Introduction to R

MATTHEW DENNY SUNDAY 19TH OCTOBER, 2014

INSTITUTE FOR SOCIAL SCIENCE RESEARCH
UNIVERSITY OF MASSACHUSETTS AMHERST

R is a statistical computing language that does everything that other programs such as SAS, Stata, SPSS and Excel can do, but with a lot more flexibility and functionality built on top. It is a full fledged programming language. More importantly, it is free and open source, and is usually the first platform to provide access to cutting-edge statistical methods. This functionality is added through user created packages, which provide additional functionality and commands to the R environment, and are often written by the inventors of a statistical method themselves. This workshop will teach you the basic skills necessary to read in an manipulate data, as well as access the breadth of functionality available in R packages and learn how to help yourself when you encounter problems.

This document provides links to the software you will need to download for this workshop and links to a number of additional resources. I have also provided a script file which will be our main focus for the duration for the workshop. In the next section, I will go over a number of useful features of the base R and RStudio graphical user interfaces. Not all of these will immediately make sense to you, but being able to easily install and load packages, search for help, create new scripts and run commands (for example) will make your life a lot easier as you get more into programming in R.

1 Resources

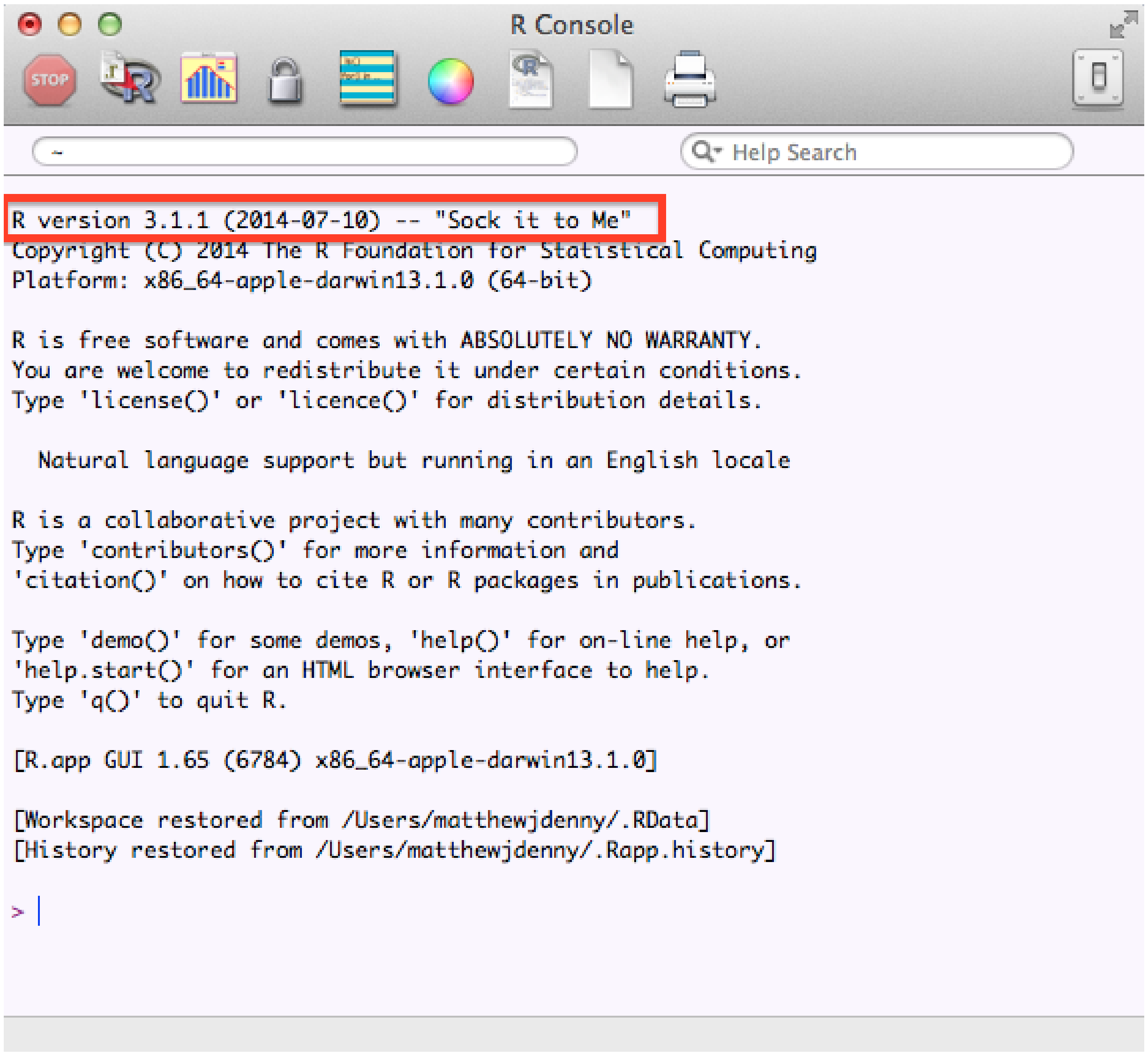
1. My favorite go-to guide for basic data management and statistics in R is the **Quick-R** website: http://www.statmethods.net/.
2. The R cookbook provides lots of helpful examples that you can adapt to do whatever you need to do: http://www.cookbook-r.com/
3. Hadley Wickham's excellent, thorough, well written ebook on advanced topics in R is available here: http://adv-r.had.co.nz/
4. The R graphics gallery is a really goo place to go to find code for making whatever crazy graph you could think of. http://rgraphgallery.blogspot.com/
5. **ggplot2** is what all of the cool kids are using these days to make their graphics in R. It is harder than the graphing functionality in base R but once you feel comfortable with base R and want to make crazier graphics, check this site out: http://docs.ggplot2.org/current/

2 Software

2.1 Base R

The R statistical programming language is available for download here: http://cran.us.r-project.org/. Select the 64 bit version if you have a computer manufactured in the last 3 years, otherwise, check online to see what version your operating system is. It is free and essential for this workshop.

