Kalamazoo College **Fall Quarter**
Math 362 **2008**

Getting Started With R

Goals

After completing this lesson, you will be able to:

- **Get data into R by several different methods.**
- **Use simple R functions.**
- **Use R to obtain function plots**
- **Modify and execute basic loops in R.**
- **Access data stored in R workspaces.**
- **Setting the working directory.**

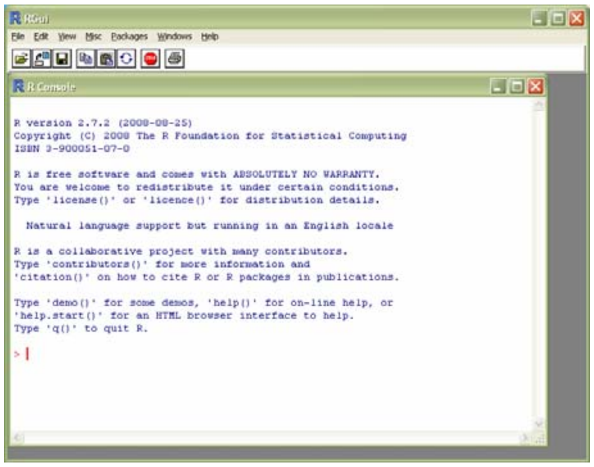
1. Preamble

For this course, we will be using the R statistical software package to explore some properties of probability distributions. R is both statistical software and a statistical programming language. We will make only very limited use of its programming capabilities. R can also be used to carry out simulation studies.

R is freeware and you are welcome to download and install the latest version on your own personal computer. Visit http://cran.us.r-project.org/ to download and follow the instructions at the website. You will most likely want to download the "Precompiled Binary Distributions" for whatever operating system you use (Windows, Linux, or Mac). You can find more links related to R at the web page for this class, http://kzoo.edu/enordmoe/math362/.

2. Beginning and Ending an R Session

Before doing any data analysis, you will need to know how to get started and how to end your session. You start an R session on one of the lab machines (or your own) just as you do most other Windows programs. Click on the "R" button on the R section of the Start…Program menu. You should see the following with a command line prompt, ">" waiting for commands from you:[1]



```
R version 2.7.2 (2008-08-25)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

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.

> |
```

In case you forget, R gives you a friendly reminder each time you start that you can quit by typing "`q()`".Why the closed parentheses, "`()`" you ask? Wouldn't "q" suffice? It turns out that R is comprised of functions typically followed by arguments enclosed in parentheses. Omitting these arguments and typing only the command itself will cause R to display the "code" executed when you request the corresponding procedure. If you're curious, go ahead and try typing "q" without the parentheses.

3. Getting Help

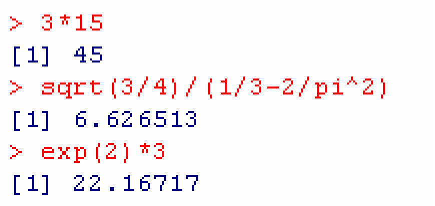
Now that you know how to get in and out of R without getting hurt, you know just enough to be dangerous. When in trouble, remember the pull-down Help menu items. The "Html help" and "Search help…" are very useful. In particular, the PDF Manual "An Introduction to R" available from the Help menu may be useful as you are getting acquainted with R. The R Data Import/Export manual is also extremely helpful. You can get help about any specific command in a

[1] This is an example of the default MDI (multiple display interface) in R. That is, by default R opens any plots or editor windows inside one parent window. The interface can be changed to use separate windows (SDI) through the GUI preferences dialog box under the Edit menu item.

variety of ways. For example, you could get help about the R function "`mean()`" by typing any of the following at the command line prompt: `help(mean)` or `?mean`. Another useful feature is to request only examples of the use of a function: `example(mean)`

4. A Glimpse of R

R may be unlike other software packages you have used in that it treats a variety of items you'll work with—data, plots, functions and even results—in the same way as *objects*. The implications of this may not be apparent for awhile, but it turns out that this is very powerful when using R as a statistical programming language. To get a glimpse at how works, take a look at the following brief session. The character ">" represents the command prompt.



```
> 3*15
[1] 45
> sqrt(3/4)/(1/3-2/pi^2)
[1] 6.626513
> exp(2)*3
[1] 22.16717
```

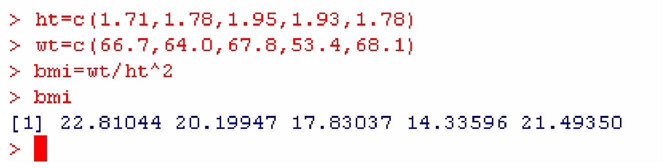
Note that R uses the standard operators for multiplication ("*") and exponentiation ("^") and contains built-in functions (e.g., `sqrt()` and `exp()`, the exponential function) and built-in constants ("`pi`" represents the famous number 3.1415…).

