**Coursera Data Science Course 5 Week 2 – Markdown & knitr**

**Coding Standards in R that are Important**

1. Always use a text file/ text editor  
2. Indent your code – just to show flow of the program – Preferences, Code editing, tab width  
four spaces, maybe even 8 spaces  
3. Limit the width of your code (around 80 columns?)  
4. Limit the length of your function – if it says read the data, make it actually read the data  
 Splitting code into logical functions makes it easier to fix

**Markdown**

Simplified markup language -> text-to-HTML conversion tool  
Very easy to be integrated with R code

Syntax:

\*text\* = italics  
\*\*text\*\* = bold  
# primary heading  
## secondary heading  
### tertiary heading

Unordered list uses -  
Ordered list uses 1 2 3 – markdown automatically orders list

Links  
[text](URL)

Advanced linking  
I spent … [text1][1] and [text2][2]  
[1]: URL “text1”  
[2]: URL “text2”

Newlines – can be created by using a double space after the end of a line

**R Markdown**  
-combination of R code and markdown  
-looks a lot like a markdown document but going to have R code  
-R code evaluated as part of processing of markdown  
-**literate statistical programming**

-use knitr package in R to convert R markdown to standard markdown  
-Markdown can be converted to HTML using markdown package  
-any basic text editor can create markdown  
-R markdown -> markdown -> HTML workflow – use R studio

R markdown demo  
uses .Rmd  
plot(cars)

We can plot and make regression models in R markdown as well.

‘’’{r}  
blah blah blah  
‘’’

Unordered list:

Use \* blah and make lists!

**Knitr**

Can have a document integrating data and code…

**“Literate Statistical Programming”**

-stream of text and code  
-analysis code in text and code “chunks”  
-need documentation language and programming language

**Reproducible..**  
-keep track of things  
-software whose operation can be coded  
-don’t save output (keep raw data)  
-save data in non-proprietary formats

Pros  
-text and code are in logical order  
-data, results automatically updated to reflect external changes  
-code is live- automatic “regression test” when building a document

Cons  
-text and code all in one place will be very lengthy  
-can also slow down processing of documents

**Knitr**-R package (built in RStudio)  
-supports RMarkdown, LaTeX, and HTML  
-can export to PDF and HTML

-need R, text editor, some support packages, and some documentation languages

We will use Markdown.

**Knitr is good for…**  
-manuals  
-short/medium-length technical documents  
-tutorials  
-reports  
-data preprocessing documents/summaries

Not good for..  
-very long research articles  
-time-consuming computations  
-documents that require precise formatting

**First knitr documents**

Some text

‘’’{r}  
***//insert code here***‘’’

Then click knit HTML and it will automatically generate HTML for you

Will show the text, code and the output in a markdown document.

-can also name your code chunk ‘’’{r –chunkname-}

Recall .Rmd -> .md -> .html , but only edit Rmd!

‘’’{r simulation, echo=FALSE} // will not print code in the actual document

To call a variable in a print statement, just do ‘r -variablename-’

Knitr produces an image tag in HTML. It embeds the entire image in HTML.

Make tables with xtable  
**library(xtable)**  
**xt <- xtable(summary(fit))** //fit is a linear regression model  
**print(xt, type = “html”)**

Set “global” options i.e. Never echo code chunks:  
**‘’’{r setoptions, echo=FALSE}  
opts\_chunk$set(echo = FALSE, results = “hide”)  
‘’’**

You can always override the default for a specific function by setting echo = TRUE

Common options:  
results: “asis”, “hide”  
echo: TRUE or FALSE

Figures  
fig.height  
fig.width

Cache = TRUE can be set on chunk by chunk basis – to not compute it over and over

Dependencies are not checked explicitly as well

Chunks with significant side effects may not be cacheable