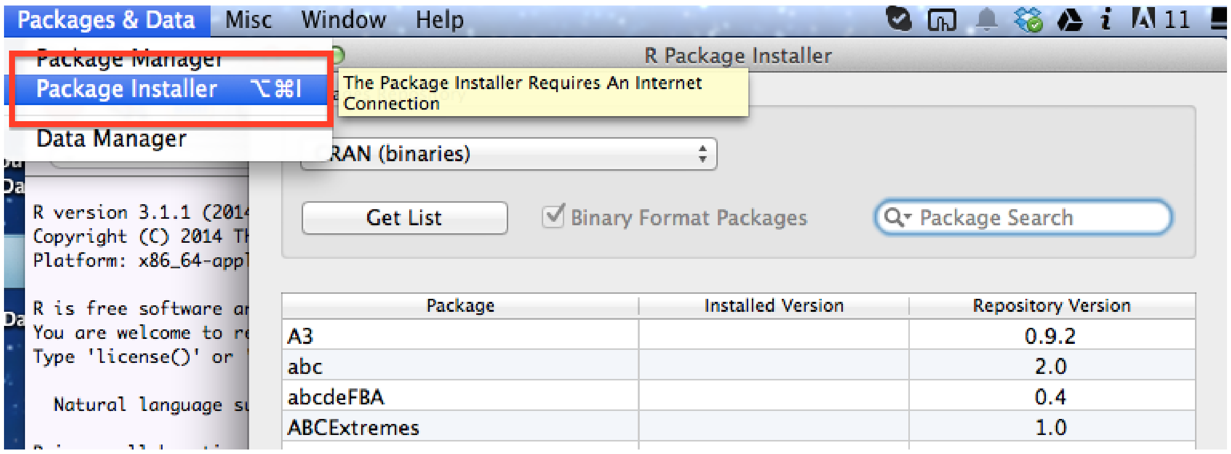
1. This is what base R looks like when you open it, just a simple console where you can enter R commands:



```
R version 3.1.1 (2014-07-10) -- "Sock it to Me"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.1.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.65 (6784) x86_64-apple-darwin13.1.0]

[Workspace restored from /Users/matthewjdenny/.RData]
[History restored from /Users/matthewjdenny/.Rapp.history]

>
```

Note that when we start R it will always tell us what version it is, which can end up being important.

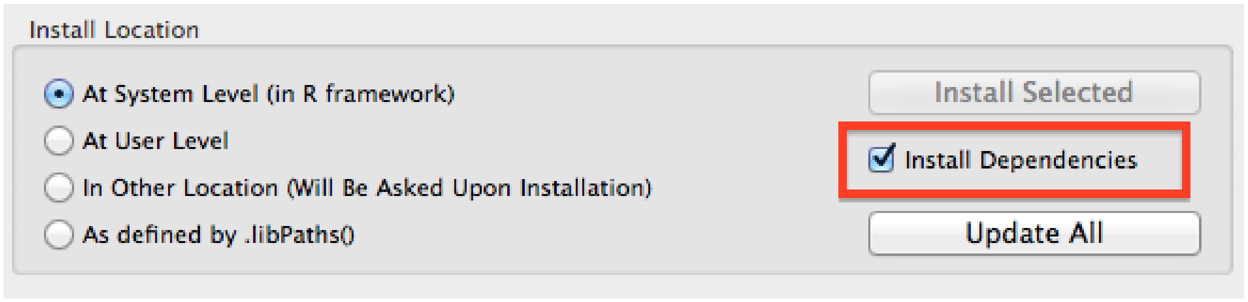
2. Base R is the best way to install user created packages as RStudio can run into some weird behavior with installing packages. For this reason I suggest you always use the included graphical interface for installing packages included in base R.



Package	Installed Version	Repository Version
A3		0.9.2
abc		2.0
abcdeFBA		0.4
ABCExtremes		1.0

This package installer lets you look for packages by name or just browse through all 2000+ packages available for R. This is where you go to download the latest functionality for new statistical analysis.

You can also download packages by using a command from the R console, but this way will often be easier as you are getting started and honestly I still do it this way myself for convenience most of the time. One important box to check is the **install dependencies** box at the bottom right of the package installer window. You will need to check this or the packages you install will often not work. This is because R encourages developers to make use of the functionality provided by other packages instead of reinventing the wheel, so just make sure you check the box!



Install Location

- [✓ At System Level (in R framework)]
- At User Level
- In Other Location (Will Be Asked Upon Installation)
- As defined by .libPaths()

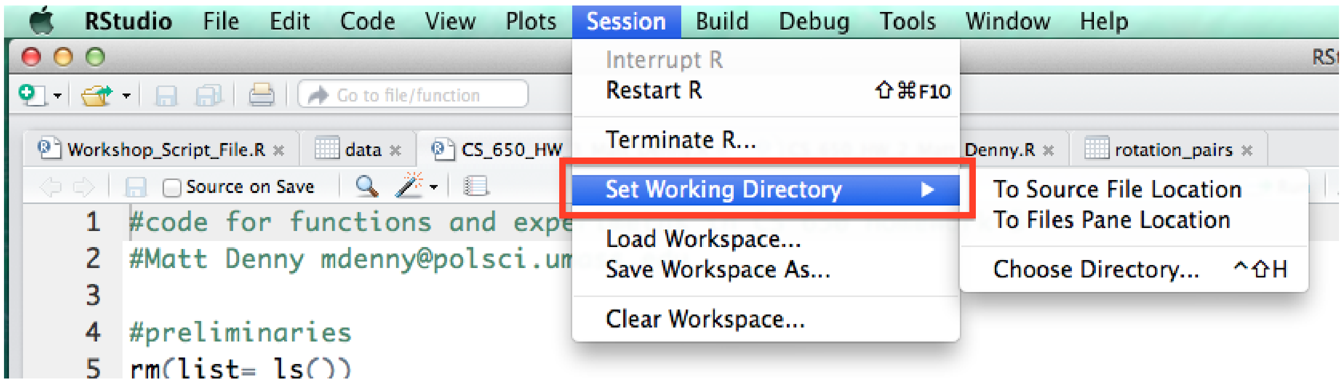
[✓ Install Dependencies]

It is also a good idea to make sure that the **At System Level** box is always checked (which it should be by default) so that you never run into problems accessing your packages.

