constant is included in the model.

b. The cut value is .500

Variables in the Equation

	B	S.E.	Wald	df	Sig.	Exp(B)
Step 0 Constant	.901	.190	22.503	1	.000	2.462

Figure 26.7: The output of the logistic regression analysis in SPSS part 1

Variables not in the Equation

		Score	df	Sig.
Step 0	Variables			
	xtcUseDosePref	19.037	1	.000
	weight_other	16.335	1	.000
Overall Statistics		27.425	2	.000

Block 1: Method = Enter**Omnibus Tests of Model Coefficients**

		Chi-square	df	Sig.
Step 1	Step	41.291	2	.000
	Block	41.291	2	.000
	Model	41.291	2	.000

Model Summary

Step	-2 Log likelihood	Cox & Snell R Square	Nagelkerke R Square
1	121.020 ^a	.264	.377

a. Estimation terminated at iteration number 6 because parameter estimates changed by less than .001.

Classification Table^a

		Predicted		Percentage Correct
		gender_bi		
Observed		Female	Male	
Step 1	gender_bi			
	Female	21	18	53.8
	Male	9	87	90.6
Overall Percentage				80.0

a. The cut value is .500

Variables in the Equation

		B	S.E.	Wald	df	Sig.	Exp(B)
Step 1 ^a	xtcUseDosePref	.018	.005	13.297	1	.000	1.018
	weight_other	.070	.025	8.124	1	.004	1.073
	Constant	-6.615	1.806	13.417	1	.000	.001

a. Variable(s) entered on step 1: xtcUseDosePref, weight_other.

Figure 26.8: The output of the logistic regression analysis in SPSS part 2

Chapter 27

Moderated mediation

Mediation refers to a causal model in which the relation between the predictor and the dependent variable is mediated by one or more other variables (“the mediators”). Note that mediation by definition entails strong assumptions regarding causality, which require longitudinal designs and almost always experimental independent manipulation of the predictor (and if possible mediator). Moderation and mediation models can also be combined. The causal path between the predictor and the mediator(s) can be moderated by a moderator. Likewise, the path between the mediators and the dependent variable can be moderated by another moderator. A moderator strengthens or weakens the relation between two variables. The path from predictor to mediator can be moderated by a moderator designated as “w”, while the path from mediator to dependent variable can be moderated by a moderator designated as “v”. The PROCESS macro (SPSS) uses different model numbers to distinguish these models. In these examples, the dataset/dataframe is called `dat`, the predictor (the first variable in the causal chain) is called `predictorVariable`, the mediators are called `mediatorVariable1`, `mediatorVariable2`, etc. (the intermediate variables in the causal chain), the dependent variable is called `dependentVariable` (the last variable in the causal chain), and covariates are called `covariate1`, `covariate2`, etc. Finally, the moderators are called `moderatorVariable1` and `moderatorVariable2`.

27.1 jamovi

First download the `jamm` module. Go to the Medmod menu and select ‘GLM mediation model’. Numerical predictor variables are dragged to the window, labeled ‘Covariates’. Mediators are dragged to the window ‘Mediators’. Example 1 is a mediation model without moderator and with two mediators.

The screenshot shows the 'GLM Mediation Model' interface. On the left, a list of variables includes 'idNumber', 'moderatorVariable1' (highlighted), 'moderatorVariable2', 'covariate1', and 'covariate2'. On the right, four sections are visible: 'Dependent Variable' containing 'dependentVariable'; 'Mediators' containing 'mediatorVariable1' and 'mediatorVariable2'; 'Factors' which is currently empty; and 'Covariates' containing 'predictorVariable'. Each section has a right-pointing arrow button to its left, and each variable in the sections has a small orange diamond icon next to it.

In example 2 a moderator of the x-m path is added to the model and only one mediator is used. The second example needs extra windows (sections). First, add the moderator as a covariate in the first section, see example 1. Second, drag the moderator in the Moderators section to the 'Mediated Effects' window. Third, remove in the Full Model section under 'Model Terms' the interaction terms 'predictorVariable * covariate1' and 'mediatorVariable * covariate1'.

GLM Mediation Model ➔

idNumber

moderatorVariable2

covariate1

mediatorVariable2

covariate2

Dependent Variable

➔ dependentVariable

Mediators

➔ mediatorVariable1

Factors

➔

Covariates

➔ predictorVariable

moderatorVariable1

Mediators Models

predictorVariable

moderatorVariable1

mediatorVariable1

Models for mediators

Mediator = mediatorVariable1

predictorVariable

moderatorVariable1

predictorVariable * moderatorVariab...

Full Model

Components

predictorVariable

moderatorVariable1

mediatorVariable1

Model Terms

mediatorVariable1

predictorVariable

moderatorVariable1

Moderators

predictorVariable

moderatorVariable1

Mediated effects

Mediator = mediatorVariable1

moderatorVariable1

By adding variables as mediators, covariates, or moderators, jamovi automatically builds a full model consisting of all terms, including interactions. By removing terms from the model window, all kinds of special models can be defined. In the output pane the defined model is visualised, so here it can be verified whether the intended model is indeed computed.

