EEB 603 – Chapter 1: Using the R Markdown language to promote Reproducible Science

Part C: Advanced R and R Markdown settings

Sven Buerki

2020-09-09

Contents

1 Learning outcomes

This tutorial is devoted to part C of chapter 1 and provides students with opportunities to learn procedures to:

- Set your working directory.
- Set global options for code chunks related to:
  - text,

  - code decoration,
  - caching,
  - plots (output format and resolution),
  - positioning figures (close to associated code chunks).

2 Associated files supporting this tutorial

Although less used in part A, a set of files are provided to support teaching of material presented in this chapter. These files are deposited in the shared Google Drive at this path:

- `Reproducible_Science/Chapters/Chapter_1/Tutorial_files`

Files are as follows:

- `EEB603_Syllabus_BUERKI.Rmd`: This is the `.Rmd` file used to compile the syllabus of this class. This file provides a good source of information for the syntax and protocols described in this tutorial.
- `Bibliography_Reproducible_Science_2.bib`: This file contains references cited in `BibTeX` format.
- `AmJBot.csl`: This citation style language (CSL) file allows formatting citations and bibliography following citation style of *American Journal of Botany*.
- `Bioinformatic workflow_PART2.pdf`: A `pdf` file containing the bioinformatic workflow taught in this class. This file will be used to learn how to incorporate a figure into R Markdown file.

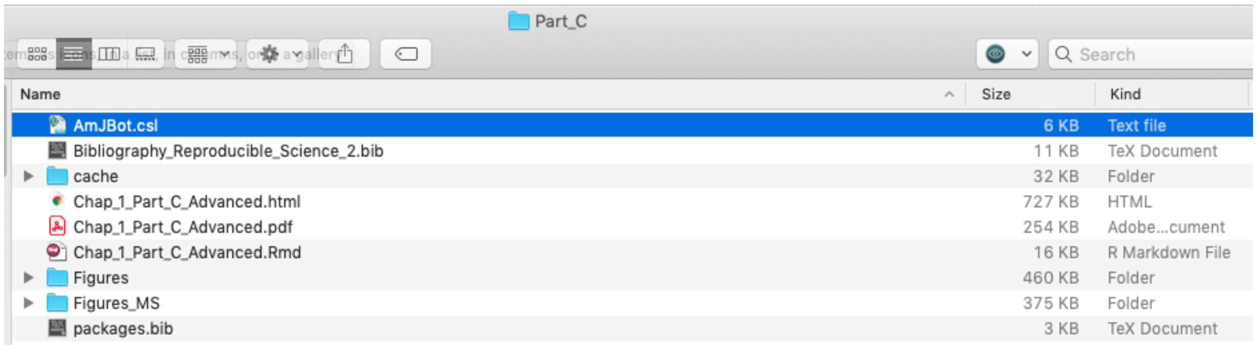
3 Set your working directory

Unlike R scripts where you have to set your working directory or provide the path to your files, the approach implemented in R Markdown document (`.Rmd`) automatically sets your working directory to the location of your `.Rmd` file. This procedure is done by `knitr` functions. `knitr` expects all declared files to be located in the same path as your `.Rmd` file or in a subfolder within this working directory. The main reason for this approach is to maximize portability of your R Markdown project, which is usually composed of a set of files (see Figure 1).

3.1 Instructions to setup your working directory

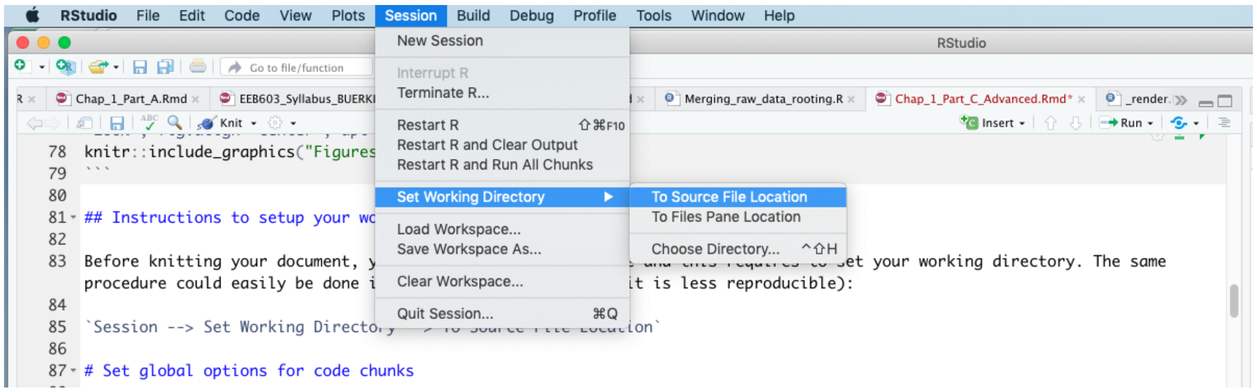
Before knitting your document, you will be testing your code and this requires to set your working directory. The can be done in RStudio by clicking (see Figure 2):

```
Session --> Set Working Directory --> To Source File Location
```



