An Introductory Tutorial:
Learning R for Quantitative Thinking in the Life Sciences

Scott C Merrill

August 28, 2012

Chapter 1

This introduction to R is derived from multiple sources including:

1. An Introduction to R which was based an original set of notes describing the S and S-Plus environments written in 1990–2 by Bill Venables and David M. Smith when at the University of Adelaide. The R team made a number of small changes to reflect differences between the R and S programs, and expanded some of the material in their Introduction. Found at: http://cran.r-project.org/doc/manuals/R-intro.html

2. N. Thompson Hobbs (2012) *An Ecological Modeler's Primer on R*. Natural Resource Ecology Laboratory and Graduate Degree Program in Ecology, Colorado State University, Fort Collins CO, 80523

3. Plant, R. E. (2012) *Spatial Data Analysis in Ecology and Agriculture Using R*. CRC Press. Taylor & Francis Group, Boca Raton, Fl. USA

Notes: http://www.r-project.org/

Why use R?
It is a very broad language, fast, powerful, can be used for statistics, modeling, graphics, data manipulation, and simulation.

The downside: it is very flexible and powerful, and thus can be very confusing compared to weak programs (e.g., jmp). The documentation can be hard to access and difficult to use. It is such a broad and powerful language that jumping into the pool can be terrifying!

The goal for this course is to provide an introduction to this language and to teach you how to access packages – not to teach you every statistical package or modeling tool (but this is a small class and so we will delve into packages that fit your specific project!)

I encourage you to create an R coding notes document. This can be a very useful document. Put in tricks etc. Also, I urge you to keep a log, *for each project*, of what you have done including: 1) date, 2) program version, and 3) what you accomplished on that date.

Downloading R
Go to: http://cran.r-project.org/mirrors.html
Select any of the US sites (Berkeley works well http://cran.cnr.Berkeley.edu).

Windows users
Select the bulleted link "Download R for Windows"
You will want to download the "base" subdirectory.
The "base" link will take you to a page that will allow you to download "Download R 2.15.1 for Windows"
Save the file "R-2.15.1-win.exe" to your computer. Note that this is an executable file
Run the downloaded "R-2.15.1-win.exe" file and select standard downloading procedures. If you have a 64-bit system definitely get this option.

For Windows help see: http://127.0.0.1:10089/doc/html/rw-FAQ.html

Mac users
Select the bulleted link "Download R for MacOS X"
Download the file package "R-2.15.1-signed.pkg"

As noted on the site, this file is the **R 2.15.1** binary for Mac OS X 10.5 (Leopard) and higher, signed package. Contains R 2.15.1 framework, R.app GUI 1.52 in 32-bit and 64-bit for Intel Macs. The above file is an Installer package which can be installed by double-clicking. Depending on your browser, you may need to press the control key and click on this link to download the file.

Towards the bottom of the downloading page is a link "R for Mac OS X FAQ." that will help with any additional questions.

