Reproducible data science

Combining RStudio, Git & R Markdown for reproducibility

Statistical Computing and Empirical Methods
Unit EMATM0061, Data Science MSc

Rihuan Ke

rihuan.ke@bristol.ac.uk

Teaching Block 1, 2024

What we will cover in this lecture

- 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.

Replicability vs. Reproducibility in data science

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.

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.

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

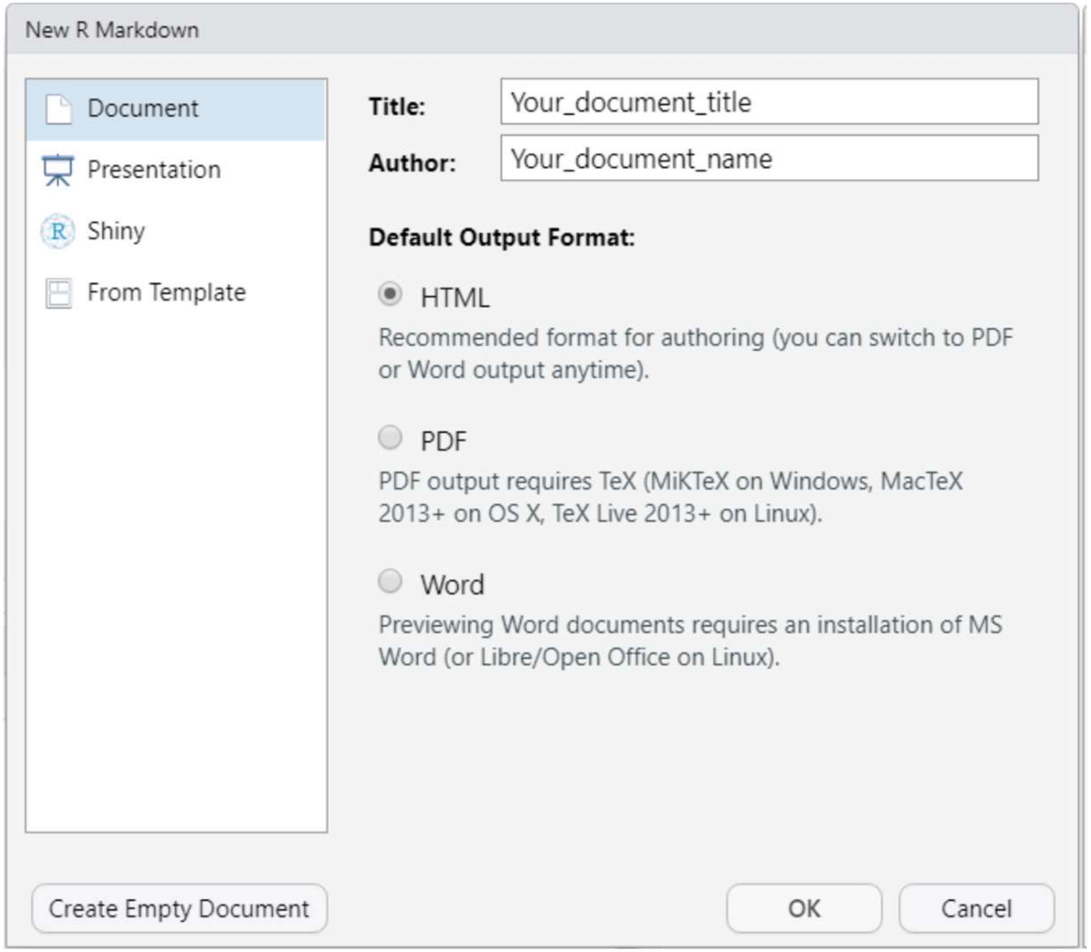
R Markdown allows us to generate a notebook style document which includes R code, plots and explanatory text in a linear format.

