Introduction to R programming

https://github.com/psboonstra/michR

MICHR Bayesian Workshop Series, Fall 2019

Learning objectives

1. Develop a basic familiarity with R and RStudio
2. Practice:

- reading in data
- wrangling data (a little bit)
- creating simple plots
- saving your work

What is R?

A free...

- ...open-source programming language and statistical environment
- ...calculator
- ...simulator: use R to mimic a roll of a dice, flip of a coin, draw from a bell curve, a time to the occurrence of some event
- ...means of turning databases, spreadsheets, text files, websites, etc. into actual data that can be manipulated, visualized, and analyzed

R dates back to the mid-90's. A non-profit foundation holds the copyright for R, and it is developed and maintained by a core group of contributors (https://www.r-project.org/contributors.html)

R gives immediate access to thousands of functions including mathematical operations, algorithms, basic plotting tools, and standard statistical techniques, as well as several dozen built-in datasets

Anyone can implement their own idea in R and contribute a *package* to share

Why use R?

- Freely available, open source
- Dynamic and state-of-the-art statistical environment
- Knowledgeable, helpful support community
- Publication-quality graphics

What is RStudio?

The name of a for-profit company (circa 2010) as well as its primary product

Why use RStudio?

- Recognizes and highlights R code syntax
- Manages all of your R projects
- Makes it easy to do reproducible research
- Provides a safety net: graceful crashing, debugging, history
- Integration withfancy data-science-y tools
- Free cloud-based account

How do you speak R?

R is imperative: you send commands to the console, R follows them, and it then it waits for more commands

In the process of following your commands, the environment may have changed

nouns:

```
1:6;
```

```
## [1] 1 2 3 4 5 6
```

1:6 denotes the set of integers from 1 to 6. The numbers are printed and then lost

verbs:

these are R's actions and usually involve functions

mean() calculates the average of whatever is provided as the argument for x:

```
mean(x = 1:6);
```

```
## [1] 3.5
```

== tests for equality between the left and the right objects:

```
(1:6) == 3;
```

```
## [1] FALSE FALSE  TRUE FALSE FALSE FALSE
```

If you want to declare something, then use the assignment operator <-:

```
x <- 3^2;
```

which means "set x to be whatever value three squared evaluates to". After following this command, there exists a new variable called x

R in action: Rats!

"the primary aim was to investigate the effect of inhibition of the production of testosterone in male Wistar Rats on their craniofacial growth" (Verbeke and Molenberghs 2009). Also analyzed in Verbeke and Lesaffre (1999), Verbeke and Molenberghs (2003), Gelman et al. (2005)

Variables:

(1) obs: observation number (integer)

(2) treat: treament group ('con': control; 'hig': high dose; 'low': low dose)

(3) rat: rat identification number (integer)

(4) age: age of the rat in days when the observation was made (integer)

(5) response: height of the rat's skull (double)

```
# The pound sign is R's comment character
# load a collection of helper functions, including read_csv
library(tidyverse);
# read in a csv file and save as object called 'rat_data'
rat_data <- read_csv(file = "rats.csv");
# look at the beginning of rat_data
rat_data;
```

```
## # A tibble: 252 x 5
##      obs treat   rat   age response
##    <int> <chr> <int> <int>    <dbl>
##  1     1 con       3    50     75.0
##  2     2 con       3    60     78.6
##  3     3 con       3    70     79.9
##  4     4 con       3    80     80.8
##  5     5 con       5    50     68.0
##  6     6 con       6    50     76.0
##  7     7 con       6    60     78.3
##  8     8 con       6    70     81.2
##  9     9 con       9    50     70.8
## 10    10 con       9    60     75.0
## # ... with 242 more rows
```

```
rat_data %>% # take the rat_data data,
  group_by(treat) %>% # group by treatment,
  # then, within each group, calculate:
  summarize(mean_response = mean(response), # average response
            sd_response = sd(response)); # sd of response
```

```
## # A tibble: 3 x 3
##   treat mean_response sd_response
##   <chr>         <dbl>       <dbl>
## 1 con            77.2        4.11
## 2 hig            76.9        4.24
## 3 low            78.4        4.68
```