R basics

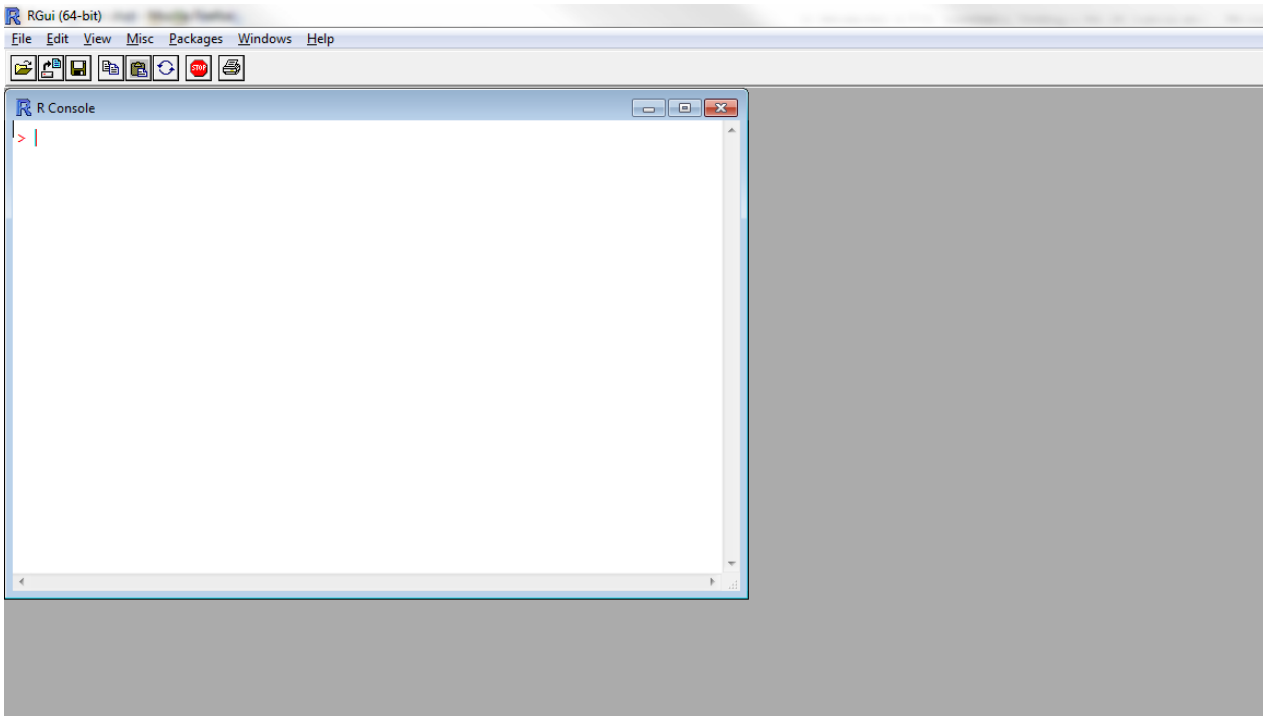
Command driven language – which means it is carried out by statements or commands

First Session

Okay, you have successfully downloaded R. Click on the R shortcut on your desktop and see if it works!

You should see something like the image below:

R console



At the prompt enter:

help.start () then press the <ENTER> key.

help.start() is a function that takes you to a website that will act as a sources for a lot of basic information. Play around on this site. You will find a lot of information here (FAQ, tutorials, etc.). Okay, now back to the R program.

The most basic functionality of R is that it can act as a calculator.

At the prompt enter:
> 8*5 then press the <ENTER> key.

[1] 40

> sqrt(16) then press the <ENTER> key.

[1] 4

> 4^2 then press the <ENTER> key.

[1] 16

> 8*5^2 then press the <ENTER> key.

[1] 200

BTW, what is happening here (order of operations)

The R console will allow you to type commands directly into the console but does not provide a good record of your work. That is, once you close the program you will not be able to retrieve / review your work.

Scripts take care of this issue. Scripts are a record of your code that, when compiled, will run your program (statistical analysis, animations, etc.).

Scripts (R Editor)

Windows
Go to the File tab, select New Script

Mac OS
Go to the File tab, select New Document

```
# This opens a new "untitled R Editor" box
# Note that "#" means what follows is a comment. That is, the computer will not compile code
#       after the # sign
# The comment # is your most important tool. Use it more than you think possibly could be
#       healthy and then use it some more. Annotate!! It is incredible how quickly one can
#       forget the thoughts that went into the stats, creating the models, etc.
```

