Stat 215B (Spring 2004)
Computing Guide: Getting started

B. M. Bolstad
bolstad@stat.berkeley.edu
http://www.stat.berkeley.edu/users/bolstad

January 22, 2004

1 Introduction

The purpose of this document is to introduce you to some of the computing resources that you should make use of for this class. You are expected to learn how to effectively operate R and/or S-plus to carry out your analysis. You are welcome to use an alternative of your own choosing, such as Matlab, but some analysis methods might be difficult to carry out in this case. Most analysis code will be interchangeable between R and S-plus, but you might find some differences. You need only use one of the packages with my recommendation being that you use R (since it is free you can install it on your own machine).

For preparing the lab reports you must use a typesetting/word processing program of some kind. I recommend you learn LaTeXof you intend to make serious use of mathematical formulae in your write-ups. But if you use Windows, you are welcome to use Word or something else that you are familiar with.

You may work on your own computer or on the network provided by the Statistics department. Henceforth, we shall call this the Statistical Computing Facility (SCF). This document is broken into two major parts. The first covers software on the SCF system. The second discusses how to install the appropriate software on your own computer.

2 The SCF system

2.1 General Information

You can work in Evans 491, 342 or 432. Make sure you are throughly familiar with the rules and regulations relating to the usage of the labs and acceptable passwords. You may view documents relating to this information by typing `help rules` and `help passwd` respectively.

2.2 Intro to UNIX

SCF computers use a UNIX operating system. Many UNIX commands have options that can be attached to the basic commands. For instance, the command `ls` lists objects in a directory. But the option -l gives a lengthier output. Several options can be added on a command simultaneously. For example, try the command `ls -alt`. For all of the available options and more details than you will ever want, try the "man pages" by typing `xman &` from inside Xwindows, or simply man command-name.

Wild-cards are also very useful in UNIX. The command `ls *.s` will list all objects in the directory ending in the extension '.s'. There are many other useful wild-cards.

command	description	example
ls	Lists objects in directory	ls -la
cp	Copy a file	cp from-file to-file
mv	Move file(s)	mv from-file to-file
rm	Remove (delete) file(s)	rm myfile
cat	prints file(s) to screen	cat myfile
more	Like cat, but you can read it	more myfile
less	Like more, but better	less myfile
mkdir	Make a sub-directory	mkdir mydir
cd	Change current directory	cd mydir
rmdir	Remove an empty directory	rmdir mydir
pwd	Show full path of current directory	pwd
lpr	Send a file to the (default) printer	lpr myfile
xterm	Open another window	xterm &
quota -v	Check your account's disk quota	quota -v
charges	You have limited free printing	charges
man	"Man page" for a command	man command
mail	Read mail using UNIX program	mail mail
	Send mail using UNIX program	mail user@host
pine	A more user-friendly mail program	pine
help	For SCF help files	help topic
help -l	Topics with help available	help -l
logout	To log out of your account	logout

Table 1: Some common UNIX commands that you may use on the SCF system

2.3 Some basic UNIX Commands

Table 1 is just the very basics to get you started. Once you figure these out, below is a list of other very useful commands. Type man command-name to figure out what they are. Of course there are literally thousands more that aren't listed here. Consult your local computer nerd for more help.

`alias, bg, chmod, compress, echo, enscript, fg, finger, ftp, grep, history, lpq, lprm, nice, ps, set, setenv, source, tar, telnet, top, vacation, zip.`

2.4 Other UNIX information

Note a few special symbols. Two consecutive dots refers to the current directory's parent directory. For example, to move from a subdirectory to the directory containing it, I type `cd ..` The tilde "," always refers to your root directory. So, if I am sitting in any directory and want to copy a file to my root, I can just type cp filename , (note that not providing a name means that the file will be copied into the directory to a file of the same name as the original)

Three final useful concepts in UNIX. First, redirections. The symbol ">" takes output from one command and redirects it to a file. For example (and you should try this one) the command `help -l > help.list` lists all of the topics for which help is available, but the output is sent to the file "help.list" instead of the screen. Note that ">>" works in the same way, except the file is appended if it already exists. The "<" symbol works the same way but for input rather than output. Second, pipes. The symbol "|" takes the output of one command and makes it the input of the next. For example, to have only the first 10 lines of a file output to the screen use the command `cat myfile | head -10`. Of course, the output of a pipe can then be redirected too. Third, background processes. Commands that take a while to execute can be run in the background, which returns control of the prompt to you while the processor is working. This is done by putting the ampersand symbol "&" on the end of the command. Compare typing `xman` with and without

an ampersand. Type `help background` for more information.

An online summary of UNIX for new users is available by typing `help summary`. Even more information can be found by typing `help learn_unix`.