2.2 RStudio

Please also download the RStudio IDE. It is also free! This program extends R and provides additional functionality such as a graphical user interface for keeping track of data objects stored in memory and debugging support. It is available here: http://www.rstudio.com/products/rstudio/download/. RStudio provides a number of extensions and upgrades to the base R user experience, some of which are menu driven. I will go over a few of these below:

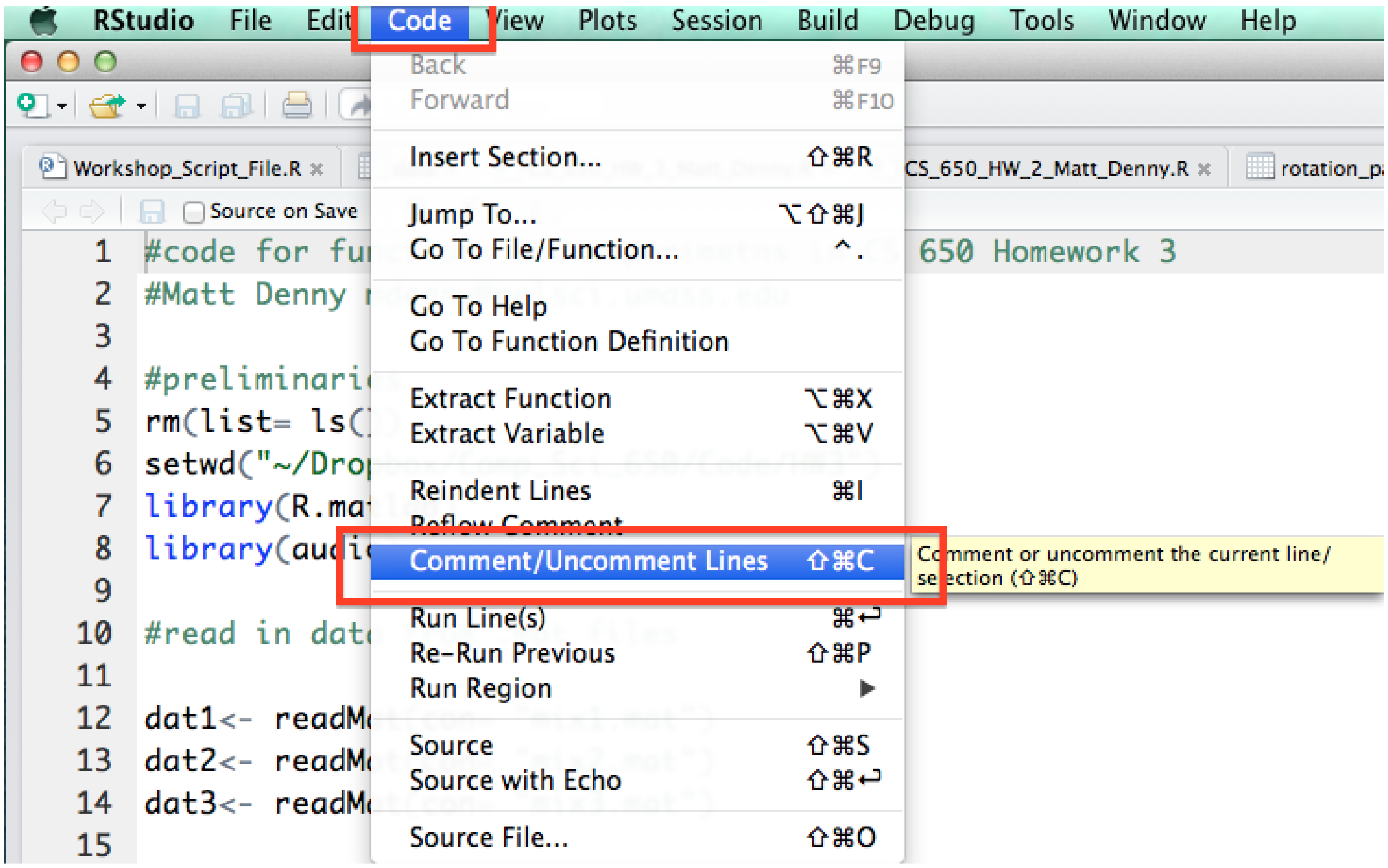
1. One of the things you will do most often is set you working directory. This is a way to tell your computer where you want to look for data by default, and where you have stored or want to store output. It is just a folder on your computer and can be set to your My Documents folder, although I suggest you create a separate folder for each project and keep all of the files, data and scripts for the project in that one folder. RStudio provides an easy menu driven way to do this:



```
#code for functions and expe
#Matt Denny mdenny@polsci.um

#preliminaries
rm(list= ls())
```

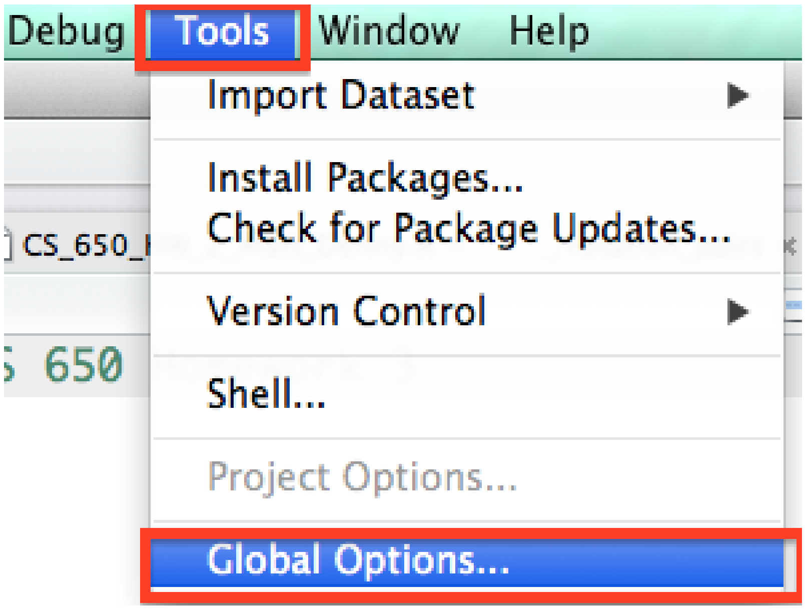
Note that you have several choices: you can choose to search for the folder if you select **Choose Directory**, or set your working directory to the folder that contains the R script file you are working on (only works if the script has already been saved and is in a folder where you have your data/want your output to go) if you select **To Source File Location**. This will also print the R command to your console pane, and then you can copy it into your script file so that you do not have to keep searching for the folder every time you open R but instead can just enter the one line R command that sets your working directory.

2. Under the code tab, R also has a handy functionality to comment or un-comment a block of code. This will come in handy if you discover that some of your code is incorrect or you just want to save it but make sure you do not run it by accident and do not want to comment the lines by hand.



