

# Creating and Disseminating Reproducible Research

- *R Packages*
- *RMarkdown Documents*
- *Writing in RMarkdown*

James Balamuta

IMSI Data Science Workshop Series Summer 2022



# Lecture Objectives

- **Create** dynamic reports using Quarto or *RMarkdown*
- **Explore** the R ecosystem for extension packages.

# *R* Packages

# Base R

... default set of features that form the *R* language...

Data

trees			mtcars				...
Girth	Height	Volume	mpg	...	gear	carb	

Help



Functions

mean(x, ...)  
sd(x, ...)  
sum(x, ...)  
prod(x, ...)  
plot(x, y, ...)  
...

# R Packages

... ~18,571 extensions to R available on [CRAN](#) ...



[CRAN](#)  
[Mirrors](#)  
[What's new?](#)  
[Task Views](#)  
[Search](#)

[About R](#)  
[R Homepage](#)  
[The R Journal](#)

[Software](#)  
[R Sources](#)  
[R Binaries](#)  
[Packages](#)  
[Other](#)

[Documentation](#)  
[Manuals](#)  
[FAQs](#)  
[Contributed](#)

[A3](#)  
[abyyR](#)  
[abc](#)  
[abc.data](#)  
[ABC.RAP](#)  
[ABCanalysis](#)  
[abcdeFBA](#)  
[ABCOptim](#)  
[ABCp2](#)  
[abcrf](#)  
[abctools](#)  
[abd](#)  
[abe](#)  
[abf2](#)  
[ABHgenotypeR](#)  
[abind](#)  
[abjutils](#)  
[abn](#)  
[abnormality](#)

## Available CRAN Packages By Name

[A](#) [B](#) [C](#) [D](#) [E](#) [F](#) [G](#) [H](#) [I](#) [J](#) [K](#) [L](#) [M](#) [N](#) [O](#) [P](#) [Q](#) [R](#) [S](#) [T](#) [U](#) [V](#) [W](#) [X](#) [Y](#) [Z](#)

Accurate, Adaptable, and Accessible Error Metrics for Predictive Models  
Access to Abbyy Optical Character Recognition (OCR) API  
Tools for Approximate Bayesian Computation (ABC)  
Data Only: Tools for Approximate Bayesian Computation (ABC)  
Array Based CpG Region Analysis Pipeline  
Computed ABC Analysis  
ABCDE\_FBA: A-Biologist-Can-Do-Everything of Flux Balance Analysis with this package  
Implementation of Artificial Bee Colony (ABC) Optimization  
Approximate Bayesian Computational Model for Estimating P2  
Approximate Bayesian Computation via Random Forests  
Tools for ABC Analyses  
The Analysis of Biological Data  
Augmented Backward Elimination  
Load Gap-Free Axon ABF2 Files  
Easy Visualization of ABH Genotypes  
Combine Multidimensional Arrays  
Useful Tools for Jurimetical Analysis Used by the Brazilian Jurimetrics Association  
Modelling Multivariate Data with Additive Bayesian Networks  
Measure a Subject's Abnormality with Respect to a Reference Population

# Task Views

## Subject Matter Expert Curated Task Views

The screenshot shows two browser windows. The left window displays the main 'CRAN Task Views' page, which lists various task views such as Bayesian, ChemPhys, ClinicalTrials, Cluster, Databases, DifferentialEquations, Distributions, Econometrics, Environmetrics, Epidemiology, ExperimentalDesign, ExtremeValue, Finance, FunctionalData, GraphicalModels, HighPerformanceComputing, Hydrology, MachineLearning, and MedicalImaging. The right window shows a detailed view of the 'Databases' task view, providing information about its maintainer, contact, version, URL, source, contributions, citation, and installation instructions.

CRAN task views aim to provide some guidance which packages on CRAN are relevant for tasks related to a certain topic. They give a brief overview of the included packages and can be automatically installed using the [ctv](#) package. The views are intended to have a sharp focus so that it is sufficiently clear which packages should be included (or excluded) - and they are *not* meant to endorse the "best" packages for a given task.

To automatically install the views, the [ctv](#) package needs to be installed, e.g., via

```
install.packages("ctv")
```

and then the views can be installed via `install.views` or `update.views` (where the latter only installs those packages are not installed).

```
ctv::install.views("Econometrics")
ctv::update.views("Econometrics")
```

The resources provided by the [CRAN Task View Initiative](#) provide further information on how to contribute to existing task views.

**Topics**

<a href="#">Bayesian</a>	Bayesian Inference
<a href="#">ChemPhys</a>	Chemometrics and Computational Physics
<a href="#">ClinicalTrials</a>	Clinical Trial Design, Monitoring, and Analysis
<a href="#">Cluster</a>	Cluster Analysis & Finite Mixture Models
<a href="#">Databases</a>	Databases with R
<a href="#">DifferentialEquations</a>	Differential Equations
<a href="#">Distributions</a>	Probability Distributions
<a href="#">Econometrics</a>	Econometrics
<a href="#">Environmetrics</a>	Analysis of Ecological and Environmental Data
<a href="#">Epidemiology</a>	Epidemiology
<a href="#">ExperimentalDesign</a>	Design of Experiments (DoE) & Analysis of Experimental Data
<a href="#">ExtremeValue</a>	Extreme Value Analysis
<a href="#">Finance</a>	Empirical Finance
<a href="#">FunctionalData</a>	Functional Data Analysis
<a href="#">GraphicalModels</a>	Graphical Models
<a href="#">HighPerformanceComputing</a>	High-Performance and Parallel Computing with R
<a href="#">Hydrology</a>	Hydrological Data and Modeling
<a href="#">MachineLearning</a>	Machine Learning & Statistical Learning
<a href="#">MedicalImaging</a>	Medical Image Analysis

**CRAN Task View: Databases with R**

**Maintainer:** Yuan Tang, James Joseph Balamuta  
**Contact:** terrytangyuan@gmail.com  
**Version:** 2022-02-21  
**URL:** <https://CRAN.R-project.org/view=Databases>  
**Source:** <https://github.com/cran-task-views/Databases/>

**Contributions:** Suggestions and improvements for this task view are very welcome and can be made through issues or pull requests on GitHub or via e-mail to the maintainer address. For further details see the [Contributing guide](#).

