

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(["https://friendly.github.io/psy6136//R/install-pkgs.R"](https://friendly.github.io/psy6136//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)

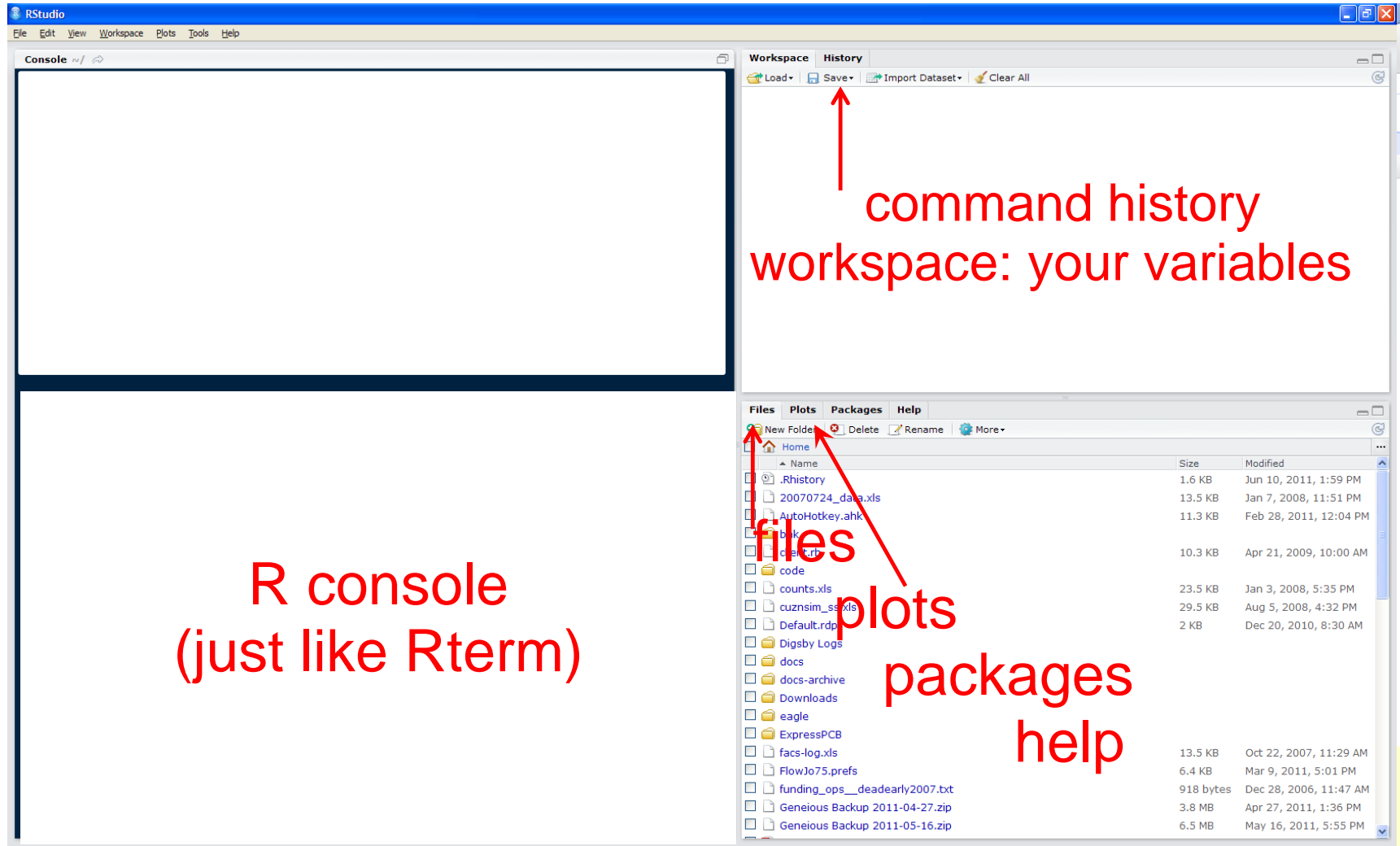


Publish: A variety of R packages make it easy to write and publish research reports and slide presentations in various formats (HTML, Word, LaTeX, ...), all within R Studio