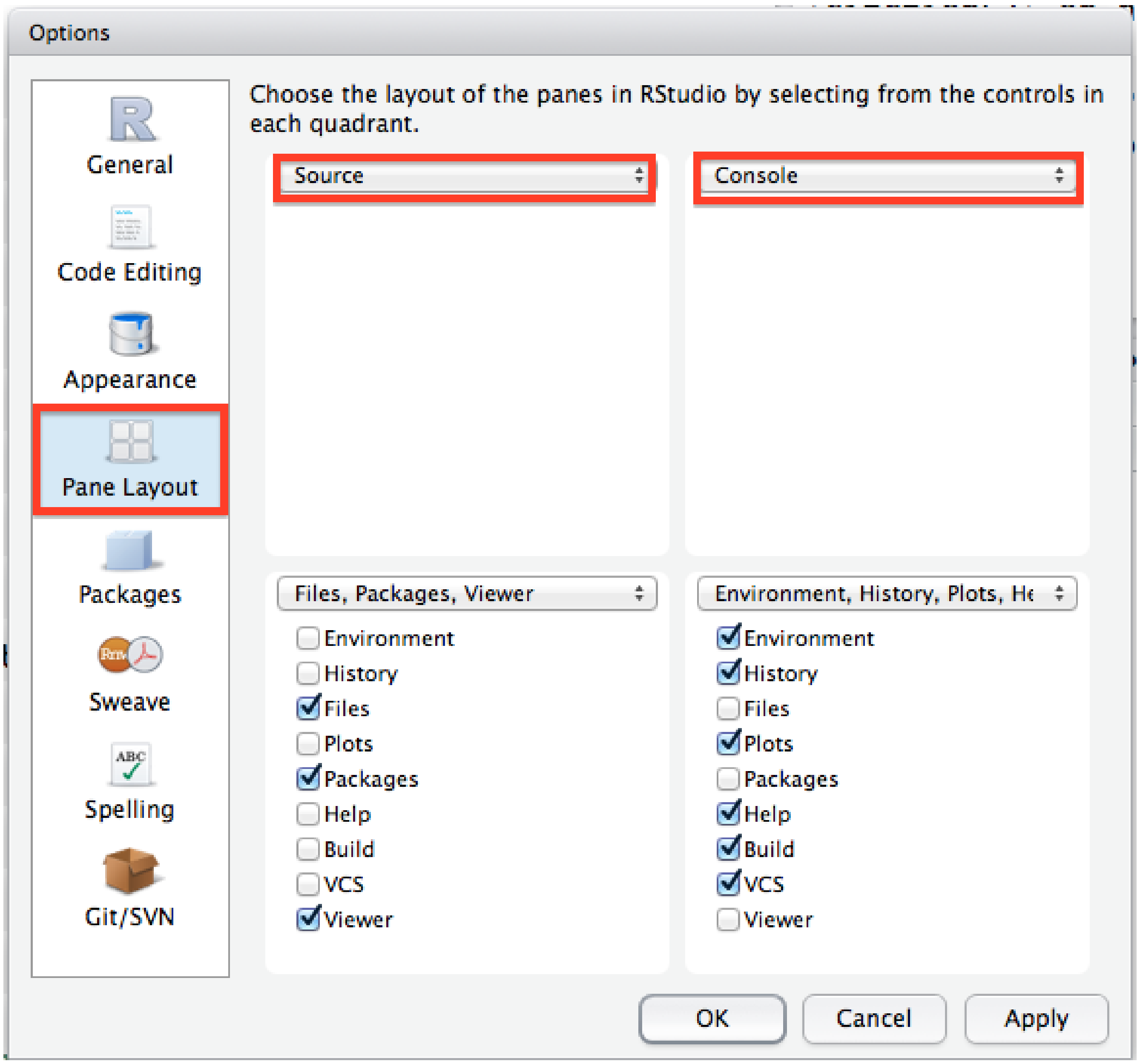
```
#code for fu                    650 Homework 3
#Matt Denny 

#preliminari
rm(list= ls(
setwd("~/Dro
library(R.ma
library(aud

#read in dat

dat1<- readM
dat2<- readM
dat3<- readM

```

3. Perhaps the most important menu item in RStudio is the button that takes you to the **Global Options** for RStudio.



From this menu, you will be able to change the overall appearance and functionality of R and RStudio.

4. One of the most relevant things you may want to do is change the window layout in RStudio. As you may notice in the workshop, I have my windows set up a different way from the default. You can change the layout on your computer by clicking to **Pane Layout** tab and checking the boxes as they are in the picture below:



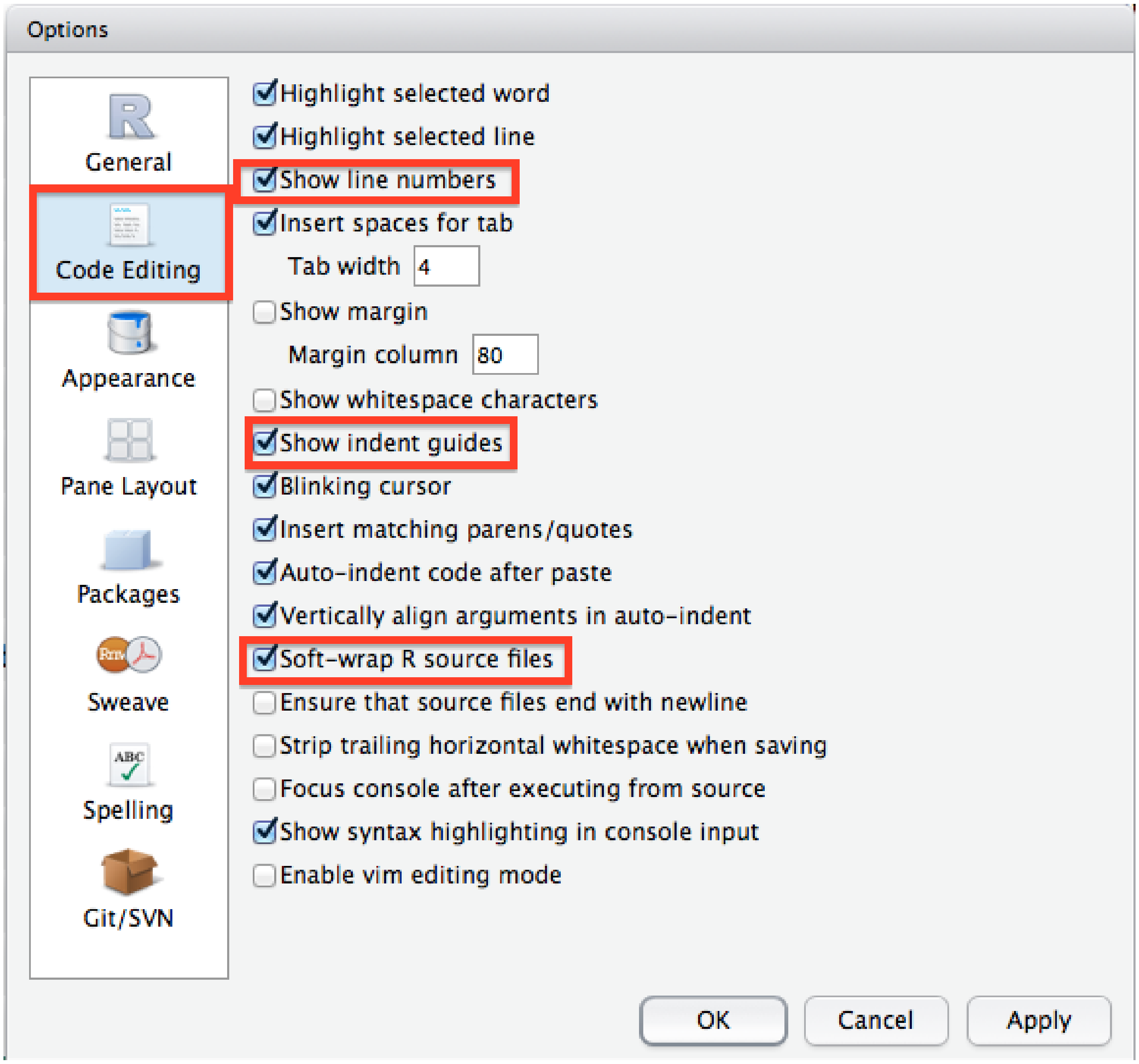
Choose the layout of the panes in RStudio by selecting from the controls in each quadrant.

