

Working with RStudio

Michael Friendly

Psych 6136

<http://friendly.github.io/psy6136>

Getting started: Tools

- To profit best, you need to install both R and R Studio on your computer



The basic R system: R console (GUI) & packages

Download: <http://cran.us.r-project.org/>

Add my recommended packages:

source(["http://euclid.psych.yorku.ca/www/psy6135/R/install-pkgs.R"](http://euclid.psych.yorku.ca/www/psy6135/R/install-pkgs.R))

The R Studio IDE: analyze, write, publish

Download:

<https://www.rstudio.com/products/rstudio/download/>

Add: R Studio-related packages, as useful



R package tools



Data prep: Tidy data makes analysis and graphing much easier.

Packages: [tidyverse](#), comprised of: [tidyr](#), [dplyr](#), [lubridate](#), ...

The tidyverse

Components



R graphics: general frameworks for making standard and custom graphics

Graphics frameworks: base graphics, [lattice](#), [ggplot2](#), [rgl](#) (3D)

Application packages: [car](#) (linear models), [vcd](#) (categorical data analysis), [heplots](#) (multivariate linear models)

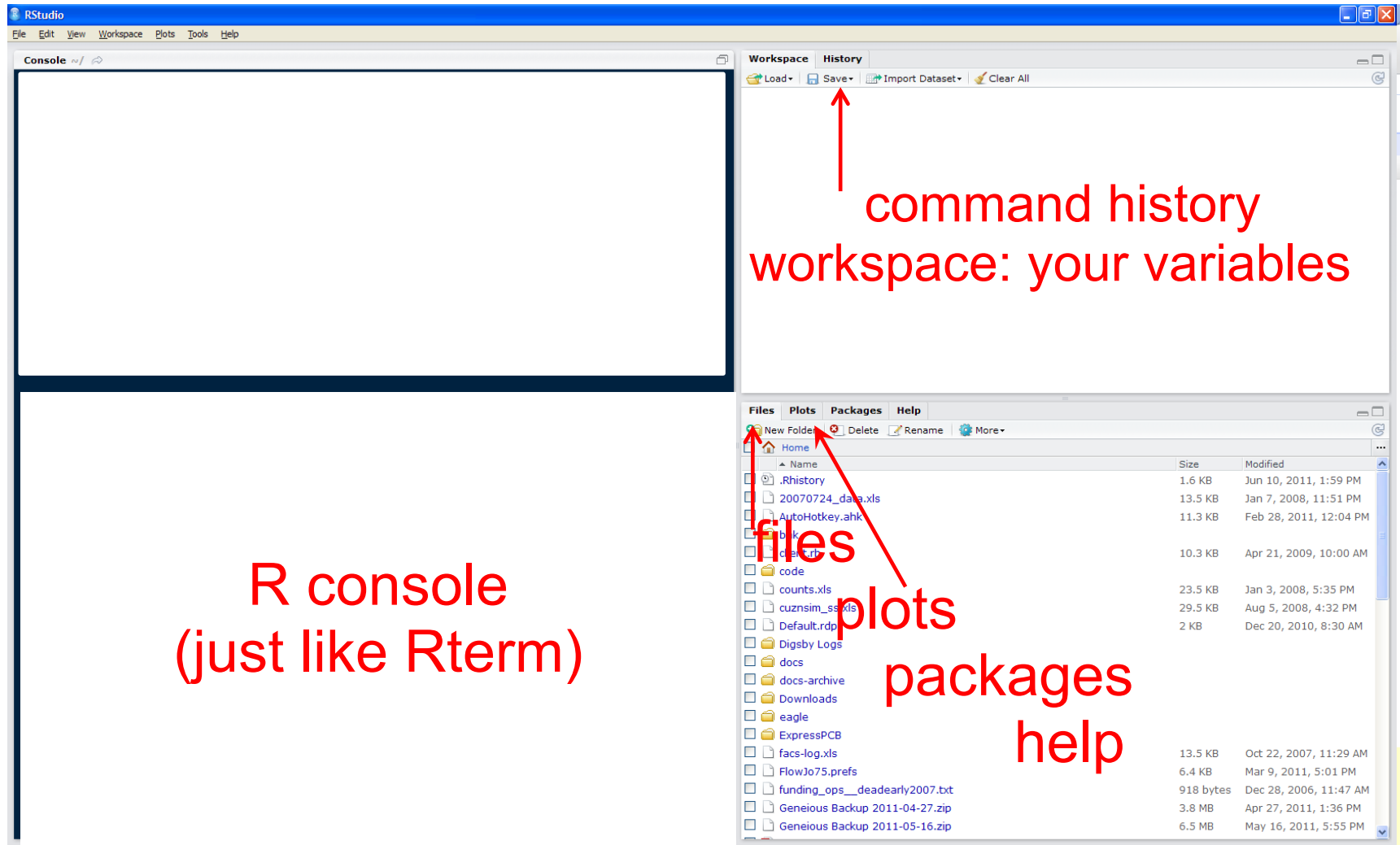


Publish: A variety of R packages make it easy to write and publish research reports and slide presentations in various formats (HTML, Word, LaTeX, ...), all within R Studio



Web apps: R now has several powerful connections to preparing dynamic, web-based data display and analysis applications.