**Citation:** Yuan Tang, James Joseph Balamuta (2022). CRAN Task View: Databases with R. Version 2022-02-21. URL <https://CRAN.R-project.org/view=Databases>.

**Installation:** The packages from this task view can be installed automatically using the [ctv](#) package. For example, `ctv::install.views("Databases", coreOnly = TRUE)` installs all the core packages or `ctv::update.views("Databases")` installs all packages that are not yet installed and up-to-date. See the [CRAN Task View Initiative](#) for more details.

This CRAN task view contains a list of packages related to accessibility of different databases. This does not include data import/export or data management. Moreover, the task view on [HighPerformanceComputing](#) and [MachineLearning](#) might provide useful information.

As datasets become larger and larger, it is impossible for people to save them in traditional file formats such as spreadsheet, raw text file, etc., which could not fit on devices with limited storage and could not be easily shared across collaborators. Instead, people nowadays tend to store data in databases for more scalable and reliable data management.

Database systems are often classified based on the [database models](#) that they support. [Relational databases](#) became dominant in the 1980s. The data in relational databases is modeled as rows and columns in a series of tables with the use of [SQL](#) to express the logic for writing and querying data. The tables are relational, e.g. you have a user who uses your softwares and those softwares have creators and contributors. Non-relational databases became popular in recent years due to huge demand in storing unstructured data with the use of [NoSQL](#) as the query language. Users generally don't need to define the data schema up front. If there are changing requirements in the applications, non-relational databases can be much easier to use and manage.

The content presented in this task view is undergoing rapid changes in industries and academia. Please send any suggestions to the maintainer via e-mail or submit an issue or pull request in the GitHub repository linked above. All suggestions and corrections by others are gratefully acknowledged.

**Relational databases**

This section includes packages that provides access to relational databases within R.

- The [DBI](#) package provides a database interface definition for communication between R and relational database management systems. It's worth noting that some packages try to follow this interface definition (DBI-compliant) but many existing packages don't.
- The [RODBC](#) package provides access to databases through an ODBC interface. This package is maintained by the R Core Team and depends only on base R. See

<https://cran.r-project.org/web/views/>

<https://cran.r-project.org/web/views/Databases.html>

# Using R Packages

... installing and loading an *R* package from [CRAN](#) ...

1.

```
# Required once per R computer / Cloud project
install.packages(c("rmarkdown","quarto"))
```

```
trying URL 'https://cran.rstudio.com/bin/macosx/el-capitan/contrib/4.2/rmarkdown_1.8.tgz'
Content type 'application/x-gzip' length 2227986 bytes (2.1 MB)
=====
downloaded 2.1 MB
```

The downloaded binary packages are in  
/var/folders/b0/vt\_1hj2d6yd8myx9lwh81pww0000gn/T/Rtmp66L8EO/downloaded\_packages

2.

```
# Required once per R session
library("rmarkdown"); library("quarto");
```

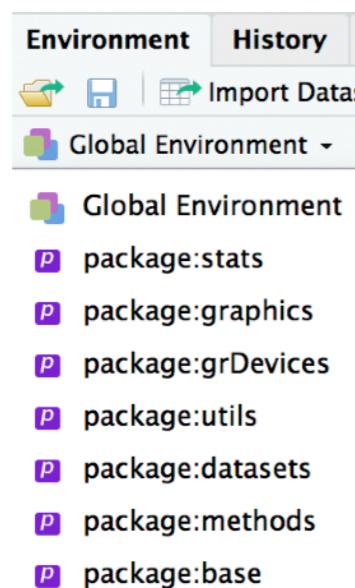
---

\* Installing an *R* package is only required **once per R installation**. Useful to include a **# install.packages("pkename")** prior to the library loads to emphasize

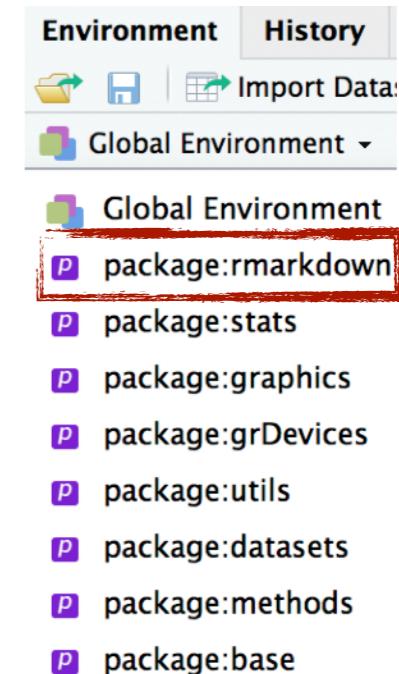
\*\* Inside of all R Markdown documents or R Scripts, you will need to call library.

# Loading an R Package

... what happens when **library()** is used to load a package ...