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

R Markdown

Create a new R Markdown document on RStudio

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

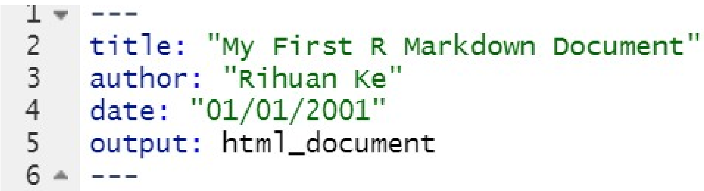
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

○ Word

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

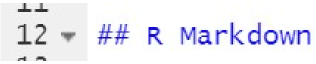
Edit R Markdown document

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



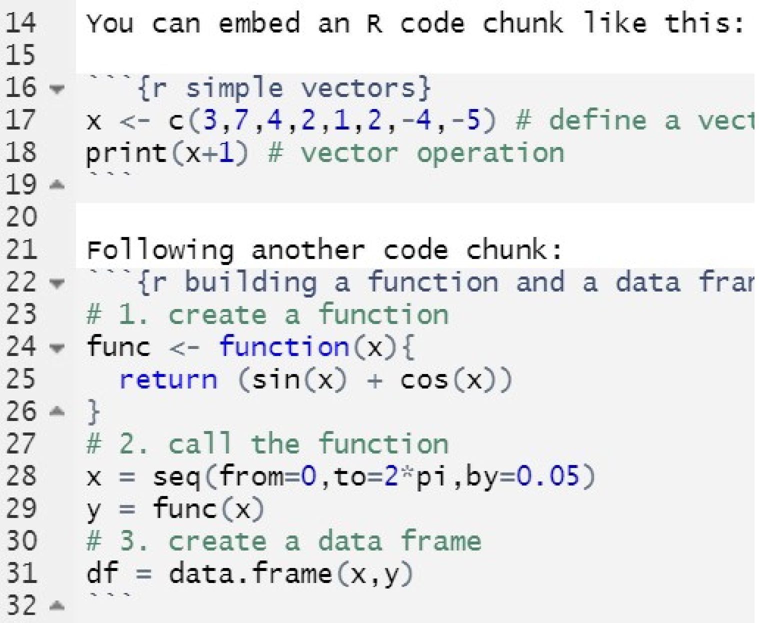
```
---
title: "My First R Markdown Document"
author: "Rihuan Ke"
date: "01/01/2001"
output: html_document
---
```

We can generate section headings:



```
## R Markdown
```

We can also embed code fragments:

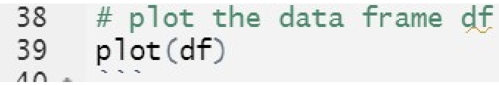


~~~
You can embed an R code chunk like this:

```{r simple vectors}
x <- c(3,7,4,2,1,2,-4,-5) # define a vect
print(x+1) # vector operation
```

Following another code chunk:
```{r building a function and a data fran
# 1. create a function
func <- function(x){
  return (sin(x) + cos(x))
}
# 2. call the function
x = seq(from=0,to=2*pi,by=0.05)
y = func(x)
# 3. create a data frame
df = data.frame(x,y)
```
~~~

