1 Installing R

The main web page for the R system is http://www.r-project.org. From the main R page, you can download a copy of R by clicking on “CRAN” (Comprehensive R Archive Network) and selecting an archive. There are versions of R for Mac OS X, Windows, and Linux. They all work fairly similarly.

Exercise 1 If you have a personal computer, install R. If you do not, verify that you have access to it (for example, in the graduate computing room). For the remaining exercises, provide hardcopy of R code and output, with written comments as requested.

2 R as a Calculator

When you start up R, you will see a “console” with some preliminary information. At the last line of text will be a “>”:

R version 2.13.1 (2011-07-08)
Copyright (C) 2011 The R Foundation for Statistical Computing

[some more information]...

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.

>

The “>” indicates the current line, where you can enter commands to R. R can be used as a calculator. For example, try typing in “1+2” and hitting return. You should see:

> 1+2
[1] 3

For now, ignore the “[1]”. The answer returned by R is 3. Try the following commands, and be sure you understand what is being returned by R:

Exercise 2 Run the following commands in R and print out the results. Explain in words what each line does.
Note: expressions like `log( )` are function calls in R. If you are unsure about a function, you can use the help facility. At the prompt, type

```
> help(log)
```

This will pop up a help page, with information about the function `log` as well as some related functions. The usage of `log` is somewhat complicated:

**Usage**

`log(x, base = exp(1))` ...

Notice that with the log function, you can actually have two inputs. The base is the base for the logarithm. So if you want to take the log in base 10 of the number 100, you could type

```
> log(100, 10)
[1] 2
```

This way of using the log function is a bit dangerous, because you might forget and accidentally switch the base and the number to take the log of. R allows you to specify the inputs by name:

```
> log(100, base=10)
[1] 2
> log(x=100, base=10)
[1] 2
> log(base=10, x=100)
[1] 2
```

This way, it's clear which is the base. Notice also that the usage says `log(x, base=exp(1))` This means that if you do not specify the base, it is taken as `exp(1)`, which is $e = 2.718282...$ We say that the default value for the base is `exp(1)`. There is no default value for `x`, so it must always be supplied when calling `log()`.

**Exercise 3** Here are some other functions. Try them out in R, using the help facility to make sure you understand what they are doing:

`factorial()`
`choose()`
`gamma()`
3 Storing Values in Variables

It's convenient to store values for later use. The general syntax is

\[
\text{variablename} \leftarrow \text{value}
\]

The symbols “\(\leftarrow\)” are meant to look like a leftward-pointing arrow (←). So we could enter:

```r
> x <- 42
```

and that will store the value 42 in a variable named \(x\). To see what value is stored in a particular variable, you can type the variable name and hit return:

```r
> x
[1] 42
```

We can later use \(x\), for example:

```r
> x+2
[1] 44
```

Variable names can be longer, for example:

```r
> this.is.a.valid.variable.name <- 100
```

In addition, the “value” on the right hand side of the assignment can be any expression that returns a numerical value:

```r
> Y <- \text{exp}(1) + x-2
> Y
[1] 42.71828
```

Exercise 4 Using variables, carry out the following computation in R and display the values of each variable:

\[
\begin{align*}
  w &= 4 \\
  x &= 2.4 \\
  y &= w^x - \log(x) \\
  z &= x + y - w
\end{align*}
\]

4 Defining Functions

You can define your own functions in R. Here is a simple example:

```r
> square <- function(x) x^2
```

This defines \(\text{square}\) as a function that takes input \(x\), and returns \(x\) squared. After defining the function, we can type in

```r
> square(4)
```

Note that the input(s) to a function can be a specific value (like 4), a variable, or an expression that returns a value:

```r
> j <- 2
> square(j)
[1] 4
> square(j+3)
[1] 25
> square(exp(1))
[1] 7.389056
```

We can also have functions that take more than one input:

```r
> add <- function(a,b) a+b
> add(23,10)
[1] 33
```

We can specify default values for the inputs:

```r
> add2 <- function(a,b=10) a+b
> add2(23,10)
[1] 33
> add2(23)
[1] 33
```

Finally, we can have the function be quite complex. If the function requires a series of calculations, we enclose them with curly braces `{ }`, and the last value calculated is returned as the output of the function.

```r
> myfun <- function(x,y){
  a <- x^2
  b <- y^3
  a+b
}
```

This defines a function that first calculates the square of \( x \), and puts it in the variable \( a \), then calculates \( y \) cubed, and puts it in \( b \), then adds the two. Since \( a+b \) is the last line, this is what the function returns as output. One thing to note is that the variables \( a \) and \( b \) are defined within the function. These variables are “local” to the function, and are not accessible outside of the function.

**Exercise 5** Define the following functions in R, and evaluate them at the values given.

1. \( f(x) = x^2 + 2x - 1 \), evaluated at \( x = 0, 1, 10 \).
2. \( f(x) = (2\pi)^{-1} \cdot \exp \left( -x^2 / 2 \right) \), evaluated at \( x = -1.96, 0, 1.96 \).
3. \( f(x) = \log(\log(x)) \), evaluated at \( x = 1, e, 4 \).
5 Quitting R

To quit R, you can use the menu item or the standard keyboard shortcut (on the Mac, command-Q). You will be asked whether you wish to “Save workspace image?” If you choose “yes,” R will save the current state (variables, function definitions) so they will be available the next time you start R. For now, you can choose not to save the workspace image.

6 More Information about R

There are many books and web resources for learning R. Here are a few recommended ones:

- Introduction to R at CRAN:
  http://cran.r-project.org/doc/manuals/R-intro.html

- R Tutorial at cyclismo.org:
  http://www.cyclismo.org/tutorial/R/

- R Links from John Bear:
  http://www.johnbear.net/JohnBearsRCchecklist.html

- Braun and Murdoch, A First Course in Statistical Programming with R:
  Amazon

- Adler, R in a Nutshell:
  Amazon