Name	Size	Kind
AmJBot.csl	6 KB	Text file
Bibliography_Reproducible_Science_2.bib	11 KB	TeX Document
cache	32 KB	Folder
Chap_1_Part_C_Advanced.html	727 KB	HTML
Chap_1_Part_C_Advanced.pdf	254 KB	Adobe...cument
Chap_1_Part_C_Advanced.Rmd	16 KB	R Markdown File
Figures	460 KB	Folder
Figures_MS	375 KB	Folder
packages.bib	3 KB	TeX Document

Figure 1: Snapshot of the project structure associated to part C of Chapter 1. External figures used in the document are in Figures, whereas those generated by the document are saved in Figures MS.



```
knitr::include_graphics("Figures
```

Instructions to setup your wo

Before knitting your document, y ... and this requires to set your working directory. The same
procedure could easily be done i ... it is less reproducible):

`Session --> Set Working Directory --> To Source File Location`

Set global options for code chunks
```

Figure 2: Snapshot of RStudio showing procedure to set your working directory to allow testing your code prior to knitting.
```

4 Set global options for code chunks

To improve code reproducibility and efficiency and to follow publication requirements, it is customed to include a "code chunk" at the beginning of your `.Rmd` file to set global options applying to the whole document. Those settings are related to the following elements of your code:

- text results.
- code decoration.
- caching code.
- plots (or figures).
- positioning figures.

These general settings will be set using the `opts_chunk()` function implemented in `knitr` (Xie, 2020). The following website contains valuable information on code chunk options:

- https://yihui.name/knitr/options/

4.1 The opts_chunk$set() function

The `knitr` function `opts_chunk$set()` is used to change the default global options in an `.Rmd` document.

Before starting, a few special notes should be known on the options:

1. Chunk options must be written in one line; no line breaks are allowed inside chunk options.
2. Avoid spaces and periods (`.`) in chunk labels and directory names.
3. All option values must be valid R expressions just like how we write function arguments.

Here we will be discussing each part of the settings individually, but those will have to be merged into one code chunk in your document entitled `setup` (please see below for more details).

4.1.1 Text results

This section deals with settings related to text results generated by code chunks.

Please find below an example of options that could be applied across code chunks:

```r
# Setup options for text results
opts_chunk$set(echo = TRUE, warning = TRUE, message = TRUE, include = TRUE)
```

4.1.1.1 Explanations of the text results options

- `echo = TRUE`: Include all R source codes in the output file.

- `warning = TRUE`: Preserve warnings (produced by `warning()`) in the output like we run R code in a terminal.
- `message = TRUE`: Preserve messages emitted by `message()` (similar to warning)
- `include = TRUE`: Include all chunk outputs in the final output document.

If you want some of the text results to have different options, please adjust those in their specific code chunks. This comment is valid for all the other general settings.

4.1.2 Code decoration

This section deals with settings related to code decoration (i.e. how it is outputted in the final `pdf` document) generated by code chunks.

Please find below an example of options that could be applied across code chunks:

```
# Setup options for code decoration
opts_chunk$set(tidy = TRUE, tidy.opts = list(blank = FALSE, width.cutoff = 60),
    highlight = TRUE)
```

4.1.2.1 Explanations of the code decoration options

- `tidy = TRUE`: Use `formatR::tidy_source()` to reformat the code. Please see `tidy.opts` below.
- `tidy.opts = list(blank = FALSE, width.cutoff = 60)`: This provides a list of options to be passed to the function determined by the `tidy` option. Here we format the code to avoid blank lines and with a width cutoff of 60 characters.
- `highlight = TRUE`: This highlights the source code.

4.1.3 Caching code

To compile your `.Rmd` document faster (especially if you have computing intensive tasks), you can cache the output of your code into files associated to each of your code chunks. This process allows compute intensive chunks to be saved and the output used later without being re-run.