5. Getting Data Into R

There are many, many ways of getting data into objects with R. In this section, I illustrate just a few.

`c()`

A versatile, if somewhat awkward way of entering a list of data in R is to use the "concatenate" command, abbreviated c() and illustrated below:

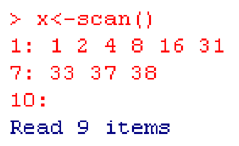


```
> ht=c(1.71,1.78,1.95,1.93,1.78)
> wt=c(66.7,64.0,67.8,53.4,68.1)
> bmi=wt/ht^2
> bmi
[1] 22.81044 20.19947 17.83037 14.33596 21.49350
> 
```

Note that the characters "<-" form the assignment operator in R, assigning the value of the expression on the right to the object, e.g., "ht" on the left. Simply using "=" instead of the "`<-`" assignment operator is also allowed. The commands input the heights (in meters) and weights (in kg) of 5 subjects and use R operators to compute the body mass index (BMI) for each. Note that R automatically uses element-by-element operations when working with vectors like these.

`scan()`

Using the command `scan()` lets you enter data into a vector or list at the command line. You can use `scan()` to create a list of values "x" as follows:



```
> x<-scan()
1: 1 2 4 8 16 31
7: 33 37 38
10:
Read 9 items
```

This command lets you continue entering values in a single list until you either leave a blank line or type Ctrl-Z at the beginning of a new line. In another version of this command, you can simply type `x<-scan(),` hit return, and then paste in a column of values from a spreadsheet package such as Excel. This is often a very good way to input a list of numbers.

`seq()`

A very simple and very useful function is `seq()` which can be used to create a sequence of numbers. For example `seq(1,10)` gives the integers from 1 to 10. Convenient shorthand for this command is to simply use 1:10 so that you could input `x=1:10` at the command line to create the same vector of integers from 1 to 10. Alternatively, a third argument may be given to generate a sequence with increments of fixed size as shown in the following display:

```
> seq(0,20,4)
[1]  0  4  8 12 16 20
```

6. Plotting

To get a quick taste of some simple plotting features of R, try the following

```
> xvar=seq(0,1,.01)
> yvar=xvar^2
> plot(xvar,yvar,type="l")
> for(j in 2:8) lines(xvar,xvar^j)
> for(j in seq(.25,.75,.5)) lines(xvar,xvar^j,col=j)
```

The first two lines simply generate a vector and plot the variable xvar on the horizontal axis and yvar on the vertical axis (i.e., plot the function $f(x) = x^2$)[2]. Specifying type="l" (where "l" is the letter "l" not the number "1") ensures that the graph is drawn as a line. The last two lines use the "for" loop command in R to add power curves to the current plot. Use of the function `lines()` requests that R add the curves to the existing plot rather than create a new plot. The for loop will be a very useful tool for investigating the effect of parameter changes on the plots of probability functions. I will give more examples of its use on future assignments so that you will know how to employ it for the specific tasks at hand. Notice that the variable j is used to assign a unique color (2 through 8) to each line.

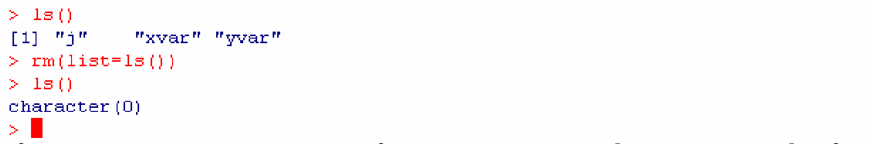
In addition to `plot()` and `lines()`, a useful function for creating graphs of functions is `curve()`. For example, to create the graph of $f(x) = e \,\hat{}\, x$ on the interval [-2,2], one simply uses the command `curve(exp(x),-2,2)` where the -2 and 2 are the lower and upper limits on the x-axis. Again, you may wish to check the help page for further documentation on this function.

7. A Little Probability

A few functions that will be useful in probability are the `combn()`, `prod()`, `choose()` and `factorial()` functions. Use the help functions described earlier to investigate the syntax and use of these functions.

