

The StartR Pack

An Introduction to Reproducible Data Analysis with R

Dr. Matthew Sigal

March 3rd, 2020

Today's Goals

- 1 Introduce R, what can it do, why you might want to learn it, and some basics of working with the program
- 2 Introduction to reproducible analysis. . .
 - a) in HTML (via Rmarkdown)
 - b) in APA style (via papaja)
- 3 Provide support links and package recommendations

Files for this workshop can be downloaded from:

<https://tinyurl.com/startr2020>

Before we begin:

- 1 Download the files from the above link and place them in a folder.
- 2 Open 0-RunFirst.R.
- 3 Highlight all of the code.
- 4 Click “Run” to install the packages required for today’s workshop.

What is R?

- Also known as: **The R Language for Statistical Computing**
- ...Is, unsurprisingly, an object-orientated programming language designed for statistical computing and graphics.
- And also has the very desirable cost of...

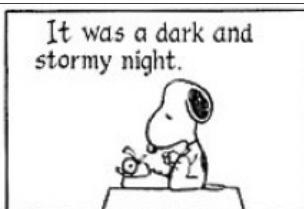


What is R?

- Also known as: **The R Language for Statistical Computing**
- ...Is, unsurprisingly, an object-orientated programming language designed for statistical computing and graphics.
- And also has the very desirable cost of...
- **FREE!**



The current version of R is
3.6.2, code-named:
"Dark and Stormy Night".



Who is R?

- From S to R:
 - S was created by John Chambers and colleagues at Bell Laboratories in the 1970s.
 - R is a re-implementation of S that began development in 1993 by Ross Ihaka and Robert Gentleman at University of Auckland, New Zealand.
- Since 1997, the R Development Core Team (which includes John Chambers) maintains and oversees updates to the R source code
- The R Foundation, a non-profit organization, oversees the design and evolution of R.

However, the R Project is actually a large scale effort and features the work thousands of contributors world-wide.

Who is R?

- From S to R:
 - S was created by John Chambers and colleagues at Bell Laboratories in the 1970s.
 - R is a re-implementation of S that began development in 1993 by Ross Ihaka and Robert Gentleman at University of Auckland, New Zealand.
- Since 1997, the R Development Core Team (which includes John Chambers) maintains and oversees updates to the R source code
- The R Foundation, a non-profit organization, oversees the design and evolution of R.

However, the R Project is actually a large scale effort and features the work thousands of contributors world-wide.

- Users are able to develop their own collections of functions that can be “packaged” and hosted for public distribution on the Comprehensive R Archive Network (<http://cran.r-project.org/>)
- As of February 2020, CRAN hosts **15,329** packages and counting!

What can R do for me?

- Open source – No black boxes!
- Primarily designed as a programming language for *statistical operations*, and almost every statistical method, especially those on the cutting edge, have R implementations.
- One of R's strengths is the ease with which well-designed publication-quality plots can be produced.
- Additional packages provide a framework for everything from running spatial analyses, to text mining, to the construction of interactive visualizations, and even to the creation of...

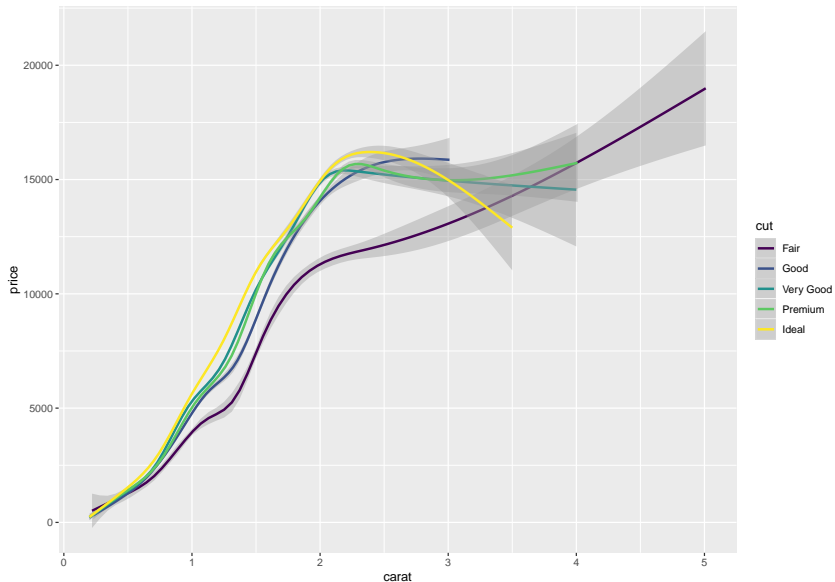
What can R do for me?

