Reporting Using R and Markdown:

Part 3: knitr

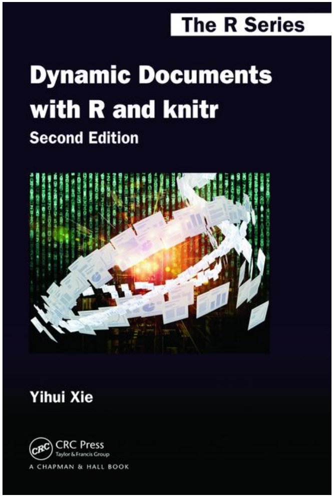
Niels Hagenbuch and Claude Renaux

February 2022

knitr

Preliminaries

Books

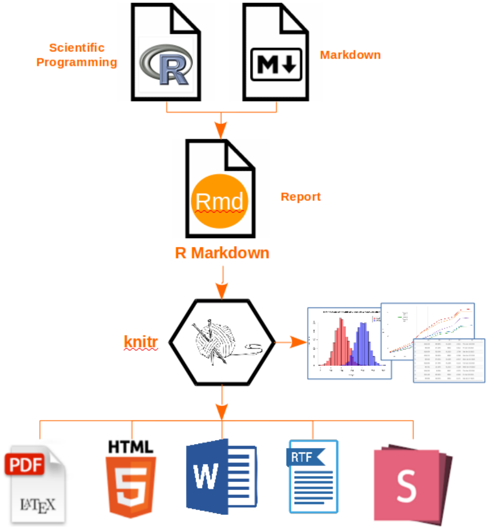


Yihui Xie
Dynamic Documents with R and knitr
2nd Edition
CRC Press, 2015

ISBN 978-1-4987-1696-3

Idea

Idea



Source:
http://applied-r.com/project-reporting-template/

Concept

- An **R Markdown file** (`.Rmd`) contains **Markdown Code** and **R Code**.
- The Markdown part contains **your**
  - Text
  - Tables
  - Pictures
  - Footnotes
- The R part contains the code for
  - Handling the Data
  - Creating Tables
  - Creating Plots
  - Statistical Analysis

Concept

Uniting these two parts is called **“to knit”**:

1. In a **First Pass** through the `.Rmd` file, the R code is executed.
2. The R code is removed and replaced by the result (a number, a table, a plot, a summary).
3. In a **Second Pass**, the R Markdown-formatted text, now containing the results from R, is converted to Word or HTML.
4. As an option, the R code can be included in the document.

To extract only the R code from an `.Rmd` file is called **“to purl”**:

1. Any R Markdown-formatted text is removed.
2. The remaining R code is properly formatted and commented.
3. The ensuing `.R` file contains the raw R part of your report and can be run separately.

Control Sequences

Control Sequences

knitr goes through an `.Rmd` document line by line.

It ignores text and markdown code, but **Recognizes and Executes R Code**, if it is enclosed within two **Control Sequences**.

- An **R Chunk** begins with ` ```{r } ` and ends with ` ``` `
- Each of these control sequences have to be on a separate line

~~~
```{r chunk name, chunk options}
x <- sqrt(2)
```
~~~

Control Sequences

- **In-line R Code** within the R Markdown document is enclosed in `` `r `` and a backtick `` ` ``
- The in-line "mini-chunk" must not contain line-breaks
- If it is too long, remove any space in the R code
- If it is still too long, put it on a separate line

```
The value of x is `r round(x, 4)` and displayed here.

After the transformation, x is
`r exp(asin(tan(cos(sin((x))))))-1/18*x^3`
and no longer recognizable.
```

R Chunks

knitr: Chunk Syntax

knitr offers many options to control the way R chunks are processed and how results are transformed into R Markdown text.

The **General Syntax** is:

~~~
```{r chunk name, chunk options}
# Your
# lines
# of
# R code
```
~~~

knitr: Chunk Names

Chunk Names

- must be **Unique**
- no spaces
- no periods
- hyphen - and underscore _ are allowed
- numbers are allowed

knitr: Chunk Options

All **Chunk Options** are explained here:

`https://yihui.org/knitr/options/`

The most important ones are:

Option	**Description**
`eval`	Evaluate the chunk of R code?
`echo`	Display the *R Code* in the document?
`include`	Display the *Results* of the R code?
`warning`	Show warning messages in the document?
`message`	Show messages in the document?
`error`	Show error messages in the document?

- All these options expect a logical as argument (e.g., `echo = FALSE`).
- The default for the options listed above is `TRUE`.

In the Beginning...

When starting an analysis that is likely to turn into a report, start with an `.Rmd` file including title, author, and date.

Then develop your code within a chunk that is neither evaluated nor shown.

~~~
```{r SANDBOX, eval = FALSE, echo = FALSE}
# Develop your
# R code
# here
```
~~~

Set General Options

The two options `echo` and `include` are usually set to `FALSE` to avoid displaying R code and unnecessary results.