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/6136"
```

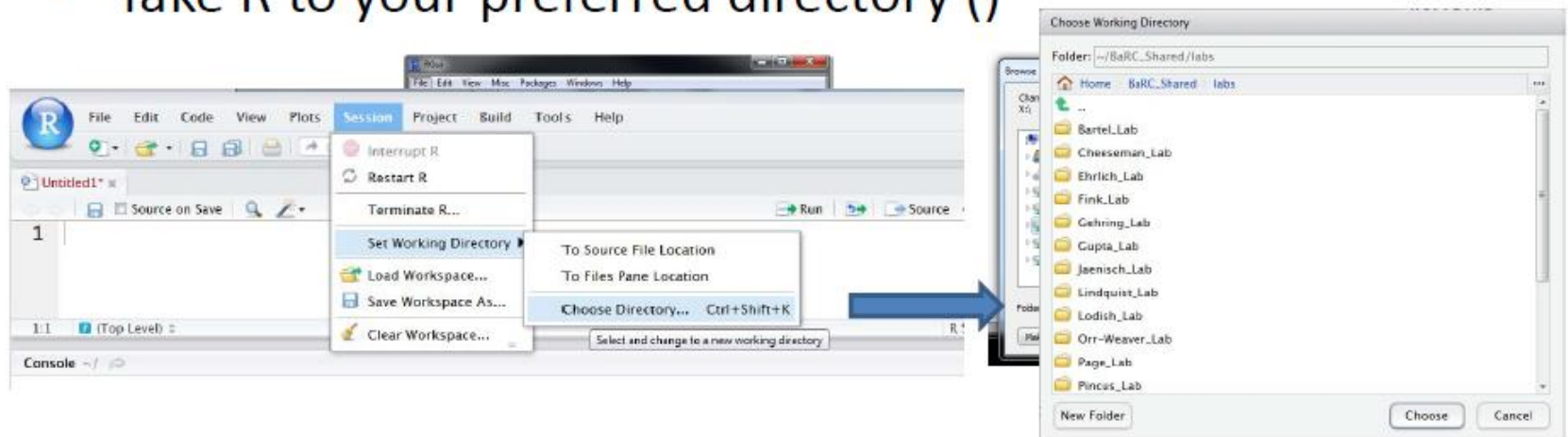
Better yet: create an R project!

- Go somewhere:

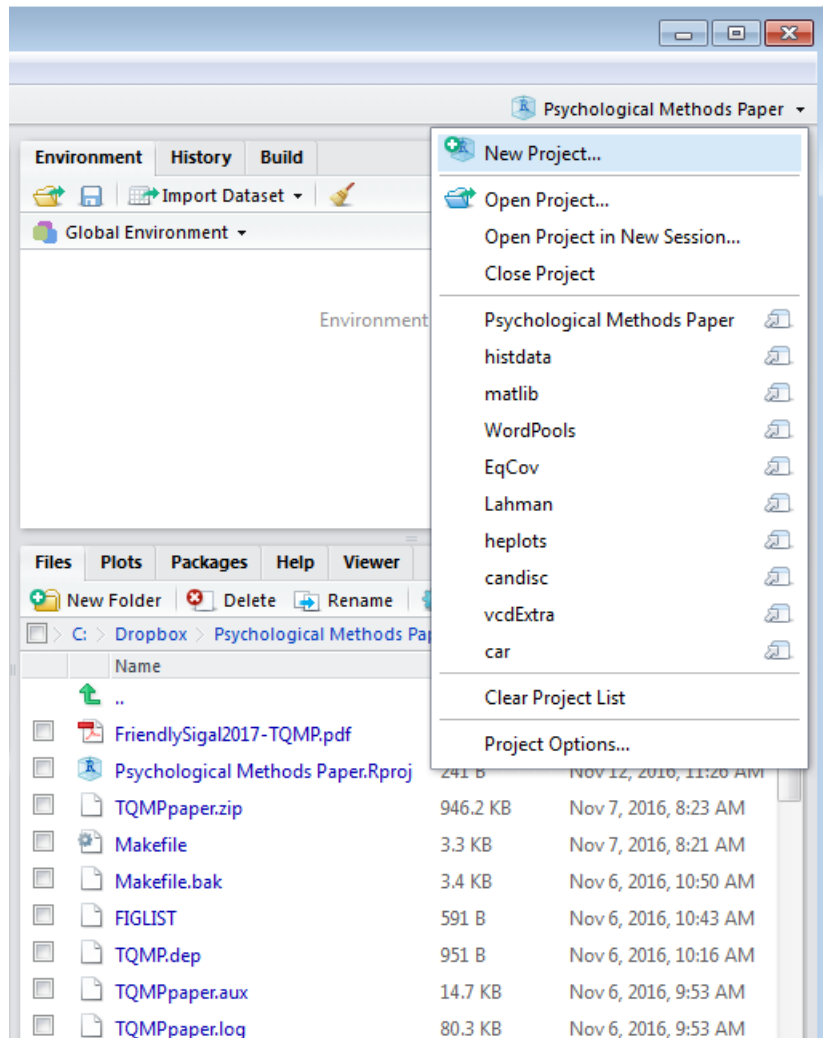
```
> setwd("C:/Dropbox")  
> setwd(file.choose())
```

R Studio GUI

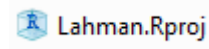
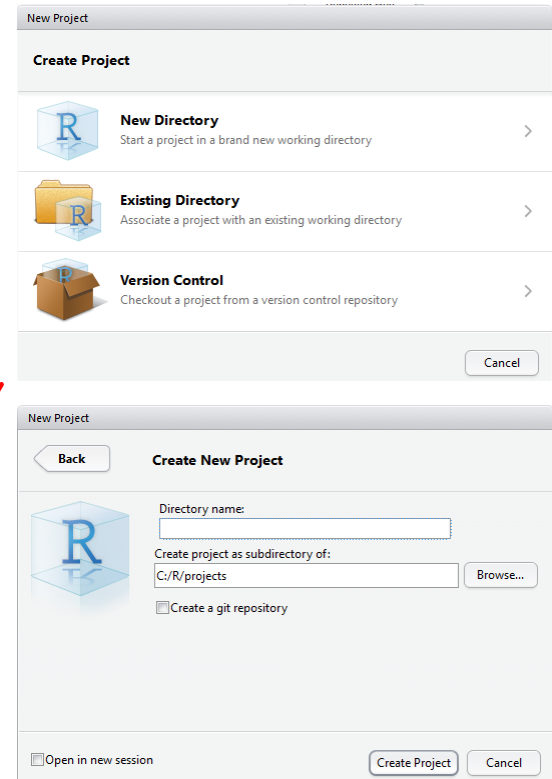
- Take R to your preferred directory ()



R Studio projects



R Studio projects are a handy way to organize your work



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”)

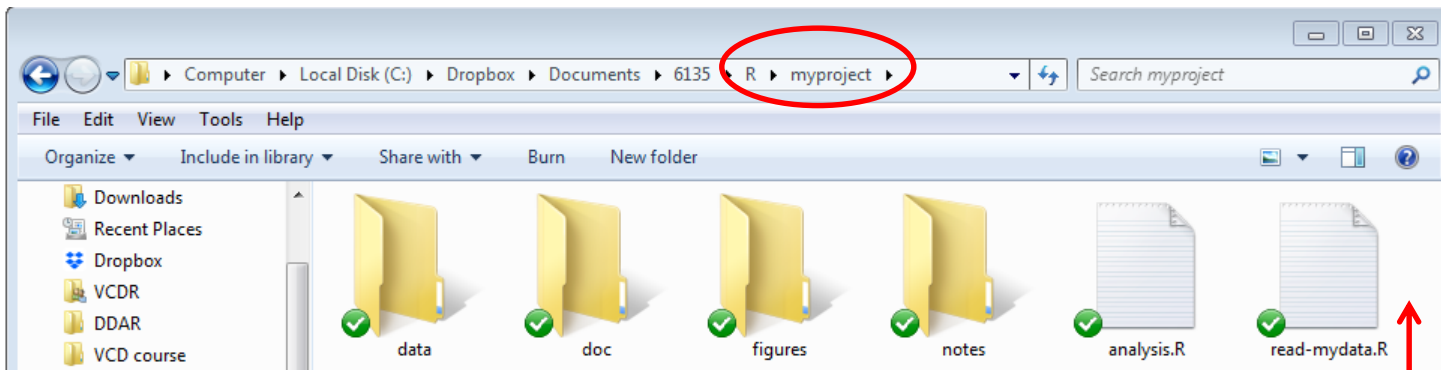
The screenshot displays the RStudio IDE interface for a project titled "Psychological Methods Paper". The main editor window shows an R Markdown file with a title "Graphical Methods for Multivariate Linear Models in Psychological Research: An R Tutorial" and author information for Michael Friendly and Matthew Sigal. The console window at the bottom shows the R