- Open source – No black boxes!
- Primarily designed as a programming language for *statistical operations*, and almost every statistical method, especially those on the cutting edge, have R implementations.
- One of R's strengths is the ease with which well-designed publication-quality plots can be produced.
- Additional packages provide a framework for everything from running spatial analyses, to text mining, to the construction of interactive visualizations, and even to the creation of...

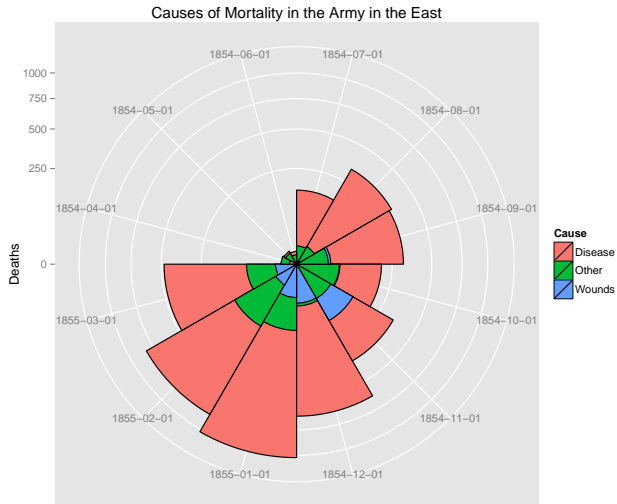
this presentation!

So... What can R do?