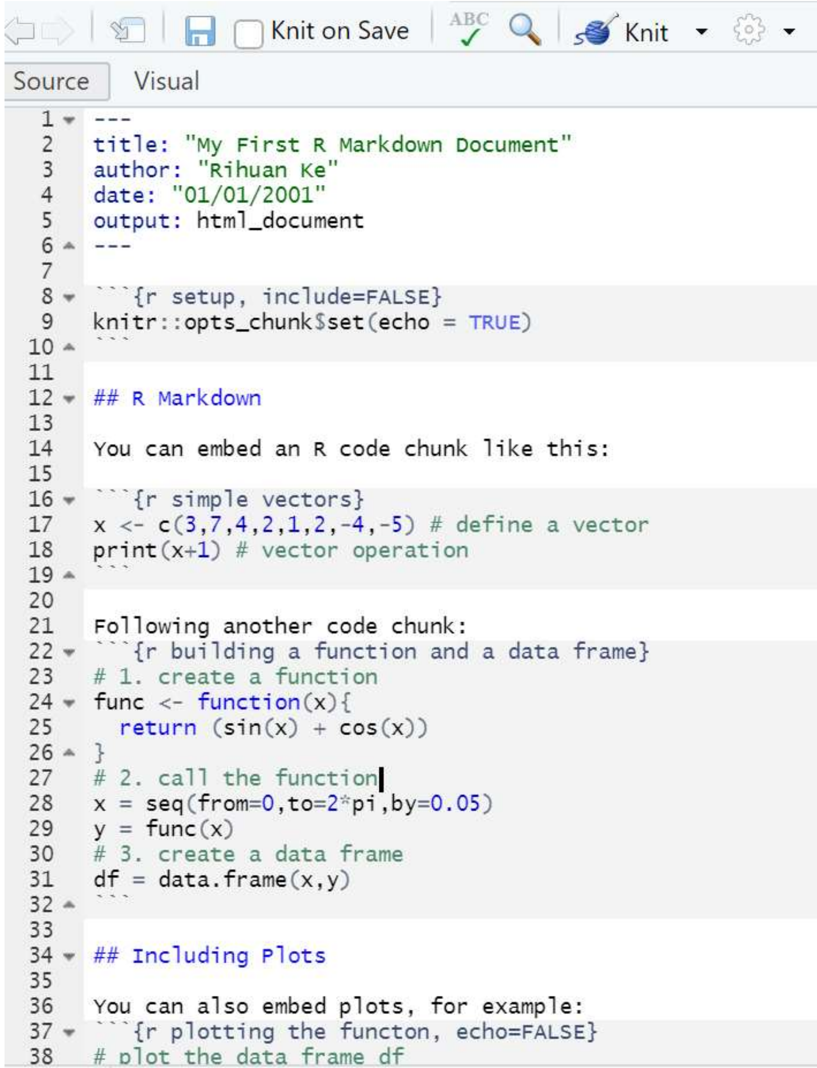
We can also include plots



~~~
# plot the data frame df
plot(df)
```
~~~

Generate html document

code → Knit → output (html format)



~~~
---
title: "My First R Markdown Document"
author: "Rihuan Ke"
date: "01/01/2001"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
~~~

R Markdown

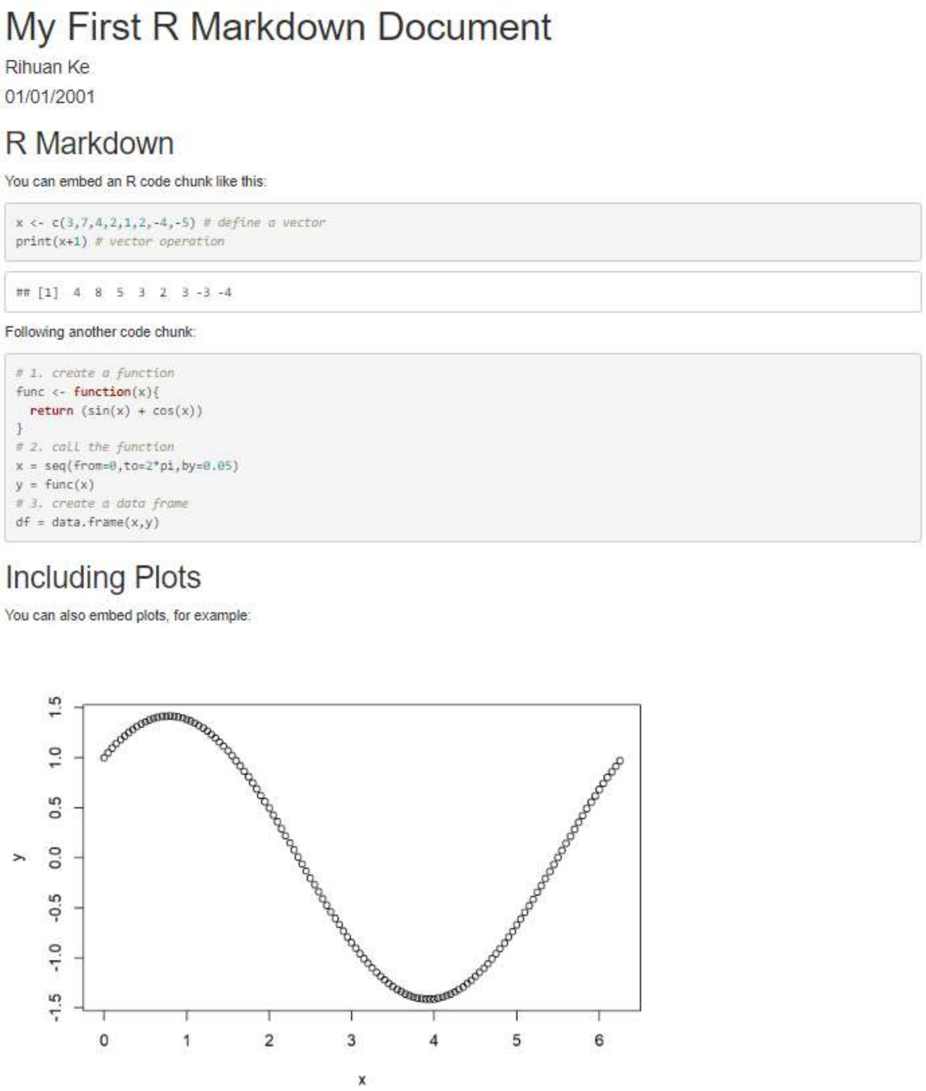
You can embed an R code chunk like this:

```{r simple vectors}
x <- c(3,7,4,2,1,2,-4,-5) # define a vector
print(x+1) # vector operation
```

Following another code chunk:
```{r building a function and a data frame}
# 1. create a function
func <- function(x){
  return (sin(x) + cos(x))
}
# 2. call the function
x = seq(from=0,to=2*pi,by=0.05)
y = func(x)
# 3. create a data frame
df = data.frame(x,y)
```

Including Plots

You can also embed plots, for example:
```{r plotting the functon, echo=FALSE}
# plot the data frame df
```



My First R Markdown Document

Rihuan Ke

01/01/2001

R Markdown

You can embed an R code chunk like this:

```
x <- c(3,7,4,2,1,2,-4,-5) # define a vector
print(x+1) # vector operation
```

```
## [1]  4  8  5  3  2  3 -3 -4
```

Following another code chunk:

```
# 1. create a function
func <- function(x){
  return (sin(x) + cos(x))
}
# 2. call the function
x = seq(from=0,to=2*pi,by=0.05)
y = func(x)
# 3. create a data frame
df = data.frame(x,y)
```

Including Plots

You can also embed plots, for example:

Version control with Git

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

Install git locally:

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

1. Go to https://github.com/ and create a new repository by pressing New

Then add an informative title, a description and include a README.

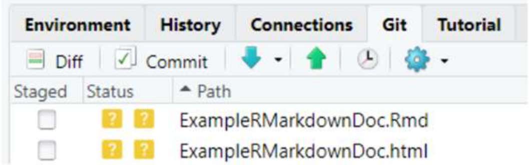
2. Within the github repo click on Code ▾ and copy the repo URL.

3. 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.

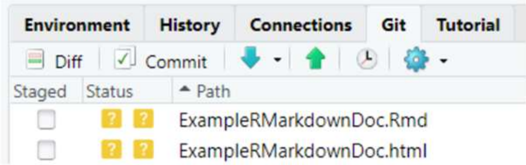
4. We can now add files, commit, push and pull using the Git panel in the top right of RStudio.



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

Set up an R project with Git version control

add files, commit, push and pull



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

☑ **Stage files**: Choose which files to include in the version history

☑ Commit **Commit**: Take a snapshot of staged files within the local git repository

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 we have covered

- We discussed the central role of replicability in data science.
- We discussed the difference between replicability and reproducibility.
- We introduced R Markdown for reproducible data analysis.
- We discussed the integration of Git with RStudio and R projects.

Thanks for listening!

Dr. Rihuan Ke
rihuan.ke@bristol.ac.uk

Statistical Computing and Empirical Methods
Unit EMATM0061, MSc Data Science