You will probably want to change your shell from csh to tcsh. This is essentially the same as csh but gives you command line editing, so you can use the cursor keys to edit commands in the X windows. To do this, give the command `chsh` followed by typing tcsh.

2.5 Text Editors

A text editor is an essential tool, so it is a good idea to get proficient at using one right away. There are three main choices: `emacs`, `jove`, and `vi`. Most people nowadays prefer emacs, and leave the other two for hackers (in the computer geek community the text editor preference is somewhat of a religious war). Type `help edit` for how to get started. This help file also tells how to get printouts of postscript files of quick reference cards for some of the editors. When running from Xwindows emacs has a useful "help" pull-down menu on the top which includes a tutorial and a list of key bindings.

2.6 Introduction to Splus

This is just enough to get you going. Phil Spector's book "An Introduction to S and S-plus" is excellent. The text for 215B, Venables and Ripley (VR), 4th edition, also has a good introduction. Before we start, there is some setting up to do. First cd into the directory from which you will work. Then, if it doesn't already exist type `mkdir .Data`. Type ls -a to see that it is there. All of your Splus objects will be stored in this subdirectory automatically. Next, type the command `setenv S.CLEDITOR emacs` (or replace emacs with vi if you prefer). Now initiate Splus with the command `Splus -e`. The option -e allows us to use editor commands inside of Splus.

Note It is very important you create a new .Data directory each time you start a new project, otherwise you may end up confusing yourself with objects and data items from your previous sessions. I suggest you do something like `mkdir lab1; mkdir lab1/.Data` before starting work on each lab

Once inside of Splus, the default prompt is ">". A good place to start is with the internal help system. This is available with the command

```
> help.start()
```

Assignments are done with the two symbols "<-". In other words, a less-than sign immediately followed by a dash. For instance, the following command assigns a vector of 10 numbers to the variable a

```
> a <- c(2:8,6,6,7)
```

Simply type the variable name to see its value. The function `c()` is for concatenate, and is used to create vectors. The expression `2:8` does the same thing as `c(2,3,4,5,6,7,8)`. Let's do some simple arithmetic. Try the following:

```
> sum(a)
> b <- 5*(a+1) - 7
> d <- b / a^2
> a>5
> e <- a[a>5]
```

Notice that when no assignment is made the results of the computation are print to the screen. Such results are stored temporarily (until the next computation) in the variable `.Last.value`. Notice that the standard arithmetic operations are performed component-wise when applied to a vector (also true for matrices), and the standard order of operations is followed. The operation > is a logical comparison, and the final expression will store in the variable e those elements of a that exceed 5 . The square brackets are used to index elements

of a vector. To see how many elements the vector `e` has, use `length(e)`. To see all of the variables in your work area type `ls()` or `objects()` for a list. Note what happens if the parenthesis are omitted.

Here are some examples of matrices:

```
> x <- matrix(1:12,3,4,byrow=T)
> y <- matrix(c(7,3,4),2,3)
> z <- matrix(rnorm(30),ncol=3)
> w <- diag(rep(1,4))
```

Notice what happens if `byrow=T` is omitted from the first example. As for the second, if the number of values divides the number of entries in the matrix evenly Splus will cycle through until the matrix is full. This can lead to some nasty bugs. The third makes a 10 by 3 matrix of random standard normals. The fourth uses the useful `rep()` function to make a 4 dimensional identity matrix. Matrix multiplication is done with the symbol `%*%` and the function `t()` transposes a matrix. A very useful function is `apply()`. It applies a function to rows or columns of a matrix. For example,

```
> apply(z,2,mean)
```

Takes the means of the three columns of the matrix `z`. See also `tapply()`, `sapply()`, and `lapply()` for cousins of this function.

Let's talk graphics. To get a histogram of `d`, we first need to open a graphics device.

```
> motif()
> hist(d)
```

The brackets `()` denote that the object is a function. The first one above does not require any arguments. One of the powerful features of Splus is the ease with which you can write your own functions. For example, here is a simple function that returns the standard deviation of a vector of numbers.

```
> sd <- function(x) {sqrt(var(x))}
```

With more complicated functions you will want to create them in an editor and then load them into Splus using the Splus function `source()`. We can now call this function just as if it was any other function. For example,

```
> sd(a)
```