startup message and copyright information. The right-hand pane shows the "Environment" tab with a list of files and folders, including "TQMPpaper.zip", "Makefile", "FIGLIST", "TQMP.dep", "TQMPpaper.aux", "TQMPpaper.log", "TQMPpaper.out", "TQMPpaper.pdf", "TQMPpaper.synctex.gz", and ".Rhistory".

Environment

| Name | Size | Modified |
|-----------------------------------|----------|-----------------------|
| .. | | |
| FriendlySigal2017-TQMP.pdf | | |
| Psychological Methods Paper.Rproj | | |
| TQMPpaper.zip | 946.2 KB | Nov 7, 2016, 8:23 AM |
| Makefile | 3.3 KB | Nov 7, 2016, 8:21 AM |
| Makefile.bak | 3.4 KB | Nov 6, 2016, 10:50 AM |
| FIGLIST | 591 B | Nov 6, 2016, 10:43 AM |
| TQMP.dep | 951 B | Nov 6, 2016, 10:16 AM |
| TQMPpaper.aux | 14.7 KB | Nov 6, 2016, 9:53 AM |
| TQMPpaper.log | 80.3 KB | Nov 6, 2016, 9:53 AM |
| TQMPpaper.out | 6 KB | Nov 6, 2016, 9:53 AM |
| TQMPpaper.pdf | 819.8 KB | Nov 6, 2016, 9:53 AM |