27.2 R

In R the function is called `gemm()`, which is in the package `rosetta`. A tutorial can be found at: https://www.academia.edu/40039588/Tutorial_of_moderated_mediation_with_SEM_the_gemm_function or DOI: 10.13140/RG.2.2.22175.10404. Example 1 is a mediation model without moderator and with two mediators; this model is called as follows, The data are in the R object `dat` again.

```
gemm(dat = dat,
      xvar = "predictorVariable",
      mvars = c("mediatorVariable1", "mediatorVariable2"),
      yvar = "dependentVariable");
```

In example 2 a moderator of the x-m path is added to the model and only one mediator is used, which is called as follows.

```
gemm(dat = dat,
      xvar = "predictorVariable",
      mvars = "mediatorVariable1",
      yvar = "dependentVariable",
      xmmod = "moderatorVariable1")
```

In model 3 covariates are added. It is possible to choose on which variable (mediators or dependent) the covariates have an assumed effect. In this example both covariates have an assumed effect on both the mediators and the dependent variable. Furthermore, the m-y path is moderated. Example 3 is called as follows.

```
gemm(dat = dat,
      xvar = "predictorVariable",
      mvars = "mediatorVariable1",
      yvar = "dependentVariable",
```

```

mymod = "moderatorVariable1",
cmvars = c("covariate1","covariate2"),
cyvars = c("covariate1","covariate2" )

```

In example 4 both x-m and m-y paths are moderated. There are two mediators. In this example the covariates have an assumed effect on both the mediators and the dependent variable. Example 4 is called as follows.

```

gemm(dat = dat,
      xvar  ="predictorVariable",
      mvars = c("mediatorVariable1","mediatorVariable2"),
      yvar  = "dependentVariable",
      xmmod = "moderatorVariable1",
      mymod = "moderatorVariable2",
      cmvars = "covariate1",
      cyvars = "covariate1" )

```

The output of the `gemm` function can be put in an R object and can then be further inspected. The `print` and three `plot` functions are developed to provide relevant output of this analysis. Plots are only relevant when there is moderation, because they show the simple slopes or index of moderated mediation.

```

output <- gemm(dat = dat,
               xvar  ="predictorVariable",
               mvars = c("mediatorVariable1", "mediatorVariable2"),
               yvar  = "dependentVariable",
               xmmod = "moderatorVariable");

print(output)
plotIMM(output);
plotSS(output);

```

27.3 SPSS

In SPSS the PROCESS macro can be used to analyse a (moderated) mediation model. Version 3 of this macro can be downloaded from <http://www.afhayes.com>. The SPSS code shown here is based on version 2. Several mediators can be added and also several covariates. A model code must be specified in this version. See Hayes(2013; 2018).

Example 1 has two mediators and no covariates.

```
PROCESS vars = dependentVariable, predictorVariable, mediatorVariable1, mediatorVariable2
/ y = dependentVariable
/ x = predictorVariable
/ m = mediatorVariable1
mediatorVariable2
/ model = 4.
```

Example 2 has one mediator, one moderator of the path from the predictor to the mediator (the x-m path) and no covariates.

```
PROCESS vars = dependentVariable, predictorVariable mediatorVariable, moderatorVariable1
/ y = dependentVariable
/ x = predictorVariable
/ m = mediatorVariable1
/ w = moderatorVariable1
/ model = 7.
```

Example 3 has one mediator, one moderator of the path from the mediator to the dependent variable (m-y path) and two covariates.

```
PROCESS vars = dependentVariable, predictorVariable mediatorVariable1, moderatorVariable1
covariate1 covariate2
/ y = dependentVariable
/ x = predictorVariable
/ m = mediatorVariable1
/ v = moderatorVariable2
/ c = covariate1 covariate2
/ model = 14.
```

Example 4 has two mediators, two moderators one for each path and two covariates.

```
PROCESS vars = dependentVariable, predictorVariable mediatorVariable1, mediatorVariable2
covariate1 covariate2
/ y = dependentVariable
/ x = predictorVariable
/ m = mediatorVariable1 mediatorVariable2
/ w = moderatorVariable1
/ v = moderatorVariable2
/ c = covariate1
/ model = 21.
```

Chapter 28

Multilevel analysis

Multilevel analysis (MLA) is used for the analysis of hierarchical data. Data at the lowest level are clustered in higher levels. An important application for MLA is in ESM/EMA data. This type of data are called intensive longitudinal data. Subjects have measurements on multiple occasions. The occasions are clustered within the subjects. In these examples, the dataset/dataframe is called `dat`, the predictor is called `predictorVariable` (more than one are possible), the clustervariable is called `idNumber`, the dependent variable is called `dependentVariable`. For longitudinal data there is an `indexVariable`, which indicates the repeated measures. Furthermore, you may need a lagged variable to control for autocorrelation: for example, a lagged one variable: `predictorVariableLag1`.

28.1 jamovi

MLA can be done in jamovi with the module “GAMLj: general analysis for linear models for jamovi”. After loading this module, the Linear Models menu must be selected and then the option MIXED Model. In the input window the variables are placed as shown in Figure 28.1. Under the heading Fixed Effects the variables for the fixed effects are selected. Here, it is also possible to define interaction effects.

Under the heading Random Effects the variables for the random effects are selected, as shown in Figure 28.2, and when there are more than one, the correlations between these effects can also be specified by checking the box `correlated`.

