University of BRISTOL

Reproducible data science

Combining R Studio, Git & R Markdown for reproducibility

Henry W J Reeve

henry.reeve@bristol.ac.uk

Statistical Computing & Empirical Methods (EMATM0061)

MSc in Data Science, Teaching block 1, 2021.

What will we cover today?

- We will discuss the three main goals of scientific research
- We will discuss the central role of replicability in science.
- We consider the “replication crisis”.
- We will understand the importance of reproducible data analysis.
- We will introduce several tools for facilitating reproducible data analysis in R:
    - R Markdown
    - R Projects
    - Git integration.

Three goals for scientific research

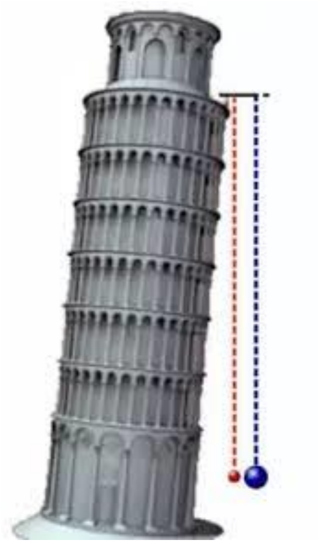
Data Science is the science of extracting information, insight and understanding from data!

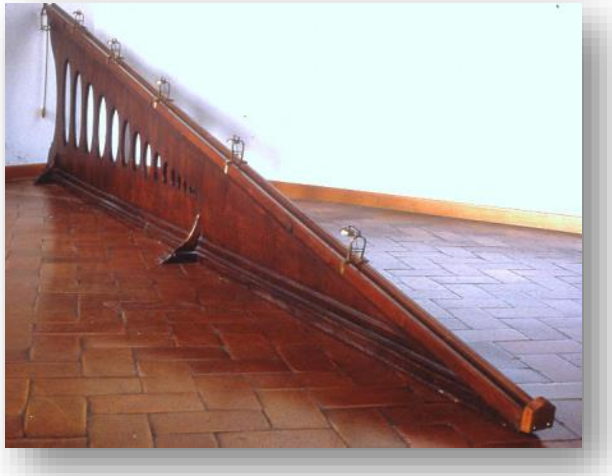
Reliable: Scientific research should be potentially replicable by other scientists.

Valid: Scientific research should use principled approaches to ensure that the results of the investigation actually support the conclusions drawn.

Importance: Scientific research should address important questions.

The importance of replication





Galileo Galilei, circa 1590:

Two objects of the same shape and material but different weights will fall at the same rate.

Replicated many times and in many locations, including on the moon – Apollo 15, 1971.

The replication crisis!

- Scientific truths should be robust to repeated replications of the same experiment.

- John Ioannidis has argued that “most published research findings are false” Nature, 2005.

- Replication studies conducted by pharmaceutical companies:

Prinz, Schlange, Asadullah from Bayer Healthcare, 2011, 67 replication projects:

“In almost two-thirds of the projects, there were inconsistencies between

published data and in-house”

Begley & Ellis from Amgen, 2012, 53 landmark studies,

“scientific findings were confirmed in only 6 cases”.

Essay

Why Most Published Research Findings Are False

It can be proven that most claimed research findings are false.

Replicability vs. Reproducibility

- Scientific truths should be robust to repeated replications of the same experiment.

- **Replicability:** Different experimenters will yield the same results from different data, when an experiment is repeated under similar conditions.

- **Reproducibility:** Different scientists will yield the same results by repeating the analysis on the same data.

- Surprisingly, reproducibility is still a challenge!

  Difficult to reproduce analysis spread across a poorly organized amalgam of code & spread sheets.

Now take a break!

Literate programming & reproducibility

- Donald Knuth emphasized the importance of literate programming:

> "Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do."

- Make your code as readable as possible – both for others and your future self!
  - Include plenty of clear comments
  - Adopt sensible naming conventions
  - Aim for a simple organizational structure with succinct functions.