| TQMPpaper.synctex.gz | 286.1 KB | Nov 6, 2016, 9:53 AM |
| .Rhistory | 13.5 KB | Nov 3, 2016, 9:35 AM |

Organizing an R project

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



data files:

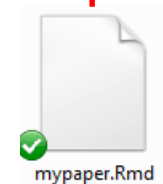
- raw data (.csv)
- saved R data
(.Rdata)

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

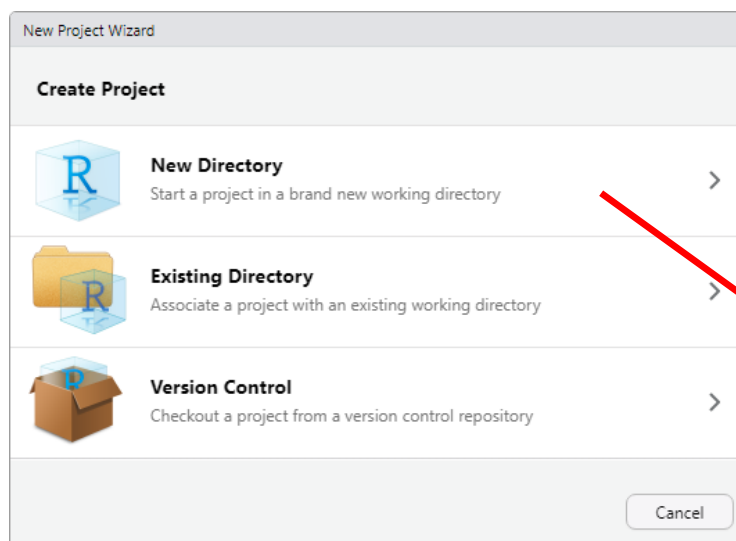
```
plot(mymod.1)
```

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

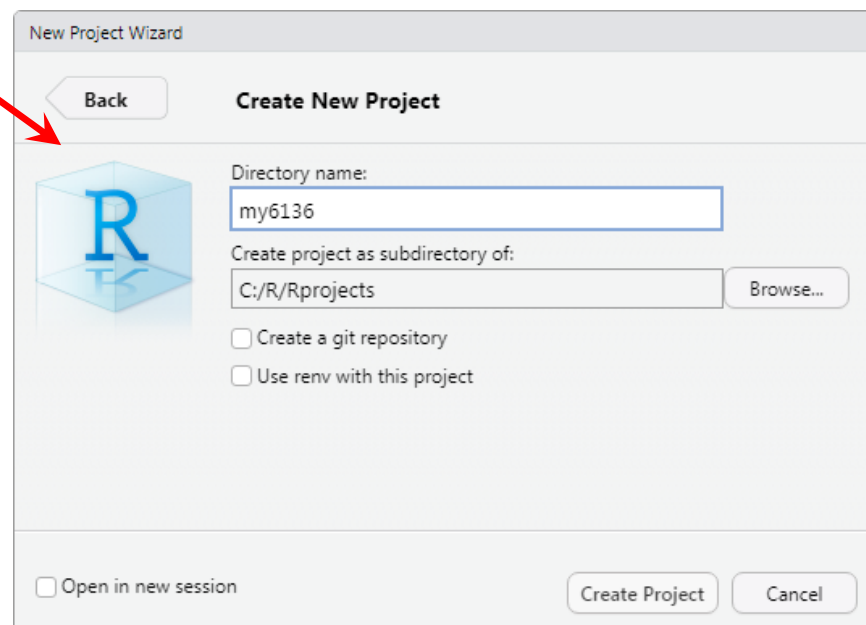
NB: #' ## is a special R comment for a H2 heading in an R “notebook” script

Your psy6136 project

I recommend that you create your own RStudio project for work in this course
This will enable you to keep all your work together, in a structured way



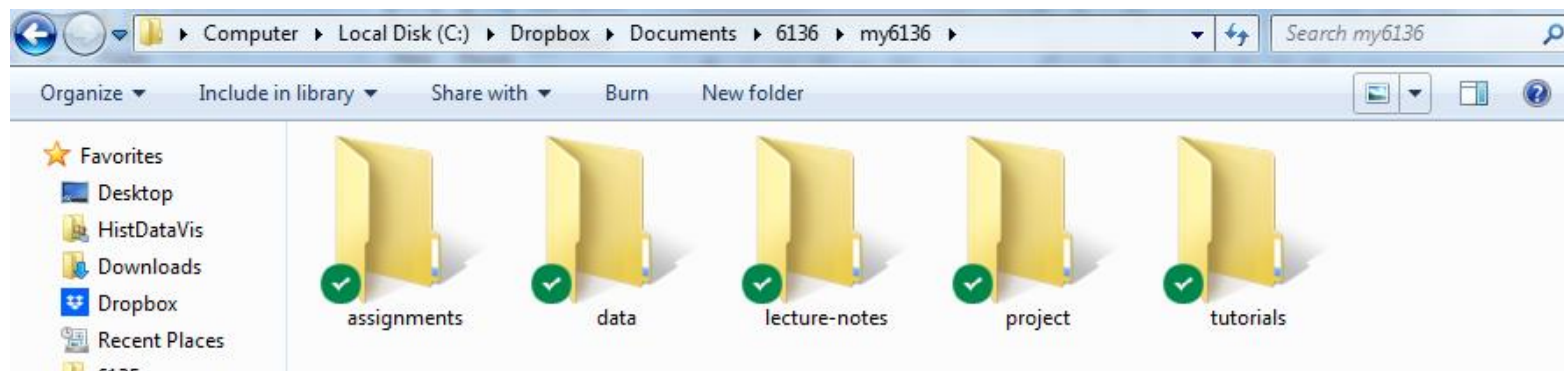
This will create a file, my6136.Rproj
Double-click on this to launch it in RStudio



Your psy6136 project