Example 2 concerns a longitudinal example. Here we have an additional variable, called `indexVariable`, which is an index of time, e.g, the days or other

Mixed Model ➔

predictorVariable2
indexVariable

Dependent Variable
➔ dependentVariable

Factors
➔

Covariates
➔ predictorVariable

Cluster variables
➔ idNumber

Estimation **Confidence Intervals**

REML Confidence intervals Interval %

▼ Fixed Effects

Components
predictorVariable

Model Terms
➔ predictorVariable
➔ ▼

Fixed Intercept

Figure 28.1: Select the variables for the analysis

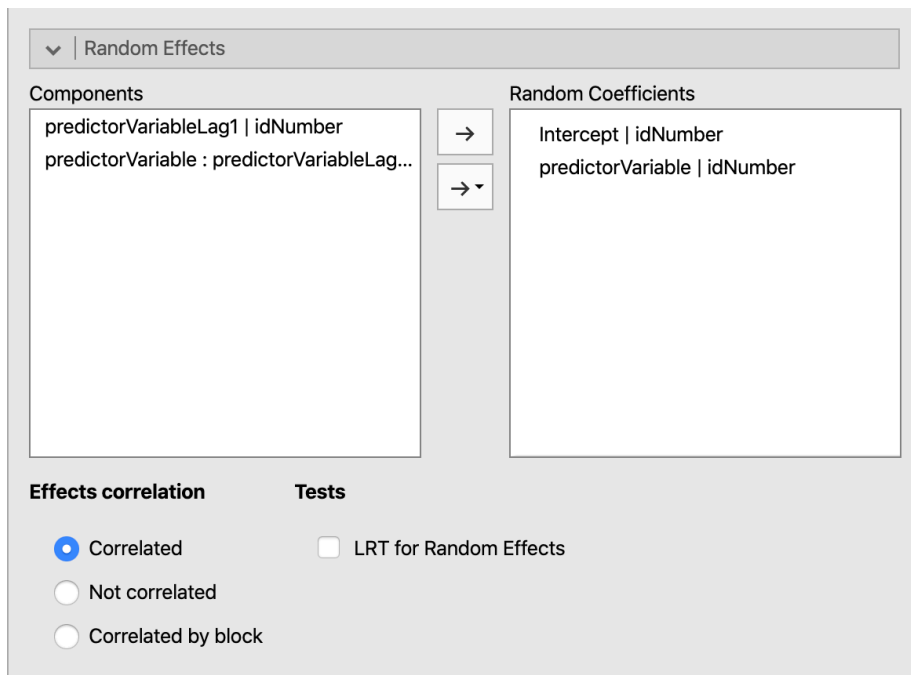


Figure 28.2: Define the random effects for the analysis

measurement moments. Figure 28.3 shows the input in jamovi, including a lagged variable to model the auto-correlation.

28.2 R

In R there are several possible packages for multilevel analysis. Here the function that is illustrated, is called `lmer`, which is in the package `lme4`. The data are in data set `dat`. Example 1 is called as follows. So, make sure to install the `lme4` package first and load it in your R session with `library(lme4)`.

```
model <- lmer(dependentVariable ~ predictorVariable +
              (1 + predictorVariable | idNumber, data = dat)
```

Example 2, for longitudinal data, is called with the following code. A lagged variable can be constructed and added to the fixed part of the model to model the auto-correlation.

Mixed Model ➔

Dependent Variable

➔ dependentVariable

Factors

➔

Covariates

➔ predictorVariable
indexVariable
predictorVariableLag1

Cluster variables

➔ idNumber

Estimation **Confidence Intervals**

REML Confidence intervals Interval %

▼ | Fixed Effects

Components

predictorVariable	1
indexVariable	
predictorVariableLag1	

Model Terms

➔ predictorVariable
indexVariable
predictorVariableLag1

Fixed Intercept

Figure 28.3: Select the variables for the longitudinal analysis


```
model <- lmer(dependentVariable ~ predictorVariable + indexVariable + predictorVariableLag1 +
              (1 + predictorVariable| idNumber, data = dat)
```

The output can be put in an object and with, for example, the `summary()` function this output can be inspected.

```
summary(model);
```

If you want to see p-values for the fixed effect, you should load the package `lmerTest`. With this package the output of the calls to `lmer` automatically provide p-values.

28.3 SPSS

In SPSS the MIXED procedure can be used to do multilevel analysis. A simple model with one predictor and two random effects, for respectively the intercept and the predictor, can be run with the following syntax.

```
MIXED dependentVariable WITH predictorVariable
  /PRINT= SOLUTION TESTCOV
  /METHOD= REML
  /FIXED= INTERCEPT predictorVariable
  /RANDOM= INTERCEPT predictorVariable| SUBJECT(idNumber) COVTYPE(UN).
```

Example 2 shows a longitudinal example again.

```
MIXED dependentVariable WITH predictorVariable indexVariable
  /PRINT=SOLUTION TESTCOV
  /FIXED=INTERCEPT predictorVariable | SSTYPE(3)
  /RANDOM=INTERCEPT predictorVariable | SUBJECT(idNumber) COVTYPE(VC)
  /REPEATED=indexvariable | SUBJECT(idNumber) COVTYPE(AR1).
```


Part VI

Sample Size Calculations

Chapter 29

Cohen's d: accuracy

29.1 Intro

Study planning includes deciding how large a sample will be collected. This decision is often informed by the required power or the required accuracy. This chapter shows how to compute both.

29.1.1 Example dataset

These sample size computations do not require data.

29.1.2 Variable(s)

These sample size computations do not require variables.