Getting started: R Studio



R Studio navigation

R folder navigation commands:

- Where am I?

```
> getwd()  
[1] "C:/Dropbox/Documents/6135"
```

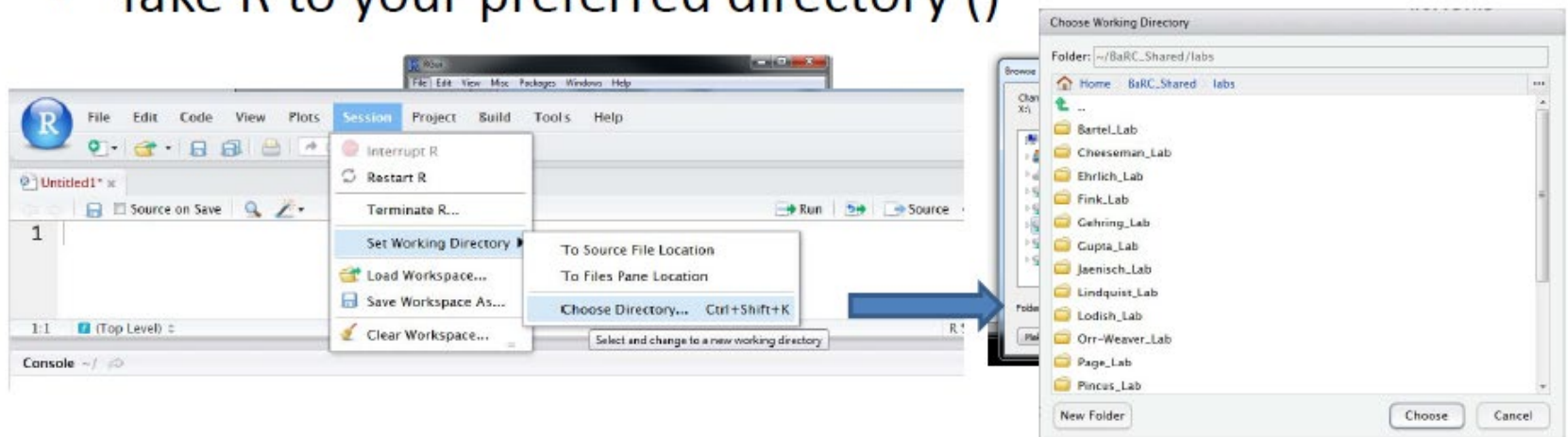
Better yet: create an R project!

- Go somewhere:

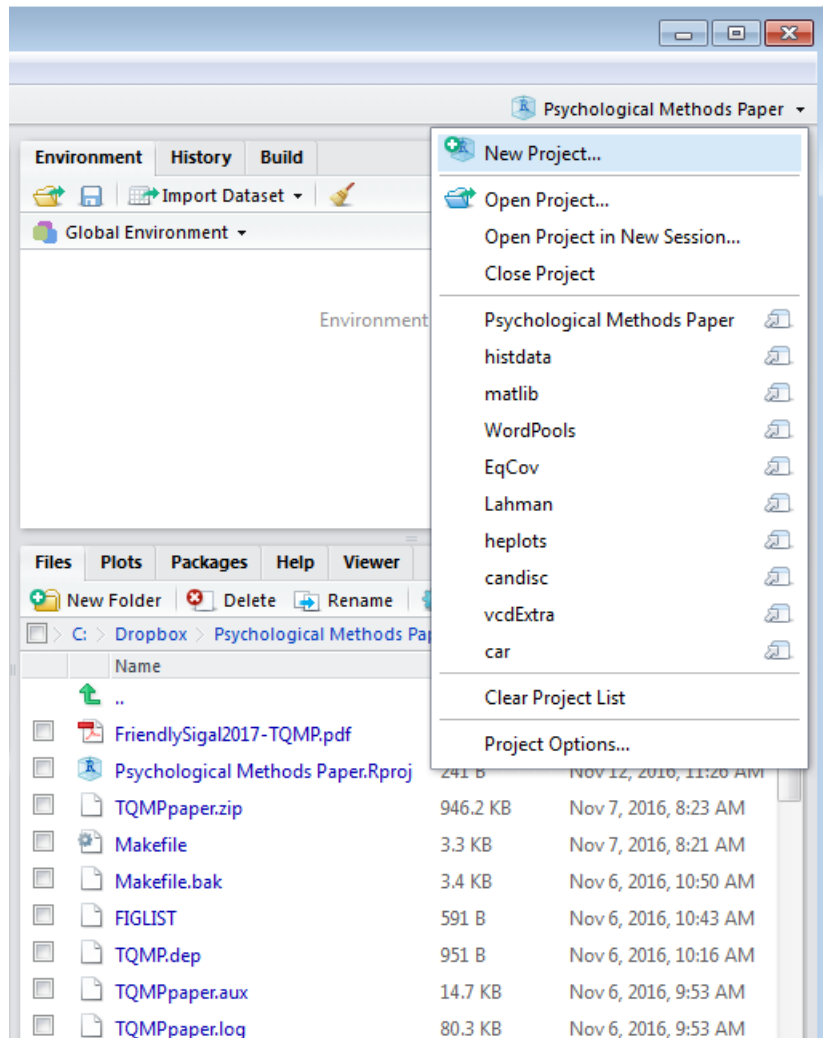
```
> setwd("C:/Dropbox")  
> setwd(file.choose())
```