I also recommend that you create sub-folders there to organize your work

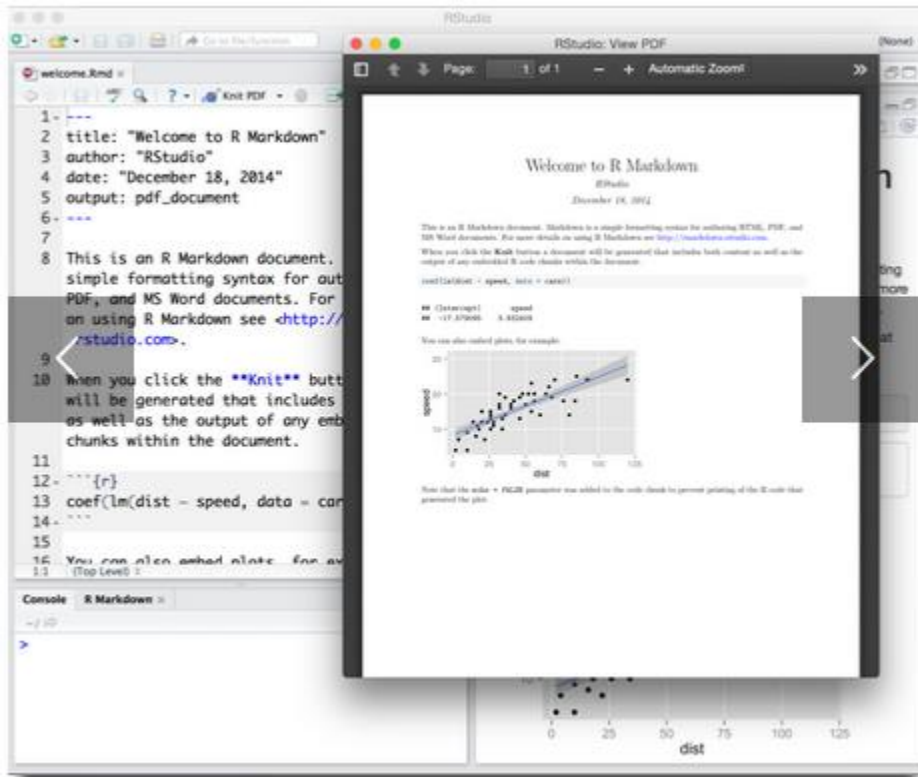
What you name them is up to you, but the main thing is for you to keep things organized by topics.



R scripts, data,
outputs for
assignments

work on
tutorials, R
examples

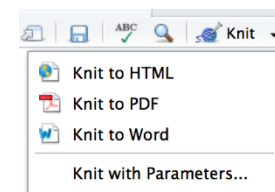
Reproducible analysis & reporting



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

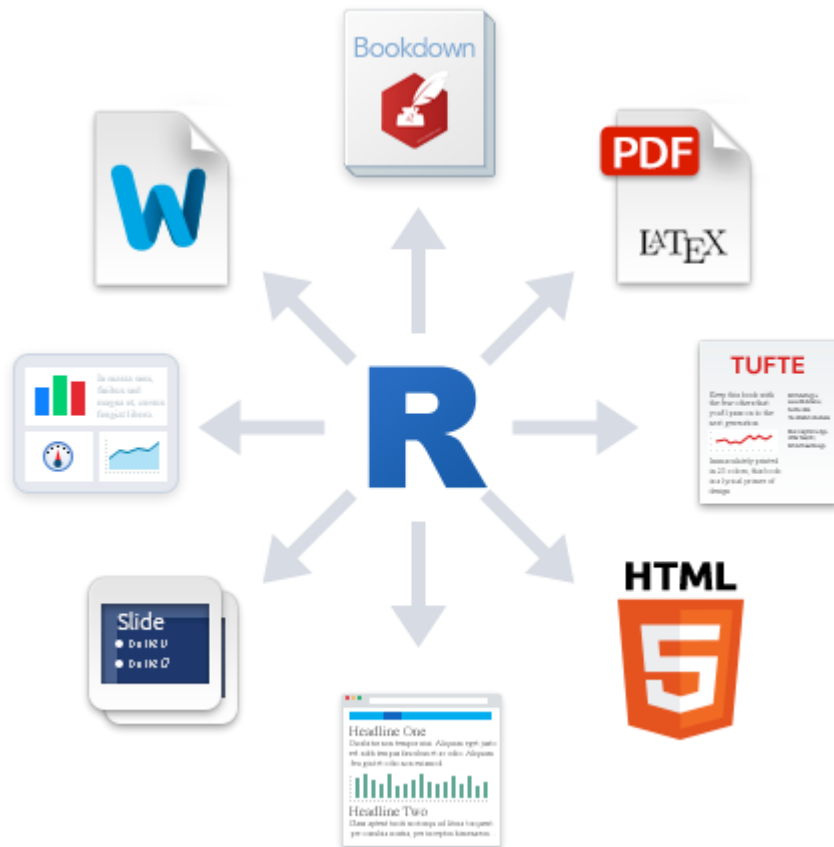
.Rmd files are just text files, using rmarkdown markup and knitr to run R on “code chunks”

A given document can be rendered in different output formats:

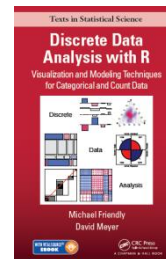


Output formats and templates

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.



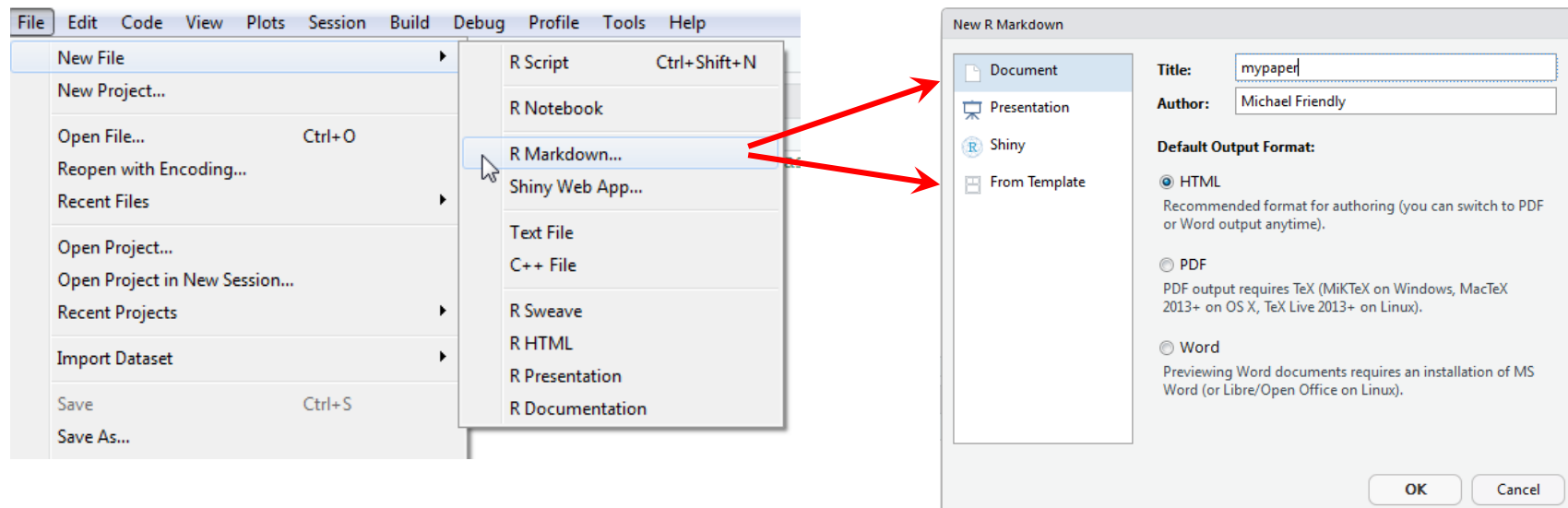
The [bookdown](#) package makes it easier to manage a book-length project – TOC, fig/table #s, cross-references, etc.

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 ```
```