Reproducibility with R & RStudio

- RStudio facilities reproducible analysis via R Projects, R Markdown and Git interface.

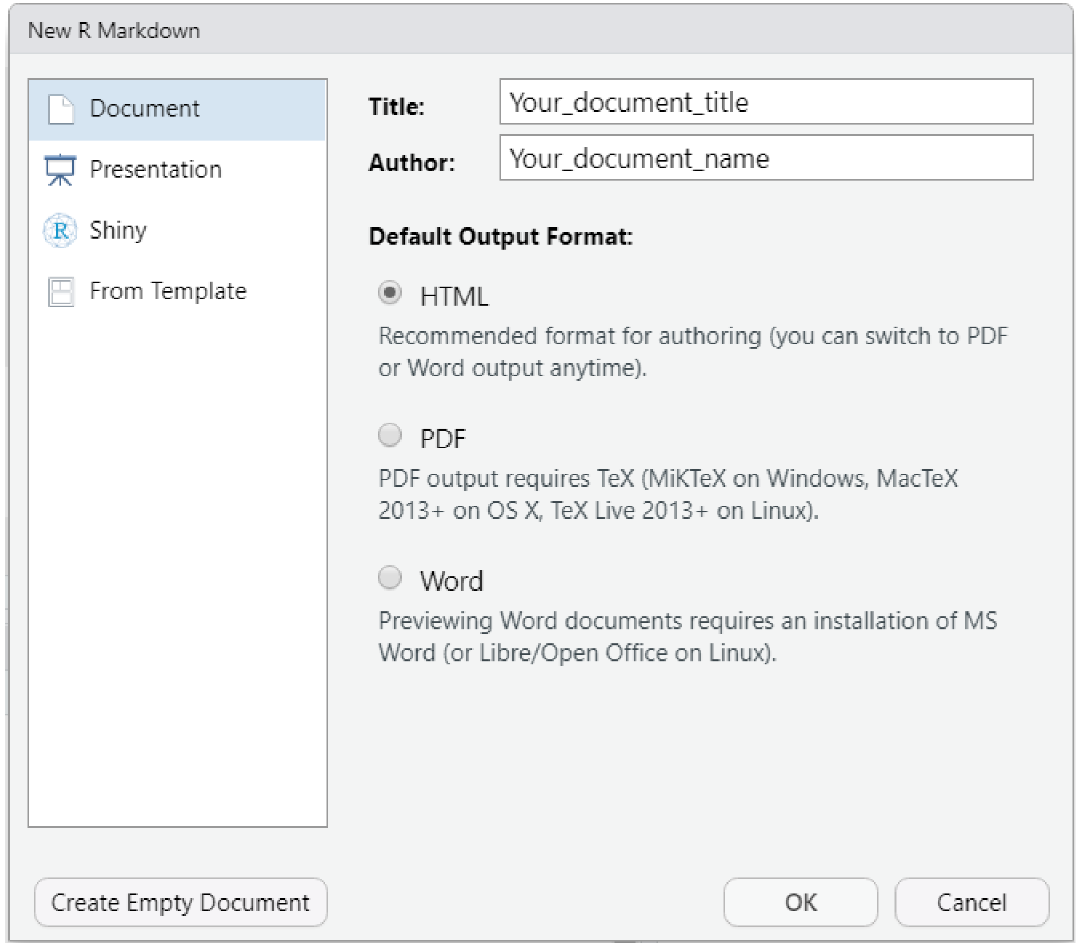
- RMarkdown allows us to generate a notebook style document which includes R code, plots and explanatory text in a linear format.

- R Projects provide a specific workspace for each project with a working directory, data & history.

- Git is a version control system which allows us to track and revert changes, and collaborate.

R Markdown

- We can create a new R Markdown document by File -> New File -> R Markdown …



Title: Your_document_title

Author: Your_document_name

Default Output Format:

● HTML

Recommended format for authoring (you can switch to PDF or Word output anytime).