8. Listing and Removing Objects

One problem with R is that it tends to be a bit of a hoarder. Everything that is created gets stored permanently on disk unless you tell it not to. The following brief session uses `ls()` to list the objects in your current local environment (session). The `rm(list=ls())` command gets rid of **everything** so be very careful when you use it. Using `ls()` again confirms everything's "gone."



```
> ls()
[1] "j"    "xvar" "yvar"
> rm(list=ls())
> ls()
character(0)
> 
```

If you want to save a transcript of your session, just select "Save to File" from the File menu in R. "Save Workspace" is also very useful. It will save all the objects (including data frames) used in your session so that you can load it again next time without re-creating everything.

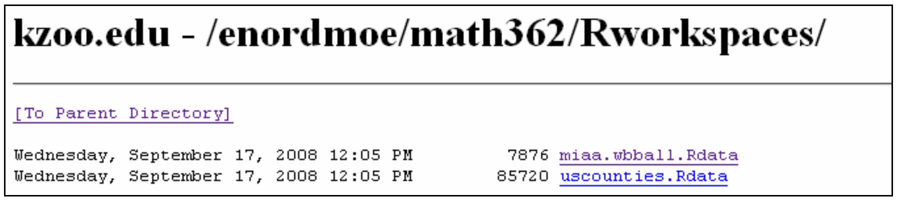
9. R Workspaces

A very useful way of storing data sets (called data frames in R) is in the form of an R *workspace*, saved with the extension `.Rdata`. Occasionally I will make data sets available to you in this format for analysis. This will save you the trouble of keying in the data or loading it from another source, e.g., a spreadsheet file.

To successfully access an R workspace I have made available through my website, R must be installed on the machine you are using. Hence, to access these workspaces on your own computer, you must first have successfully complete installation of R. *Assuming R has been installed on your computer*, the following steps illustrate how to access an R workspace.

First, open your browser to the R workspace sub-directory on my website: http://kzoo.edu/enordmoe/math362/Rworkspaces/. A sample image is shown below:

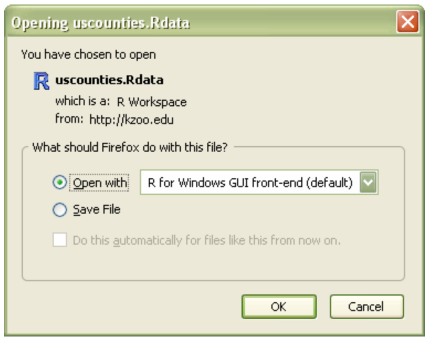
[2] Note that there is nothing unique about the names xvar and yvar. You could certainly use other variable names, as long as they don't include spaces or certain special characters (though periods are fine, e.g., x.var and y.var.



kzoo.edu - /enordmoe/math362/Rworkspaces/

```
[To Parent Directory]

Wednesday, September 17, 2008 12:05 PM         7876 miaa.wbball.Rdata
Wednesday, September 17, 2008 12:05 PM        85720 uscounties.Rdata
```

To open the R workspace containing statistics for all US counties, click on `uscounties. Rdata`. Your browser should recognize this as an R workspace and prompt you to open it in R.



Opening uscounties.Rdata

You have chosen to open

uscounties.Rdata

which is a: R Workspace

from: http://kzoo.edu

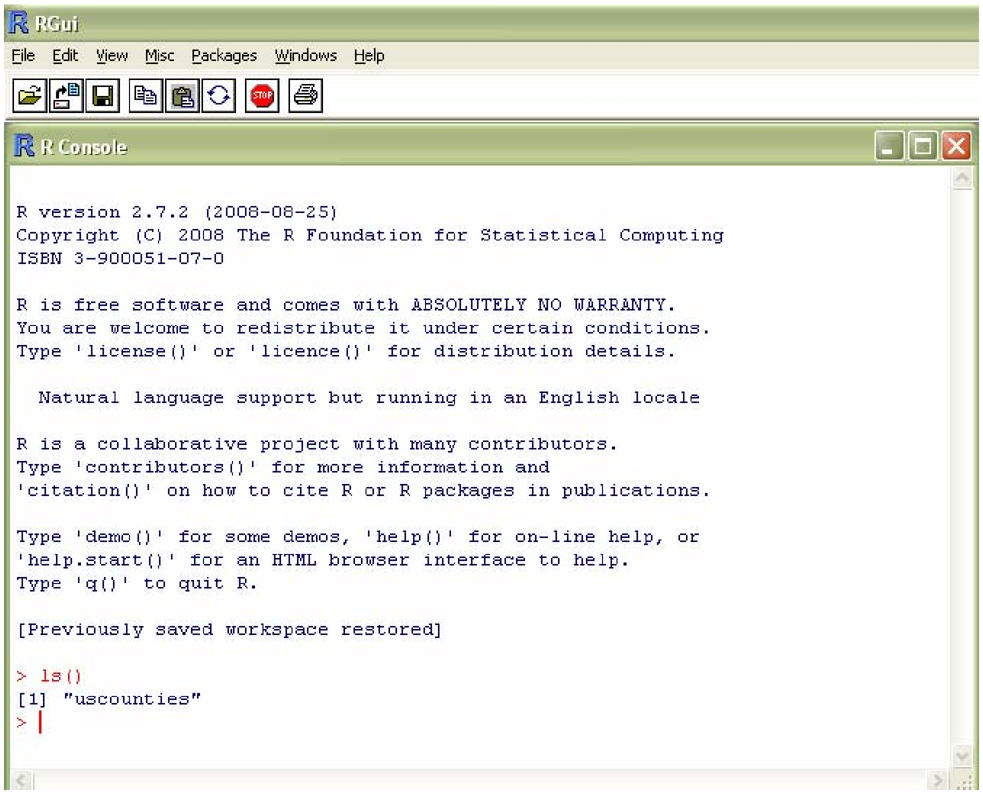
What should Firefox do with this file?