R Studio GUI

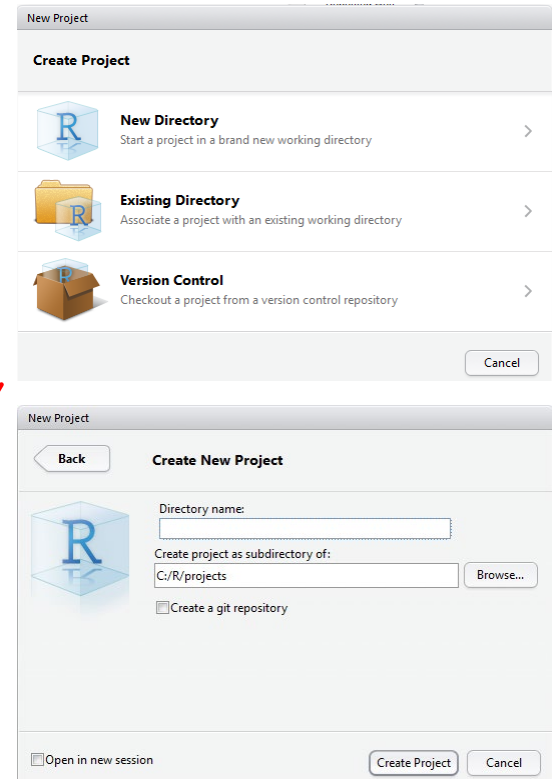
- Take R to your preferred directory ()



R Studio projects



R Studio projects are a handy way to organize your work



Lahman.Rproj

The .Rproj item opens the project in R Studio

R Studio projects

An R Studio project for a **research paper**: R files (scripts), Rmd files (text, R “chunks”)

The screenshot displays the R Studio environment with a project titled "Psychological Methods Paper". The main editor shows an R Markdown file with the following content:

```
1 ---
2 title: "Graphical Methods for Multivariate Linear Models in
3 Psychological Research: An R Tutorial"
4 shorttitle: "Graphical Methods for MLMs"
5 author:
6   - name: Michael Friendly
7     affiliation: 1
8     corresponding: yes # Define only one corresponding author
9     address: Psychology Department, York University, Toronto, Ontario,
10   Canada, M3J1P3
11   email: friendly@yorku.ca
12   - name: Matthew Sigal
13     affiliation: 1
14     id: 1
15     institution: York University
16 abstract: |
17   This paper is designed as a tutorial to highlight some
18   recent developments for visualizing the relationships among
19   response and predictor variables in multivariate linear
```

The console at the bottom shows the R startup message and the R version (3.2.2) and platform (x86_64-w64-mingw32/x64 (64-bit)).

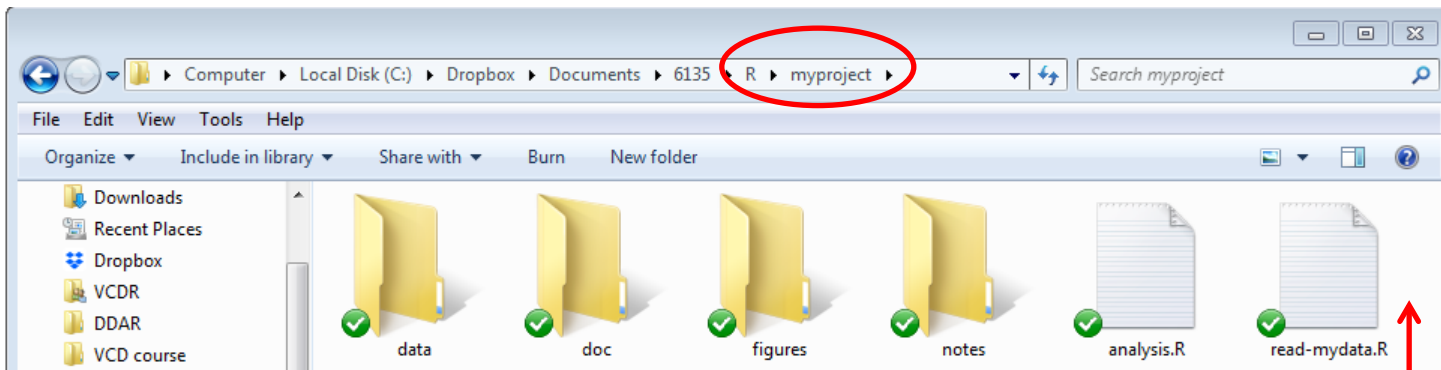
The Environment pane on the right shows the Global Environment. The Files pane on the left shows the project files, including:

- Psychological Methods Paper.Rproj
- histdata
- matlib
- WordPools
- EqCov
- Lahman
- heplots
- candisc
- vcdExtra
- car

The Project Options... dialog box is open, showing the project name and the location of the project files.

Organizing an R project

- Use a separate folder for each project
- Use sub-folders for various parts



data files:

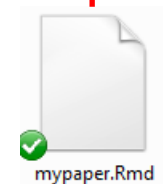
- raw data (.csv)
- saved R data (.Rdata)

figures:

- diagrams
- analysis plots

R files:

- data import
- analysis



Write up files will go here (.Rmd, .docx, .pdf)

This project, saved in a **Dropbox** folder automatically syncs with all my computers & collaborators. I use Git & **GitHub** for more serious work.