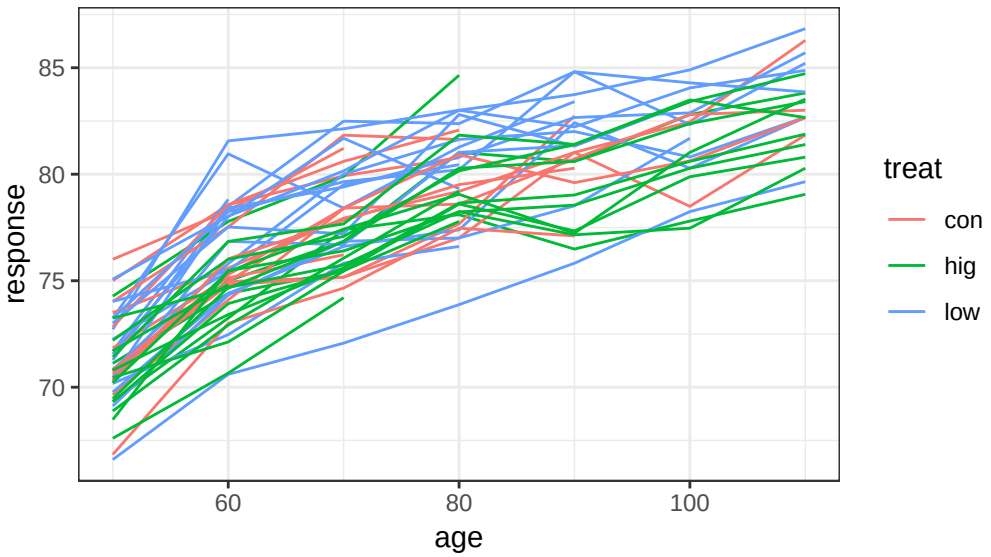
```
rat_data %>% # take rat_data,
  group_by(age) %>% # group by age at observation,
  # within each group, calculate
  summarize(mean_response = mean(response), # average response
            sd_response = sd(response)); # sd of response
```

```
## # A tibble: 7 x 3
##     age mean_response sd_response
##   <int>         <dbl>       <dbl>
## 1    50          71.2        2.12
## 2    60          75.7        2.40
## 3    70          77.7        2.36
## 4    80          79.8        2.20
## 5    90          80.5        2.47
## 6   100          81.5        2.11
## 7   110          83.0        2.08
```

```
rat_data %>% # take rat_data,
  filter(age == 50) %>% #limit only to observations taken at day 50
  summarize(mean_response = mean(response), # average response
            sd_response = sd(response)); # sd of response
```

```
## # A tibble: 1 x 2
##   mean_response sd_response
##           <dbl>       <dbl>
## 1          71.2        2.12
```

```
ggplot(data = rat_data) +
  geom_line(mapping =
              aes(x = age, # x axis
                  y = response, # y axis
                  color = treat, # color by treatment
                  group = rat)); # group by rat
```



