Reproducilble Research

Malay (malay@uab.edu)

November 10, 2016

Contents

What is reproducible research

In science replicability is of prime importance. Reproducible research is a concept where data and code are packaged together into one live document. It provides the facility to reproduce the data analysis step exactly on another computer.

Text processing vs word processing

In word processing (like in Microsoft Word), the environment is WYSIWYG (What You See Is What You Get). You exactly see the formatting of the document live. In text processing, you think about the overall structure of the document and `markup` the document is special syntax then use a software to convert the document in more presentation format, like PDF. The prime software for text processing in scientific computing is LaTeX.

LaTeX

LaTeX has been a mainstay of scientific publishing. A very simple document is LaTeX is:

```
\documentclass{article}
\title{My first document}
\author{Malay}
\begin{document}
\maketitle
\section{This is the first section}
Hello World!
\end{document}
```

Exercise

Type the above code block in a text file. Name the file as `hello.tex`. You can compile the document in LaTeX using

```
xelatex hello.tex
```

Pandoc

LaTeX is a type-setting software and no one should use it directly. `Pandoc` (http://johnmacfarlane.net/pandoc/) is a highly useful text conversion software that can convert any form of text to another. Additionally, it supports a variant of a very popular and simple text markup language called `markdown`. When you convert a `markdown` formatted file to PDF using `pandoc`, it uses LaTeX behind the scene. This makes our job is crating LaTeX a lot easier.

Let's create our first `pandoc` file.

```
---
title: My first document
author: Malay
---
# This is the first section
Hello World!
```

Type the above code in a file called `hello1.md`. Then type the following:

```
pandoc --number-sections -o hello1.pdf hello1.md
```

Interesting thing is that the same source can be now used to create an HTML document:

```
pandoc -s --number-sections -o hello1.html hello1.md
```

Also a Microsof Word file:

```
pandoc -o hello1.docx hello1.md
```

Literate programming

1. Originally created by Don Knuth.
2. Document that describes the data analysis and the code are in the same text file.
3. You **weave** a document to create the human readable text.
4. You **tangle** a document ot create the machine readable code.

knitr

knitr is a package in R to mix markdown formatted document with R code called, Rmarkdown(http://rmarkdown.rstudio.com/). The R code is live and executed on the fly to generate the final document with the result of the R already embedded in. The overall steps are:

```
.Rmd -> (use knitr)-> .md -> (use pandoc)-> HTML or PDF
```

Within RStudio all these steps are put together under one button click.

Exercise

Type the following code into RStudio:

````
---
title: My first knitr document
author: Me
---

This is some text (text chunk).

Here is some code (code chunk).

```{r}
set.seed(1)
x<-rnorm(100)
mean(x)
```
````

Chunk options

For a list of chunk options see http://yihui.name/knitr/options#chunk_options. There are several options you can use to control what should be included in the final document. For example, echo=FALSE with not include the code int he final document.

````
```{r echo=FALSE}
set.seed(1)
x<-rnorm(100)
mean(x)
```
````

results="hide" will hide the output but the code will still be shown.

````
```{r results="hide"}
set.seed(1)
x<-rnorm(100)
mean(x)
```
````

include=FALSE will suppress both the output and the code, but the code will will be executed.

````
```{r include=FALSE}
set.seed(1)
x<-rnorm(100)
mean(x)
```
````

```{r}
head(x)
```

To prevent the execution of the code, use `eval=FALSE`.

Global chunk option

You can set a chunk option globally for every chunk.

```{r include=FLASE}
knitr::opts_chunk$set(message=FALSE)
```

One important option for chunk is caching R calculations.

```{r include=FLASE}
knitr::opts_chunk$set(cache=TRUE)
```

In-line R

You can include a piece of data from the calculation inside a text chunk.

`We have `r length(x)` rows in our data.`

Nice tables

You can use R package `xtable` for creating nice looking table.

```{r fitmodel}
library(datasets)
data(airquality)
fit <- lm(Ozone ~ Wind, data=airquality)
```

We can now print a nice looking table of regression coefficients.

```{r results="asis"}
library(xtable)
options(xtable.comment=FALSE)
xt <- xtable( summary(fit) )
print (xt, type="pandoc")
```

You can also use the `knitr` function `kable` to create a table.

```{r}
knitr::kable(summary(airquality))
```