Figure 29.1: A screenshot placeholder

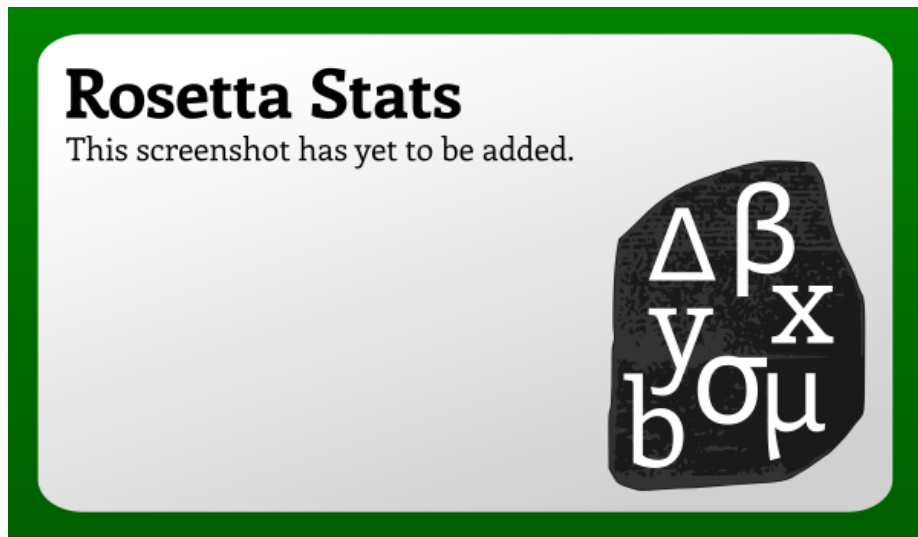


Figure 29.2: A screenshot placeholder



Figure 29.3: A screenshot placeholder

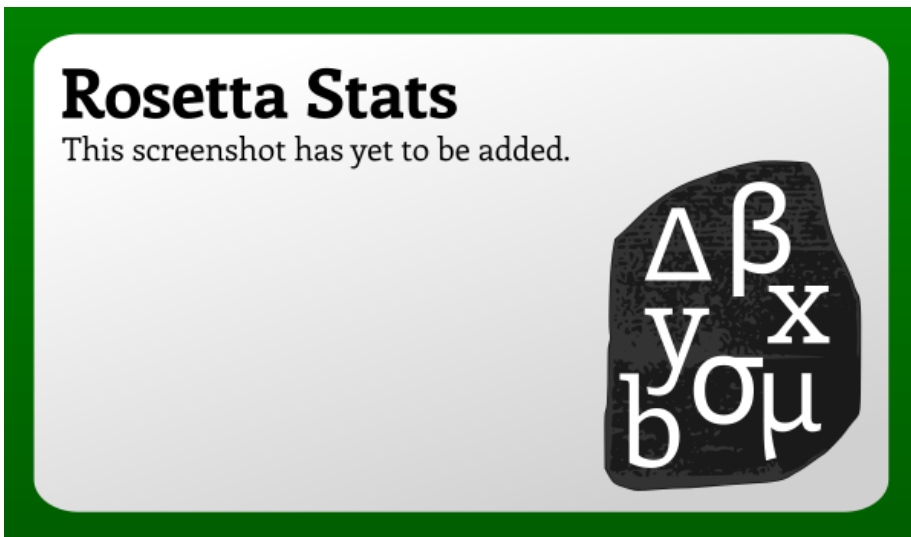


Figure 29.4: A screenshot placeholder

29.2 Input: jamovi

29.3 Input: R

29.4 Input: SPSS

29.5 Output: jamovi

29.6 Output: R



Figure 29.5: A screenshot placeholder

29.7 Output: SPSS

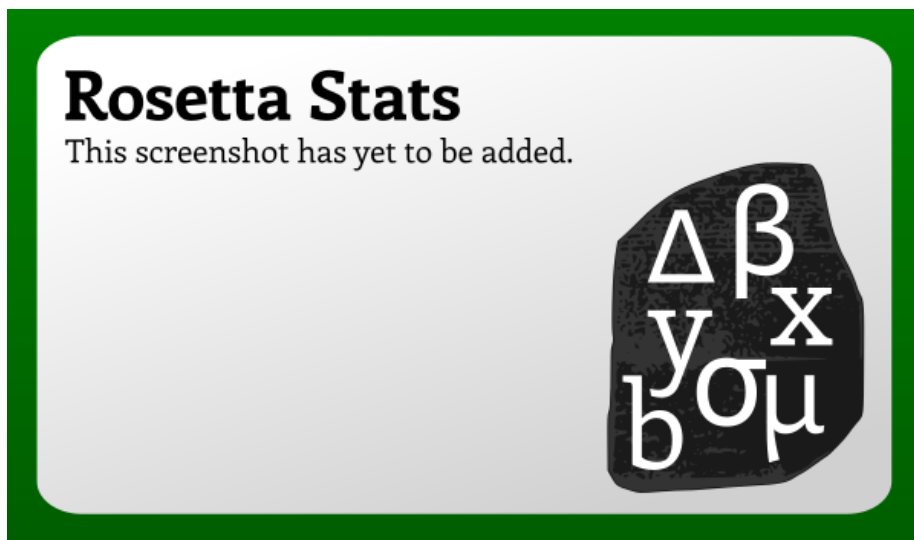


Figure 29.6: A screenshot placeholder

Part VII

Appendices and references

Chapter 30

Authors

Because Rosetta Stats is not a traditional book, the list of authors is dynamic. Therefore, they are listed on this page. In addition, this page also lists the roles everybody has.

30.1 Roles

To avoid diffusion of responsibility, the Rosetta Stats project has a set of roles that designate the primary responsibility in a number of domains. These are listed here.

30.1.1 Overarching roles

These are a number of overarching roles relating to the Rosetta Stats project. These are the following.