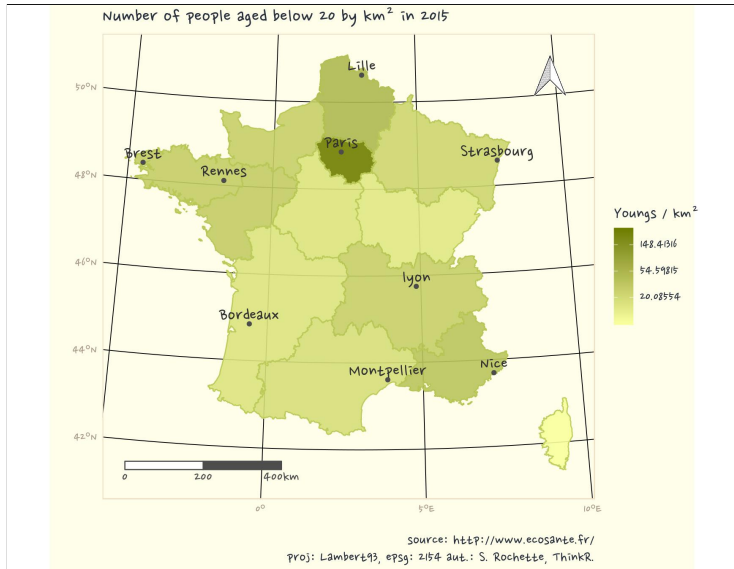
Non-Linear Regression



Polar Coordinate Plots

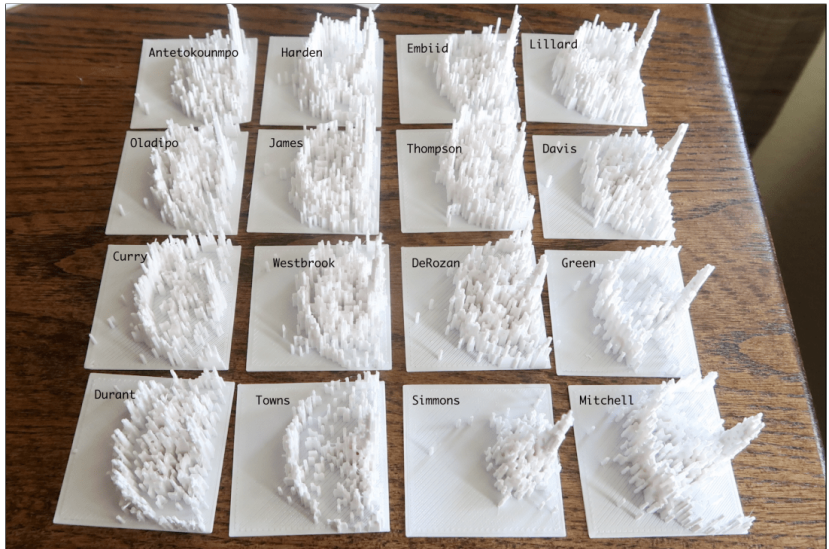


Pretty Maps



From statnmap.com

3-D Printing



From flowingdata.com

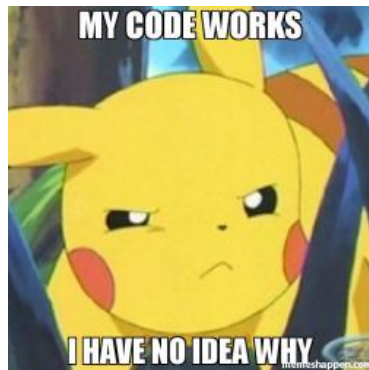
Construct and Solve Mazes in Minecraft



From revolutionanalytics.com

Oh, R...

- A challenge and strength of R is that it is entirely syntax based.
- You need to become familiar with handling objects, calling functions, and writing scripts.
- While R operates on spreadsheets, it is not 'spreadsheet software'.
- Learning R can will cause some frustration, however it is well worth the effort!



Get R

- R is available for Windows, Mac OS X, and Linux.
- Base R can be downloaded from the R project website cran.r-project.org/

The Comprehensive R Archive Network

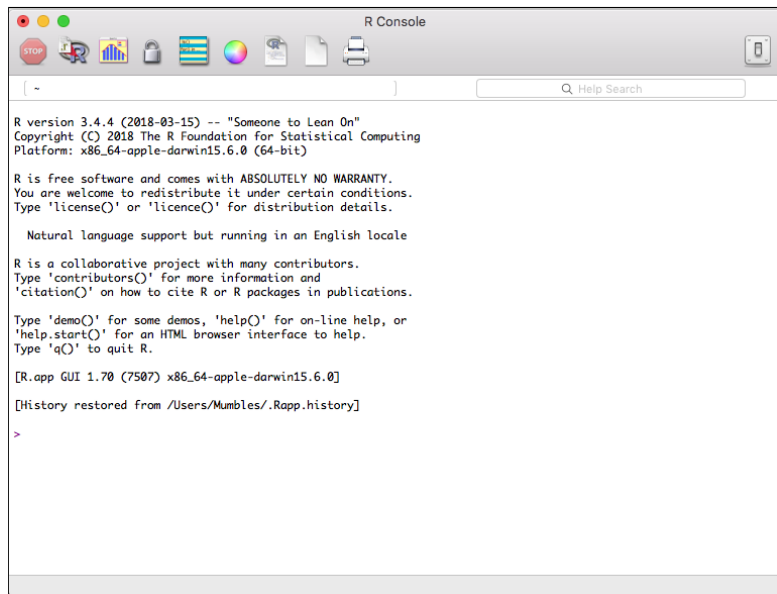
Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Introduction to R



```
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (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.

[R.app GUI 1.70 (7507) x86_64-apple-darwin15.6.0]

[History restored from /Users/Mumbles/.Rapp.history]

>
```

Software, Revisited

Unfortunately, base R leaves a lot to be desired. Fortunately, we have something better: **RStudio**.



- A graphical user interface (GUI) for R
- <http://www.rstudio.com/>
- Currently on version 1.2.5033
- Available for Windows, Mac OS, and Linux.