→ `library("rmarkdown")` →



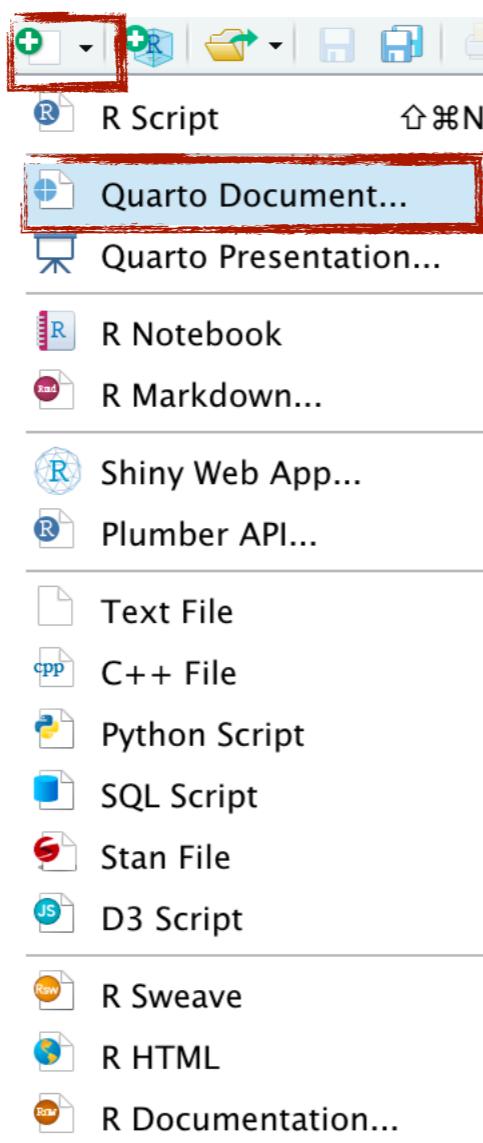
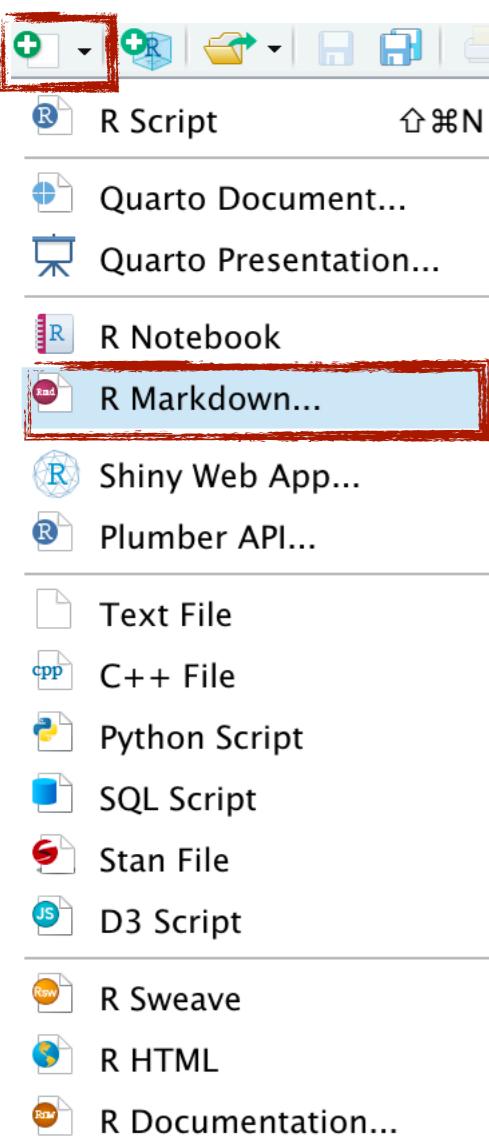
Ask Yourself: "Is the package loaded?"



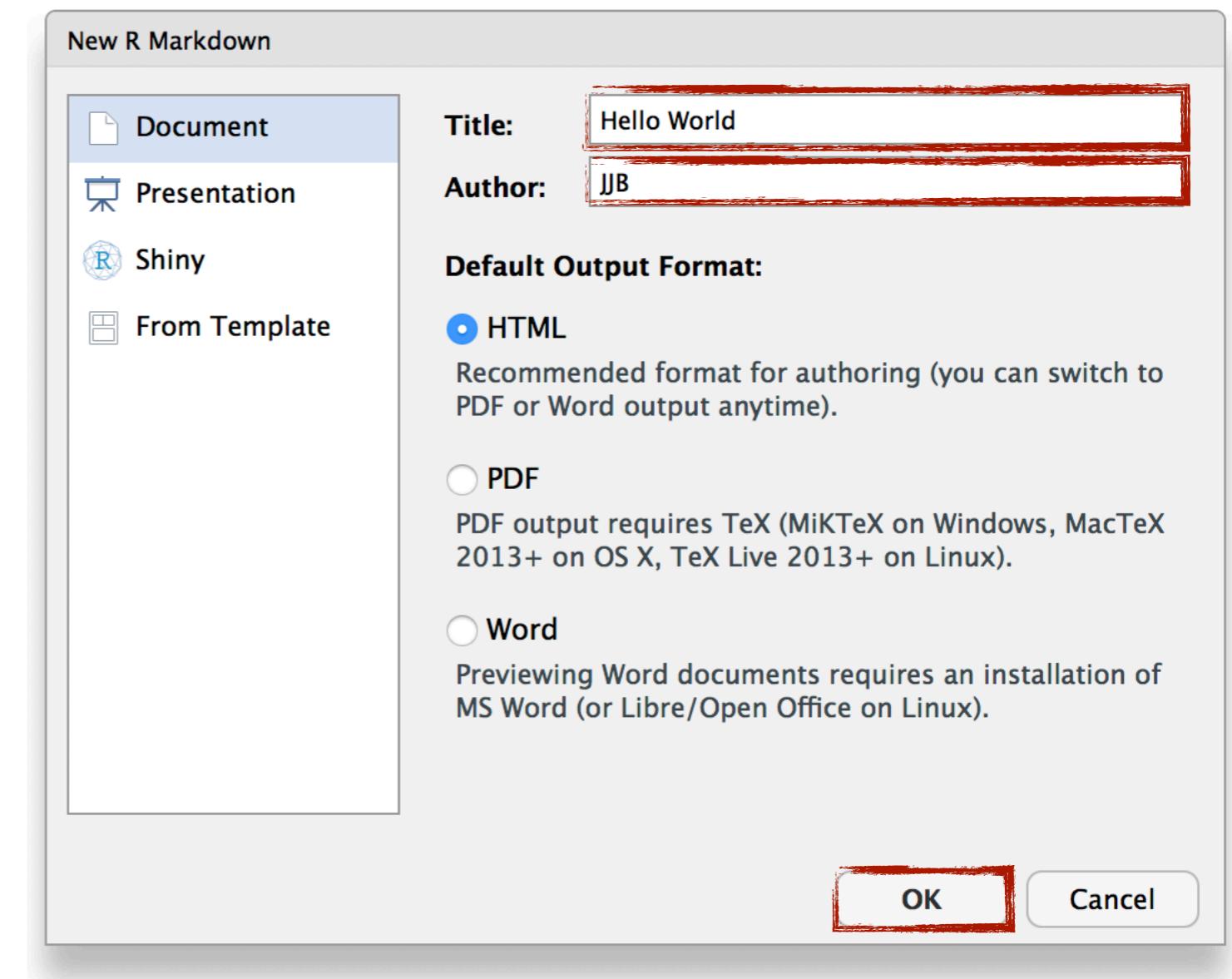
# RMarkdown and Quarto Documents

# Create an RMarkdown Document

Click the White Plus  
Select **R Markdown** or **Quarto** ...