finds the SD of the vector a. See VR chapter 4 for more on functions. // Here is an example of a more complicated analysis. Suppose that a data file called "fish.dat" exists in the same directory that I am working in and contains two columns of data. The first is the weight for 68 trout caught in Bear Lake, and the second is the length. We want to perform a linear regression of weight on length and produce residual plots all on the same figure. Use your editor to generate your own data if you want to practice this excercise. The symbol # is the comment symbol. Use `help.start` for details regarding the functions.

```
> fish <- read.table("fish.dat") # read in data
> dimnames(fish) <- list(NULL, c("weight","length")) # give labels
> fish <- as.data.frame(fish) # makes life easier below
> fit1 <-lm(weight ~ length, data=fish) # fit the linear model
> fit1.sum <- summary(fit1) # get results from model fit
> sink("fish.fit") # output fit results to a UNIX
> print(fit1.sum) # file called "fish.fit"
> sink() # close the file when done.
> motif() # open a graphics device
> par(mfrow=c(2,2), oma=c(0,0,2,0)) # partition graphics region
> plot(fish$length, fish$weight) # scatter plot
> abline(fit1) # add regression line
```

```
> plot(fit1$fitted, fit1$resid) # residual plots
> plot(fish$length, fit1$resid) # more residual plots
> qqnorm(fit1$resid) # normal quantile plot
> qqline(fit1$resid) # best line for quantile plot
> mtext(outer=T,side=3,"Linear Regression of Weight and Height") # title
```

While in Splus you can still use UNIX by inserting the "!" sign before the usual UNIX commands. For example,

```
> !ls
```

will list your UNIX files and directories. Or, if you want to open another window from UNIX, just type

```
> xterm &
```

Finally, we will make use of Splus libraries. Try the following

```
> library()
> library(help=mass)
> library(mass)
```

The first command lists the available libraries for our system. The second displays the help file for the library called mass, and the final line attaches that library to your search list. Now, all of the functions and data sets contained in that library are available for the remainder of your session. Consider creating a `.First()` function to attach important libraries automatically whenever you initiate Splus (see VR page 57).

To leave Splus, use

```
> q()
```

Your variables will be saved automatically in the subdirectory .Data.

2.7 R

You can start it by typing `R` at the command line. You will find it very similar to S-plus and most of the above commands should work perfectly. Start `help.start()` to start the help system.

2.8 LaTeX

Latex is a popular typesetting program, unlike other programs with which you may be familiar it is not WYSIWYG. It requires some learning of a command language, but you should find that this disadvantage is outweighed by the power you'll get in return. Latex is particularly useful for typesetting equations. Assuming that you already have a latex file named `report.tex` (there is a sample on the section homepage, which it is suggested you examine with the use of an editor) the command to process the file is

```
$ latex2e report.tex
```

note that this uses LaTeX2e as the processor, you may use `latex` to use the older LaTeX2.09 (I don't recommend it unless you are already used to this version). If there are any errors in your latex file they will be reported and you can edit your file and reprocess it again. When you are ready to view your file use the command

```
$ xdvi report.dvi
```

if you then want to print your file use the command

```
dvips -o report.ps report.dvi
```

this will output your dvi file to a postscript file. You can now either view it with the command `gv report.ps` or `ghostview report.ps`. Alternatively you can send it directly to the printer using the command

```
$ lpr report.ps
```

Note that while you will not be required to use LaTeXto produce your reports it may be to your advantage to learn at least a little. In any case it will be expected that you submit word processed reports.

2.9 Other Software on the SCF system

For a full list of available non-statistical software, type `help software`. A list of statistical software can be found by typing `help statistics`.

3 Installing requisite software on your own computer

3.1 Downloading and installing R

The R webpage is `http://www.r-project.org`. At this website you will find information about the R package.

3.1.1 Windows Systems

1. You need to download the Windows installer from CRAN. Go to either `http://cran.r-project.org` or one of the US mirrors `http://cran.us.r-project.org/` `http://cran.stat.ucla.edu/`
2. Click on `Windows (95 and later)`
3. Click on `Base`
4. Click on `rw1081.exe` and save the download at a location you can find.
5. When it has finished downloading, double click `rw1081.exe` to start the setup program.
6. Follow the on screen setup instructions. This should pretty much be a matter of clicking next through a series of screens.
7. Find R either via the start menu or the icon on the desktop.
8. You should now see something similar to 1 on your screen. Congratulations you are done.

3.1.2 Linux Systems

On Linux you have two options. One is to download a pre-compiled version for your specific distribution. I will give details for the second option which is to download and compile the source code for yourself.

1. You need to download the source code from CRAN. Go to either `http://cran.r-project.org` or one of the US mirrors `http://cran.us.r-project.org/` `http://cran.stat.ucla.edu/`
2. Click on `R-1.8.1.tgz` to download the source code and save it somewhere.
3. Go to the location you saved the downloaded file and type `tar xzvf R-1.8.1.tgz`. This will uncompress and extract the R source code.
4. Change into the source code directory `cd R-1.8.1/`
5. You might want to read the `INSTALL` file, but this is not completely necessary.