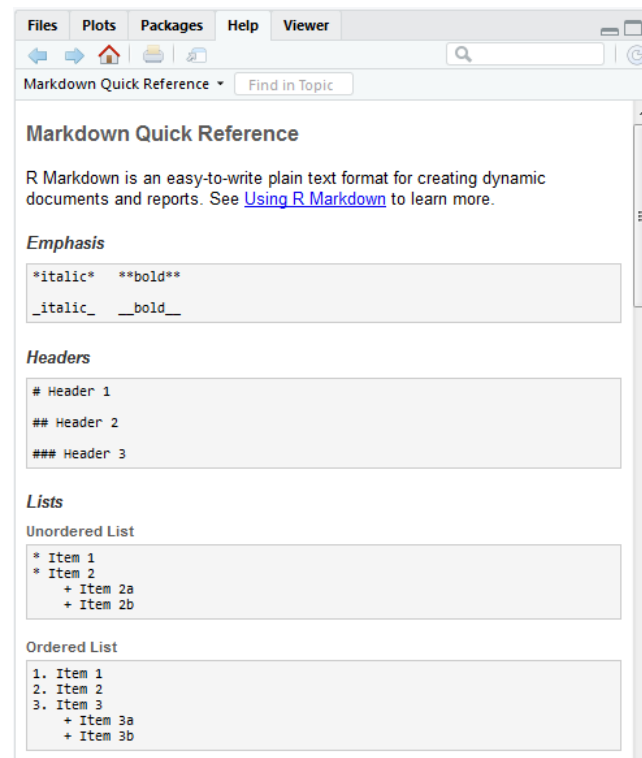
yaml header

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 shows a side-by-side comparison of an R Markdown source file and its rendered HTML output.