Source | Console

Files, Packages, Viewer
- [ ] Environment
- [ ] History
- [✓] Files
- [ ] Plots
- [✓] Packages
- [ ] Help
- [ ] Build
- [ ] VCS
- [✓] Viewer

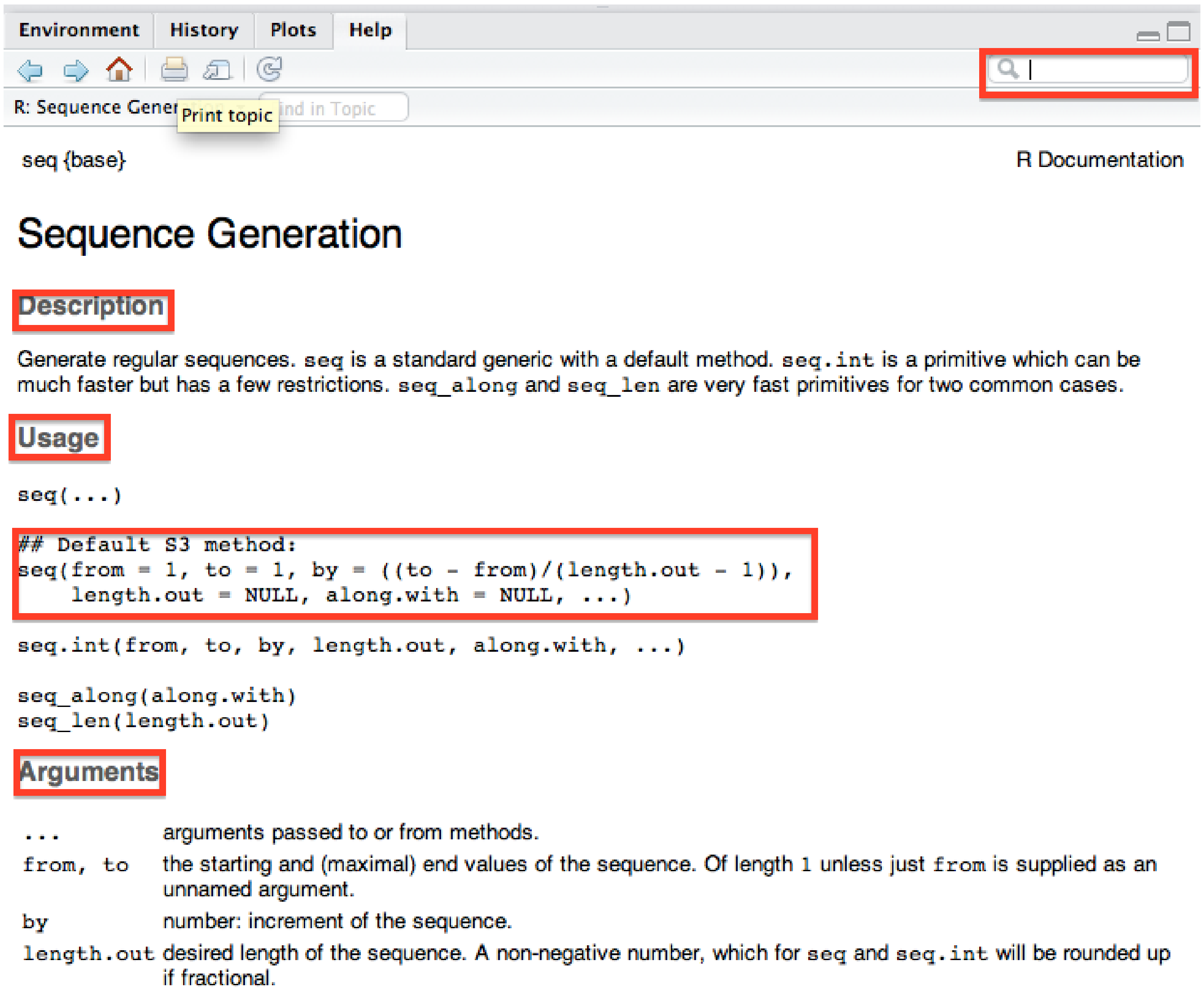
Environment, History, Plots, He
- [✓] Environment
- [✓] History
- [ ] Files
- [✓] Plots
- [ ] Packages
- [✓] Help
- [✓] Build
- [✓] VCS
- [ ] Viewer

5. There are another couple of options I highly suggest you change from their default values by clicking through to the **Code Editing** tab in the options window. I strongly suggest you check the box to **soft-wrap R source files** as this will make very long lines of R code wrap around to the next line so you do not have to keep side scrolling (extremely useful). I would also enable **Show line numbers** and **Show indent guides** as they will be helpful when you start writing longer programs. The line numbers are particularly helpful when you are trying to get help from somebody else.


