ESM 505/ESR 605 : Data Management, Spring 2014

Week 1: Introduction

1 Course Description and Objectives

This course is designed to explore the concept of reproducible research for environmental sciences. We will cover proper data storage and the use of the R statistical computing environment to manipulate that data for statistical and graphical analysis. A background in a computer language and statistics is a plus but not required.

1.1 What is "Reproducible Research"?

In a very general sense, reproducible research is one of the fundamental aims of science. When we write a paper we include methods and results sections so that, in theory, other researchers can potentially reproduce and confirm our results.

Lately, the term "Reproducible Research" has been used to really mean "Reproducible Data Analysis". Every step of data analysis (often in the form of code) is prodvided.. * Import, manipulate and transform raw data and meta-data into processed data.
Run statistical analysis. * Create plots and tables. * Incorporate analysis and plots into a report.

1.2 Benefits of "Reproducible Research".

Many journals currently have data sharing and code sharing policies in place and that number has been increasing in recent years (Stodden et al 2013).

Even if you aren't planning on publishing in a journal with such policies, there is much to be said about being able to reproduce your own data analysis in as little effort as possible.

1.2.1 Example of non-reproducible data analysis

1. Enter raw data into Excel
2. Hand edit data for QA/QC
3. Copy and paste data subsets of data into new worksheet as "processed data"
4. Do analysis in Excel or export/import data into other analysis software (R, SAS, Sigmaplot, etc..)
5. Point-n-click your way through your analysis

6. Save formatted plots and table output
7. Insert formatted plots and tables into Word Document

What happens if you obtain new data or find an error in your raw data?

You start over from step 1 or 2 and spend potentially just as much time as you initially did running your analysis.

1.2.2 Example of reproducible data analysis

1. Enter raw data into database, Excel, or CSV files
2. Use R to assist QA/QC
3. Use R to process your data for further analysis
4. Use R to preform statistical and graphical analysis
5. Use R markdown script to incorporate plots and tables into final report
6. Use Pandoc command to convert R markdown into a final report in html, word doc, PDF, etc..
7. Create a master script that combines all of these steps into one.

What happens now if you get more data or need to change something?

You edit your data and scripts as needed and rerun the master script. Yes, you spent a bit more time setting things up initially but now rerunning analyses is trivial.

1.2 CRAN Task View on Reproducible Research

For an overview of R packages for Reproducible Research see the following..

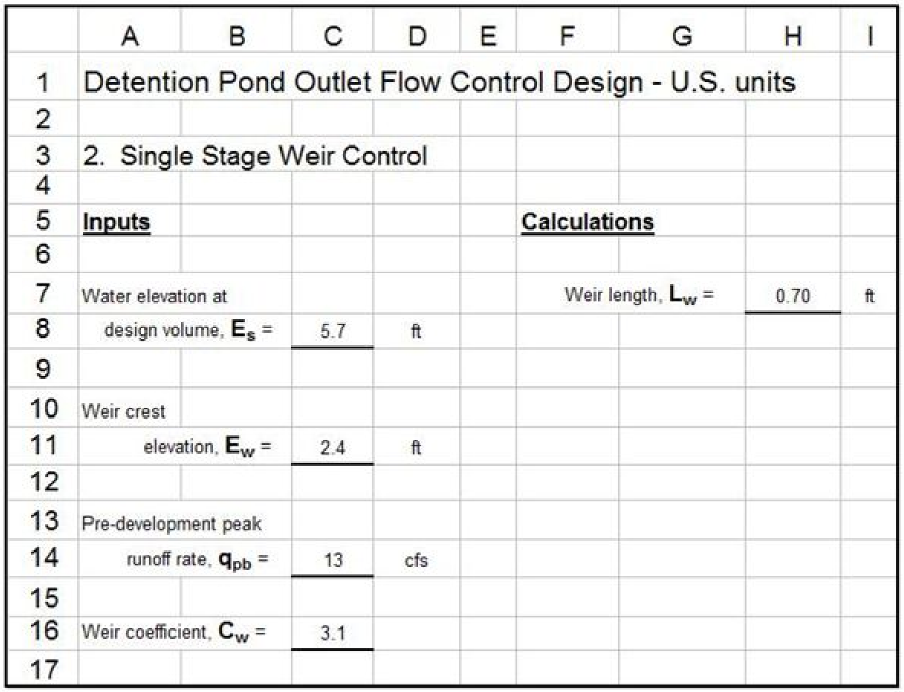
[CRAN Task View: Reproducible Research] (http://cran.r-project.org/web/views/ReproducibleResearch.html)

2 Data Storage

2.1 Spreadsheets

Microsoft Excel and other spreadsheets (such as Google Speadsheets or Open Office) are probably the most common method for data entry and data sharing. However, many people use spreadsheets in a way that makes the data practially unusable to others software.