○ PDF

PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

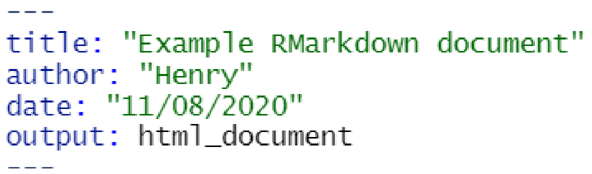
○ Word

Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

R Markdown

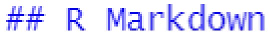
- We can create a new R Markdown document by File -> New File -> R Markdown …

- We can edit the title, author, date, output:



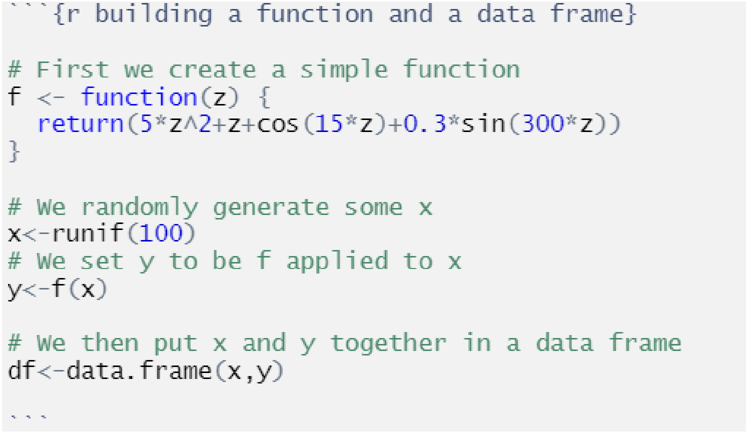
```
---
title: "Example RMarkdown document"
author: "Henry"
date: "11/08/2020"
output: html_document
---
```

- Section headings are generated by



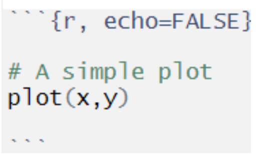
```
## R Markdown
```

- We can also embed code fragments:



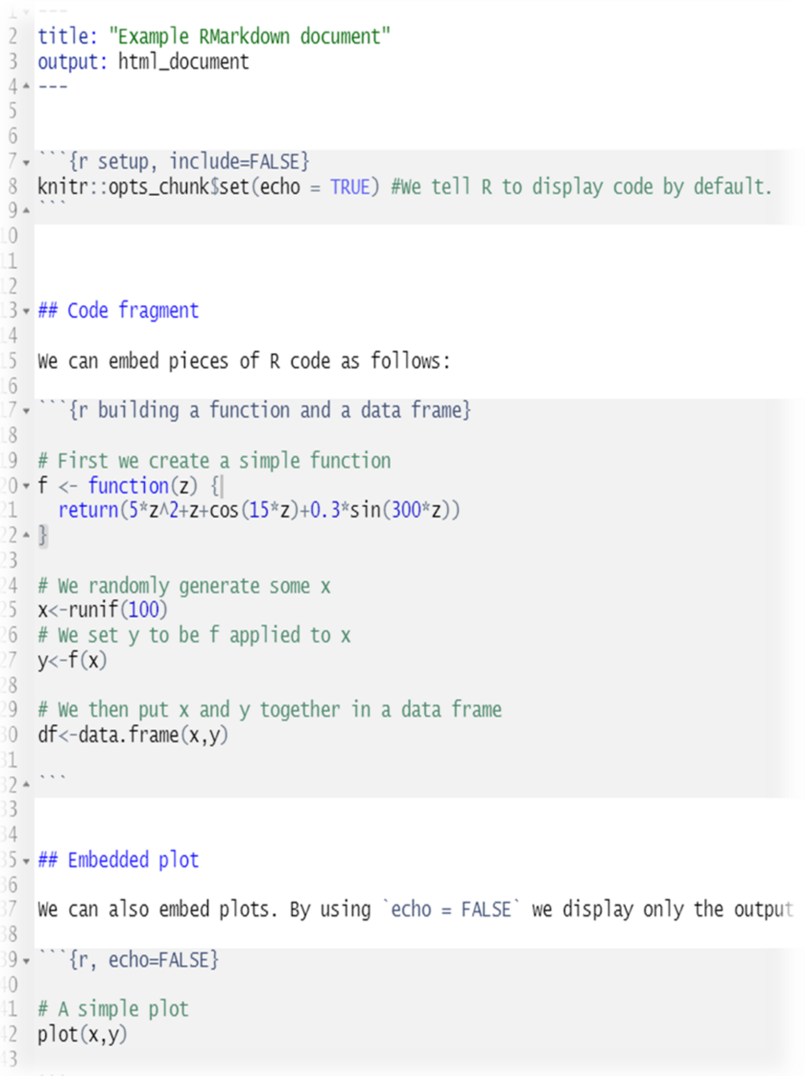
~~~
```{r building a function and a data frame}