Options

- [✓] Highlight selected word
- [✓] Highlight selected line
- [✓] Show line numbers
- [✓] Insert spaces for tab
- Tab width 4
- [ ] Show margin
- Margin column 80
- [ ] Show whitespace characters
- [✓] Show indent guides
- [✓] Blinking cursor
- [✓] Insert matching parens/quotes
- [✓] Auto-indent code after paste
- [✓] Vertically align arguments in auto-indent
- [✓] Soft-wrap R source files
- [ ] Ensure that source files end with newline
- [ ] Strip trailing horizontal whitespace when saving
- [ ] Focus console after executing from source
- [✓] Show syntax highlighting in console input
- [ ] Enable vim editing mode

6. Another useful feature of RStudio not found in the Options window is its integrated help window. On my setup, this can be found by clicking on the **Help** tab in the bottom right pane. In the top right of the help window there is a search bow where you can type in a function name and RStudio will display a bunch of useful information including a description of the function, its usage and arguments which you will need to actually use it in you own code, and all the way at the bottom there will be working examples which are often really helpful!



seq {base} R Documentation

Sequence Generation

Description

Generate regular sequences. `seq` is a standard generic with a default method. `seq.int` is a primitive which can be much faster but has a few restrictions. `seq_along` and `seq_len` are very fast primitives for two common cases.

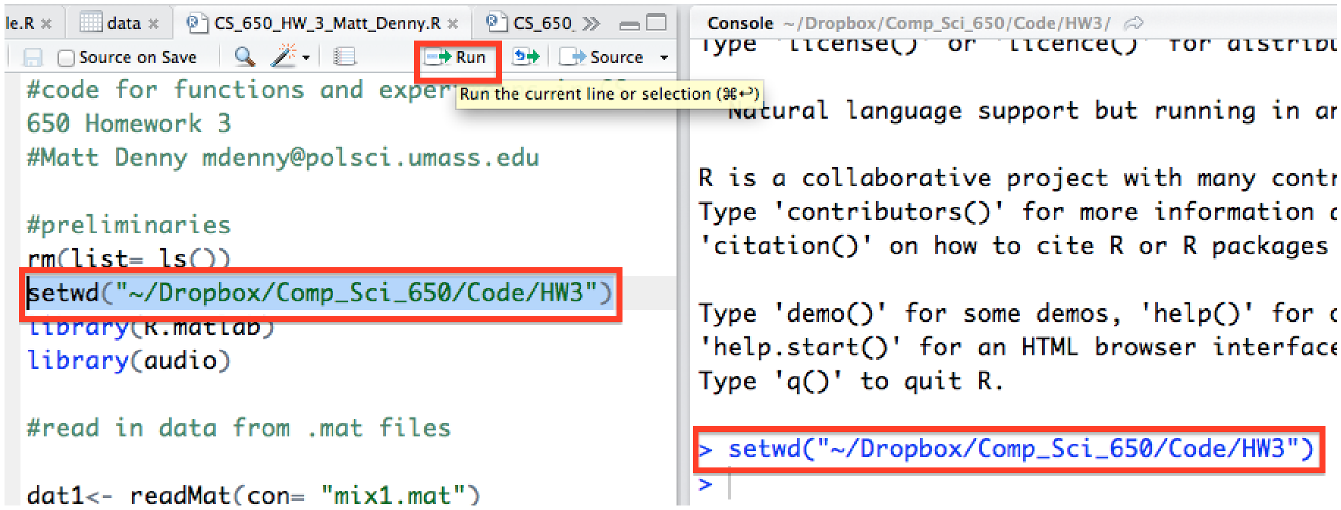
Usage

```
seq(...)

## Default S3 method:
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)),
    length.out = NULL, along.with = NULL, ...)

seq.int(from, to, by, length.out, along.with, ...)

seq_along(along.with)
seq_len(length.out)
```

Arguments

`...`	arguments passed to or from methods.
`from, to`	the starting and (maximal) end values of the sequence. Of length `1` unless just `from` is supplied as an unnamed argument.
`by`	number: increment of the sequence.
`length.out`	desired length of the sequence. A non-negative number, which for `seq` and `seq.int` will be rounded up if fractional.

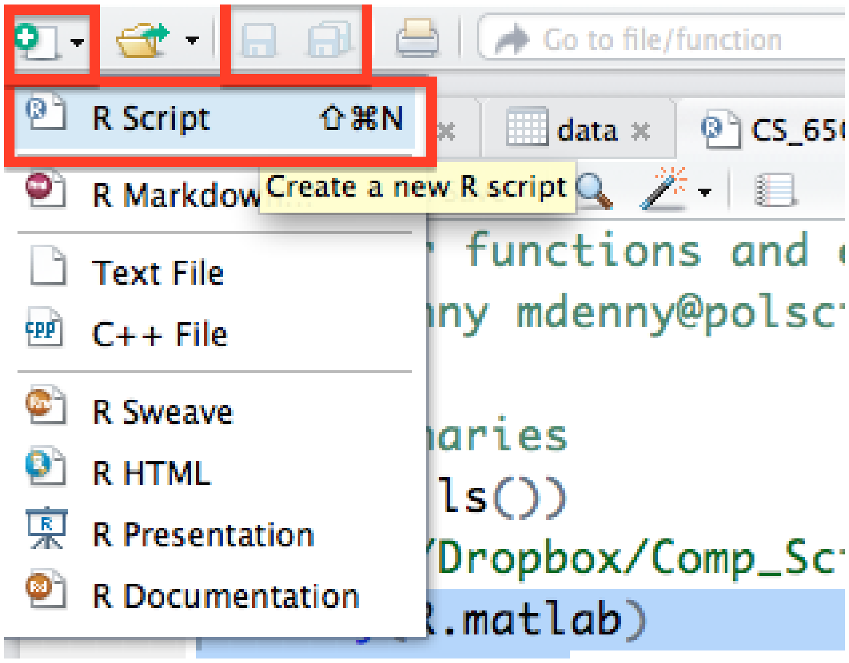
7. One other really useful feature of RStudio is the Run button, which much like its Stata counterpart, allows you to run a selected line or lines of code from your script file without having to copy the code into the console. You can also do this by using **Control + Enter** with text in the script file highlighted (Windows/ Linux) or **Command + Enter** (Mac). These are huge time savers.