Software, Revisited

Unfortunately, base R leaves a lot to be desired. Fortunately, we have something better: **RStudio**.



- A graphical user interface (GUI) for R
- <http://www.rstudio.com/>
- Currently on version 1.2.5033
- Available for Windows, Mac OS, and Linux.
- **FREE!**

RStudio

Project: (None)

Console

```
~/SFUVault/CURRENT_WORK/CURRENT_TEACHING/2020-01-PSYC301/Lectures/ ➤  
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.  
  
> |
```

Environment

Global Environment

Environment is empty

Files

R: The Binomial Distribution

Binomial (stats)

The Binomial Distribution

Description

Density, distribution function, quantile function and random generation for the binomial distribution with parameters size and prob.

This is conventionally interpreted as the number of 'successes' in size trials.

Usage

```
dbinom(x, size, prob, log = FALSE)  
pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)  
qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)  
rbinom(n, size, prob)
```

Arguments

x, q	vector of quantiles.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

9:41 (Top Level) R Script

Source

```
1 diamonds %>%  
2   factor(. $carat,  
3     levels = c(0.7),  
4     labels = c("Good")) -> diamonds2  
5 head  
6  
7  
8 diamonds %>%  
9   fct_recode(. $cut, "Good" = "GOOD") %>%  
10  str  
11  
12 fct_recode(diamonds$cut, GOOD = "Good")  
13  
14 diamonds %>%  
15   fct_recode(. $cut, GOOD = "Good") %>%  
16  str
```

Tips for Installing R and RStudio

- When running the installers for both applications, the defaults are fine.
- Upon launching RStudio, it is useful to change some options (Tools → Global Options)
- For example, on the General options tab, set “Save Workspace to .RData on Exit” to Never
- Put Console in upper left corner, Source in lower left in Pane Layout
- Choose an awesome hacker theme from the Appearance tab
- If you need more detailed instructions installing R and RStudio, watch [this DataCamp video](#) and see [this link](#) for useful keyboard shortcuts.

at first I was like...



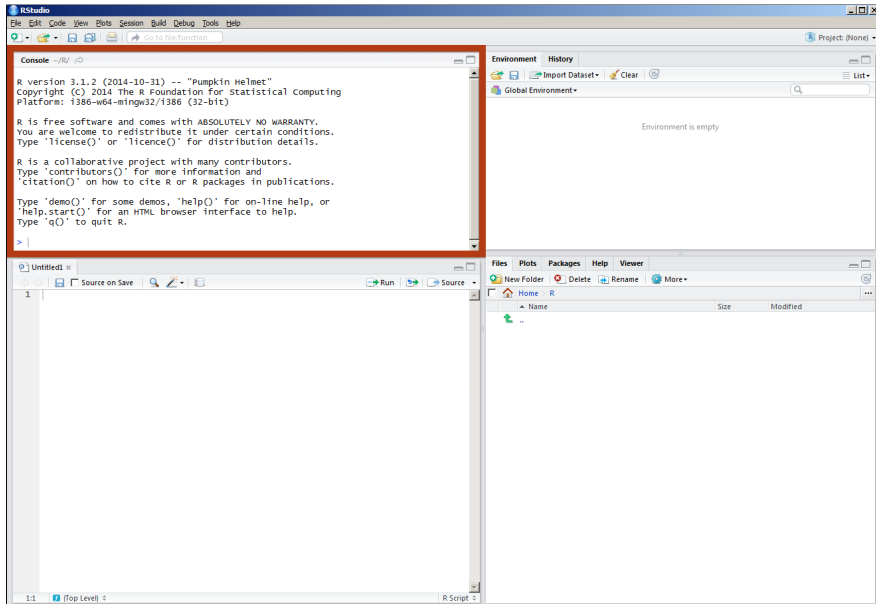
Illustration by Allison Horst.

...but now it's like...



Illustration by Allison Horst.

Introduction to RStudio: The Console



The Console

The main window in R is called the Console. This is where R syntax is executed and you will see the results of the submitted code.