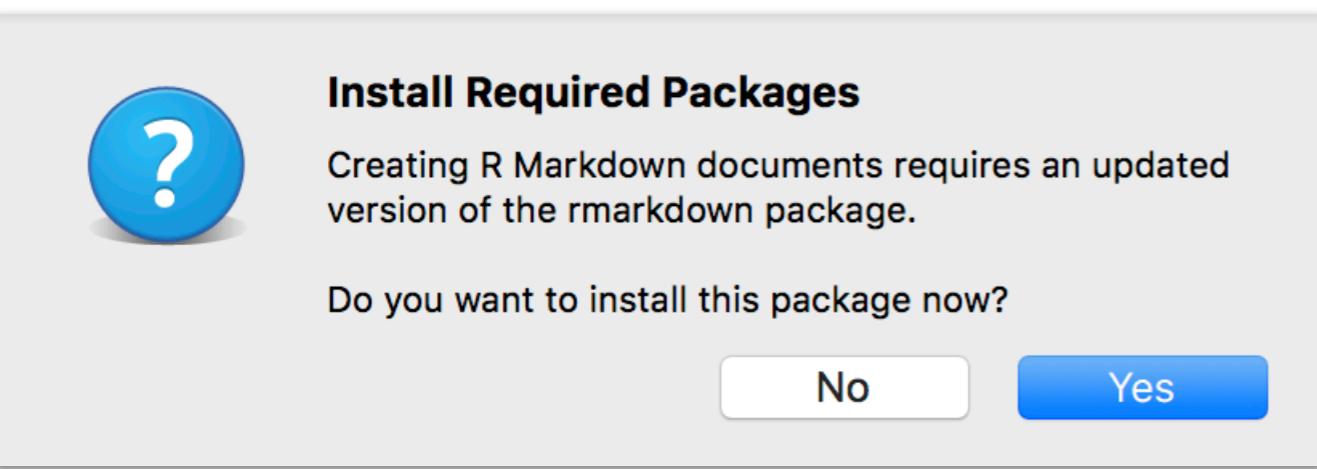


Enter Document Title and Author  
Press **OK**



# Oopsie

... something went wrong ...



```
library("rmarkdown")  
Error in library("rmarkdown") :  
there is no package called  
'rmarkdown'
```

... the package wasn't *installed*, so use ...

```
install.packages("rmarkdown")
```

# RMarkdown Document Sections

Create the output document [Cmd/Cntrl + Shift + K]

Code  
Chunks

```
1 ---  
2 title: "Hello World"  
3 author: "JJB"  
4 date: "1/19/2018"  
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 syntax for  
authoring HTML, PDF, and MS Word documents. For more details on using R  
Markdown see <http://rmarkdown.rstudio.com>.  
15  
16 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:  
17  
18 ```{r cars}  
19 summary(cars)  
20 ...  
21  
22 ## Including Plots  
23  
24 You can also embed plots, for example:  
25  
26 ```{r pressure, echo=FALSE}  
27 plot(pressure)  
28 ...  
29  
30 Note that the `echo = FALSE` parameter was added to the code chunk to  
prevent printing of the R code that generated the plot.  
31
```

Run this Code Chunk  
[Cmd/ Cntrl + Shift + Enter]

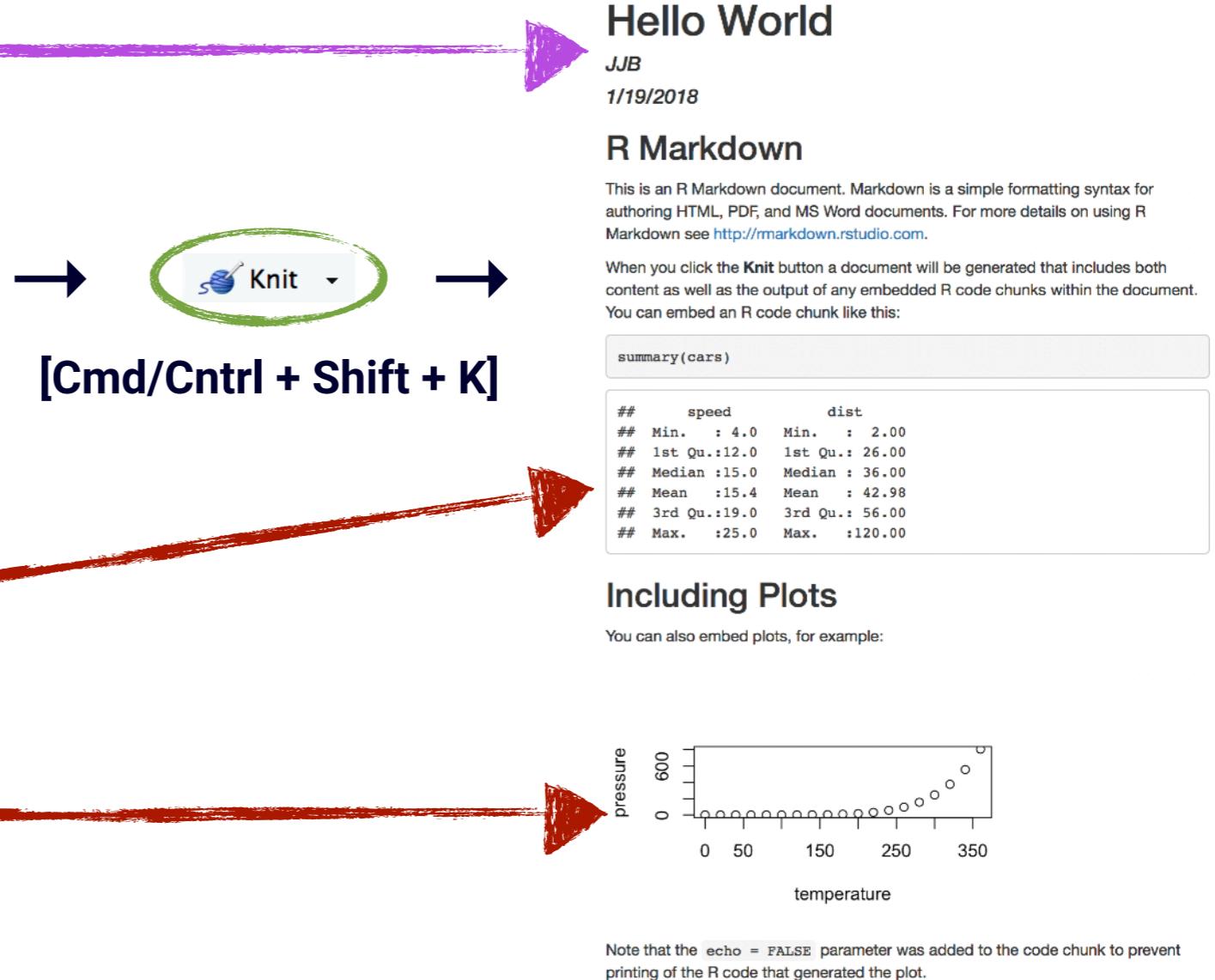
Text with formatting  
that describes what is  
contained in the  
document