[✓ Open with] R for Windows GUI front-end (default)

Save File

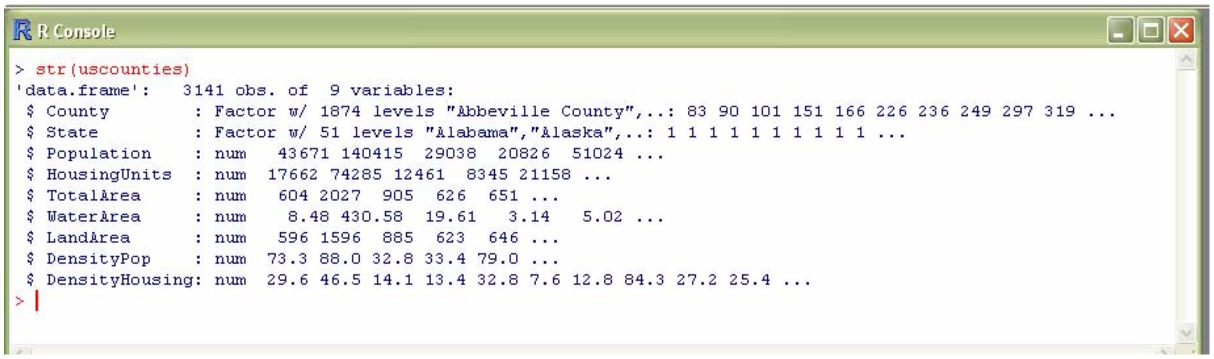
Do this automatically for files like this from now on.

Click OK and R should open with the workspace already loaded as in the figure below. The command `ls()` displays the contents of the workspace, in this case the *data frame* called `uscounties`, the same name given to the workspace.



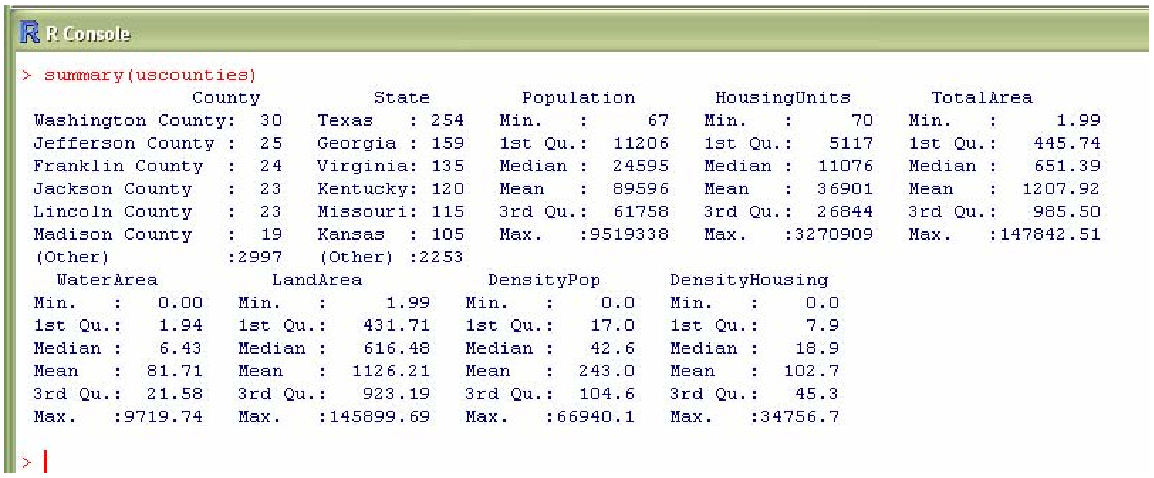
```
R version 2.7.2 (2008-08-25)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

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.

[Previously saved workspace restored]

> ls()
[1] "uscounties"
> 
```