We can interact with R directly through the Console window by typing commands at the `>` prompt and pressing `<return>`.

```
5 + 16  
## [1] 21
```

The Console

Using the Console, we can also use **functions** and create **variables**. Every time you see `()`, you are using a function. For instance:

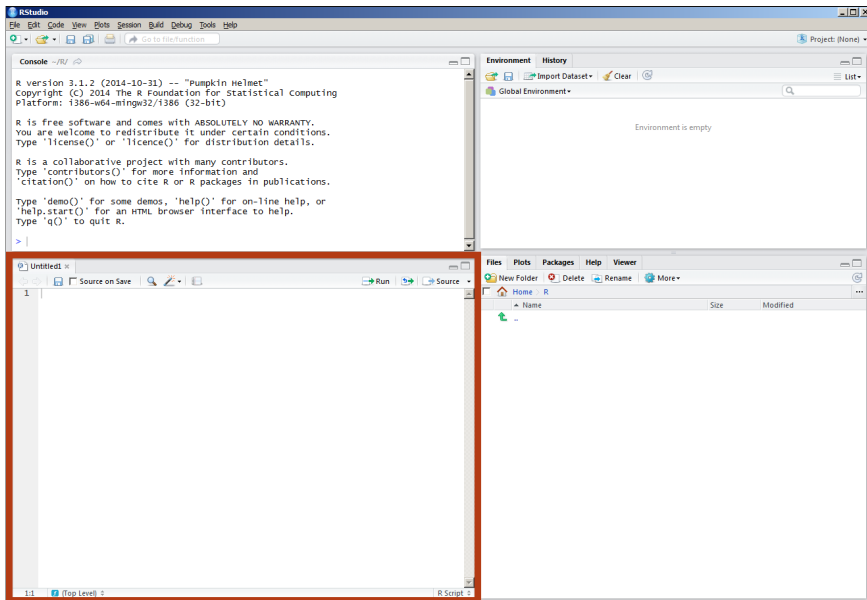
```
x <- c(3, 5, 5, 7, 9) # this calls the function "c"  
x  
## [1] 3 5 5 7 9
```

Assigning Values in R

Note: We typically use `<-` to assign values to variables rather than `=`.¹

¹While you can use `=` as an assignment operator, it sometimes can lead to strange behaviours. So, as a heuristic, it is safer to use the conventional `<-`.

The Source Window (or Editor)



The Source Window (or Editor)

Upon launching RStudio, this may be hidden until you create a new script via `File → New File → R Script` or use `File → Open File`.

In the Editor, we can write the R commands that we want to run:

- code can be run line-by-line via `<ctrl>+Enter` (Windows) or `<command>+Return` (Mac).
- run multiple lines by highlighting them and using the same shortcut.
- can also use the Run drop-down from the top-right of the Editor.

On R Files:

.R script files are plain text: they can be modified with any text editor, cross-platform.

Coding Conventions

Comments in R

```
# The # symbol is used to demark comments in R  
x <- c(3, 5, 5, 7, 9)  
# y <- c(1, 2, 3, 4, 5)  
x # print X  
## [1] 3 5 5 7 9  
y # print Y  
## Error in eval(expr, envir, enclos): object 'y' not found
```

Commenting code inline is **highly encouraged**, and doubly so for collaborative projects.

Note: We will soon discuss a better way of managing comments and code via **Markdown**.

Coding Conventions

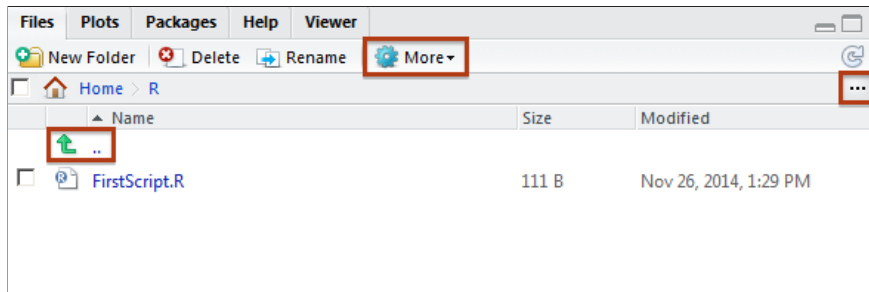
Spacing in R Syntax

R is almost always oblivious to spacing. There are a few notable exceptions, but they tend to come out in more advanced features (e.g., using `if` statements and embedded functions). The following will all produce the same results:

```
x <- c(3, 5, 5, 7, 9)
x <- c(3,5,5,7,9)
x <- c (3,5,5,7,9)
x <- c( 3, 5, 5, 7, 9 )
```

If you want to follow some leading sources on style, [Google](#) and [Hadley Wickham](#) both have R-specific style guides.