Run Code Chunks  
up to here  
[Cmd/Cntrl + Shift/Opt + P]

Change Code Chunk  
options

# RMarkdown Output

```
rmarkdown-sample.Rmd x
1 ---  
2 title: "Hello World"  
3 author: "JJB"  
4 date: "1/19/2018"  
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 syntax for  
authoring HTML, PDF, and MS Word documents. For more details on using R  
Markdown see <http://rmarkdown.rstudio.com>.  
15  
16 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:  
17  
18 ```{r cars}  
19 summary(cars)  
20 ````  
21  
22 ## Including Plots  
23  
24 You can also embed plots, for example:  
25  
26 ```{r pressure, echo=FALSE}  
27 plot(pressure)  
28 ````  
29  
30 Note that the `echo = FALSE` parameter was added to the code chunk to  
prevent printing of the R code that generated the plot.  
31  
2:1 # Hello World R Markdown
```



\* Press "Knit" is the same as typing: `rmarkdown::render("rmarkdown-sample.Rmd")`

# What You See Is What You Get

Write in a Word-like style, but retain the Markdown syntax.

Click



Source Visual Outline

```
1 ---  
2 title: "Hello World"  
3 author: "JJB"  
4 output: html_document  
5 ---  
6  
7 ```{r setup, include=FALSE}  
8 knitr::opts_chunk$set(echo = TRUE)  
9 ---  
10  
11 ## R Markdown  
12  
13 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>.  
14  
15 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:  
16  
17 ```{r cars}  
18 summary(cars)  
19 ---  
20  
21 ## Including Plots  
22  
23 You can also embed plots, for example:  
8:35 C Chunk 1: setup
```

R Markdown

Source Visual B I <> Normal | E F Format Insert Table Outline

R Markdown Including Plots

```
---  
title: "Hello World"  
author: "JJB"  
output: html_document  
---  
  
{r setup, include=FALSE}  
knitr::opts_chunk$set(echo = TRUE)
```

## 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:

```
{r cars}  
summary(cars)
```

C Chunk 1: setup

R Markdown

Markdown Language Editor

WYSIWYG Editor

# Navigate Document

The screenshot shows the RStudio interface with the 'Source' tab selected in the top bar. The main area displays the following R Markdown code:

```
1 ---  
2 title: "Hello World"  
3 author: "JJB"  
4 output: html_document  
5 ---  
6 ---  
7 ```{r setup, include=FALSE}  
8 knitr::opts_chunk$set(echo = TRUE)  
9 ---  
10 ---  
11 ## R Markdown  
12 ---  
13 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>.  
14 ---  
15 When you click the **Knit** button a document will be generated that includes both  
16 code output of any embedded R code chunks within the document.  
17 ---  
18 ---  
19 ---  
20 ---  
21 ---
```

The 'Outline' tab is highlighted with a red box. The 'Outline' pane on the right shows a tree structure of the document's contents, including 'R Markdown Including Plots'. The 'Hello World' section is expanded, showing 'Chunk 1: setup', 'R Markdown', 'Chunk 2: cars', 'Including Plots', and 'Chunk 3: pressure'. A red box highlights the 'Hello World' section in the outline tree.

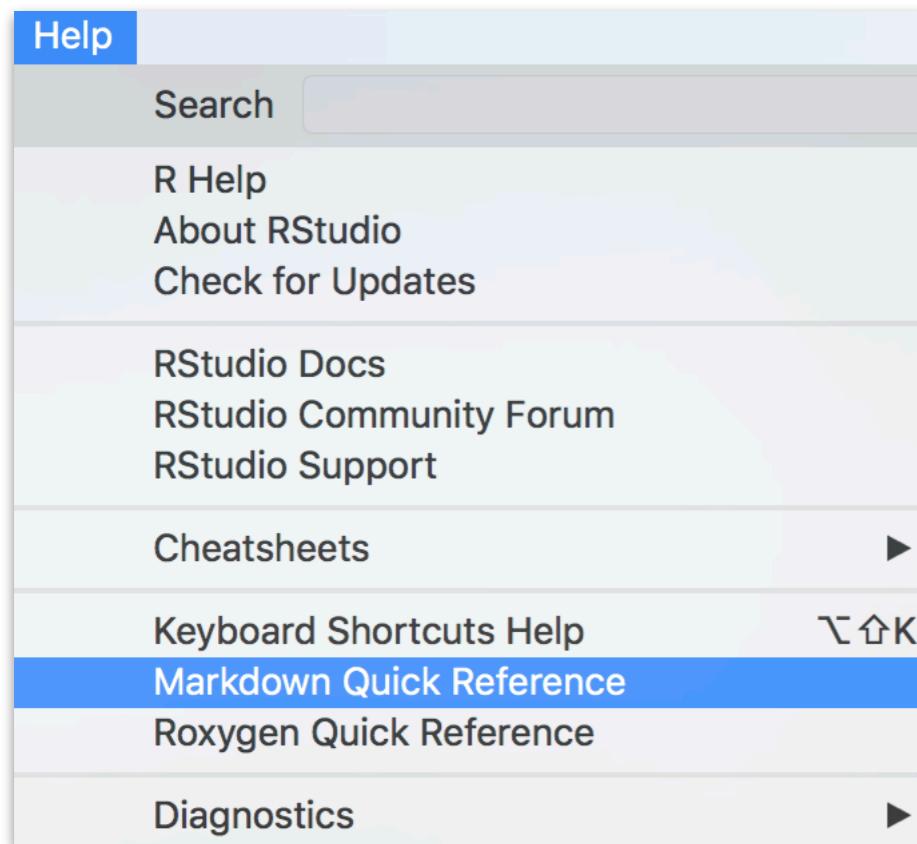
Outline  
form

Navigator form

# Writing in RMarkdown

# Help Writing

... markdown syntax guide built-in to RStudio ...