Organizing an R project

- Use separate R files for different steps:
 - Data import, data cleaning, ... → save as an RData file
 - Analysis: load RData, ...

read-mydata.R

```
# read the data; better yet: use RStudio File -> Import Dataset ...
```

```
mydata <- read.csv("data/mydata.csv")
```

```
# data cleaning:
```

```
#   filter missing, make factors, transform variables, ....
```

```
# save the current state
```

```
save("data/mydata.RData")
```

Organizing an R project

- Use separate R files for different steps:
 - Data import, data cleaning, ... → save as an RData file
 - Analysis: load RData, ...

analyse.R

#' ## load the data

```
load("data/mydata.RData")
```

#' ## do the analysis – exploratory plots

```
plot(mydata)
```

#' ## fit models

```
mymod.1 <- lm(y ~ X1 + X2 + X3, data=mydata)
```

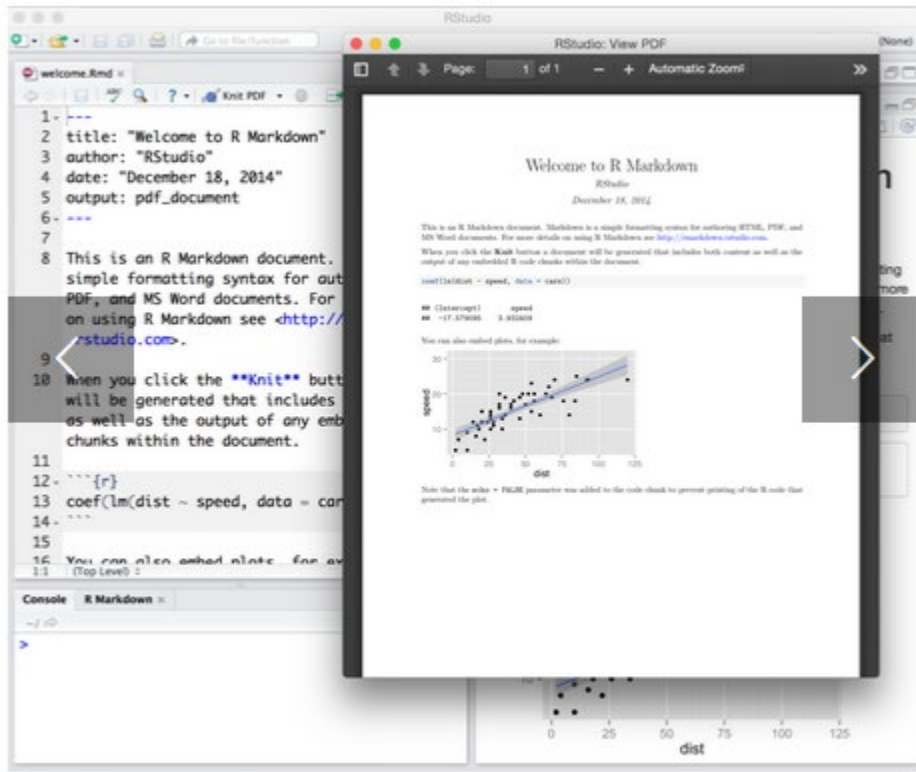
#' ## plot models, extract model summaries

```
plot(mymod.1)
```

```
summary(mymod.1)
```

NB: #' ## is a special R comment for a H2 heading in an R “notebook” script

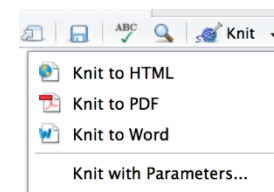
Reproducible analysis & reporting



R Studio, together with the knitr and rmarkdown packages provide an easy way to combine writing, analysis, and R output into complete documents

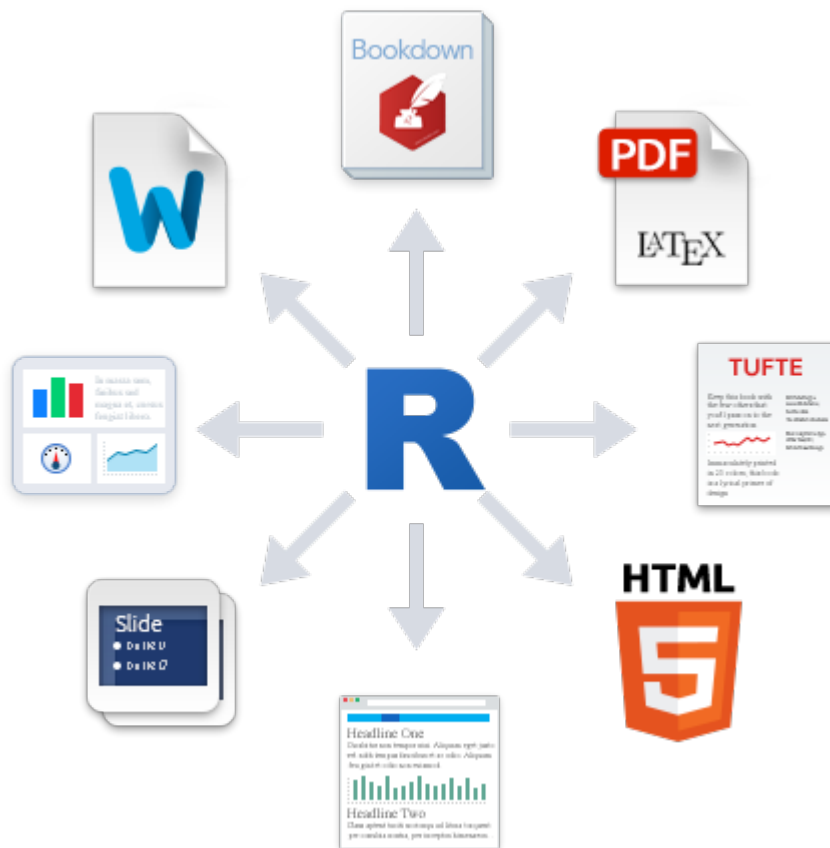
.Rmd files are just text files, using rmarkdown markup and knitr to run R on “code chunks”