To set knitr options for the entire document, use the function `opts_chunk$Set()`

An `.Rmd` file usually begins with the following R chunks:

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

~~~
```{r R-setup}
##  More settings and "global variables".
rnd <- 2   ##  Number of decimals to round.
```
~~~

Example

R code and output to be displayed must be “turned on manually”:

~~~
The following two linear models were fitted in R:

```{r Regressions, echo = TRUE}
fm1 <- lm(No.of.Births ~ 1 + No.of.Birds, data = Baby)
fm2 <- lm(No.of.Births ~ 1 + No.of.Birds + Year, data = Baby)
```

Model 1 estimated the effects as follows:

```{r RegrModel1_Summary, include = TRUE}
summary(fm1)
```
~~~

R Tables

Tables

Creating tables is a bit of a challenge.

The two most comprehensive packages are sjPlot and stargazer:
`https://dmyee.files.wordpress.com/2016/03/table_workshop.pdf`

The following smaller functions are briefly mentioned in this course:

Package	**Function**	**Description**
knitr	`kable`	Simple tables in Word, HTML (and LaTeX)
kableExtra	`kable_styling`	Enhence `kable` with different styles in HTML (and LaTeX)
flextable	`regulartable`	Complex tables specific in Word
tableone	`CreateTableOne`	"Table 1" for biomedical papers
table1	`table1`	"Table 1" for biomedical papers, HTML only
broom	`tidy`	Extract estimates, std. errors, and p-values from statistical models

Very Simple Tables with kable

knitr includes the function `kable()`, which produces simple tables.

~~~
```{r Kable, include = TRUE}
iris$noise <- rnorm(n = nrow(iris))
kable(iris[1:10, -5])
```
~~~

Very Simple Tables with kable

Useful arguments:

- `digits` controls the number of decimals
- `align` controls the alignment of the columns (string of `l`, `c`, and `r`)
- `caption` prints a caption above the table

~~~
```{r Kable2, include = TRUE}
kable(iris[1:10, -5], digits = 4,
      align = c("llccr"),
      caption = "Tab. 1: First 10 observations
                 in the famous iris data set.")
```
~~~

Tidy Up Your Output

The package broom offers two helpful generic functions:

- `tidy()` converts summary output of any conceivable statistical model into a data frame
- `glance()` gives a summary of the model, *e.g.* overall p-value, R^2, AIC, BIC, etc.

https://cran.r-project.org/web/packages/broom/vignettes/broom.html

~~~
```{r Regression, echo = TRUE}
fm <- lm(mpg ~ 1 + wt, data = mtcars)
```
~~~

Tidy Up Your Output

~~~
```{r RegressionOutputRaw, include = TRUE}
summary(fm)
```
~~~

~~~
```{r RegressionOutputTidy, include = TRUE}
library(broom)
tidy(fm)
```
~~~

~~~
```{r RegressionModelSummary, include = TRUE}
glance(fm)
```
~~~

Tidy Up Your Output

In general, it is easier to access the estimated coefficients, their standard errors, and *p*-values from a tidy-object.

However, the names of the variables are no longer rownames, but entries in the column `term`.

```
##  In R:
td <- tidy(fm)

##  Correct:
td[td$term == "wt", "estimate"]
```

```
##  Wrong:
td["wt", "estimate"]
```

R Plots

Plots

To include plots in the document, the chunk options need to be set to `include = TRUE`.

The plot is directly included in the HTML/Word document.

~~~
```{r Sine, include = TRUE}
plot(sin, -pi, 2 * pi)
title(main = substitute(paste("Sine from ", -pi, " to 2", pi)))
```
~~~

Plots: File

A better way is use the options `fig.path` (where to store the file) and `dev` (file format; see `https://yihui.org/knitr/options/#plots`) in the initial knitr-setup chunk.

~~~
```{r knitr-setup2, include = FALSE}
opts_chunk$set(echo = FALSE, include = FALSE,
               fig.path = "Figures/", dev = "jpeg")
```
~~~

The **Chunk Name** will be used as **File Name**, and the plot is in addition available in the directory indicated by `fig.path`.

~~~
```{r Sine2, include = TRUE}
plot(sin, -pi, 2 * pi)
title(main = substitute(paste("Sine from ", -pi, " to 2", pi)))
```
~~~

Plots: Size

The size (in inches) is controlled by the two options `fig.height` and `fig.width`.

~~~
```{r SineSmall, include = TRUE, fig.height = 3, fig.width = 3}
plot(sin, -pi, 2 * pi)
```
~~~

Plots: Caption

A caption can be included using the option `fig.cap`.

~~~
```{r Cosine, include = TRUE, fig.cap = "Fig. 2: A Cosine Curve."}
plot(cos, -pi, 2 * pi)
```
~~~

Caching

Caching

The option `cache` allows knitr to store the results of the chunk.