File Manager



- Navigate by clicking on folders or the double dots/green arrow to go up a level.
- Use the ellipsis to open a file browser.
- Under **More**, lives the very useful Set As Working Directory command.

File Manager

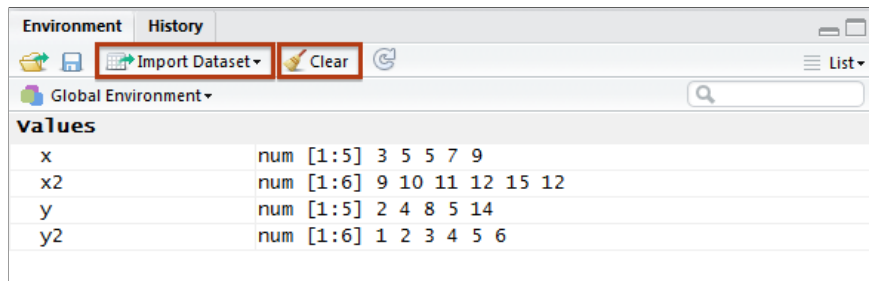
- What you see in the file pane shows R's **working directory**.
- This is very important to keep track of, as this is the directory R will look in when you specify a filename!

Tips and Tricks:

- If you open RStudio using the default shortcut, the working directory is set under **Tools** → **Global Options**
- This usually is something like: `C:/Users/your.name/Documents/R/`
- However, if you open RStudio by double clicking an `.R` or `.Rmd` file, the working directory will *automatically* be set to the same directory the file is in.

Always keep a project's datafiles and 'R' scripts in the same directory.

Environment

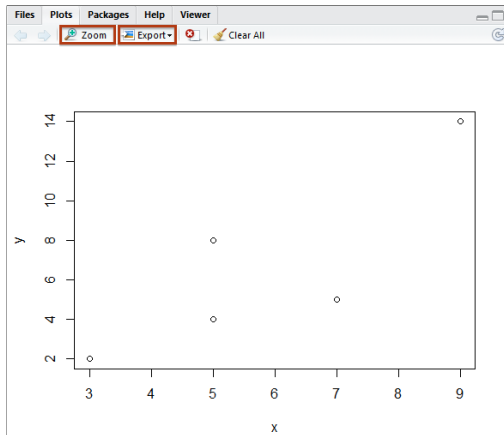


The screenshot shows the RStudio Environment pane. At the top, there are tabs for 'Environment' and 'History'. Below the tabs is a toolbar with icons for file operations and a search bar. The 'Import Dataset' and 'Clear' buttons are highlighted with red boxes. The 'Global Environment' is selected, and a search bar is visible. Below the search bar, the 'values' section lists variables in the workspace:

Variable	Type	Length	First 10 Elements
x	num	[1:5]	3 5 5 7 9
x2	num	[1:6]	9 10 11 12 15 12
y	num	[1:5]	2 4 8 5 14
y2	num	[1:6]	1 2 3 4 5 6