Source File (Left Pane):

```
1 # Header 1
2
3 This is an R Markdown document. Markdown is a
4 simple formatting syntax for authoring webpages.
5 Use an asterisk mark to provide emphasis, such
6 as italics or bold.
7 Create lists with a dash:
8
9 - Item 1
10 - Item 2
11 - Item 3
12
13 ```
14 Use back ticks to
15 create a block of code
16 ```
17
18 Embed LaTeX or MathML equations,
19  $\frac{1}{n} \sum_{i=1}^n x_i$ 
20
21 Or even footnotes, citations, and a
22 bibliography. [^1]
23
24 [^1]: Markdown is great.
```

Rendered Document (Right Pane):

Header 1

This is an R Markdown document. Markdown is a simple formatting syntax for authoring web pages.

Use an asterisk mark to provide emphasis, such as *italics* or **bold**.

Create lists with a dash:

- Item 1
- Item 2
- Item 3

Use back ticks to create a block of code

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

Or even footnotes, citations, and a bibliography. ¹

1. Markdown is great.↩

Red arrows indicate the mapping from source to rendered output:

- Line 1: `# Header 1` →

Header 1
- Line 7: `Create lists with a dash:` →

Create lists with a dash:
- Line 18: `Embed LaTeX or MathML equations,` →

Embed LaTeX or MathML equations,

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, a file named 'chunks.Rmd' is open, displaying an R code chunk. The chunk starts with a title 'R Code Chunks' followed by a horizontal line. The code inside the chunk includes a comment about R Markdown, a knitr chunk header, a library call for 'ggplot2', a summary of the 'cars' dataset, and a ggplot of 'dist' vs 'speed' with a smoothed line. A red box highlights the chunk header and code, with a red arrow pointing to the rendered output in the preview window. The preview window, titled 'RStudio: Preview HTML', shows the rendered HTML. It includes the title 'R Code Chunks', the same comment, the summary of the 'cars' dataset, and the ggplot. The plot shows 'dist' on the y-axis (0 to 100) and 'speed' on the x-axis (5 to 25). The plot includes a blue smoothed line and a grey shaded confidence interval. A red arrow points from the code in the Rmd file to the plot 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()
```

R Code Chunks

With R Markdown, you can insert R code chunks including plots:

```
# quick summary and plot
library(ggplot2)
summary(cars)
```

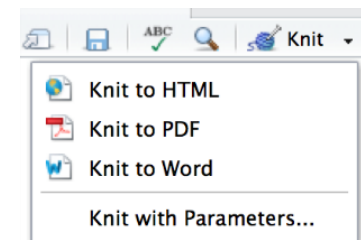
| ## | speed | dist |
|----|---------------|-------------|
| ## | Min. : 4.0 | Min. : 2 |
| ## | 1st Qu.: 12.0 | 1st Qu.: 26 |
| ## | Median : 15.0 | Median : 36 |
| ## | Mean : 15.4 | Mean : 43 |
| ## | 3rd Qu.: 19.0 | 3rd Qu.: 56 |
| ## | Max. : 25.0 | Max. : 120 |

```
qplot(speed, dist, data = cars) + geom_smooth()
```

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

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, RTT, 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. See 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:
 - [GitHub.com](#)
 - [Shinyapps.io](#)
 - RStudio Connect
 - Refresh Document
 - File path to output document
- 6 **Examine build log** in R Markdown console.
- 7 **Use output file** that is saved alongside .Rmd.

.Rmd structure

YAML Header

Optional section of header to e.g. provide options similar to 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 save the location of the .Rmd file as the working directory

Embed code with knitr syntax

Inline code

Insert with `<code>`. Results appear as text without code.

Code chunks

One or more lines surrounded with ````{r}` and `````. Place chunk options within curly braces, after ````{r}`.

Global options

Set with knitr: `opts_chunk$set()`, e.g.