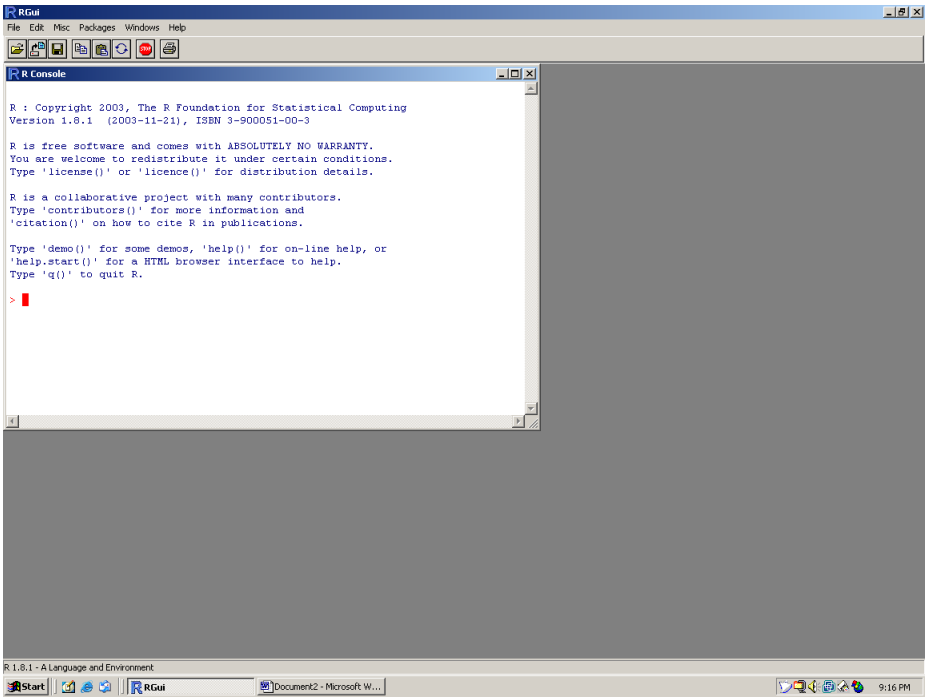

Figure 1: A typical R session on a Windows machine

6. If you have root access on your machine just type `./configure`. If don't have root access and you have sufficient disk space somewhere type `./configure --prefix=/path/to/install/location` where of course you replace `/path/to/install/location` with your install location.

7. Now type `make`. It will start compiling. This might take awhile, depending on the speed of your machine.

8. Type `make install`. This will install R.

9. You may need to add the `bin` subdirectory of your install location to your path. Use `setenv` (csh/tcsh) or `export` (bash) to do this.

10. type `R` at the command-line.

11. If all goes well you should have a working R installed.

3.1.3 Installing additional R packages

There may come a time where you want to install an additional package to your R installation because the base install does not have a function that you need (not necessarily during this particular class). You can find many packages on CRAN.

On Windows you use the "packages" menu (see figure 2 to install new packages. You have two options: Either download the file from CRAN (make sure you get the file with the ".zip" extension) and use the "install from local zip file" option or Choose the "Install package(s) from CRAN" and select the package you want (it will be downloaded and installed automatically).

On Linux/UNIX machines you use `R CMD INSTALL` to install packages. eg `R CMD INSTALL packagename_1.0.0.tar.gz`. You may need to set the `R_LIBS` environment variable.

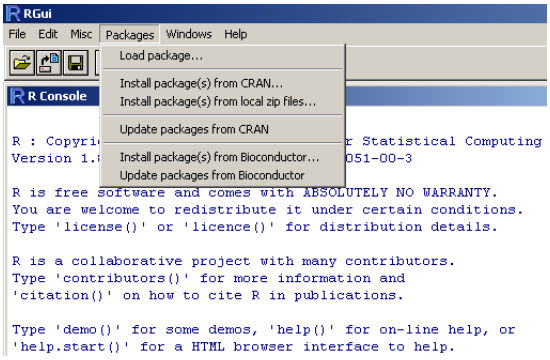

Figure 2: The install menu on a Windows R

3.2 LaTeX

3.2.1 Windows Systems

Easy to follow instructions are given at `http://www.math.auk.dk/~dethlef/Tips/introduction.html`.

3.2.2 Linux Systems

Installing on Linux is more complicated. It generally depends on your distribution. Often it is already installed for you. Consult the documentation for your particular distribution for installing LaTeXsoftware.

4 For More Help

This document as well as other useful computing information and online copies of the labs can be found on the Stat 215b section homepage at `http://www.stat.berkeley.edu/users/bolstad/Stat215b/`. You should not hesitate to consult me by e-mail or in person if you are having any computing difficulties. I can handle most of the problems with the S215b SCF class accounts.

The Netscape browser is available for accessing the web. If you have a mechanical problem in the computer room, send e-mail to `trouble`. For other computing questions, you can e-mail `consult`, but this may not be immediately answered. Finally, do not forget about the man pages or system help files.