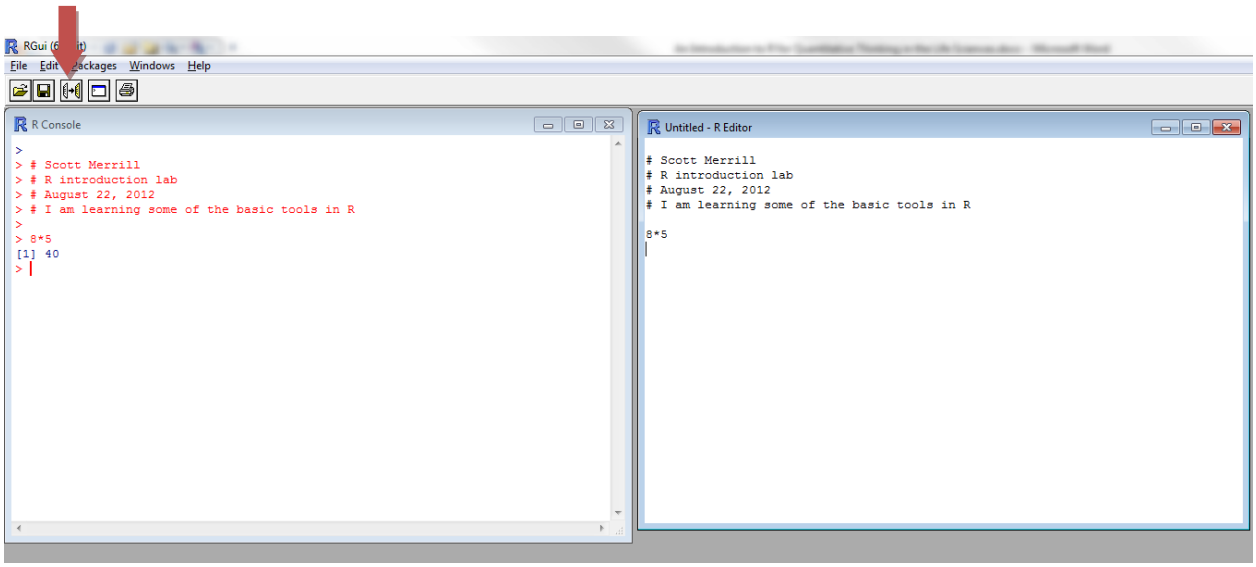
In the R editor box, start by entering as comments your name, and a title, the date and a brief annotation about what you are doing:
For example:

```
# Scott Merrill
# R introduction lab
# August 22, 2012
# I am learning some of the basic tools in R
```

Then type in your basic calculator function again:

8*5 then press the <ENTER> key.

You will note that nothing has happened. No response has been elicited from the console.
You will need to run or compile the code to get a response.



There are two ways of running your code. 1) click on the third icon (indicated by the red arrow above). When you click on this icon one line of code will run based on the location of the curser. If you highlight / select a portion or all of the code in the Editor then press this icon, all of the selected code will run. 2) you can use cheats instead of pressing the icon. Windows: highlight code to run and then press "control + r". Mac OS: highlight code to run "cmd + enter".

More on programming in a bit, but first we want to learn how to save our script and set up a directory that holds your R work.

Saving your work and setting up your working directory
Create an R working directory folder on your computer (or in a cloud if you have good and consistent access to the cloud). This will be a location for your R programs(code) and results, and perhaps assignments and notes as well.

From Hobbs (2012)

"As with all projects, it is helpful to organize your work into folders. So I suggest that you make folder named something like [PSS381] with a subfolder called Code. You will probably want to add additional subfolders to this to hold different exercises. You could also organize this by lab numbers; how to fit this into your overall organizational scheme is up to you—there is nothing magic about this. Now put some file, any file including a blank one in the folder where you want to store things. Once you have done this, type in an exceedingly useful command at the prompt:

```
> file.choose()
```

R will respond with finder in Mac OS and Windows Explorer in Windows. Navigate to your working folder and click on the file that is in it. R responds something like

```
> file.choose()

 [1] "C:\\Research\\R_Course_Directory\\PSS381_Code\\Population_Growth.xlsx"
```

Now select everything in the path except the file name and paste it, with quotes into the following command at the >

```
> setwd("C:\\Research\\R_Course_Directory\\PSS381_Code")
```

Now when you save a document (Under the File tab or with the cheat "cntr + s" or "cmd + s"), the document will start its save folder tree in your directory (you could still save it elsewhere if you would like).