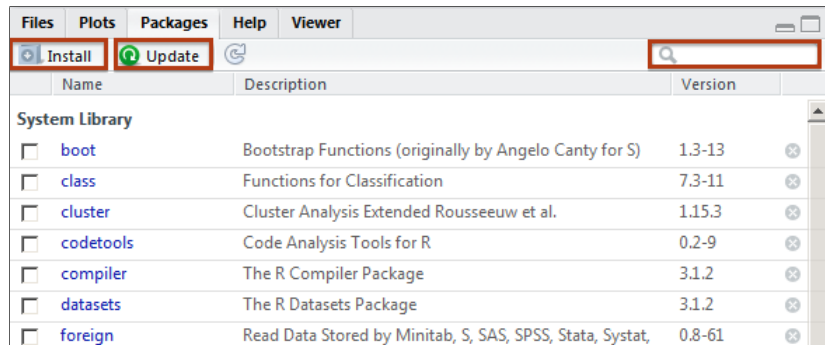
- Live updating, shows everything currently in the workspace.
- Headings for **Data**, **Values**, and **Functions**
- Denotes variable type, length, and the first 10 elements.
- Can click on a complex objects to see what they contain.
- Handy commands: Import Dataset and Clear to clean the workspace.

Plot Panel



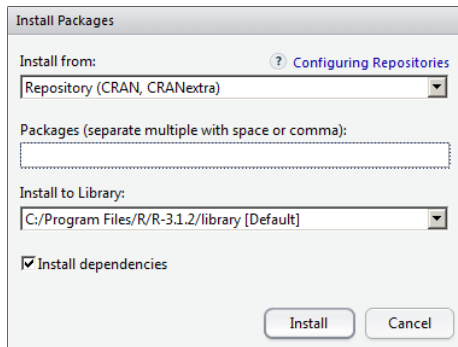
- Unlike base R, we can navigate through many plots via arrows.
- Use Zoom to see plot in higher resolution.
- Use Export to save current plot in a variety of formats.

Packages



- Can use the Install button to get packages from CRAN
- Use Update periodically to keep them current

Installing Packages



The screenshot shows the 'Install Packages' dialog box. It has a title bar 'Install Packages'. Below the title bar, there is a section 'Install from:' with a help icon (?) and a link 'Configuring Repositories'. Below this is a dropdown menu showing 'Repository (CRAN, CRANextra)'. The next section is 'Packages (separate multiple with space or comma):' followed by an empty text input field. Below that is 'Install to Library:' with a dropdown menu showing 'C:/Program Files/R/R-3.1.2/library [Default]'. At the bottom, there is a checked checkbox 'Install dependencies'. At the very bottom are two buttons: 'Install' and 'Cancel'.

- Text input live updates with available CRAN packages
- Always leave `Install dependencies` checked
- Can also use: `install.packages("NAME", dependencies = TRUE)` in the console.

Installing Packages

`install.packages("polycor", dep = TRUE)` installs: `polycor`, `mvtnorm`, and `sfsmisc`.

Note: Just because a package has been installed, doesn't mean that it is available for use! We need to load packages that we want to use.

<input type="checkbox"/>	polycor	Polychoric and Polyserial Correlations	0.7-8	✕
--------------------------	---------	--	-------	---

The package appears in our list, but is unchecked.

Running `library(polycor)` will make `polycor` available for use.

<input checked="" type="checkbox"/>	polycor	Polychoric and Polyserial Correlations	0.7-8	✕
-------------------------------------	---------	--	-------	---

Getting Help

In the console, use `?` and `??` to search for a particular function name, and for pattern matching, respectively. For example:

```
?t.test
```

```
??noise
```

Notes:

- The `?` approach only searches **loaded packages**, and only for an exact match (e.g., `?ttest` gives an error)
- The `??` approach searches all packages installed on your computer for the requested keyword

Getting Help



The screenshot shows the R Documentation window for the `t.test` function. The window has a menu bar with 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. Below the menu bar is a toolbar with icons for back, forward, home, print, and search. The main content area displays the function signature `t.test {stats}` and the title 'Student's t-Test'. The 'Description' section states that it performs one and two sample t-tests on vectors of data. The 'Usage' section shows the function signature `t.test(x, ...)` and two examples: the default S3 method and the S3 method for class 'formula'. The 'Arguments' section lists the parameters: `x` (a (non-empty) numeric vector of data values), `y` (an optional (non-empty) numeric vector of data values), `alternative` (a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter), and `mu` (a number indicating the true value of the mean (or difference in means if you are performing a two sample

R Documentation

`t.test {stats}`

Student's t-Test

Description

Performs one and two sample t-tests on vectors of data.

