STAT 571A — Advanced Statistical Regression Analysis

Introduction to R — NOTES

Computer Implementation

- **Regression analysis can often be performed by hand calculation, but this is almost always tedious (and sometimes difficult!).**
- **Instead, we turn to the computer. Available "packages" include SAS, SPSS, Stata, Minitab, etc. In STAT 571A, we focus on the (freeware) program 'R'.**

R

The R language has many qualities:

- **possibly the most powerful combination of flexibility and functionality in modern statistical computing**
- **highly effective graphics subsystem**
- **extensive user group creates external "packages" (we will appeal to some of these)**
- **it's free (!): download at**

http://cran.r-project.org

R Overview

- **This short overview introduces R and a few of its features.**
- **Students are expected to develop skills with R separate from the course's core material. (Sorry, but this isn't an R class.)**
- **A recommended source is *The R Guide* by W.J. Owen (version 2.5):**

http://cran.r-project.org/doc/contrib/Owen-TheRGuide.pdf

Command-Line Driven

- One setback with R is that it uses command-line input. (Holdover from the 70's...)
- So, very few "point and click" features.
- Can try the "R-commander" add-on for more of a GUI feel, if desired (see the *Rcmdr* package). Use at own risk.

What is R?

- R is an object-oriented statistical programming environment. (And, a useful tool for understanding statistical thinking...)
- After loading R on your computer, start R by double-clicking the R icon.
- This brings up the CONSOLE WINDOW within which the CURRENT WORKSPACE is accessed.

R Workspace

- To quite R, type

```
> q()
```

  or use mouse clicks on the program GUI.

- Notice the `>` to represent the R input prompt.

- While in R, we essentially are manipulating the “workspace” with the data we enter and analyze.

Data Entry

- The simplest way to enter data in R is via the `c()` (for concatenate) command:

```
> four.numbers <- c(7, 9, 14, pi)
```

- This assigns the four numbers {7, 9, 14, π} to the object (here, a vector) `four.numbers`
- Notice the assignment operator: `<-`
  - sort of like a ‘left arrow’
  - can also use `=`

R commands

- R can act on an object in a variety of ways:

```
> ls()   #lists components of the workspace
> four.numbers/2  #divide all elements by 2
> new.numbers = four.numbers+3    #add 3 to
      all elements & assign to new object
> rm(new.numbers)                 #remove object
```

- It is recommended to keep the workspace clean of unneeded/old objects.

R help

- To get HELP in R, use

```
> help(log)   #get help on log function
```

  or just

```
> ?log
```

- Can also do keyword searching:

```
> apropos("log")   #lists all functions
                       with string "log"
> ??log
```

R Functions

- A **function** in R is an operation (or set thereof) that acts on other R objects.
- There are LOTS of R functions already inbuilt (and, you can always write your own; see Owen's *RGuide*, §8.3)
- E.g., `matrix()` creates matrices (see `?matrix`)

matrix()

```
> a.matrix = matrix(
          c(1,2,3,4,5,6,7,8), nrow=2,
          ncol=4, byrow=FALSE )

> print( a.matrix )
```

will create and display the 2x4 matrix:

```
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
```

Output

- **To print output in R, you can:**
  - **print directly from the R console with "Print" from the File menu (which prints everything in the window...) on Windows or MacOS**
  - **copy portions of your work to a word processor—use Courier font— and edit/print from there (better option!)**
- **To save a workspace, choose "Save Workspace" from File menu (a good idea!)**

A calculator on steroids...

- R can act as a basic calculator, with some extra features. Try these:

```
> 2+3
> 4^2-3*2
> abs(2-4)
> log(0)
> sum(four.numbers)
```

```
> 2^3
> sqrt(2)
> atan(4*pi)
> factorial(6)
> prod(four.numbers)
```

- Can also do matrix arithmetic (Owen, §2.3)

Other Handy Functions

```
> seq(1,9)                    #sequence from 1 to 9
> seq(1,5, by=.5)             #increment by 0.5
> 1:9                         #shortcut for seq(1,9)
> 1.5:10
> rep(9, times=4)             #repeat 9 four times
> rep( c("A","B","C"), 2)                  #can do
                                        characters too
> length( four.numbers)
> round( pi, digits=4 )                    #rounding
```

The I() function

- If you need to operate on a variable within an R formula, the basic arithmetic operations don't work.
- Need to use R's "Inhibit Interpretation" function, I():

```
> X = c(1.1, -2.2, 5.25, pi)
> formula = ( Y ~ X + I(X^2) )
> I(X^2)

[1]  1.2100  4.8400 27.5625  9.8696
```

Entering Data, Redux

Other ways to enter data in R:

- **Scan in keyboard input until 2 consecutive carriage returns are entered:**

```
> scan()
```

- **Scan in from external data file via dialog window:**

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

- **Scan in from external TXT file (notice forward slash):**

```
> scan( “C:/datafile.txt” )
```

Entering Data, Redux

- **Read in formatted data (from TXT file):**

```
> read.table( file.choose(),
                        header=TRUE )
```

- **Read in comma-separated values (from TXT file):**

```
> read.csv( file.choose(),
                        header=TRUE )
```

- **Read in external R source code from TXT file (tricky; see CRAN online manuals):**

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

R packages

To access the many useful (well, *some* are useful) external R packages:

- **First download the package and install it on your hard drive. (May need to select a mirror site.) Do this only once per machine.**
  - → **use `install.packages()` command or use menu options**
- **Once installed, load the package every time you need it in your R session.**
  - → **use `library()` or `require()` or use menu options**

R Graphics

- **R's graphical subsystem is felt by some to be its strongest feature:**
  - **high-quality outputs**
  - **very flexible (if you known all the sub-commands)**
  - **easy to save as PDF, EPS, JPG, TIFF, etc., or just "Print" from File menu**
- **Default is to overwrite every new graphic, so plot device needs careful management.**
- **See Owen's *RGuide* (§4).**