- *Rosetta Stats Coordination Team*: These people are responsible for the general coordination of the Rosetta Stats project, enforcing the Code of Conduct, deciding to include new languages, et cetera. Currently, these are:
 - Danny Katz
 - Ron Pat-El
 - Gjalt-Jorn Peters
 - Peter Verboon
 - Melissa Wolf
 - Juul Coumans

- *Git, bookdown & CI*: These people are responsible for making sure the git repository, the Bookdown project, and GitLab's continuous integration functionality all keep doing their thing. Currently, these are:
 - Gjalt-Jorn Peters
- *Zotero group management*: These people are responsible for managing the admin memberships of the public Zotero group associated to this project (see https://www.zotero.org/groups/2419994/rosetta_stats). Currently, these are:
 - Gjalt-Jorn Peters

30.1.2 Language-specific roles

This list lists who is responsible for each specific language.

- *jamovi*: Gjalt-Jorn Peters
- *R rosetta*: Gjalt-Jorn Peters
- *R tidyverse*: Melissa Wolf & Danny Katz
- *SAS*: ...
- *Stata*: ...
- *SPSS GUI*: Melissa Wolf
- *SPSS syntax*: Ron Pat-El

30.2 People

These are the people who contribute to Rosetta Stats.

30.2.1 Danny Katz

Danny is a PhD candidate in the school of Education at the University of California, Santa Barbara. His advisor is Andy Maul.

30.2.2 Ron Pat-El

Ron Pat-El works at the Methodology and Statistics department of the faculty of Psychology at the Open University of the Netherlands. ...

30.2.3 Gjalt-Jorn Ygram Peters

Gjalt-Jorn works at the Methodology and Statistics department of the faculty of Psychology at the Open University of the Netherlands. He enjoys leveraging R and other programming languages to address operational, methodological, and theoretical challenges in psychology. A list of the R packages he is involved in is available at <https://gitlab.com/r-packages>. Gjalt-Jorn's area of research of behavior change, and on those relatively rare occasions where he does substantive research, he applies this to nightlife-related risk behavior. His Twitter handle is matherion.

30.2.4 Peter Verboon

Peter Verboon works at the Methodology and Statistics department of the faculty of Psychology at the Open University of the Netherlands. He received his PhD long a ago at the Leiden University in the Netherlands, for research on robust nonlinear multivariate analysis. His present area of interest is in the analysis of longitudinal data, in particular obtained from single case designs (<https://gitlab.com/Verboon/scda---single-case-design-analyses.git>) or ecological momentary assessments (EMA) designs. He developed several R packages: <https://github.com/PeterVerboon/lagnetw.git>, <https://github.com/PeterVerboon/cyclicpkg.git>, and <https://gitlab.com/r-packages/scda.git>.

30.2.5 Melissa Gordon Wolf

Melissa is a PhD student in the school of Education at the University of California, Santa Barbara. She is interested in the design, validation, and analysis of self-report surveys. Her advisor is Andy Maul. To learn more, visit her website: <https://www.melissagwolf.com/>.

Chapter 31

Contribution Guidelines

Rosetta Stats is an Open Science project, both as a product, and in its organisation. That means that you are warmly invited to join in and collaborate.

31.1 How can I contribute?

There are lots of ways to contribute, but most contributions will fall in one of three categories: filling in blanks, adding new analyses, and adding new software.

31.1.1 Filling in blanks

Especially in the early days, there will be many blanks in Rosetta Stats. If you are familiar with a software package and analysis, and you have some time to fill in one of those blanks, that would be great!

31.1.2 Adding new analyses

...

31.1.3 Adding new software

...

31.2 Some rules

31.2.1 Inform yourself

Because of the open nature of this project, enabling both collaboration and long-term maintenance requires implementing some conventions that contributors have to follow. In the case of Rosetta Stats, contributors have to comply with both our style guide and the code of conduct, listed in chapters 32 and 35 respectively. This also implies that these have to be read completely before you start contributing.

31.2.2 No textbook

Rosetta Stats is explicitly *not* a textbook. However, every chapter *does* include a “Read more” section where links to other sources with more background information can be added. Ideally, these are Open Access sources; if you link to a Closed Access source, please indicate that status explicitly.

In addition, the Rosetta Stats Coordination Team (see Appendix 30) may at some point decide to add a section with some theoretical background to complement this Read More section. At the moment, however, Rosetta Stats is strictly a chrestomathy.

Chapter 32

Style Guide

32.1 Filenames

Every chapter is in a file that contains only digits (0–9), lowercases letters (a–z), and dashes (-), followed by a period and the extension, which is always `.Rmd`. The digits determine the location of the chapter in the book. The first digit determines the Part of the book where the chapter is located:

1. **Basics:** Preliminary information, for example about software.
2. **Univariate analyses:** Analyses including exactly one variable, such as means and histograms.
3. **Psychometrics:** Analyses used when developing or evaluating operationalizations of psychological constructs (i.e. measurement instruments or manipulations).
4. **Bivariate analyses:** Analyses including exactly two variables, such as Pearson correlation coefficients, Cohen's d , and t -tests.
5. **Multivariate analyses:** Analyses including three or more variables, either as predictors (e.g. multivariate regression analysis or two-way anova) or criteria (e.g. manova).
6. **Appendices:** Information that is not important for readers, such as this style guide.