Unlike a lot of programs these days, *R does not have an automatic save function*. Save often, and then again just to make sure!

Back to getting to know how to program in R…

Recall the "help.start ()" function that took you to a website with lots of information? Try typing and running/compiling the following instead of "help.start()":

At the prompt type:

```
>  Help.start()
```

I imagine you got an interesting error. The first error probably looked like:

```
Error: could not find function "Help.start"
```

Why? R is case sensitive. This is one of the biggest sources of error found when programming in R. For example, if you said "x = 4" then "X*2" thinking that you would get 8, you would instead get an error because the R program thinks x and X are different objects and has no idea what value X has. Specifically, R thinks Help.start() is a different function than help.start().

Work through Tom Hobb's Chapter 3.3 on programming (directly below). This is a great introduction to coding.

3.3 Writing programs[3]

Most of your work in R will be done with programs, or strictly speaking, scripts[4]. To illustrate, go to File and click on New script for Windows or File and New Document in Mac OS. A window opens that allows you to type in programs, collections of statements that can be submitted entirely or in bits and saved for later use. For example, enter the following lines of in your script window:

```
r = .2

N0 = 100

N1 = N0*exp(r)
```

```
N1
```

Now do two things. First, put your cursor at the top of the file and press cntrl-r if you are using Windows R (or cmd-enter if you use a Mac) at each line, observing the console each time. Your output should look something like:

```
> r = .2

> N0 = 100

> N1 = N0*exp(r)

> N1 [1] 122.1403
```

Note that when you entered N1 at the console, R returned its value. If it had not been assigned a value then R would return:, Error: object "N1" not found. To illustrate, enter N0 (which has been assigned the value 100) then N2 (which has not been assigned any value. More about this message shortly.

Now select the first two lines with the mouse and again press cntrl-r, then do cntrl-a followed by cntrl-r, observing the results each time. In Mac R, you would use cmd-enter and cmd-a[5]. You have now learned 3 ways to input from a script to R: one line at a time, a few lines selected, or all of the lines in the widow selected. All three of these methods will be used heavily in this course.

Commenting your code

[SCM: I left this in here to reinforce how important commenting is!]
This is important. R allows you to make little notes to yourself with code to help you remember what you did when you come back to a program after six months. Or, these comments allow someone else to understand what you are doing. Making a habit of commenting your code is one of the best things you can learn in this course. To add a comment, simply put a # in front of whatever your write. So, for example, to add appropriate comments to the code you just wrote:

```
# intrinsic rate of increase
r = .2

# initial population size
N0 = 100

# Calculate new population size after one unit of time
N1 = N0*exp(r)

# Output new population size to console
N1
```

Ok, ok, ok—these are baby comments. But you get the idea. The comment symbol can also be useful to prevent a line of code from executing[6] We will talk about this as the opportunity arises in lab. No programmer likes to take the time to write comments and I am not terribly good at it. However, if you fail to do so, "the future your will be really mad at the past you."

[3]I will assume that you will use the standard R editor, which is what I have always done, but there are much fancier ones. If you want something with more features then you should look into EMACS ESS, Tinn-R or the commercial editor, BBEdit

[4]Strictly speaking because R is an interpreted language and the instructions given to its interpreter are called scripts. The instructions given to a compiler (for complied languages like C) are, strictly speaking, called programs. But I rarely make this distinction.

[5]Ok—when I originally wrote this document I used Windows. I have tried to add text throughout the document reminding Mac users that the Mac equivalent of cntrl-key is cmd-key, but I may have missed some spots. I will simply assume that Mac users, being a cunningly clever bunch, will be able to do this mental translation without my help.

Congrats, you have made it through Chapter 1 this tutorial. If you have any questions come by my office (217 Jeffords), e-mail, ask a classmate, or bring them to class on Wednesday.