Fundamental template for ggplots

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>));
```

From Master the Tidyverse, 01-Visualize-Data.pdf (https://github.com/rstudio/master-the-tidyverse/)

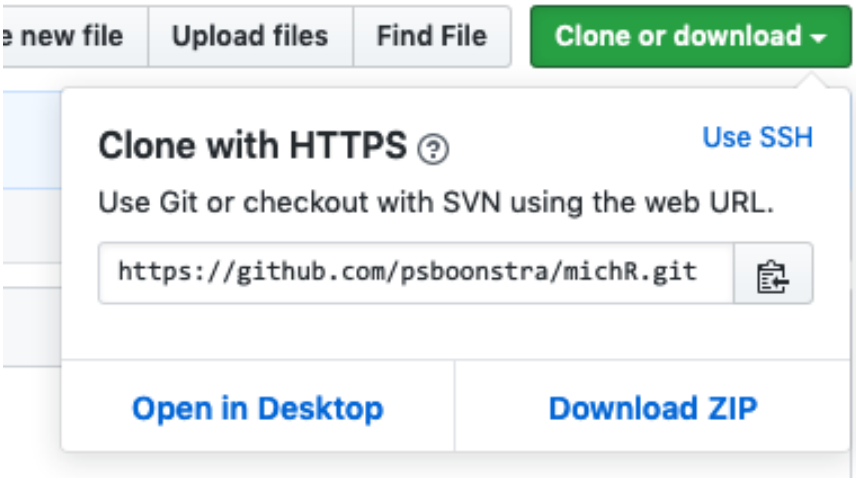
`<DATA>` is the dataset object you want to plot from, with observations in rows variables in columns

`<GEOM_FUNCTION>` describes the type of plot you want: `geom_point`, `geom_line`, `geom_boxplot`, `geom_histogram`, etc

`<MAPPINGS>` give the comma separated list of which variables to map to which aesthetics, e.g. `x = age` means to use age for the aesthetic corresponding to the xaxis

Try it out: Option 1

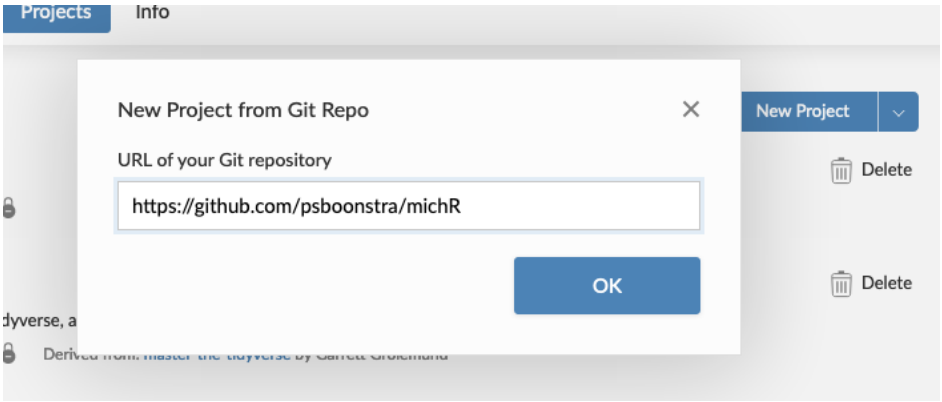
a. Download `R` (https://cran.r-project.org/)
b. Download `RStudio` to interface with `R` (https://www.rstudio.com/)
c. Go to the workshop URL: https://github.com/psboonstra/michR and click 'Clone or download' > 'Download ZIP'



d. Look for the folder 'michR-master.zip' in your downloads folder, unzip
e. Double-click the file 'michR.Rproj'

Try it out: Option 2

a. Go to https://rstudio.cloud/ > Get Started
b. Create an account
c. Click the dropdown menu *next to* the New Project button, select 'New Project from Git Repo', and enter the workshop URL of the workshop repository: https://github.com/psboonstra/michR



Exercise 1

You should be in `RStudio` now (either of the two options above)

Press and hold *Cmd+O* (*Ctrl+O* for Windows) and you should see the file `01-introduction.Rmd`

Take 15 minutes to work through `01-introduction.Rmd`. Anything with an `<XXX>` is something that requires an answer from you

15:00

1.

Run the above code chunk again. What happened? Why do you think this happened?

Nothing should have happened when you ran the code a second time, because running `!require(tidyverse)` now returns `FALSE`, causing the contents in the subsequent brackets to be skipped

2.

For the above two plots, what 'geom' function was used, and what aesthetics were provided? Is there anything you would change about either of the plots?