le.R
data
CS_650_HW_3_Matt_Denny.R
CS_650
Source on Save
Run
Source
Run the current line or selection (⌘↩)
Console ~/Dropbox/Comp_Sci_650/Code/HW3/

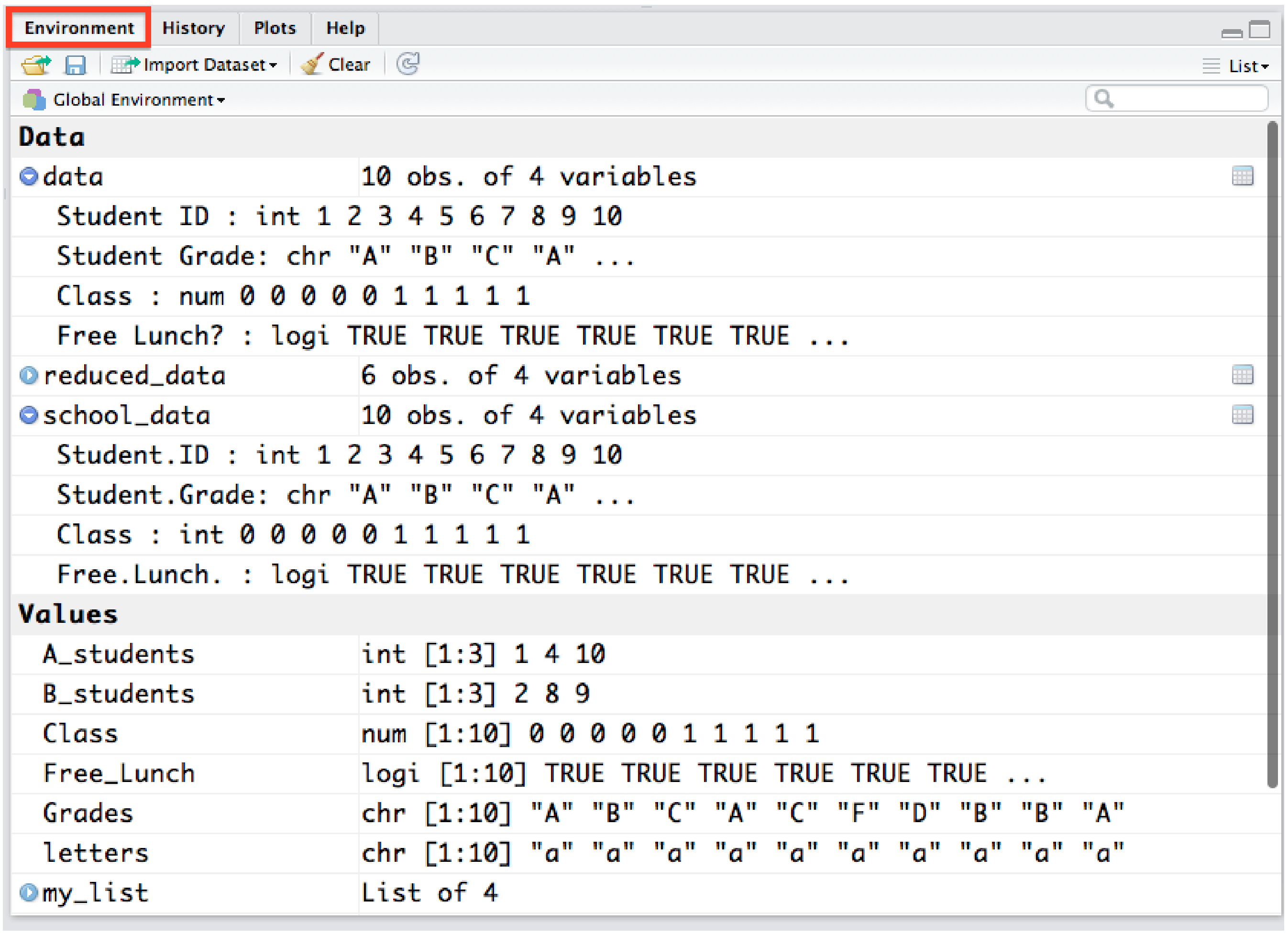
```
#code for functions and exper
650 Homework 3
#Matt Denny mdenny@polsci.umass.edu

#preliminaries
rm(list= ls())
setwd("~/Dropbox/Comp_Sci_650/Code/HW3")
library(R.matlab)
library(audio)

#read in data from .mat files

dat1<- readMat(con= "mix1.mat")
```

```
Type 'license()' or 'licence()' for distribu
Natural language support but running in ar

R is a collaborative project with many contr
Type 'contributors()' for more information c
'citation()' on how to cite R or R packages

Type 'demo()' for some demos, 'help()' for c
'help.start()' for an HTML browser interface
Type 'q()' to quit R.

> setwd("~/Dropbox/Comp_Sci_650/Code/HW3")
>
```

8. Before you go off coding your brains out, you will need to be able to create a new R script file, which you can do from the top left corner of the **Source** pane and then by selecting **R Script**. You will then want to immediately save this file which can be done by going to the file menu or by clicking on the floppy disk icon right next tot eh new file icon (which will only become active and light up when you have unsaved changes to the current script).