The *knitr* package has options to only evaluate cached chunks when necessary, but this has to be set by users. Such procedure creates a unique MD5 digest (= a data storage technique) of each chunk to track when changes are present. When the option `cache=TRUE` (there are other, more granular settings; see below), the chunk will only be evaluated in the following scenarios:

- There are no cached results (either this is the first time running or the results were moved/deleted).
- The code chunk has been modified.

The following code allows implementing this procedure to your document:

```
# Setup options for code cache
opts_chunk$set(cache = 2, cache.path = "cache/")
```

4.1.3.1 Explanations of the caching code options

- Besides `TRUE` and `FALSE` for the chunk option `cache`, advanced users can also consider more granular cache by using numeric values for `cache = 0, 1, 2, 3`. `0` means `FALSE`, and `3` is equivalent to `TRUE`. For `cache = 1`, the results of the computation are loaded from the cache, so the code is not evaluated again, but everything else is still executed, such the output hooks and saving recorded plots to files. For `cache = 2` (used here), it is very similar to `1`, and the only difference is that the recorded plots will not be re-saved to files when the plot files already exist, which might save some time when the plots are big.
- `cache.path = "cache/"`: Directory where cache files will be saved. You don't have to create the directory before executing the code, it will be created automatically by *knitr*.

4.1.4 Plots

Plots are a major element of your research and they are at the core of your figures. We can take advantage of options implemented in the `knitr` package to output plots meeting publication requirements. This will save precious time during the writing phase of your research (= no need to fiddle with the size and resolution of figures to meet journal policies).

Please find below an example of options that could be applied across code chunks:

```
# Setup options for plots
opts_chunk$set(fig.path = "Figures_MS/", dev = c("pdf", "png"),
    dpi = 300)
```

4.1.4.1 Explanations of the caching plots options

- `fig.path = "Figures_MS/""`: Set directory to save figures generated by the R Markdown document. As above, this folder doesn't need to exist prior to executing the code chunks. Files will be save based on code chunk title and assigned figure number.
- `dev = c('pdf', 'png')`: Save figures in both `pdf` and `png` formats.
- `dpi = 300`: The DPI (dots per inch) for bitmap devices (dpi * inches = pixels). Please look at publishing requirements to set this parameter appropriately.

It is worth noting that you might be using external figures in your `.Rmd` document. To avoid confusions between figures generated by the `.Rmd` document and those coming from outside, it is best practice to have them saved in two different subfolders (see Figure 1 for more details).

4.1.4.2 Additional plots options

Some journals have specific requirements on figure dimensions. You can easily set these by using the following option:

- `fig.dim`: (NULL; numeric) if a numeric vector of length 2, it gives `fig.width` and `fig.height`, e.g., `fig.dim = c(5, 7)`.

4.1.5 Positioning figures

Positioning figures close to their code chunks is critical and can get sorted by adding another `opts_chunk$set()` code line in your `setup` R code chunk. This is done by invoking the `fig.pos` argument and setting it to `"H"`. **Warning:** Setting this argument might generate errors when documents are knitted as `pdf` documents. If it happens, please comment this line using `#` and knit again.

```
## Locate figures as close as possible to requested position
## (=code)
opts_chunk$set(fig.pos = "H")
```

4.2 Applying global changes to all code chunks

In this section, we will collate all global settings discussed above into a code chunk entitled `setup`, which will be placed under the YAML metadata section. In addition to containing the global settings, it is advisable to also include a code section devoted to loading required R packages in the `setup` code chunk.

Please find below the code for the `setup` code chunk based on the options presented above:

```
# Load packages
## Add any packages specific to your code
library("knitr")
library("bookdown")
# Chunk options: see http://yihui.name/knitr/options/ ###
## Text results
opts_chunk$set(echo = TRUE, warning = TRUE, message = TRUE, include = TRUE)
## Code decoration
opts_chunk$set(tidy = TRUE, tidy.opts = list(blank = FALSE, width.cutoff = 60),
    highlight = TRUE)
## Caching code
opts_chunk$set(cache = 2, cache.path = "cache/")
## Plots
opts_chunk$set(fig.path = "Figures_MS/", dev = c("pdf", "png"),
    dpi = 300)
## Locate figures as close as possible to requested position
## (=code)
opts_chunk$set(fig.pos = "H")
```

4.2.1 Settings of the `setup` R code chunk

When inserting the above code into an R code chunk, please set the options of the chunk as follows:

- `setup`: Unique ID of the code chunk.
- `include = FALSE`: Nothing will be written into the output document, but the code will be evaluated and plot files will be generated (if there are any plots in the chunk).
- `cache = FALSE`: Code chunk will not be cached (see above for more details).
- `message = FALSE`: Messages emitted by `message()` will not be preserved.