In the first plot, the `geom_boxplot` function was used, with the `mort30` and `hour` variables being assigned to the `x` and `y` aesthetics, respectively

In the second plot, the `geom_smooth` function was used, with the `hour` and `complication` variables being assigned to the `x` and `y` aesthetics, respectively. The complication variable had to first be coerced to a numeric variable so that it could be smoothed

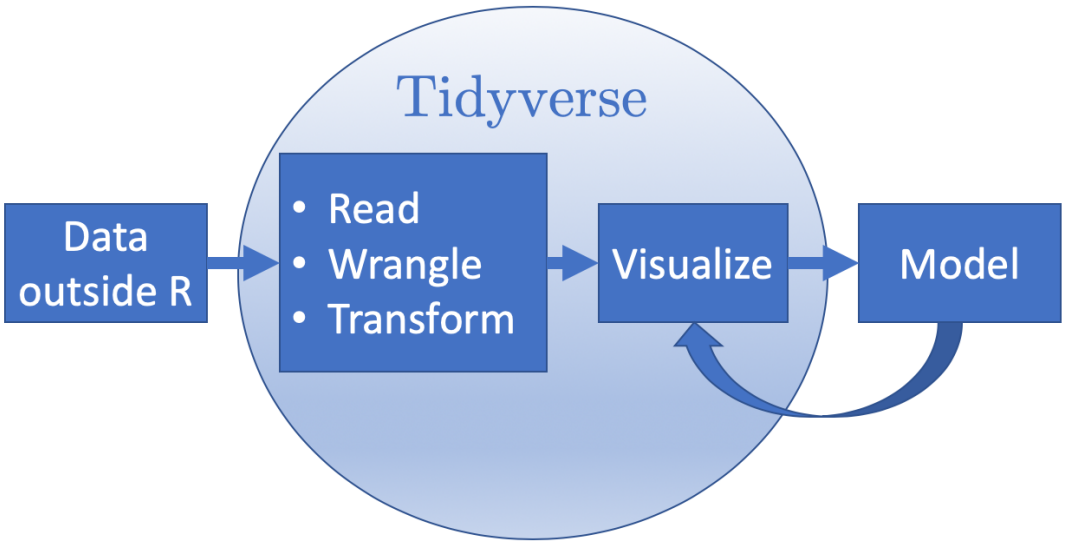
In both cases, I'd probably suggest to make the axis labels and tick marks more informative.

3.

What is the difference between the above two barplots? In other words, what is the effect of adding 'position = "fill"'?

Adding position = "fill" tells ggplot to make the height of each bar proportional to 1.

Note that this is distinct from the fill aesthetic, which tells R how to differentially color the bars.



Visual adapted from **Introduction**, Wickham and Grolemund (2016)

The `tidyverse` helps get data from the spreadsheet into `R` and wrangles it in such a way that it can be plotted, summarized, and modeled in the right way

Most `tidyverse` functions are centered around manipulating a special **noun** called a `data.frame`, which is the main way that `R` stores data

The `tidyverse` has a special `data.frame` called a `tibble`, which is nearly identical but with more thoughtful features

There are six especially useful functions (**verbs**) from the `tidyverse` specifically designed for `tibbles` (and `data.frames`):

1. `filter()`: pick observations (rows) base on values
2. `arrange()`: reorder the observations (rows)
3. `summarize()`: collapse many observations (rows) down to a single summary
4. `mutate()`: create new variables (columns) from existing variables
5. `select()`: pick variables (columns) based on names
6. `group_by()`: group all subsequent analyses by the group definitions specified herein.

Paraphrased from **Data transformation**, Wickham and Grolemund (2016)

Exercise 2

Take 10 minutes to work through `02-tidyverse_verbs.Rmd`

As before, anything with an `<XXX>` requires an answer from you

10:00

1.

describe what each of the following chunks does