The next time the file is knitted, the results are obtained from the cache instead of running the R code again.

`cache = TRUE`:

- If the chunk is run for the first time, the results are stored.
- If stored results exists, they will be used. No code is run.
- If the R code is modified, it will be re-run and the results are stored (Note: ANY change in the code is considered a modification, even the removal of a space or a blank line).

`cache = FALSE`:

- R code is run, irrespective of existing cached results.
- No results are cached.

Caching

Cached results can save a lot of time!

~~~
```{r MultipleImputation, cache = TRUE}
imp <- mice(data.na, pred = pred.mat, meth = mthds,
            seed = 123, maxit = 50, m = 500)
fm <- with(imp, lmer(Y ~ 1 + X1 * X2 * X3 + (1 | ID)))
```
~~~

Caching

Cached results can cause nasty errors!

A change in `Chunk-1` will not be propagated into `Chunk-2` and `Chunk-3`:

~~~
```{r Chunk-1, cache = TRUE}
res <- foo(x)
```


```{r Chunk-2, cache = TRUE}
plot(res)
```


```{r Chunk-3, cache = TRUE}
summary(res)
```
~~~

Caching

- Use a global flag and set it when required.
- Delete the content of the cache directory regularly.

~~~
```{r knitr-setup, include = FALSE}
library(knitr)

opts_chunk$set(echo = FALSE, include = FALSE)

##  Global option for chunk caching.
cache = FALSE
# cache = TRUE

##  Global option for figure caching.
fig.cache = FALSE
# fig.cache = TRUE
```
~~~

Caching

Use a global flag.

~~~
```{r Chunk-1, cache = cache}
res <- foo(x)
```


```{r Chunk-2, cache = fig.cache}
plot(res)
```


```{r Chunk-3, cache = cache}
summary(res)
```
~~~

References

Xie, Yihui, J. J. Allaire, and Garrett Grolemund (2018). *R Markdown. The Definitive Guide.* CRC Press. isbn: 978-1-138-35933-8. url: `https://bookdown.org/yihui/rmarkdown/`.

Xie, Yihui (2015). *Dynamic Documents with R and knitr* CRC Press. isbn: 978-1-4987-1696-3.

Advanced Use of Options

Advanced Options: Evaluation and Display of Lines

Instead of `eval = TRUE`, single lines of code can be addressed.

- Specific **Inclusion**: `eval = c(1, 3, 18:29)`
- Specific **Exclusion**: `eval = -c(4:7, 19)`

The option `echo` can be used in a similar fashion.

- Specific **Inclusion**: `echo = c(1:10, 31)`
- Specific **Exclusion**: `echo = -(10:23)`

Advanced Options: Tables for Word with Flextable

The package flextable has several functions to create, modify, and tailor tables directly in Word format.

`https://mran.microsoft.com/package/flextable`

Start with the function `regulartable()`.

~~~
```{r FlexTable, include = TRUE}
library(flextable)

##  Create table, and indicate column names to be included
##  with the argument col_keys (nothing is printed yet).
rt <- regulartable(head(mtcars), col_keys =
                   c("hp", "wt", "gear", "mpg"))
```
~~~

Advanced Options: Tables for Word with Flextable

Then adjust formating, merge cells, change size, etc.

~~~
```{r FlexTable2, include = TRUE}
##  Give columns proper names in the table.
rt <- set_header_labels(rt, hp = "Horsepower",
                        wt = "Weight",
                        gear = "No. of Gears",
                        mpg = "Miles/Gallon")

##  Make header bold and outcome italic.
rt <- bold(rt, part = "header")
rt <- italic(rt, j = 4)   ##  Make column 4 italic.

##  Adjust widths and heights of cells.
rt <- autofit(rt)
rt
```
~~~

Advanced Options: “Table 1” with tableone

The package tableone constructs the “Table 1,” summarizing patient baseline characteristics in biomedical research papers.

- Continuous and categorical variables can be shown in one table.
- Continuous variables are summarized with means and standard deviations or medians and interquartile ranges.
- Categorical variables are shown as counts and/or percentages.
- The output can then be exported to Excel for editing, and then to Word.

`https://cran.r-project.org/web/packages/tableone/vignettes/introduction.html`

Advanced Options: “Table 1” with tableone

~~~
```{r TableOne1, include = TRUE}
library(tableone)

CreateTableOne(data = iris, strata = "Species",
               test = FALSE)
```
~~~

~~~
```{r TableOne2, include = TRUE}
tab <- CreateTableOne(data = iris, strata = "Species",
                      test = FALSE)
print(tab, nonnormal = c("Petal.Length", "Petal.Width"))
```
~~~

Advanced Options: "Table 1" with table1

table1 produces HTML output only.

https://cran.r-project.org/web/packages/table1/vignettes/table1-examples.html

~~~
```{r Table1, include = TRUE}
library(table1)
table1(~ . | Species, data = iris)
```
~~~