The screenshot shows the RStudio Viewer pane displaying the 'Markdown Quick Reference' guide. The page includes sections for Emphasis, Headers, Lists, and Examples.

**Emphasis**

```
*italic* **bold**  
_italic_ __bold__
```

**Headers**

```
# Header 1  
## Header 2  
### Header 3
```

**Lists**

Unordered List

```
* Item 1  
* Item 2  
  + Item 2a  
  + Item 2b
```

Ordered List

```
1. Item 1  
2. Item 2  
3. Item 3  
  + Item 3a  
  + Item 3b
```

# Code Demo

... writing in rmarkdown ...

# Text Emphasis & Links

... making text stand out and pointing to external resources ...

## Code

Writing text with emphasis in  
\*italics\*, \*\*bold\*\* and `code style`.

## Output

Writing text with emphasis in  
*italics*, **bold** and code style.

Line breaks create a new paragraph.

Links can be hidden e.g.  
[illinois](www.illinois.edu) or  
not <<http://illinois.edu>> .

Line breaks create a new paragraph.

Links can be hidden e.g.  
[illinois](#) or not <http://illinois.edu>.

# Lists

... ordered and unordered ...

## Code

My \*\*un\*\*ordered list:

- Write Selection Simulation
- Conference Abstracts
  - UseR
  - Learning at Scale



## Output

My **unordered** list:

- Write Selection Simulation
- Conference Abstracts
  - UseR
  - Learning at Scale

My \*\*ordered\*\* list:

1. Apples
1. Bananas
1. Chobani
  - 1. Pineapple
  - 1. Everything else



My **ordered** list:

1. Apples
2. Bananas
3. Chobani
  - 1. Pineapple
  - 2. Everything else

---

\* Make sure a new line (space) exists between text and the first list item.

\*\* ~~\_\_\_\_\_~~ Indent four spaces to create a new level in the list.

# Quotes & Math

... imparting wisdom and funky symbols ...

## Code

```
> "Never gonna give you up,  
> never gonna let you down..."  
>  
> --- Rick Astley
```



## Output

"Never gonna give you up, never  
gonna let you down..."  
– Rick Astley

Inline math  $a^2 + b^2 = c^2$



Display math (centered math) 
$$1 - x = y$$

Inline math  $a^2 + b^2 = c^2$

Display math (centered math)

$$a^2 + b^2 = c^2$$

# Tables \* \*\*

... organizing output in a rectangular form ...

## Code

Left	Center	Right
Hey, check it out	Colons provide	873
its <b>Markdown</b>	alignment thus	1000
right in the table	<i>*centered*</i> text	



## Output

Left	Center	Right
Hey, check it out	Colons provide	873
its <b>Markdown</b>	alignment thus	1000
right in the table	<i>centered</i> text	

\* Markdown table generator: [https://www.tablesgenerator.com/markdown\\_tables](https://www.tablesgenerator.com/markdown_tables)

\*\* Other supported markdown table styles:

[http://rmarkdown.rstudio.com/authoring\\_pandoc\\_markdown.html#tables](http://rmarkdown.rstudio.com/authoring_pandoc_markdown.html#tables)

# Code Chunks

... how we embed code into the document ...

## Example

```
```{r chunk-label}  
# Your R code here  
...``
```

## Code

We're embedding \_R\_ code  
**\*\*into\*\*** a report!!

```
```{r add_nums}  
1 + 2  
...``
```

## Output

We're embedding *R* code  
**into** a report!!