Usage

```
t.test(x, ...)
```

```
## Default S3 method:
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)
```

```
## S3 method for class 'formula'
t.test(formula, data, subset, na.action, ...)
```

Arguments

<code>x</code>	a (non-empty) numeric vector of data values.
<code>y</code>	an optional (non-empty) numeric vector of data values.
<code>alternative</code>	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
<code>mu</code>	a number indicating the true value of the mean (or difference in means if you are performing a two sample

Getting Help

R help files generally follow a template:

- **Function Name {Package}**: e.g. - `t.test {stats}`
- **Description**: Brief idea of what the function does.
- **Usage**: Function call, with defaults indicated
- `t.test(x, y = NULL, alternative = c("two.sided", "less", "greater"), mu = 0, paired = FALSE, var.equal = FALSE, conf.level = 0.95, ...)`
- **Arguments**: Detailed list of function's parameters.
- **Details**: Generally technical details about function's behaviour.
- **Value**: Description of the output generated from the function.
- **See Also**: A list of other functions that you may find useful.
- **Examples**: The best part of a help file!

On R Scripts...

.R files (like SPSS syntax files) only ever have the plain text commands and comments that you decide to include in them. This means they are missing a variety of things, such as...

- No output!
- No plots!
- No narrative!

This is...

On R Scripts...

.R files (like SPSS syntax files) only ever have the plain text commands and comments that you decide to include in them. This means they are missing a variety of things, such as...

- No output!
- No plots!
- No narrative!

This is...

Not great!

R + RStudio + Markdown = Dynamic Documents

A substantially more powerful method of working is to work with **R Markdown (.Rmd) files** instead of R scripts (.R). The benefits of working with Markdown are:

- Flexibility in text formatting (titles, subtitles, quotes)
- The ability to include lists, links, and images
- The ability to output the document to HTML, PDF, even DOCX!
- But, most of all: the ability to include raw R code (“R chunks”) and R output in the same document!

R + RStudio + Markdown = Dynamic Documents

A substantially more powerful method of working is to work with **R Markdown (.Rmd) files** instead of R scripts (.R). The benefits of working with Markdown are:

- Flexibility in text formatting (titles, subtitles, quotes)
- The ability to include lists, links, and images
- The ability to output the document to HTML, PDF, even DOCX!
- But, most of all: the ability to include raw R code (“R chunks”) and R output in the same document!

Reproducible research is the best research!

Rmarkdown

TEXT. CODE. OUTPUT.
(GET IT TOGETHER, PEOPLE.)



An RMarkdown Example (.Rmd file)

Create a new RMarkdown file via File → New File → R Markdown.

```
1 ---
2 |title: "My Awesome Title"
3 |author: "My Best Self"
4 |date: '2020-01-06'
5 |output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for
15 authoring HTML, PDF, and MS Word documents. For more details on using R
16 Markdown see <http://rmarkdown.rstudio.com>.
17
18 When you click the Knit button a document will be generated that
19 includes both content as well as the output of any embedded R code chunks
20 within the document. You can embed an R code chunk like this:
21
22 ```{r cars}
23 summary(cars)
24 ```
```

Markdown Example: Knit an .html File

My Awesome Title

My Best Self

2020-01-06

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed      dist
##  Min.   : 4.0    Min.   : 2.00
##  1st Qu.:12.0    1st Qu.: 26.00
##  Median :15.0    Median : 36.00
##  Mean   :15.4    Mean   : 42.98
##  3rd Qu.:19.0    3rd Qu.: 56.00
##  Max.   :25.0    Max.   :120.00
```

Including Plots

Markdown Tips and Tricks



- May need to run: `install.packages("knitr")`.
- Remember, R code always goes inside a “chunk”
- Knit to PDF requires a LaTeX distribution installed.
 - If you only want to do this, install the package `tinytex`
- Look at the Markdown Quick Reference page (under “Cheatsheets” in the Help menu) to see how to:
 - Create lists (unordered via asterisks, ordered via numbers)
 - Incorporate links, images, LaTeX equations, page breaks, tables.