A given document can be rendered in different output formats:

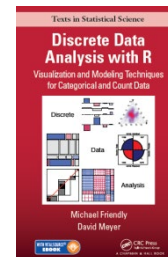


Output formats and templates

The integration of R, R Studio, knitr, rmarkdown and other tools is now highly advanced.



My last book was written entirely in R Studio, using .Rnw syntax → LaTeX → PDF → camera ready copy



The ggplot2 book was written using .Rmd format.



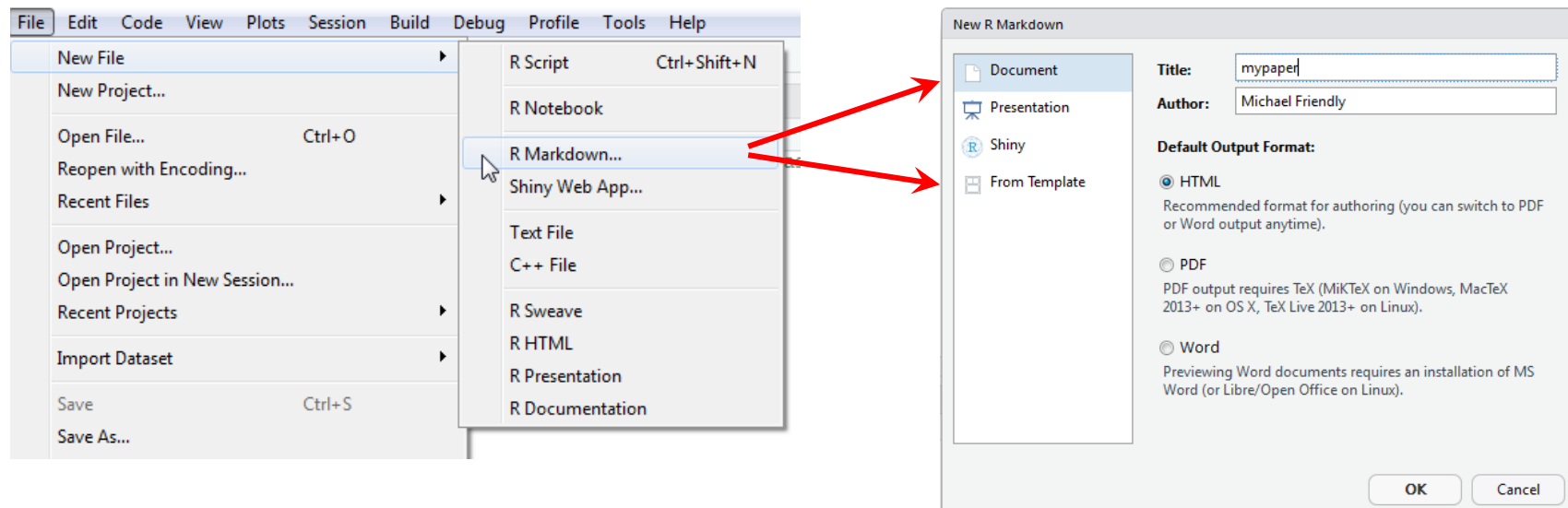
The [bookdown](#) package makes it easier to manage a book-length project – TOC, fig/table #s, cross-references, etc.

Also: [blogdown](#), [posterdown](#), ...

Templates are available for APA papers, slides, handouts, entire web sites, etc.

Writing it up

- In R Studio, create a .Rmd file to use R Markdown for your write-up
 - lots of options: HTML, Word, PDF (needs LaTeX)
 - templates for various pub types



Writing it up

- Use simple Markdown to write text
- Include code chunks for analysis & graphs

mypaper.Rmd, created from a template

```
1 ---
2 title: "mypaper"
3 author: "Michael Friendly"
4 date: "January 29, 2018"
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
15 details on using R Markdown see <http://rmarkdown.rstudio.com>.
16
17 when you click the Knit button a document will be generated
18 R code chunks within the document. You can embed an R code chunk
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
25
26 You can also embed plots, for example:
27
28 ```{r pressure, echo=FALSE}
29 plot(pressure)
30 ```
```