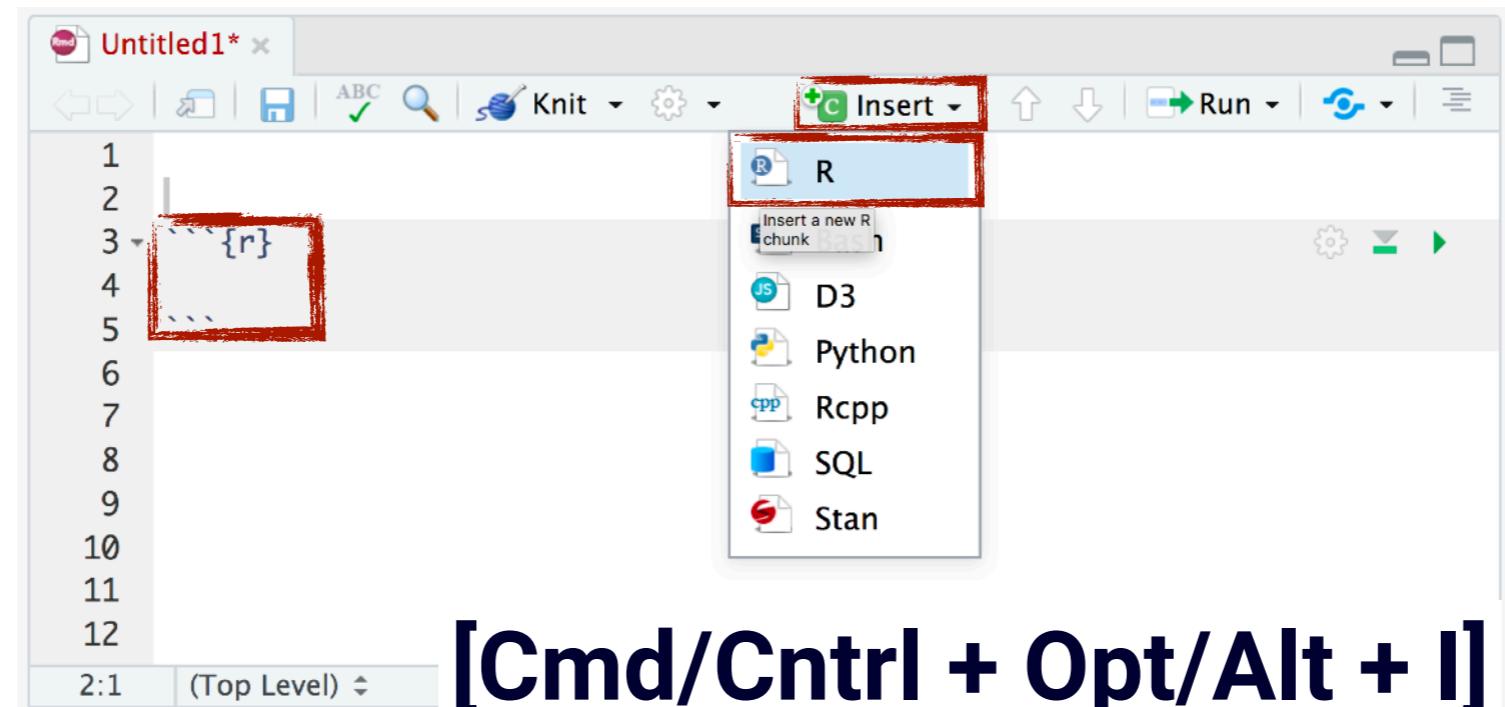
1 + 2

## [1] 3

\* Syntax or code highlighting is retained.

# Ways to Create a Code Chunk

... dropdown menus, short cuts, and keyboard movements ...



# Pets or Livestock

... to label or not to label ...

Console Terminal **R Markdown** Jobs

processing file: rmarkdown-sample-names.Rmd

|.....  
ordinary text without R code

|.....  
**label: add\_num** ←  
|.....  
ordinary text without R code

|.....| 100%  
**label: unnamed-chunk-1** ←

| 25%

| 50%

| 75%

| 100%

```{r add\_nums}

1 + 2

```



```{r}

1 + 2

```



# Code Chunk Options

... customizing how the embedded code/results appear ...

```
```{r chunk_label, chunkopt1 = value1, chunkopt2 = value2}
# Code here
````
```

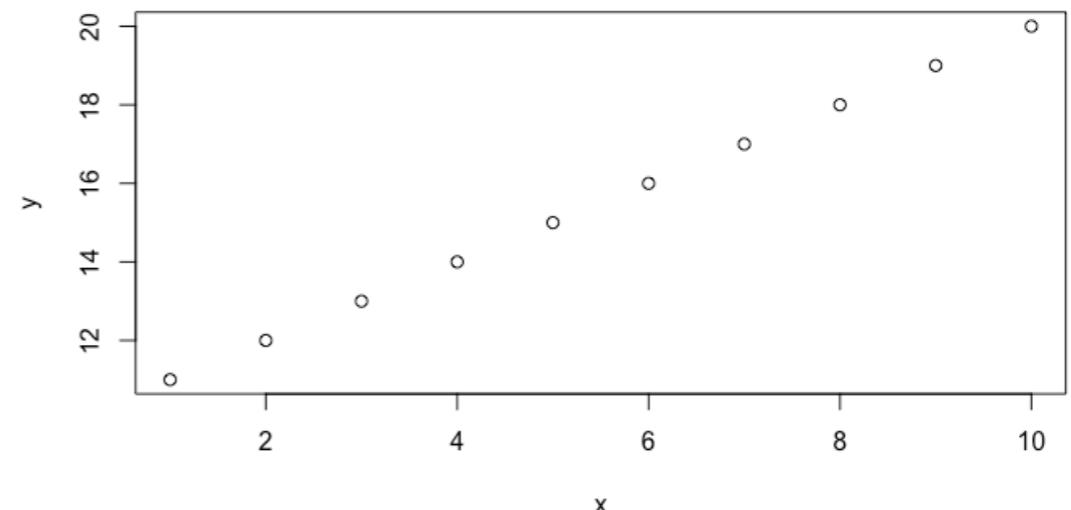
## Code

Let's hide the R code from showing up in the report!

```
```{r ex-hide, echo = FALSE}
x = 1:10
y = 11:20
plot(x, y)
````
```

## Output

Let's hide the *R* code from showing up in the report!



# Common Chunk Options\*

... most frequently used options ...

| Chunk Option         | Description                       |
|----------------------|-----------------------------------|
| cache = TRUE         | Stores to file code chunk results |
| eval = FALSE         | Doesn't evaluate/run the code     |
| eval = 2:3           | Evaluate only lines 2 and 3       |
| echo = FALSE         | Doesn't display the code          |
| echo = -c(1,3,4)     | Hide only lines 1, 3 and 4        |
| results = 'hide'     | Doesn't display code output       |
| include = FALSE      | Doesn't display code or output    |
| fig.path = 'img/'    | File path to store images         |
| fig.align = 'center' | Align image in the center         |
| fig.width = #        | Sets width of figure              |
| fig.height = #       | Sets height of figure             |
| warning = FALSE      | Hides R's warning messages        |
| message = FALSE      | Hides R's note messages           |