# First we create a simple function
f <- function(z) {
  return(5*z^2+z+cos(15*z)+0.3*sin(300*z))
}

# We randomly generate some x
x<-runif(100)
# We set y to be f applied to x
y<-f(x)

# We then put x and y together in a data frame
df<-data.frame(x,y)

```
~~~

- We can also include plots:



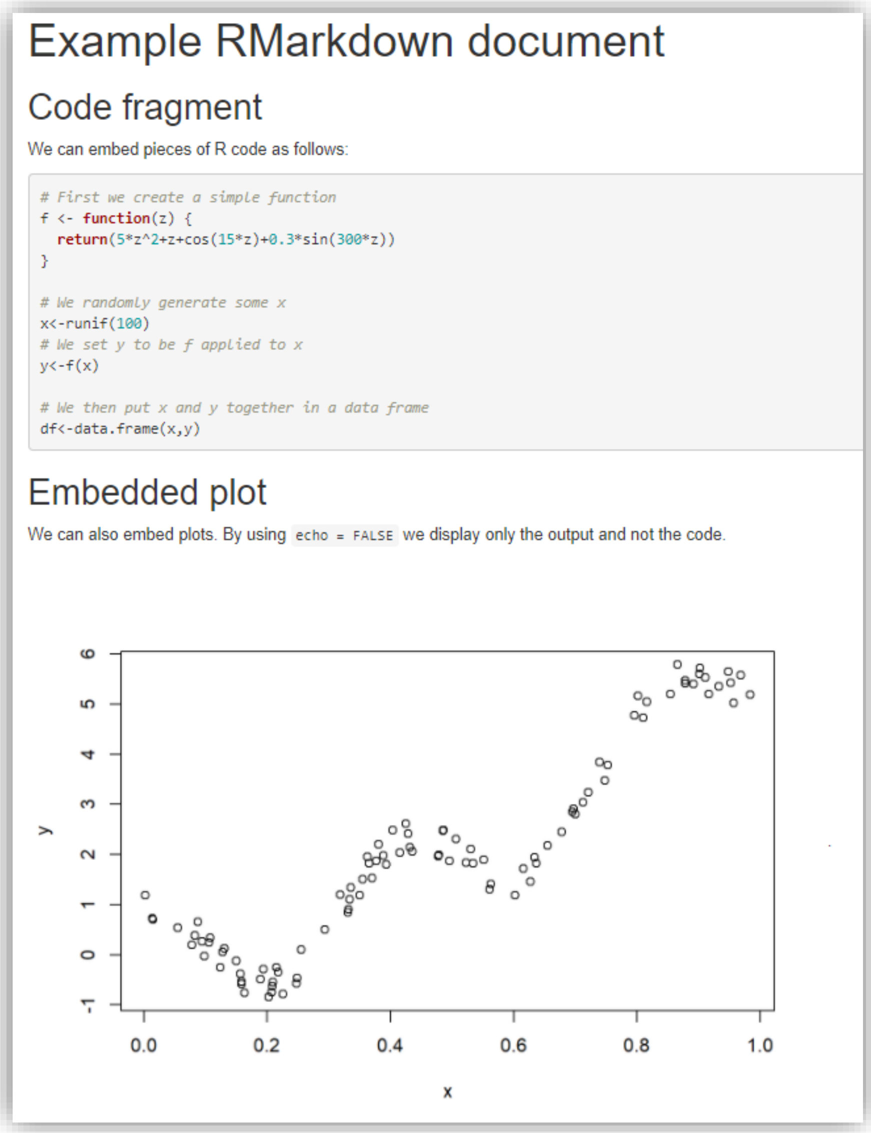
~~~
```{r, echo=FALSE}

# A simple plot
plot(x,y)

```
~~~

R Markdown



~~~
---
title: "Example RMarkdown document"
output: html_document
---


```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE) #We tell R to display code by default.
```



## Code fragment

We can embed pieces of R code as follows:

```{r building a function and a data frame}

# First we create a simple function
f <- function(z) {
  return(5*z^2+z+cos(15*z)+0.3*sin(300*z))
}

# We randomly generate some x
x<-runif(100)
# We set y to be f applied to x
y<-f(x)

# We then put x and y together in a data frame
df<-data.frame(x,y)

```


## Embedded plot

We can also embed plots. By using `echo = FALSE` we display only the output

```{r, echo=FALSE}

# A simple plot
plot(x,y)
```
~~~

Knit



Example RMarkdown document

Code fragment

We can embed pieces of R code as follows:

```
# First we create a simple function
f <- function(z) {
  return(5*z^2+z+cos(15*z)+0.3*sin(300*z))
}

# We randomly generate some x
x<-runif(100)
# We set y to be f applied to x
y<-f(x)

# We then put x and y together in a data frame
df<-data.frame(x,y)
```

Embedded plot

We can also embed plots. By using `echo = FALSE` we display only the output and not the code.

Now take a break!

Version control with Git

- Go to https://github.com and register for free GitHub account.

- Install git locally:

Windows:	https://gitforwindows.org
Mac OSX:	`xcode-select --install`
Ubuntu/Debian:	`sudo apt-get install git`
Fedora/Redhat:	`sudo yum install git`

- Connect to your Git account within R:

```