In this example the spreadsheet is formatted like a data entry sheet and/or reporting table which is fine, unless you need to share the raw data with other software.



	A	B	C	D	E	F	G	H	I
1	Detention Pond Outlet Flow Control Design - U.S. units								
2									
3	2. Single Stage Weir Control								
4									
5	**Inputs**					**Calculations**			
6									
7	Water elevation at						Weir length, $\mathbf{L_w}$ =	0.70	ft
8		design volume, $\mathbf{E_s}$ =	5.7	ft					
9									
10	Weir crest								
11		elevation, $\mathbf{E_w}$ =	2.4	ft					
12									
13	Pre-development peak								
14		runoff rate, $\mathbf{q_{pb}}$ =	13	cfs					
15									
16	Weir coefficient, $\mathbf{C_w}$ =		3.1						
17									

Figure 1: Example of a bad spreadsheet design

A proper data spreadsheet should look more like a database tabale with unique data as rows and variables as columns with the column headers listing those variables.

One other complication to using spreadsheets in this fashion is that there can be incompatibilites based on which version of the spreadsheet software is being used.

2.2 Databases (Microsoft Access, MySQL, PostgreSQL, etc..)

Sometimes data is too big and too complicated to store in a simple spreadsheet. When this happens databases are needed.

2.3 CSV (comma seperated values)

Almost every spreadsheet and database can export properly formatted data into CSV files. Likewise, just about every program used for data analysis can import CSV files. Because CSV files are simple text files, they are the *lingua franca* of data formats. However, large amounts of data in text format can take awhile to load.

The following code shows how long it takes to load a 44.6 Mb file with 332,633 rows of data and 13 variables.

```
csv_speed <- system.time(data <- read.csv("~/WSA/Data/WSA_data.csv"))
print(csv_speed)
```

```
##    user  system elapsed
##   5.150   0.000   5.123
```

Here I've wrapped the 'system.time' function call around the 'read.csv' function that loads the data file so that I can time it. This can be useful if you ever want to know how long a particular part of your analysis takes.

2.4 Binary Files (e.g. '.Rdata')

One option to speed this process up in R is to save the data in a binary Rdata file. This file format compresses the data in such a way that it can be reopened at least 10 times faster then .

```
save(data, file = "WSA_data.Rdata")
binary_speed <- system.time(load(file = "WSA_data.Rdata"))
print(binary_speed)
```

```
##    user  system elapsed
##   0.090   0.000   0.097
```

The loading of the binary data file was over **53** times faster than loading the csv file!

2.5 R Objects

R Object Types

Dimensions	Homogeneous	Heterogeneous
1d	Atomic Vector	List
2d	Matrix	Data Frame
3d	array	

R Object Structures

Vectors The most basic variable structure is a vector (1d, homogeneous). Generally created using "c()" to concatenate values of the same type.

Matrices

Arrays

Data Frames

Lists

Factors

Numeric

Iteger

Character

Date

Data Format (Wide vs Narrow)

Most people are familiar with the “wide” data format where each row represents an observation and each column represents all the various attributes for that observation.

Site	Date	species1	species2	species3	species4
A	2009-03-10	1	0	0	4
B	2009-03-10	2	7	0	0
C	2009-03-10	0	2	2	0
A	2009-04-13	0	1	1	5
B	2009-04-13	0	2	1	2
C	2009-04-13	1	5	2	0

id	Site	Date	Speices	Count
1	A	2009-03-10	species1	1
2	B	2009-03-10	species1	2
6	C	2009-04-13	species1	1
8	B	2009-03-10	species2	7
9	C	2009-03-10	species2	2
10	A	2009-04-13	species2	1
11	B	2009-04-13	species2	2
12	C	2009-04-13	species2	5
15	C	2009-03-10	species3	2
16	A	2009-04-13	species3	1
17	B	2009-04-13	species3	1
18	C	2009-04-13	species3	2
19	A	2009-03-10	species4	4

22	A	2009-04-13	species4	5
23	B	2009-04-13	species4	2

Meta-data

Citations

[1] Stodden V, Guo P, Ma Z (2013) Toward Reproducible Computational Research: An Empirical Analysis of Data and Code Policy Adoption by Journals. PLoS ONE 8(6): e67111. doi:10.1371/journal.pone.0067111 [link] (http://www.plosone.org/article/info%3Adoi%2F10.1371%2Fjournal.pone.0067111)