Appendix A. Getting started with R

1. Why R?

I use R and recommend that others use R because it is:

(a) a powerful statistics package
- good at reading data
- wide range of statistical tests and techniques
- good graphics
- very flexible

(b) a usable package
- available for many platforms (PC, Mac, Unix, Linux....)
- programmable
- user community for support

(c) it is noncommercial
- distributed under the GNU "copyleft"
- maintained by a community of users
- upgrades happen because the users need improvements, not because the company needs more money.

2. What is R?

From the R FAQ (frequently asked questions) at the R project page - http://www.r-project.org/.

"R is a system for statistical computation and graphics. It consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files."

3. How do I get R on my computer?

(a) go to the R project web site (www.r-project.org).
(b) click the CRAN link to see the download servers on the Comprehensive R Archive Network.
(c) choose a download server near your location.
(d) choose your platform - click to download and follow instructions.

4. How do learn how to use R?

The R project web site has several R books available in PDF format. I can recommend "R for Beginners" by Emmanuel Paradis as a particularly good starting point. I also use the internal "help" documentation in R all the time. At the R prompt you can type:

```
> help(command.name)    # if you know the name of the command you want to use
> help.search("topic")  # to find commands that relate to a topic
```

5. Now what?

The best way to get to know R is to use it, so here's a little demo of some features. This demo assumes that you've installed R on your computer. Also, for step (c) you may need to use a web browser to download my "central.limits" script.

(a) Make a directory (folder) for this R session

```
[n] mkdir R_demo
```

(b) Make that new directory the current working directory

```
[n] cd R_demo
```

(c) get the script "central.limit" from "http://ling.ohio-state.edu/~kjohnson/ling795q/scripts"
The "." at the end of this command is the destination, so the command means 'copy the file /home.../central.limit to here - the current working directory'

```
[n] cp /home/kjohnson/public_html/ling795q/scripts/central.limit .
```

(d) open it in the emacs editor - we'll look at it and change it. The "&" causes the command to start the editor and then return a command prompt to us.

```
[n] emacs central.limit &
```

(e) Start R

```
[n] R
```

(f) Now we type commands to the R prompt ">" -- load the script into R

```
>source("central.limit")
```

(g) See what it does when you run it with the default parameters

```
>central.limit()
```

(h) Look at the emacs file to see how the input paramters are named - try chaning one.

```
>central.limit(n=50)
>central.limit(50)
```

(i) what does "signif() do? remove this function by changing the text in the emacs window

```
find the expression “signif(sd(means),3)”  and change it to “sd(means)”
“<ctrl>x<ctrl>s”               # to save the change in emacs
>source(“central.limit”)    # to load the now changed script
>central.limit()                # to see the consequence of the change
```

(j) get some output from the function The script has a structure like this:

```
central.limit = function() {
        .......
        if () {
        ......
        } else {
        ........
        }
}
```

Now add a line between the last two closing curly brackets:

```
        }
        return(means)  # spew out the vector of mean values
}
```

Then, save the change, read the script into R again, and run it.

```
<ctrl>x<ctrl>s
>source(“central.limit”)
>central.limit()
```

(k) Now that we have the script return the array of means whenever we run the script we get a long list of numbers, the means of our (m=1000) samples. We can capture this list of numbers in a vector.

```
>mymeans = central.limit()
>sd(mymeans)   # this should give the same value as the one printed as “standard error”
                       #       on the figure
```

try this:

```
>sd(central.limit())      # a new plot, and standard deviation number, without storing
                              # “means” in a vector.
```