---

\* More options: <http://yihui.name/knitr/options>

\*\* Caching is very useful and dangerous, more on this later

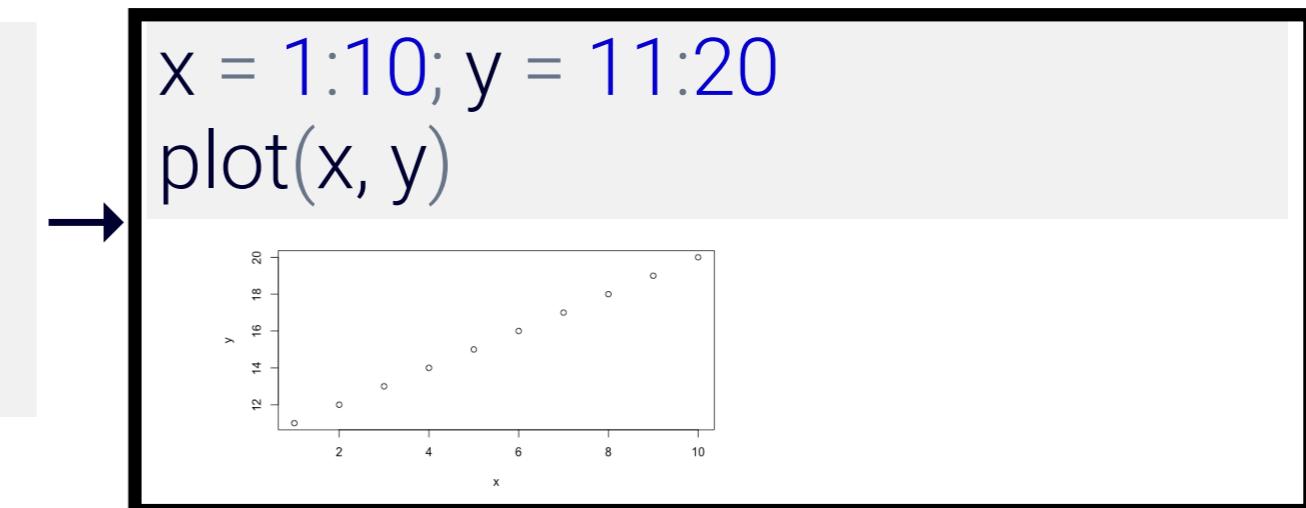
# eval vs. echo

... difference between code options ...

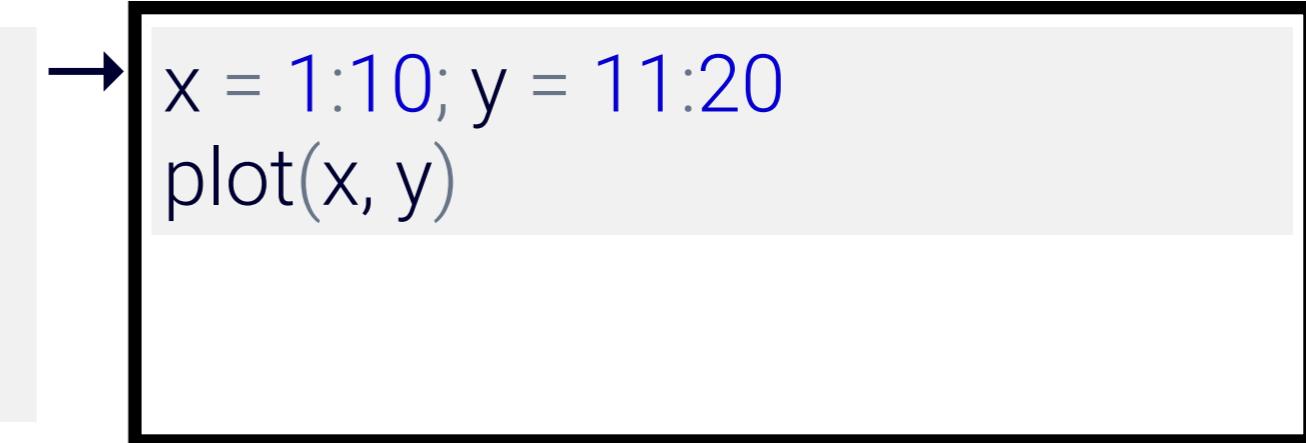
## Code

```
```{r ex-orig}  
x = 1:10; y = 11:20  
plot(x, y)  
````
```

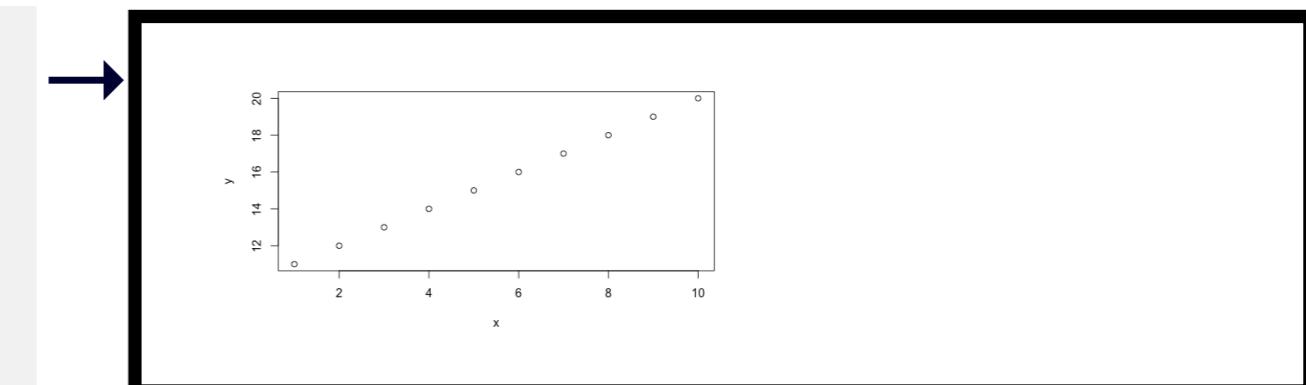
## Output



```
```{r ex-not-run, eval = FALSE}  
x = 1:10; y = 11:20  
plot(x, y)  
````
```



```
```{r ex-hide-code, echo = FALSE}  
x = 1:10; y = 11:20  
plot(x, y)  
````
```



# Embedding In-line R Code

... evaluating an *R* expression without a code chunk ...

Inline \_R\_ code is possible with **`r expression\_here`**

## Code

```
```{r calc-values, echo = FALSE}
x = 1:10
x_mu = mean(x)
x_sd = sd(x)
````
```



Pre-compute values  
in code chunks

## Text

The \_mean\_ of \*\*x\*\* is `r x\_mu` and  
the \_standard deviation\_ is `r x\_sd`.



## Output

The mean of **x** is 5.5 and the  
standard deviation is 3.02765.

**Access** variable contents

# Inserting Images

... adding non-R produced images ...

## Code

![Here is a Relative Path](img/block-i.png)

Caption

Path to File



**Relative Path**

## Output



![Here is an Absolute path](C:/jjb/img/block-i.png)

**Absolute Path**



- 
- \* Relative paths are the *best* to use if the goal is to share your work with others as they are operating system independent. More next!
  - \*\* For instance, do you have a user called "jjb" on your computer with a folder "img"?

# Recap

- **R Packages are extensions to R**
  - `install.packages('pkg')` run **once per** computer / cloud project.
  - `library('pkg')` use for **every** analysis to enable the extension.
- **RMarkdown**
  - Combine *R* code with analysis in a reproducible manner.

This work is licensed under the  
Creative Commons  
Attribution-NonCommercial-  
ShareAlike 4.0 International  
License