Within these Parts, the chapters are ordered based on the second and third digits. However, we only use this to manually set the order for Parts 1 and 6. In the Parts with most of the book's content, Parts 2-4, the chapters should be sorted alphabetically. These chapters should all be named for the analysis they cover. This way, readers can easily find a given analysis. These filenames should always have 10 as second and third digits.

The first three digits should always be followed by exactly three dashes. For example, the chapters about anova and scatterplots are named `410---anova.Rmd` and `410---scatterplot.Rmd`, respectively. Spaces are replaced by one dash. For example, the chapter about coefficient Alpha is named `310---coefficient-alpha.Rmd`.

32.2 Symbols and statistical characters

Symbols and statistics characters should always use LaTeX Math (which can also be used for equations, if necessary). For example, the “d” in Cohen’s d should be placed between single dollar symbols: `d`. This ensures consistency in typography between use in equations in in the running text.

32.3 Chapter and section identifiers

Always *manually* specify the identifier for chapters and sections. Although Bookdown can produce such identifiers automatically, it may sometimes be necessary or desirable to rename chapters or sections, and manually specifying the identifiers allows doing this without breaking cross-references. Identifiers are added after the corresponding heading in between curly braces (accolades), and preceded by a hash (`#`). For example, the identifier of this chapter is specified using `{#style-guide}`.

Identifiers of chapters must always be identical to the filename of the chapter, but omitting the prefixed number and the extension. For example, the filename of this chapter is `960---style-guide.Rmd`.

All section identifiers much start with their chapter identifier, separated by a dash. For example,

32.4 Cross-references

To refer to another chapter or section within a chapter, always use the chapter or section number. These numbers are automatically produced in Bookdown by referring to the identifiers of the chapter or section like this: `\@ref(identifier)`, where `identifier` is replaced by the identifier of the target chapter or section. Note that you will have to include “Chapter” or “Section” in the text yourself, if you want to.

Cross-references should always use this method (i.e. using the number of the chapter or section that is referenced). Even though this is slightly less user-friendly for readers reading the HTML version (compared to named links), using

numbers for cross-referring is the only way that also works for readers of other formats (e.g. the PDF or EPUB versions, which users may print).

32.5 Citations and references

Citations and references use Bookdowns native format. This requires the references to be present in a BibTeX file, and this BibTeX file is automatically generated by the script `download_and_save_bibliography.R` that is ran automatically before the book is rendered. It downloads the references in the Zotero groups “Rosetta Stats”, which is located at https://www.zotero.org/groups/2419994/rosetta_stats. If you contribute to Rosetta Stats, and you need to cite a source not yet in that library, join the group and request

To add a reference, place its unique identifier, prefixed with an @, between straight brackets: `[@navarro_learning_2018]`. You can see the identifier for a reference by running `download_and_save_bibliography.R`. This will show a list of the identifiers in the R console. It will also update `bibliography.bib`, which is then uploaded to the Git repository and is always available for inspection at <https://gitlab.com/sci-ops/rosetta-stats/blob/master/bibliography.bib>.

32.6 Figures

For software that relies on graphical user interfaces Figures are a great way to illustrate how to do something, especially for software using a GUI. When using figures, stick to the following conventions.

32.6.1 Try to avoid figures

This may seem ironic, but is important nonetheless. Whenever content can be presented using text, for example tables, statistical commands, or output, avoid figures. There are two reasons for this.

First, people who are visually impaired or blind rely on screen readers, and if you succumb to the relative effortlessness of screenshotting a table, that virtually makes the table inaccessible to them. However, if the table is specified using Markdown or HTML, screen readers can parse its structure, present it to readers, and facilitate their navigation through the table.

Second, especially for statistical commands, readers will often come to this book exactly to copy-paste commands. This is not directly possible from figures (although OCR software exists to facilitate the process, of course).

32.6.2 Screenshots

If you include screenshots, make sure they do not exceed 1000 pixels in width (ideally even 800 at most) and 1200 pixels in height. If you make them larger, the fonts will no longer be comfortably legible for everybody in the online version, the PDF ebook, or the EPUB ebook.

For software to support easily resizing window, and easily screenshot windows or a part of the screen, see Chapter 34.

32.6.3 Path and filenames

All figures must be placed in the `/img` directory. Their filename must always start with the filename of the chapter where they appear (so figures in this appendix would have `960-style-guide` as a prefix), followed by a unique identifier for the figure within that chapter. For example, section 2.2 contains a figure called `120-software-basics-jamovi-fresh.png`.

32.6.4 How to include a figure

To include figures, always use the `knitr::include_graphics()` function to ensure proper embedding of the figure in all output formats (see the Bookdown book for more details). Call this function in an R chunk that has as a chunk label the filename of the figure without the extension and prefixed number. Therefore, the R chunk used in section 2.2 to embed the figure from file `120-software-basics-jamovi-fresh.png` has as its label `software-basics-jamovi-fresh`.

Also, always include caption for every figure. The caption should ideally describe the figure, with blind readers and readers with impaired eyesight in mind. In addition to this reason for supplying figure captions, this also triggers Bookdown's automatic numbering of the figures, which in turn enables cross-referring to figures.

This is a full example, lifted from section 2.2:

```
```${r software-basics-jamovi-fresh, fig.cap='A fresh install of jamovi.'}
knitr::include_graphics(here::here("img", "120-software-basics-jamovi-fresh.png"));
```
```

It is important to note that chunk labels can only contain single dashes; they cannot include two dashes following each other. For the rest, the chunk labels conform to the rules for chapter names: only lower case letters (a-z), digits (0-9), and (single) dashes (-).