```
knitr::opts_chunk$set(echo = TRUE)
```

Important chunk options:

- cache** - cache results for future knits (default = FALSE)
- cache.path** - directory to save cached results in (default = ".cache/")
- child** - files to knit and then include (default = NULL)
- collapse** - collapse all output into single block (default = FALSE)
- comment** - prefix for each line of results (default = "#")
- dependson** - 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)

Parameters

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

- 1 **Add parameters**
- 2 **Call parameters**
- 3 **Set parameters**

Render

Render the document to HTML, PDF, Word, etc.

Render to HTML

Use `render("doc.Rmd")` to render the document to HTML.

Render to PDF

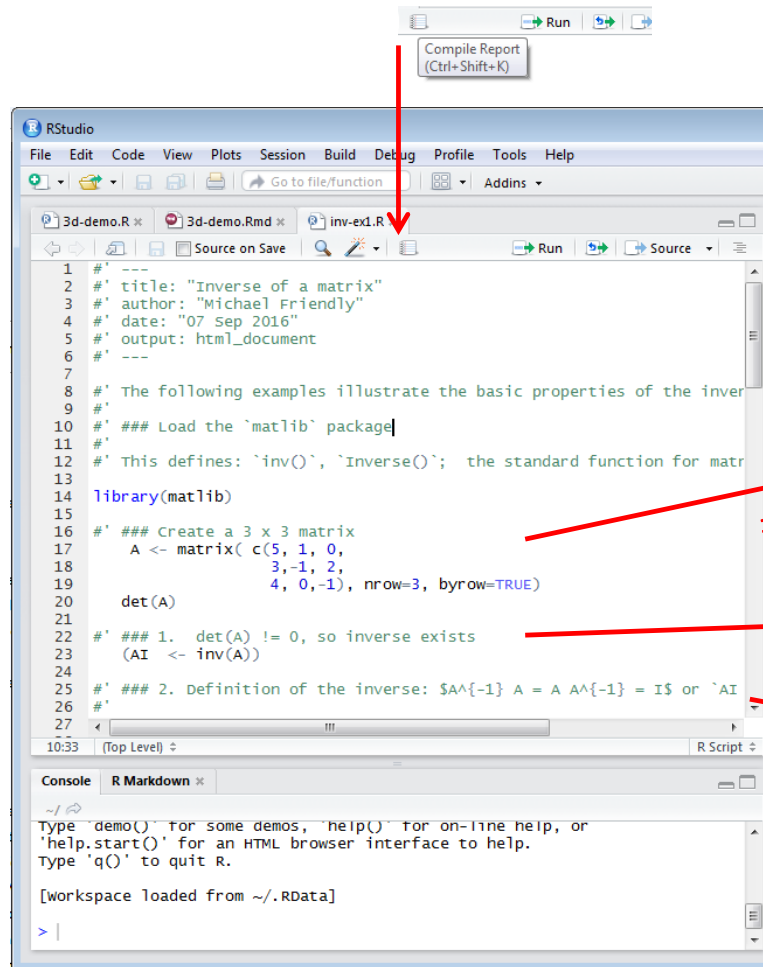
Use `render("doc.Rmd", pdf = TRUE)` to render the document to PDF.

Render to Word

Use `render("doc.Rmd", word = TRUE)` to render the document to Word.

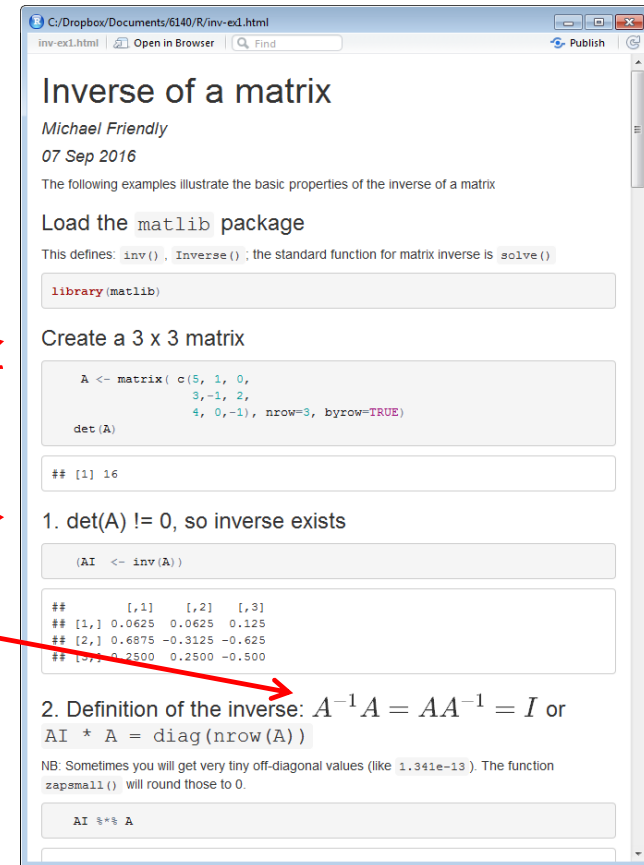
R notebooks

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



The screenshot shows the RStudio interface with a script editor containing R code for creating a 3x3 matrix and calculating its inverse. A red arrow points from the 'Compile Report (Ctrl+Shift+K)' button in the top toolbar to the script editor. The script includes comments and code for loading the 'matlib' package, creating a matrix 'A', and calculating its determinant and inverse.

```
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  $AI = I$  or  $AI = I$ 
27
```



The screenshot shows the compiled HTML report titled "Inverse of a matrix" by Michael Friendly, dated 07 Sep 2016. The report includes the same R code as the script editor, but with the output of the code embedded in the HTML. Red arrows point from the script editor to the HTML report, highlighting the transformation of the code into a readable format.

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 $AI = I$ 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
```