Options (and their associated arguments) in the code chunk have to be separated by commas.

5 An exercise to work on learning objectives

Please conduct the above exercise to get accustomed with the material presented in the tutorial. This exercise is divided into six steps as follows:

1. Open RStudio, create and save a `.Rmd` document (**Tip:** save the file in a new folder).
2. Set your working directory according to file location.
3. Insert an R code chunk (using the `Insert` button) and entitle it `setup`. This code chunk will be used to define the global settings for the following options as implemented in the `opts_chunk$set()` function:

- Text results.
- Code decoration.
- Caching code.
- Plots.
- Positioning of figures.

4. Use the R code provided below (which generates a plot) to learn more about the effect of global settings on code outputs.
5. Once the file is completed, compile your document using the `Knit` button. Please pay attention to the outputs in your folder.
6. Use the `bookdown \@ref()` function to cite your figure/plot in the text.

5.1 An R code to generate a plot & cross-reference it in text

The R code provided below is associated to step 4 of the exercise and it produces the plot displayed in Figure 3.

To get there: 1. Insert an R code chunk (using the `Insert` button) and set the following options and associated arguments:

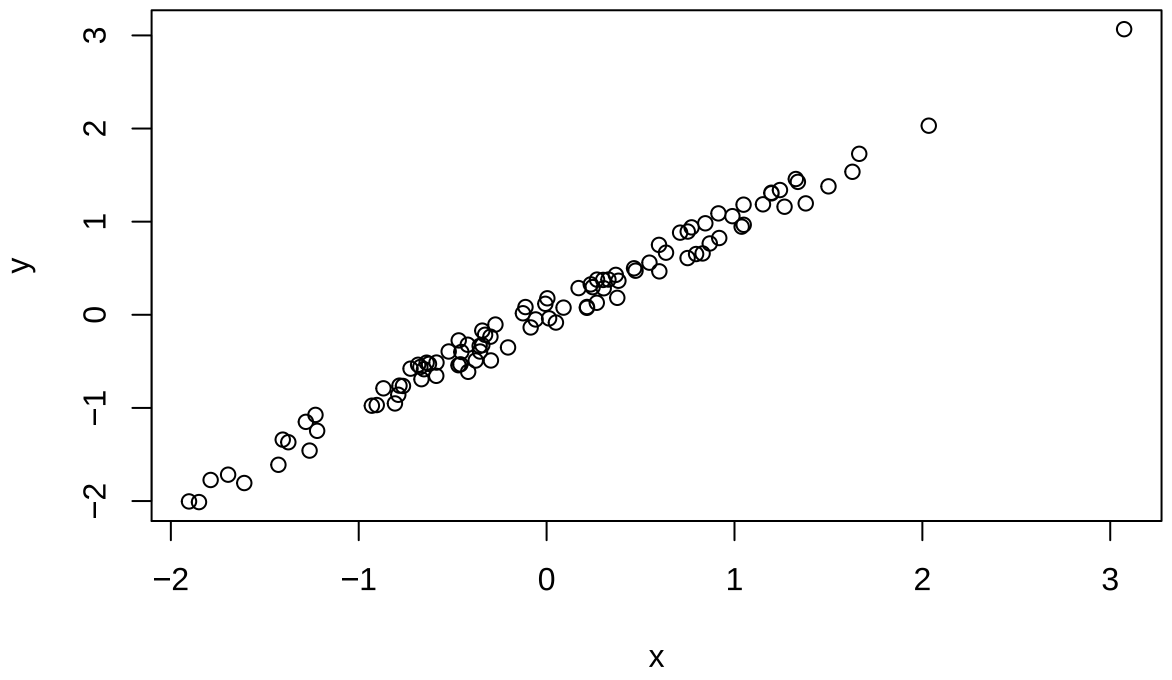


Figure 3: Plot of y ~ x.

- `Plot`: Unique ID of code chunk.
- `fig.cap="Plot of y ~ x."`: Figure caption.
- `fig.show="asis"`: Figure display.
- `out.width = '100%'`: Figure width on the page.