```
surgery %>%
  group_by(baseline_cancer, baseline_cvd) %>%
  summarize(number = n(),
            mean_mortality = mean(mort30 == "1"),
            mean_complication = mean(complication == "1"));
```

```
## # A tibble: 4 x 5
## # Groups:   baseline_cancer [?]
##   baseline_cancer baseline_cvd number mean_mortality mean_complication
##   <chr>           <chr>         <int>          <dbl>             <dbl>
## 1 0               0             10963        0.00192             0.113
## 2 0               1             10080        0.00427             0.134
## 3 1               0              4862        0.00720             0.144
```

```
## 4 1                1              6096        0.00640             0.159
```

This groups observations into four categories: whether or not the patient had baseline cancer crossed with whether or not the patient had baseline cvd and, within each group, counts the number of surgeries and the average mortality and complication rate

2.

describe what each of the following chunks does

```
surgery %>%
  filter(!is.na(bmi)) %>%
  group_by(hour <= 8) %>%
  summarize(number = n(),
            mean_bmi = mean(bmi),
            mean_mortality = mean(mort30 == "1"),
            mean_complication = mean(complication == "1"));
```

```
## # A tibble: 2 x 5
##   `hour <= 8` number mean_bmi mean_mortality mean_complication
##   <lgl>        <int>    <dbl>          <dbl>              <dbl>
## 1 FALSE        18606     29.4        0.00478              0.137
## 2 TRUE         10105     29.5        0.00277              0.124
```

Removes the observations with a missing BMI and groups the remaining observations based upon whether the surgery started no later than 8am. These groups are summarized with respect to total counts, the average BMI of the patient, the average mortality rate, and the average complication rate

3.

describe what each of the following chunks does

```
surgery %>%
  filter(baseline_cvd == "1" | baseline_diabetes == "1" | baseline_pulmonary == "1") %>%
  summarize(number = n(),
            mean_mortality = mean(mort30 == "1"),
            mean_complication = mean(complication == "1"));
```

```
## # A tibble: 1 x 3
##   number mean_mortality mean_complication
##    <int>          <dbl>             <dbl>
## 1  18170        0.00512             0.145
```

Limited only to those patients with either baseline CVD, diabetes, or pulmonary disease, report the total count, the average mortality rate, and the average complication rate.

4.

describe what each of the following chunks does

```
surgery %>%
  mutate(complication_rsi_groups = cut(complication_rsi, breaks = quantile(complication_rsi), include.lowest =
  group_by(complication_rsi_groups) %>%
  summarize(mean_complication = mean(complication == "1"));
```

```
## # A tibble: 4 x 2
##   complication_rsi_groups mean_complication
##   <fct>                             <dbl>
## 1 [-4.72,-0.84]                    0.0953
## 2 (-0.84,-0.27]                    0.111
## 3 (-0.27,0]                        0.0924
```

```
## 4 (0,13.3]                          0.324
```

Create a new variable called `complication_rsi_groups`, which is a categorical variable defined by the quartiles of the numerical complication risk stratification index. Then, grouping based upon this variable, report the mean complication rate

5.

Do any of the above chunks store their results as a new object? If so, what is the name of the new object? If not, what would you need to add to the chunk to save the result in a new object?

No, none of them do. You would need to put the assignment operator, `<-` or `=`, at the beginning of the expression and save the result to a variable name.

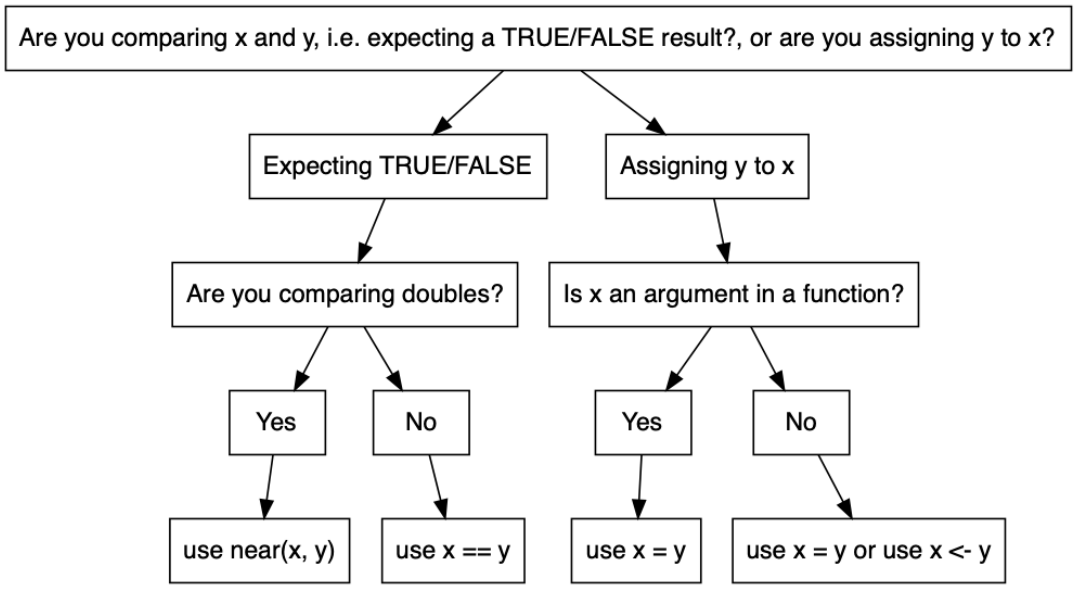
6.

Comparing those surgeries that occurred in July or August versus all other months, which group had the higher mortality rate? Complete the following chunk and use the results to answer the question.