install.packages("usethis")
library(usethis)
use_git_config(user.name = “Bob Smith", user.email = “bob@example.org")
```

Set up an R project with Git version control

- Go to https://github.com & create a new repository by pressing New .

  Add an informative title, a description and include a README.

- Within the github repo click on Code and copy the repo URL.

- Create a new project within R Studio:

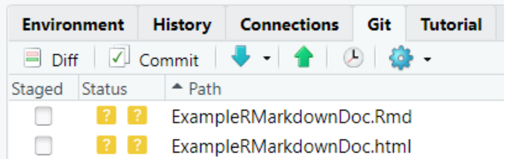
  File -> New Project -> Version Control -> Git -> Enter repo URL + Project name.

  Check “Open in new session” and then create the project Create Project .

- We can now add files, commit, push and pull using the Git panel in the top right of RStudio.

Set up an R project with Git version control

- We can now add files, commit, push and pull using the Git panel in the top right of RStudio.



Staged	Status	Path
□	? ?	ExampleRMarkdownDoc.Rmd
□	? ?	ExampleRMarkdownDoc.html

- **Stage files**: Choose which files to include in the version history by clicking ☑ by the file name.
- **Commit**: Take a snapshot of staged files within the local git repository by ☑ Commit .

  Remember to include a succinct but informative commit message.
- **Push**: Send your local changes to the master branch ⬆ .
- **Pull**: Copy changes made by your collaborators onto your local repository ⬇ .
- An excellent resource for more information from Jenny Bryan : https://happygitwithr.com

What have we covered?

- We discussed the central role of replicability in science.
- We considered the idea of a “replication crisis”.
- We discussed the difference between replicability and reproducibility.
- We introduced RMarkdown for reproducible data analysis.
- We discussed the integration of Git with RStudio and R projects.

University of BRISTOL

Thanks for listening,

… now onto the assignment!

henry.reeve@bristol.ac.uk

Include EMATM0061 in the subject of your email.