Demonstrating Literate R

Jim Tyson

4 April 2016

Contents

Literate Programming and Reproducible Research

Reproducibility is the minimum standard for research. Broman says

> Computational work is reproducible if one can take the data and code and produce the same set of results. Replicable is more stringent: can someone repeat the experiment and get the same results? (see Broman, n.d.)

And Peng in his presentation *Make the World a Better Place through Reproducible Research* speaks of *Reproducible Research: A Minimum Standard* with the following characteristics

- Published research where the following are made available:
- Analytic data
- Computer code implementing methods
- Documentation about code/data All are distributed using standard means

(see Peng 2006)

In this context, *literate programming* (see Knuth 1984) is seen as part of a workflow for *reproducible research* by allowing a report on the results of research to be disseminated along with the computational code that produced the analysis.

Some Principles

Use packrat and always create projects

Always create a project - either manually or using RStudio. Keep all the files required for your product within the project folder structure. Use packrat to ensure that your project is truly portable and won't fail if moved to an environment where some packages have not been installed.

Treat data as read only

Read the raw data and process it by script each time. You can programmatically create a new datafile if you wish or directly create and work with the processed data as a dataframe.

Script everything

All R processes should be run from scripts. This means that you always know exactly what has been done to the data. You can easily correct errors. Your audience can see exactly what you did. You can repeat your data preparation and analysis easily.

Be modular

There is no absolute rule about modularity, but as an example, I have separated the data preparation the production of graphs into separate scripts. This means that I have quite small code files to deal with and I can run only the code that needs to be run easily. This becomes much more useful when you *automate your workflow*.

Automate the workflow

Create a *make* file to specify how your product - usually a report - is to be produced. The unix **make** utility specifies *target files* to be produced, their *dependices* and the *recipe* to create them. When you execute the *Makefile* the utility checks timestamps and only runs the files that have changed since the target was last produced.

Use version control

If you have **git** installed, it is very easy to place your project under version control. There is an excellent **git** tutorial at http://gitimmersion.com/.

Make all products available to your audience

Reproducibility is only achieved if you make the code and data accessible to your audience. Github provides an excellent way to do this. Create a repository at

Make, Rstudio and Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents.

This document combines text R code that may create plots and tables, analyse data, and insert the results in to nicely formatted PDF documents. This approach is often called *literate programming*.

While in this example we are using *markdown* as the document production language, but you can equally use LaTeX. We are outputing a PDF file but you can equally produce HTML and Microsoft Word documents. To use markdown with RStudio I use the package *knitr*. To use LaTeX with RStudio, I would use the package *sweave*.

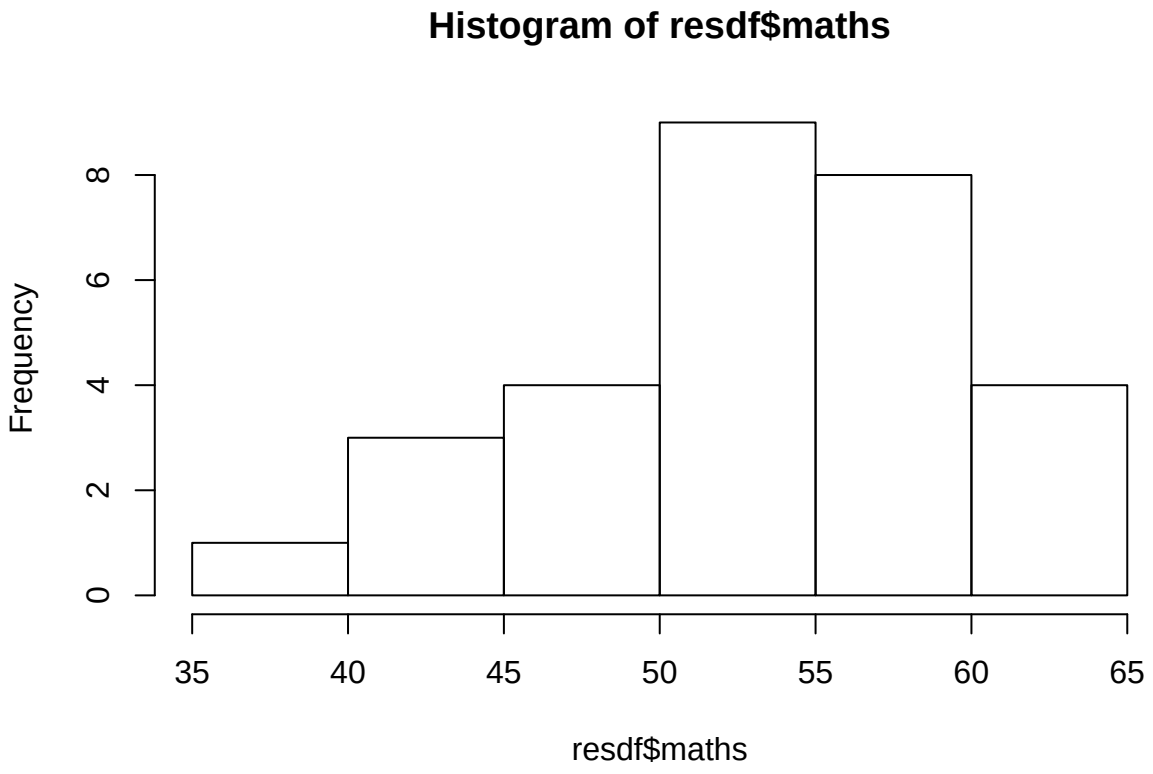
Rstudio is used to write the markdown file containing the R code and to produce the R scripts that are used in the data prepration and alaysis.