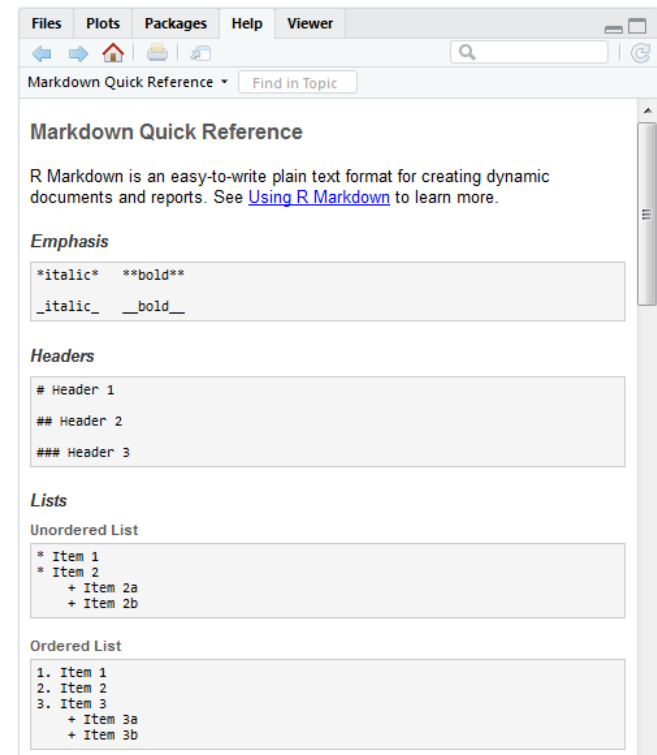
yaml header

Header 2

output code chunk

plot code chunk

Help -> Markdown quick reference



rmarkdown basics

rmarkdown uses simple formatting for all standard document elements

The image displays two windows side-by-side, illustrating the transformation of R Markdown source code into a rendered HTML document.

Left Window (Source Code): The file is named `example.Rmd`. The code includes:

- Line 1: `# Header 1`
- Line 3: `This is an R Markdown document. Markdown is a simple formatting syntax for authoring webpages.`
- Line 5: `Use an asterisk mark to provide emphasis, such as italics or bold.`
- Line 7: `Create lists with a dash:`
- Lines 9-11: A list with three items: `- Item 1`, `- Item 2`, and `- Item 3`.
- Line 13: `````
- Line 14: `Use back ticks to create a block of code`
- Line 16: `````
- Line 18: `Embed LaTeX or MathML equations,`
- Line 19: `$\frac{1}{n} \sum_{i=1}^n x_i$`
- Line 21: `Or even footnotes, citations, and a bibliography. [^1]`
- Line 23: `[^1]: Markdown is great.`

Right Window (Rendered HTML): The file is named `example.html`. The rendered output shows:

- A large heading:

Header 1
- A paragraph:

This is an R Markdown document. Markdown is a simple formatting syntax for authoring web pages.
- A paragraph:

Use an asterisk mark to provide emphasis, such as *italics* or **bold**.
- A paragraph:

Create lists with a dash:
- A list:
 - Item 1
 - Item 2
 - Item 3
- A code block (preformatted text):

```
Use back ticks to
create a block of code
```
- A paragraph:

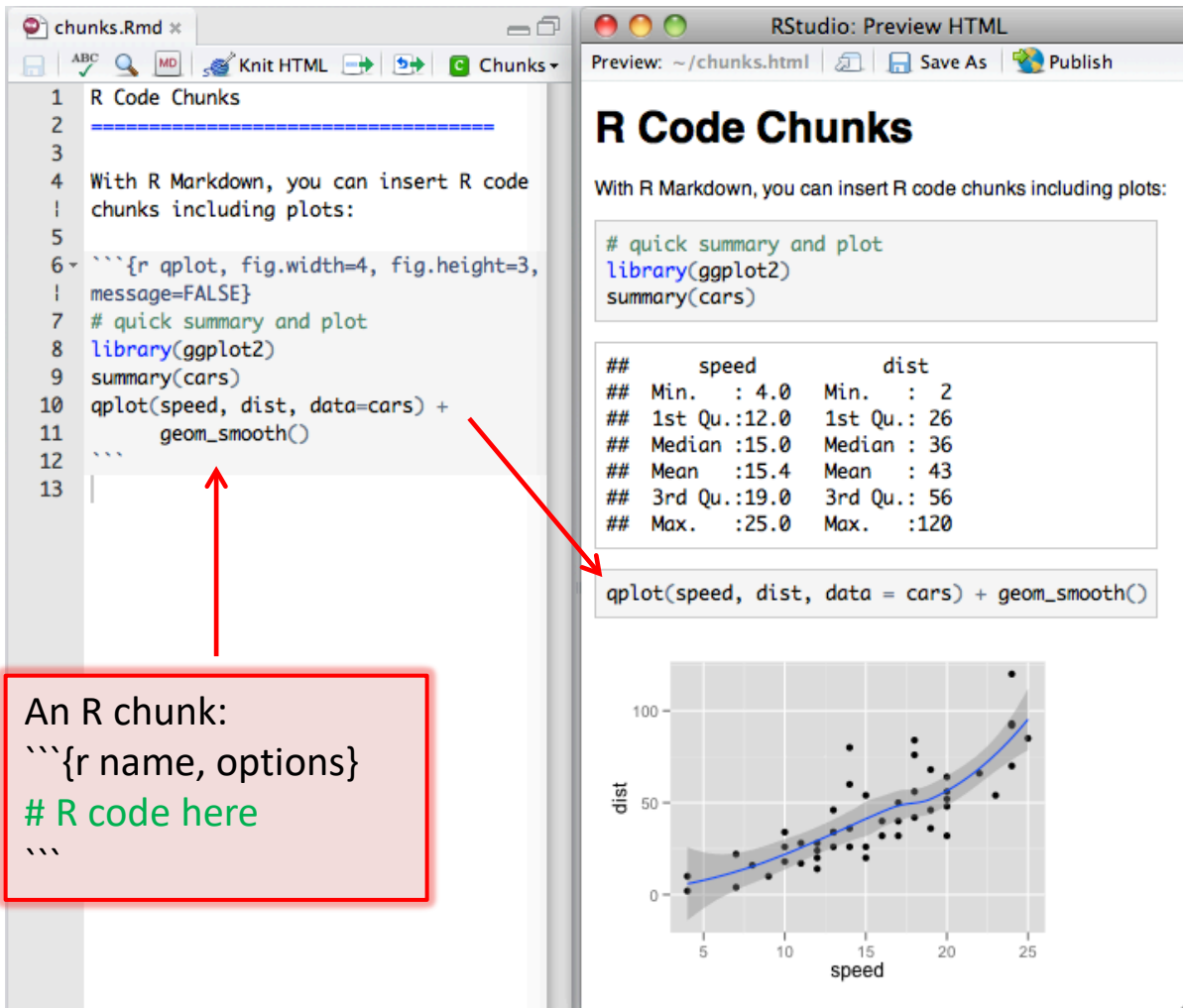
Embed LaTeX or MathML equations, $\frac{1}{n} \sum_{i=1}^n x_i$
- A paragraph:

Or even footnotes, citations, and a bibliography. ¹
- A footnote:

1. Markdown is great.↩

R code chunks

R code chunks are run by [knitr](#), and the results are inserted in the output document



The screenshot shows the RStudio interface. On the left, the 'chunks.Rmd' file is open, showing an R code chunk. The code includes a comment, a library call, a summary, and a plot. A red box highlights the chunk syntax, and a red arrow points from it to the rendered output in the preview window.

```
1 R Code Chunks
2 =====
3
4 With R Markdown, you can insert R code
5 chunks including plots:
6
7 ```{r qplot, fig.width=4, fig.height=3,
8   message=FALSE}
9 # quick summary and plot
10 library(ggplot2)
11 summary(cars)
12 qplot(speed, dist, data=cars) +
13   geom_smooth()
```

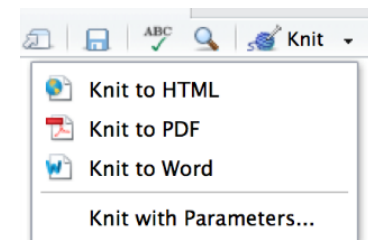
The preview window shows the rendered HTML output. It includes the title 'R Code Chunks', a subtitle, the summary of the 'cars' dataset, and a scatter plot with a smoothed regression line. The plot shows 'dist' on the y-axis and 'speed' on the x-axis.

##	speed	dist
##	Min. : 4.0	Min. : 2
##	1st Qu.: 12.0	1st Qu.: 26
##	Median : 15.0	Median : 36
##	Mean : 15.4	Mean : 43
##	3rd Qu.: 19.0	3rd Qu.: 56
##	Max. : 25.0	Max. : 120

An R chunk:
```\${r name, options}``  
# R code here  
```\n

There are many options for controlling the details of chunk output – numbers, tables, graphs

Choose the output format:



The R Markdown Cheat Sheet provides most of the details

<https://www.rstudio.com/wp-content/uploads/2016/03/rmarkdown-cheatsheet-2.0.pdf>

R Markdown Cheat Sheet

learn more at rmarkdown.rstudio.com



.Rmd files
An R Markdown (.Rmd) file is a record of your research. It contains the code that a scientist needs to reproduce your work along with the narration that a reader needs to understand your work.



Reproducible Research
At the click of a button, or the type of a command, you can rerun the code in an R Markdown file to reproduce your work and export the results as a finished report.



Dynamic Documents
You can choose to export the finished report as a html, pdf, MS Word, ODT, RTF, or markdown document, or as a html or pdf based slide show.

Workflow

- 1 Open a new .Rmd file** at File > New File > R Markdown. Use the wizard that opens to pre-populate the file with a template.
- 2 Write document** by editing template.
- 3 Knit document to create report** Use knit button or `render()` to knit.
- 4 Preview Output** in IDE window.
- 5 Publish** (optional) to web or server. Sync publish button to accounts at:
 - rpubs.com
 - shinyapps.io
 - RStudio Connect
 Reload document. Find in document. File path to output document.

Interactive Documents

Turn your report into an interactive Shiny document in 4 steps

- 1 Add runtime:** shiny to the YAML header.
- 2 Call Shiny input functions** to embed input objects.
- 3 Call Shiny render functions** to embed reactive output.
- 4 Render with `markdown::run`** or click **Run Document** in RStudio IDE.