2. Enter the following R code in the code chunk.

```
# Generate a set of observations (n=100) that have a normal
# distribution
x <- rnorm(100)
# Add a small amount of noise to x to generate a new vector
# (y)
y <- jitter(x, 1000)
# Plot y ~ x
plot(x, y)
```

3. Test your code to make sure it does what it is supposed to do (using the `Run` Run button).
4. Go to step 5 above and complete the exercise.

6 References

Xie, Y. 2020. Knitr: A general-purpose package for dynamic report generation in r. Available at: https://CRAN.R-project.org/package=knitr.

A Appendix 1

Citations of all R packages used to generate this report.

[1] J. Allaire, Y. Xie, J. McPherson, et al. *rmarkdown: Dynamic Documents for R*. R package version 2.1. 2020. <URL: https://CRAN.R-project.org/package=rmarkdown>.

[2] C. Boettiger. *knitcitations: Citations for 'Knitr' Markdown Files*. R package version 1.0.10. 2019. <URL: https://CRAN.R-project.org/package=knitcitations>.

[3] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2019. <URL: https://www.R-project.org/>.

[4] H. Wickham and J. Bryan. *usethis: Automate Package and Project Setup*. R package version 1.5.1. 2019. <URL: https://CRAN.R-project.org/package=usethis>.

[5] H. Wickham, J. Hester, and W. Chang. *devtools: Tools to Make Developing R Packages Easier*. R package version 2.2.1. 2019. <URL: https://CRAN.R-project.org/package=devtools>.

[6] Y. Xie. *bookdown: Authoring Books and Technical Documents with R Markdown*. ISBN 978-1138700109. Boca Raton, Florida: Chapman and Hall/CRC, 2016. <URL: https://github.com/rstudio/bookdown>.

[7] Y. Xie. *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.17. 2020. <URL: https://CRAN.R-project.org/package=bookdown>.

[8] Y. Xie. *Dynamic Documents with R and knitr*. 2nd. ISBN 978-1498716963. Boca Raton, Florida: Chapman and Hall/CRC, 2015. <URL: https://yihui.org/knitr/>.

[9] Y. Xie. *formatR: Format R Code Automatically*. R package version 1.7. 2019. <URL: https://CRAN.R-project.org/package=formatR>.

[10] Y. Xie. "knitr: A Comprehensive Tool for Reproducible Research in R". In: *Implementing Reproducible Computational Research*. Ed. by V. Stodden, F. Leisch and R. D. Peng. ISBN 978-1466561595. Chapman and Hall/CRC, 2014. <URL: http://www.crcpress.com/product/isbn/9781466561595>.

[11] Y. Xie. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.29. 2020. <URL: https://CRAN.R-project.org/package=knitr>.

[12] Y. Xie, J. Allaire, and G. Grolemund. *R Markdown: The Definitive Guide*. ISBN 9781138359338. Boca Raton, Florida: Chapman and Hall/CRC, 2018. <URL: https://bookdown.org/yihui/rmarkdown>.

B Appendix 2

Version information about R, the operating system (OS) and attached or R loaded packages. This appendix was generated using `sessionInfo()`.

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.6
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] devtools_2.2.1       usethis_1.5.1        formatR_1.7
## [4] knitcitations_1.0.10 bookdown_0.17        rmarkdown_2.1
## [7] knitr_1.29
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.3        highr_0.8         compiler_3.6.1    plyr_1.8.5
##  [5] remotes_2.1.0     prettyunits_1.1.1 tools_3.6.1       testthat_2.3.1
##  [9] digest_0.6.23     pkgbuild_1.0.6    pkgload_1.0.2     lubridate_1.7.4
## [13] jsonlite_1.6      evaluate_0.14     memoise_1.1.0     rlang_0.4.3
## [17] bibtex_0.4.2.2    cli_2.0.1         magick_2.4.0      yaml_2.2.0
## [21] xfun_0.16         RefManageR_1.2.12 withr_2.1.2       httr_1.4.1
## [25] stringr_1.4.0     xml2_1.2.2        desc_1.2.0        fs_1.3.1
## [29] rprojroot_1.3-2   glue_1.3.1        R6_2.4.1          processx_3.4.1
## [33] fansi_0.4.1       sessioninfo_1.1.1 callr_3.4.1       magrittr_1.5
## [37] codetools_0.2-16  backports_1.1.5   ps_1.3.0          htmltools_0.4.0
## [41] ellipsis_0.3.0    assertthat_0.2.1  stringi_1.4.5     crayon_1.3.4
```