32.6.5 Referring to figures

Refer to figures using the same method as you use for referring to chapters and sections, using the chunk labels with `fig:` prepended as identifier. For example, to refer to the figure from the example from section 2.2, we use `\@ref(fig:software-basics-jamovi-fresh)`, which Bookdown then automatically turn into a number (and hyperlink for the HTML version of the book), which we can then prepend with “Figure” to get a nice reference to Figure 2.1.

32.7 Software-specific guidelines

This section lists conventions for specific software packages.

32.7.1 R

32.7.1.1 Package parsimony

Only use base R or functions from the `rosetta` package that accompanies this book (to which you can of course contribute; the repo is at <https://r-packages.gitlab.com/rosetta>). You should experience a very, very high threshold to require the user to use any other packages than `rosetta` (the only exception to this rule is `ggplot2`, which is already installed with `rosetta`). This convention is based on two things.

First, Rosetta Stats should make it as easy as possible for people to transition to R, so the number of different packages we requires users to use should be somewhat aggressively kept to a minimum. If another package has functionality that is not available in base R or `rosetta`, write a wrapper function in `rosetta`, optimized to mimick the way that specific analysis is specified in other statistical packages, and mimicking the output generated in other statistical packages.

Second, everybody has different preferences for doing things. For example, at the moment (early 2020), the Tidyverse is a popular approach within R, and some people believe the Tidyverse to be better, and others to be worse, than using base R. In five or ten years, different trends will emerge. Rosetta Stats is meant to be agnostic concerning such preferences. This is why we do not use the tidyverse in the R sections.

If you think that despite these considerations, it is warranted to use another package anyway (for example, because developing a wrapper in the `rosetta` package is somehow not possible or desirable), contact the person responsible for the R bits in Rosetta Stats (see Appendix 30).

One exception is the Tidyverse itself. Because the Tidyverse is very popular, but also fundamentally different in how learns to think about working with objects

in R, the Tidyverse has a special section, as if it were a different statistical package.

32.7.1.2 Calling::functions

When providing R code, always prepend functions with the corresponding namespace (i.e. package name). This is an important convention to avoid clashes for users who have loaded packages with the same function names. An added benefit is that this eliminates the need to let users attach the package's namespace using `library()` or `require()`, simplifying the code to provide.

So, instead of using, for example:

```
library(dplyr);  
dat <- rename(dat, newVariableName = oldVariableName);
```

Use:

```
dat <- dplyr::rename(dat, newVariableName = oldVariableName);
```

In addition, whenever using commands that are not part of base R, such as are included in `rosetta`, `ggplot2`, or `tidyverse` packages, caution users that they must have that package installed.

Chapter 33

Empty Chapter Template

33.1 Intro

Enter a brief introduction here

33.1.1 Example dataset

Here, explain which dataset people should open to copy the examples. For example:

This example uses the Rosetta Stats example dataset “pp15” (see Chapter 1 for information about the datasets and Chapter 3 for an explanation of how to load datasets).

33.1.2 Variable(s)

Here, explain which variable(s) people should open to copy the examples. For example:

From this dataset, this example uses variable `highDose_IntentionRAA_intention`.

33.2 Input: jamovi

33.3 Input: R

In R, there are roughly three approaches. Many analyses can be done with base R without installing additional packages. The `rosetta` package accompanies



Figure 33.1: A screenshot placeholder

this book and aims to provide output similar to jamovi and SPSS with simple commands. Finally, the tidyverse is a popular collection of packages that try to work together consistently but implement a different underlying logic than base R (and so, the `rosetta` package).

33.3.1 R: base R

33.3.2 R: rosetta

33.3.3 R: tidyverse

33.4 Input: SPSS

For SPSS, there are two approaches: using the Graphical User Interface (GUI) or specify an analysis script, which in SPSS are called “syntax”.



Figure 33.2: A screenshot placeholder



Figure 33.3: A screenshot placeholder



Figure 33.4: A screenshot placeholder

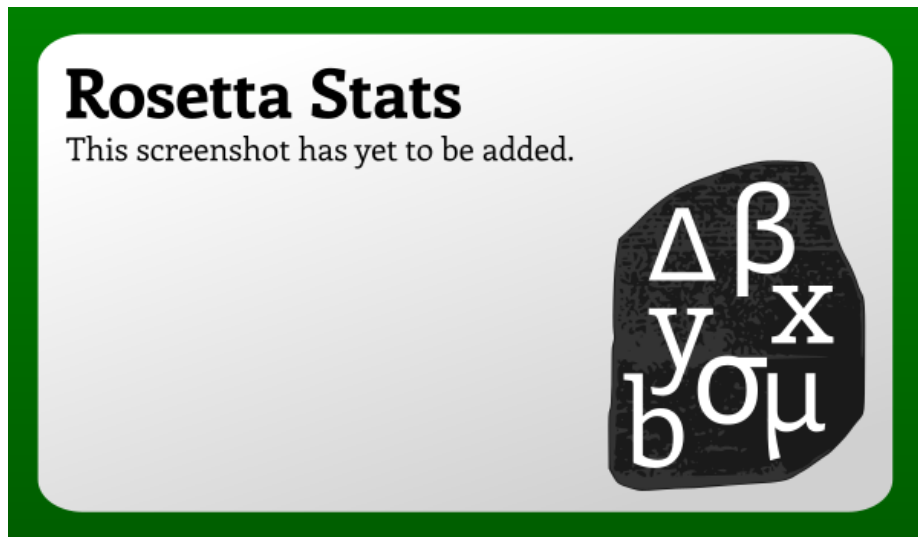


Figure 33.5: A screenshot placeholder



Figure 33.6: A screenshot placeholder



Figure 33.7: A screenshot placeholder

33.4.1 SPSS: GUI

33.4.2 SPSS: Syntax

33.5 Output: jamovi

33.6 Output: R



Figure 33.8: A screenshot placeholder

33.7 Output: SPSS

33.8 Read more

Here, you can list one or more sources with background reading, for example:

If you would like more background on this topic, you can read more in these sources:

- Learning Statistics with R (Navarro, 2018): section XXXXXXXXXXXX, page XXXXXXXXXXXX (or follow this link for the Bookdown version)



Figure 33.9: A screenshot placeholder

Chapter 34

Software and technologies

This section lists the technologies and software we use to produce Rosetta Stats.

34.1 Bookdown

This book is produced using Bookdown. Bookdown is a FLOSS R package that takes a set of Markdown files (optionally including R chunks) and renders them into an online book as well as a PDF and EPUB version. The Bookdown book is Open Access and available at <https://bookdown.org/yihui/bookdown/>

34.2 Zotero

Zotero is an excellent FLOSS reference manager. The references in this book are read from the Zotero group at https://www.zotero.org/groups/2419994/rosetta_stats using Zotero's API.

34.3 Git

We use git for version control. Git is a decentralized FLOSS version control system.

34.4 GitLab

We use GitLab as a git repository management suite. GitLab is a FLOSS alternative to another well-known git repository management suite, GitHub.

34.5 Greenshot

To create screenshots, you can use the FLOSS package Greenshot: <https://getgreenshot.org/>

34.6 Sizer

To realise consistent figure sizes throughout the book, you can use the freeware application Sizer: <http://www.brianapps.net/sizer4/>

It allows you to easily resize a window to 800x600 (or another resolution), for example by pressing the shortcut key (by default, CTRL-WIN-Z).

Chapter 35

Code of Conduct

For now, we simply adhere to the Contributor Covenant Code of Conduct.

35.1 Our Pledge

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone, regardless of age, body size, visible or invisible disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

We pledge to act and interact in ways that contribute to an open, welcoming, diverse, inclusive, and healthy community.

35.2 Our Standards

Examples of behavior that contributes to a positive environment for our community include:

- Demonstrating empathy and kindness toward other people
- Being respectful of differing opinions, viewpoints, and experiences
- Giving and gracefully accepting constructive feedback
- Accepting responsibility and apologizing to those affected by our mistakes, and learning from the experience
- Focusing on what is best not just for us as individuals, but for the overall community

Examples of unacceptable behavior include:

- The use of sexualized language or imagery, and sexual attention or advances of any kind
- Trolling, insulting or derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or email address, without their explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

35.3 Enforcement Responsibilities

Community leaders are responsible for clarifying and enforcing our standards of acceptable behavior and will take appropriate and fair corrective action in response to any behavior that they deem inappropriate, threatening, offensive, or harmful.

Community leaders have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, and will communicate reasons for moderation decisions when appropriate.

35.4 Scope

This Code of Conduct applies within all community spaces, and also applies when an individual is officially representing the community in public spaces. Examples of representing our community include using an official e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event.

35.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported to the community leaders responsible for enforcement at [INSERT CONTACT METHOD]. All complaints will be reviewed and investigated promptly and fairly.

All community leaders are obligated to respect the privacy and security of the reporter of any incident.

35.6 Enforcement Guidelines

Community leaders will follow these Community Impact Guidelines in determining the consequences for any action they deem in violation of this Code of Conduct:

35.6.1 1. Correction

Community Impact: Use of inappropriate language or other behavior deemed unprofessional or unwelcome in the community.

Consequence: A private, written warning from community leaders, providing clarity around the nature of the violation and an explanation of why the behavior was inappropriate. A public apology may be requested.

35.6.2 2. Warning

Community Impact: A violation through a single incident or series of actions.

Consequence: A warning with consequences for continued behavior. No interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, for a specified period of time. This includes avoiding interactions in community spaces as well as external channels like social media. Violating these terms may lead to a temporary or permanent ban.

35.6.3 3. Temporary Ban

Community Impact: A serious violation of community standards, including sustained inappropriate behavior.

Consequence: A temporary ban from any sort of interaction or public communication with the community for a specified period of time. No public or private interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, is allowed during this period. Violating these terms may lead to a permanent ban.

35.6.4 4. Permanent Ban

Community Impact: Demonstrating a pattern of violation of community standards, including sustained inappropriate behavior, harassment of an individual, or aggression toward or disparagement of classes of individuals.

Consequence: A permanent ban from any sort of public interaction within the project community.

35.7 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 2.0, available at https://www.contributor-covenant.org/version/2/0/code_of_conduct.html.

Community Impact Guidelines were inspired by Mozilla's code of conduct enforcement ladder.

For answers to common questions about this code of conduct, see the FAQ at <https://www.contributor-covenant.org/faq>. Translations are available at <https://www.contributor-covenant.org/translations>.

Chapter 36

References

Bibliography

Cronbach, L. J. and Shavelson, R. J. (2004). My Current Thoughts on Coefficient Alpha and Successor Procedures. *Educational and Psychological Measurement*, 64(3):391–418.

Navarro, D. (2018). *Learning Statistics with R*. New South Wales, Australia, 0.6 edition.