```
#This chunk will not run until you've replaced the <XXX> below!
#After you've done so, replace the ```{r, eval = FALSE} above with ```{r}
surgery %>%
  mutate(jul_aug_surgery = month %in% c(7, 8)) %>%
  group_by(jul_aug_surgery) %>%
  summarize(mean_mortality = mean(mort30 == "1"));
```

It would appear that the hypothesis of the study is not true, namely that the mortality rate in the July / August months (when new surgical residents start) is not appreciably different.

== or = or <- or something else?



Example When to use ==

Write `filter(surgery, gender == "1")` or `surgery %>% filter(gender == "1")` to tell `R` to "take the dataset `surgery` and limit it to the rows for which the variable `gender` equals `1` (male)"

```
filter(surgery, gender == "1");
```

```
## # A tibble: 14,768 x 25
##    ahrq_ccs   age gender race  asa_status   bmi baseline_cancer
##    <chr>    <dbl> <chr>  <chr> <chr>      <dbl> <chr>
##  1 <Other>   67.8 1      1     1           28.0 0
##  2 <Other>   71   1      1     2           32.2 0
##  3 <Other>   56.3 1      2     1           24.3 1
##  4 <Other>   56.6 1      3     3           64.6 0
##  5 <Other>   66.2 1      1     2           28.0 0
##  6 <Other>   49.4 1      1     2           41.3 0
##  7 <Other>   38.8 1      1     1           26.7 0
##  8 <Other>   53.1 1      1     1           25.0 0
##  9 <Other>   66.1 1      1     1           25.7 0
## 10 <Other>   57.6 1      1     2           36.3 0
## # ... with 14,758 more rows, and 18 more variables: baseline_cvd <chr>,
## #   baseline_dementia <chr>, baseline_diabetes <chr>,
## #   baseline_digestive <chr>, baseline_osteoart <chr>,
## #   baseline_psych <chr>, baseline_pulmonary <chr>,
## #   baseline_charlson <chr>, mortality_rsi <dbl>, complication_rsi <dbl>,
## #   ccsMort30Rate <dbl>, ccsComplicationRate <dbl>, hour <dbl>, dow <int>,
## #   month <int>, moonphase <chr>, mort30 <chr>, complication <chr>
```

Example When to use `near`

```
# this is mathematically incorrect:
1/3 == (1/2 - 1/6);
```

```
## [1] FALSE
```

```
# because the comparison loses precision:
1/3 - (1/2 - 1/6);
```

```
## [1] -5.55e-17
```

```
# do this instead:
near(1/3, 1/2 - 1/6);
```

```
## [1] TRUE
```

Example When to use `=`

Write `summarize(surgery, mean_complication_rsi = mean(complication_rsi))` to tell `R` to "reduce the dataset `surgery` to a 1 x 1 `tibble` (`data.frame`) containing the mean of `complication_rsi` in a column called `mean_complication_rsi`"

```
summarize(surgery, mean_complication_rsi = mean(complication_rsi));
```

```
## # A tibble: 1 x 1
##   mean_complication_rsi
##                   <dbl>
## 1                -0.409
```

Example When to use `<-` or `=`