.Rmd structure

YAML Header
Optional section of render (e.g. pandoc) options written as key-value pairs (YAML).
• At start of file
• Between lines of ---

Text
Narration formatted with markdown, mixed with:

Code chunks
Chunks of embedded code. Each chunk:
• Begins with ````[r]`
• Ends with `````
R Markdown will run the code and append the results to the doc.
It will use the location of the .Rmd file as the working directory.

render()

Use `markdown::render()` to render knit at cmd line. Important args:
input - file to render
output_format - List of render options (as in YAML)
output_file
output_dir
params - list of params to use
envir - environment to evaluate code chunks in
encoding - all input file

Embed code with knitr syntax

Inline code
Insert with `"r<code>"`. Results appear as text without code.
Built with `r<code>getVersion()` → Built with 3.2.3

Code chunks
One or more lines surrounded with ````[r]` and `````. Place chunk options within curly braces, after `r`. Insert with ````[r] ... ````

Global options
Set with knitr: `opts_chunk$set()`, e.g.
````[r] (r: true) FALSE````  
`knitr::opts_chunk$set(echo = TRUE)`

### Parameters

Parameterize your documents to reuse with different inputs (e.g., data sets, values, etc.)

- 1 Add parameters**  
Create and set parameters in the header as sub-values of `params`.
- 2 Call parameters**  
Call parameter values in code as `params$<name>`.
- 3 Set parameters**  
Set values with `Knit with parameters` or the `params` argument of `render()`.

`render("doc.Rmd", params = list(n = 1, d = as.Date("2015-01-01")))`

### Important chunk options

- cache** - cache results for future knits (default = FALSE)
- cache.path** - directory to save cached results in (default = "cache/")
- child** - file(s) to knit and then include (default = NULL)
- collapse** - collapse all output into single block (default = FALSE)
- comment** - prefix for each line of results (default = "##")
- dependencies** - chunk dependencies for caching (default = NULL)
- echo** - Display code in output document (default = TRUE)
- engine** - code language used in chunk (default = "R")
- error** - Display error messages in doc (TRUE) or stop render when errors occur (FALSE) (default = TRUE)
- eval** - Run code in chunk (default = TRUE)
- fig.align** - "left", "right", or "center" (default = "center")
- fig.cap** - figure caption as character string (default = NULL)
- fig.height**, **fig.width** - Dimensions of plots in inches
- highlight** - highlight source code (default = TRUE)
- include** - Include chunk in doc after running (default = TRUE)
- message** - display code messages in document (default = TRUE)
- results** - (default = "markup")  
"asis" - passthrough results  
"hide" - do not display results  
"hold" - put all results below all code
- tidy** - tidy code for display (default = FALSE)
- warning** - display code warnings in document (default = TRUE)

Options not listed above: R options, asis.opts, autodep, background, cache.comments, cache.lazy, cache.rebuild, cache.vars, dev, dev.args, dpi, engine.opts, engine.path, fig.asp, fig.env, fig.ext, fig.keep, fig.la, fig.path, fig.pos, fig.preview, fig.retina, fig.scale, fig.show, fig.showtext, fig.subcap, internal, out.height, out.width, prompt, post, ref.capt, relsize, reset, size, style, tidy.opts

# R notebooks

Often, you just want to “compile” an R script, and get the output embedded in the result, in HTML, Word, or PDF. Just type Ctrl-Shift-K or tap the **Compile Report** button

```
1 #' ---
2 #' title: "Inverse of a matrix"
3 #' author: "Michael Friendly"
4 #' date: "07 Sep 2016"
5 #' output: html_document
6 #' ---
7
8 #' The following examples illustrate the basic properties of the inverse
9 #'
10 #' ### Load the `matlib` package
11 #'
12 #' This defines: `inv()`, `Inverse()`; the standard function for matrix inverse is `solve()`
13
14 library(matlib)
15
16 #' ### Create a 3 x 3 matrix
17 A <- matrix(c(5, 1, 0,
18 3, -1, 2,
19 4, 0, -1), nrow=3, byrow=TRUE)
20
21 det(A)
22
23 #' ### 1. det(A) != 0, so inverse exists
24 (AI <- inv(A))
25
26 #' ### 2. Definition of the inverse: $AA^{-1} = A^{-1}A = I$ or $AI = I$
27
```

## Inverse of a matrix

Michael Friendly  
07 Sep 2016

The following examples illustrate the basic properties of the inverse of a matrix

### Load the `matlib` package

This defines: `inv()`, `Inverse()`; the standard function for matrix inverse is `solve()`

```
library(matlib)
```

### Create a 3 x 3 matrix

```
A <- matrix(c(5, 1, 0,
 3, -1, 2,
 4, 0, -1), nrow=3, byrow=TRUE)
```

```
det(A)
```

```
[1] 16
```

### 1. $\det(A) \neq 0$ , so inverse exists

```
(AI <- inv(A))
```

```
[,1] [,2] [,3]
[1,] 0.0625 0.0625 0.125
[2,] 0.6875 -0.3125 -0.625
[3,] 0.2500 0.2500 -0.500
```

### 2. Definition of the inverse: $A^{-1}A = AA^{-1} = I$ or $AI = I$

```
AI * A = diag(nrow(A))
```

NB: Sometimes you will get very tiny off-diagonal values (like `1.341e-13`). The function `zapsmall()` will round those to 0.

```
AI %>% A
```