Finally, the UNIX make utility is used to automate the process of running any scripts that are required to produce the report and to manage the dependencies between files, ensuring that only scripts that need to be run are run each time.

The files in this demonstration

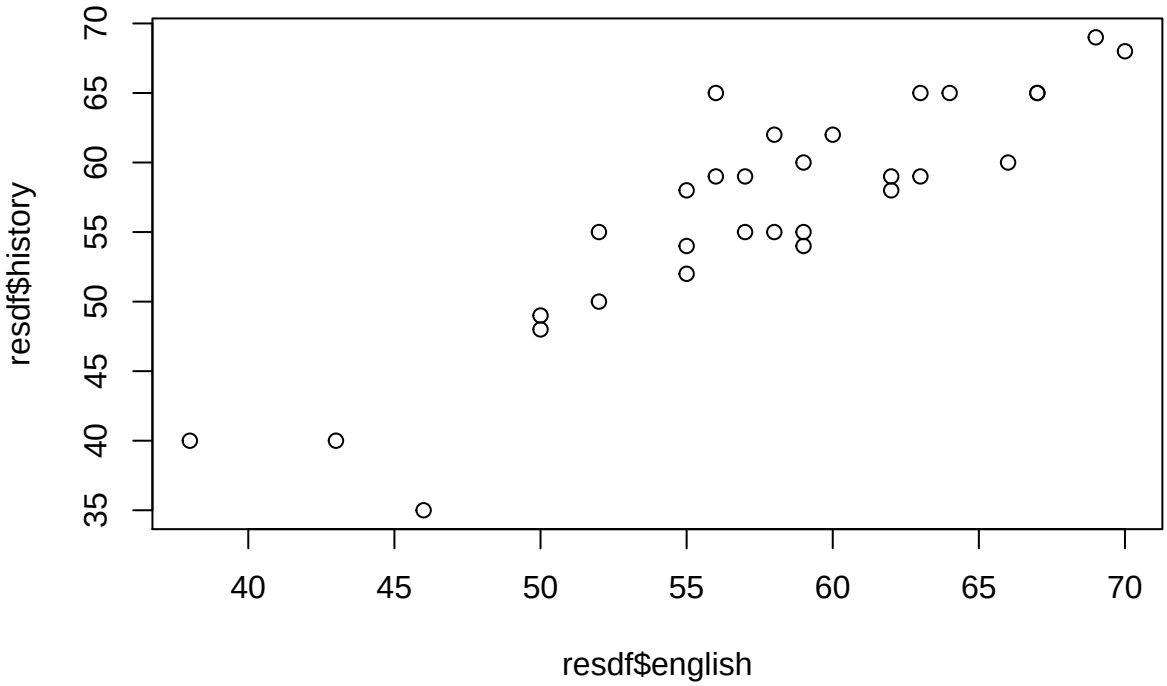
The main product of this demonstraion is the PDF. That file is produced from the script *newtesting.Rmd*; that file contains graphs produced by *graphs.R*; *newtesting.PDF* and *graphs.R* depend on a clean data file called *cleandata.csv*; *cleandata.csv* is produced from the raw data which is read from a website and prepped by a script *readandclean.R*.

A histogram



Scatter Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Custome table

Left align	Right align	Center align
This	This	This
column	column	column
will	will	will
be	be	be
left	right	center
aligned	aligned	aligned

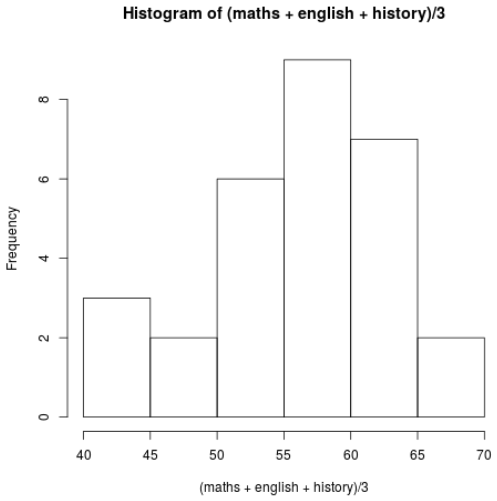
Regression

In this simple linear regression model, I introduce the r package *xtable* which extends the functionality of r and allows for the creation of well formatted LaTeX coded tables. This works because an rmarkdown file and recognise and interpret LaTeX code.

```
modenghist<-lm(resdf$english~resdf$history)
library(xtable)
coeffs<-xtable(summary(modenghist)$coef)
print(coeffs)
```

% latex table generated in R 3.2.4 by xtable 1.7-4 package % Sat Apr 9 13:03:15 2016

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	11.98	4.43	2.71	0.01
resdf$history	0.80	0.08	10.40	0.00



References

Broman, Karl. n.d. "Tools for Reproducible Research."

Knuth, Donald Ervin. 1984. "Literate Programming." *The Computer Journal* 27 (2): 97–111. http://comjnl.oxfordjournals.org/content/27/2/97.short.

Peng, Roger D. 2006. "Make the World a Better Place Through Reproducible Research."