```
# Both of these options do the same thing:
foo = summarize(surgery, mean_complication_rsi = mean(complication_rsi));

foo <- summarize(surgery, mean_complication_rsi = mean(complication_rsi));
```

```
# But don't do this:
foo <- summarize(surgery, mean_complication_rsi <- mean(complication_rsi));
```

Saving your work

Notebooks (File > New File > R Notebook creates a new file ending in `.Rmd.`) are one way to develop your work:

- `R` code is identified with code chunks
- You can wax eloquently in between your fancy statistical analysis using free text
- Knit your `.Rmd` file into a beautiful finished product (html, pdf, word, etc)

The older and probably more common means of doing `R` analyses is the `R` script: File > New File > R Script:

Main difference between notebooks and scripts is that in scripts everything that is not commented must be proper `R` code

Exercise 3

Take 10 minutes to work through `03-saving_work.R`

10:00

How does Phil use `R`?

For research

I will create a new project that contains:

(i) one script for functions that I write that implement my new methodology

(ii) other script(s) for functions that analyze the data or conduct the simulation study

(iii) sometimes I will write the manuscript directly in an `.Rmd` file

For teaching

I will create an `.Rmd` notebook for each lecture. I will knit it into html slides to project and knit the identical notebook into a pdf handout

For consulting

I will usually write my report in an `.R` script but make generous use of knitting functionality

How to find help

- Type `?<function_name>` in the `R` console, e.g. `?mean`
- *Master the tidyverse* workshop slides by Garret Grolemund (https://github.com/rstudio-education/master-the-tidyverse)
- Search in Stack Overflow (http://stackoverflow.com/questions/tagged/r)
- Google your question, e.g. "how to invert a matrix R"
- Cheat sheets: https://www.rstudio.com/resources/cheatsheets
- "The `R` Inferno" (Burns 2010) (all of the terrible mistakes you can make in `R`)
- Books

- `R` for Data Science (Wickham and Grolemund 2016) (https://r4ds.had.co.nz/)
- `ggplot2`: Elegant Graphics for Data Analysis (Wickham 2009) (https://t.co/ZQ78SgbAe1)
- Advanced `R` (Wickham 2014) (http://adv-r.had.co.nz/)
- Introductory Statistics with `R` (Dalgaard 2008) (https://link.springer.com/book/10.1007%2F978-0-387-79054-1)
- `lme4`: Mixed-effects modeling with `R` (Bates 2010) (http://lme4.r-forge.r-project.org/book/)
- Linear mixed-effects models using `R`: A step-by-step approach (Gałecki and Burzykowski 2013) (https://link.springer.com/book/10.1007%2F978-1-4614-3900-4)
- Software for Data Analysis: Programming with `R` (Chambers 2008) (https://link.springer.com/book/10.1007%2F978-0-387-75936-4)
- Other freely downloadable books: https://www.r-statistics.com/2009/10/free-statistics-e-books-for-download/

References

Bates, Douglas M. 2010. *Lme4: Mixed-Effects Modeling with R*. Berlin.

Burns, Patrick. 2010. "The R Inferno." http://www.burns-stat.com/pages/Tutor/R_inferno.pdf.

Chambers, John. 2008. *Software for Data Analysis: Programming with R*. Springer Science & Business Media.

Dalgaard, Peter. 2008. *Introductory Statistics with R*. Springer Science & Business Media.

Gałecki, Andrzej, and Tomasz Burzykowski. 2013. *Linear Mixed-Effects Models Using R: A Step-by-Step Approach*. Springer Science & Business Media.

Gelman, Andrew, Iven Van Mechelen, Geert Verbeke, Daniel F Heitjan, and Michel Meulders. 2005. "Multiple Imputation for Model Checking: Completed-Data Plots with Missing and Latent Data." *Biometrics* 61 (1). Wiley Online Library: 74–85.

Verbeke, Geert, and Emmanuel Lesaffre. 1999. "The Effect of Drop-Out on the Efficiency of Longitudinal Experiments." *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 48 (3). Wiley Online Library: 363–75.

Verbeke, Geert, and Geert Molenberghs. 2003. "The Use of Score Tests for Inference on Variance Components." *Biometrics* 59 (2). Wiley Online Library: 254–62.

———. 2009. *Linear Mixed Models for Longitudinal Data*. Springer Science & Business Media.

Wickham, Hadley. 2009. *Ggplot2: Elegant Graphics for Data Analysis*. Springer Science & Business Media.

———. 2014. *Advanced R*. CRC Press.

Wickham, Hadley, and Garrett Grolemund. 2016. *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. " O'Reilly Media, Inc."