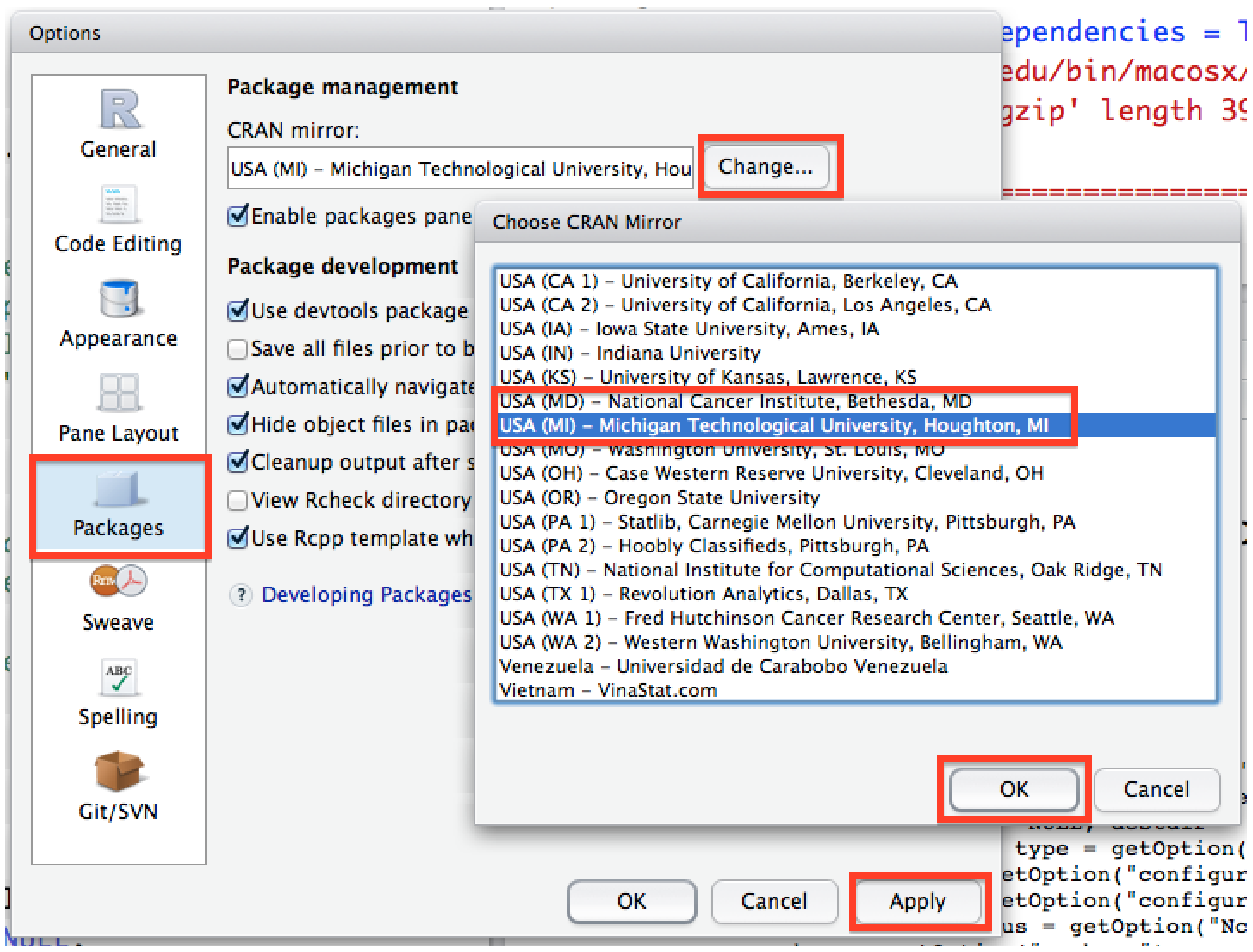


9. The environment pane is probably the single most useful feature of RStudio. It lets you see a visual representation of all of the variables and objects you have currently stored in memory. For the objects in the **Data** and **Values** tabs with a little blue arrow next to them, you can click on them and they will appear in spreadsheet form over in your **Source** pane so you can actually see the values. You can also see the variable type (character, integer, numeric, etc.) of a variable, vector, matrix, etc. which can be very useful in diagnosing problems with your data (for example numbers being read in as characters – which the regression functions in R do not like!).



```
Data
data                    10 obs. of 4 variables
  Student ID : int 1 2 3 4 5 6 7 8 9 10
  Student Grade: chr "A" "B" "C" "A" ...
  Class : num 0 0 0 0 0 1 1 1 1 1
  Free Lunch? : logi TRUE TRUE TRUE TRUE TRUE TRUE ...
reduced_data            6 obs. of 4 variables
school_data             10 obs. of 4 variables
  Student.ID : int 1 2 3 4 5 6 7 8 9 10
  Student.Grade: chr "A" "B" "C" "A" ...
  Class : int 0 0 0 0 0 1 1 1 1 1
  Free.Lunch. : logi TRUE TRUE TRUE TRUE TRUE TRUE ...
Values
A_students              int [1:3] 1 4 10
B_students              int [1:3] 2 8 9
Class                   num [1:10] 0 0 0 0 0 1 1 1 1 1
Free_Lunch              logi [1:10] TRUE TRUE TRUE TRUE TRUE TRUE ...
Grades                  chr [1:10] "A" "B" "C" "A" "C" "F" "D" "B" "B" "A"
letters                 chr [1:10] "a" "a" "a" "a" "a" "a" "a" "a" "a" "a"
my_list                 List of 4
```

10. One other very important thing you will need to do in RStudio is set the location where it should search for packages before you download them. In the **Options** window, go to the **Packages** tab, click change under the **CRAN mirror** heading and select either **USA(MD)** or **USA(MI)** (I find these work best for the east coast). Then click **OK** and then **Apply**. This will make sure that R knows where to look for packages when you ask it to download them.



Package management

CRAN mirror: USA (MI) – Michigan Technological University, Hou [Change...]

☑ Enable packages pane

Package development

☑ Use devtools package

☐ Save all files prior to b

☑ Automatically navigate

☑ Hide object files in pa

☑ Cleanup output after s

☐ View Rcheck directory

☑ Use Rcpp template wh

Developing Packages

Choose CRAN Mirror

USA (CA 1) – University of California, Berkeley, CA
USA (CA 2) – University of California, Los Angeles, CA
USA (IA) – Iowa State University, Ames, IA
USA (IN) – Indiana University
USA (KS) – University of Kansas, Lawrence, KS
USA (MD) – National Cancer Institute, Bethesda, MD
USA (MI) – Michigan Technological University, Houghton, MI
USA (MO) – Washington University, St. Louis, MO
USA (OH) – Case Western Reserve University, Cleveland, OH
USA (OR) – Oregon State University
USA (PA 1) – Statlib, Carnegie Mellon University, Pittsburgh, PA
USA (PA 2) – Hoobly Classifieds, Pittsburgh, PA
USA (TN) – National Institute for Computational Sciences, Oak Ridge, TN
USA (TX 1) – Revolution Analytics, Dallas, TX
USA (WA 1) – Fred Hutchinson Cancer Research Center, Seattle, WA
USA (WA 2) – Western Washington University, Bellingham, WA
Venezuela – Universidad de Carabobo Venezuela
Vietnam – VinaStat.com

```
ependencies = T
edu/bin/macosx/
gzip' length 39
type = getOption(
etOption("configur
etOption("configur
us = getOption("Nc
```