To see the contents of the data frame itself, the commands `str()` and `summary()` are useful as shown below:



```
> str(uscounties)
'data.frame':   3141 obs. of  9 variables:
 $ County        : Factor w/ 1874 levels "Abbeville County",..: 83 90 101 151 166 226 236 249 297 319 ...
 $ State         : Factor w/ 51 levels "Alabama","Alaska",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ Population    : num   43671 140415  29038  20826  51024 ...
 $ HousingUnits  : num  17662 74285 12461  8345 21158 ...
 $ TotalArea     : num   604 2027  905  626  651 ...
 $ WaterArea     : num    8.48 430.58  19.61   3.14   5.02 ...
 $ LandArea      : num   596 1596  885  623  646 ...
 $ DensityPop    : num  73.3 88.0 32.8 33.4 79.0 ...
 $ DensityHousing: num  29.6 46.5 14.1 13.4 32.8 7.6 12.8 84.3 27.2 25.4 ...
>
```

The `str()` command displays the variables and types (e.g., Factor or numeric) contained in the data frame. Note that this data frame is comprised of 3141 observations (counties) and 9 variables (characteristics for each county). The variables are preceded by a $ sign indicating that they can be accessed directly using the format `uscounties$County` or `uscounties$LandArea`.[3] The numbers to the right are the first few values of the variable. This command just provides basic information about the data frame; it does not give a statistical summary. For that purpose, the command `summary()` is useful:



```
> summary(uscounties)
              County           State         Population       HousingUnits       TotalArea
 Washington County:  30   Texas   : 254   Min.   :     67   Min.   :     70   Min.   :     1.99
 Jefferson County :  25   Georgia : 159   1st Qu.:  11206   1st Qu.:   5117   1st Qu.:   445.74
 Franklin County  :  24   Virginia: 135   Median :  24595   Median :  11076   Median :   651.39
 Jackson County   :  23   Kentucky: 120   Mean   :  89596   Mean   :  36901   Mean   :  1207.92
 Lincoln County   :  23   Missouri: 115   3rd Qu.:  61758   3rd Qu.:  26844   3rd Qu.:   985.50
 Madison County   :  19   Kansas  : 105   Max.   :9519338   Max.   :3270909   Max.   :147842.51
 (Other)          :2997   (Other) :2253
   WaterArea          LandArea           DensityPop       DensityHousing
 Min.   :   0.00   Min.   :     1.99   Min.   :    0.0   Min.   :    0.0
 1st Qu.:   1.94   1st Qu.:   431.71   1st Qu.:   17.0   1st Qu.:    7.9
 Median :   6.43   Median :   616.48   Median :   42.6   Median :   18.9
 Mean   :  81.71   Mean   :  1126.21   Mean   :  243.0   Mean   :  102.7
 3rd Qu.:  21.58   3rd Qu.:   923.19   3rd Qu.:  104.6   3rd Qu.:   45.3
 Max.   :9719.74   Max.   :145899.69   Max.   :66940.1   Max.   :34756.7

>
```

For categorical variables, the summary command gives the number of times the most frequent values occurred while for quantitative variables it gives the 5-number summary (min, 1st quartile, median, 3rd quartile, and maximum) and the mean. Other functions for more detailed data description are available in R but these provide a quick summary of the information in an R data frame.

10. R Working Directory and Shortcuts

One feature of the default R installation of which you should be aware is that, by default, it keeps anything you save in the installation directory or a temp directory. You may find it more convenient to make the working directory something easier to find like `C:\Documents and Settings\myname\My Documents\Rstuff`. To do so, simply choose `Change Dir`… from the `File` menu within R. Specify the working directory you would like to use and proceed. Anything saved will, by default, be stored in the working directory.

One option you may find convenient is to modify the startup working directory for R using the shortcut icon created by the installation program. To do so, right-click on the R icon, and select properties to open the properties dialog box. Change the "Start in" directory to `"C:\Documents and Settings\myname\My Documents\Rstuff"` or whatever you wish the working directory to be. Be sure that you have correctly entered the full directory name. Click OK and the default working directory will be retained for future use.

[3] To eliminate the need to use the dataframe_name$variable_name convention to access variables, first give the command attach(dataframe_name). After so doing, individual variables can be accessed directly by their names, e.g. County, Stats, Population, etc.

11. A Disclaimer

One final word. Unfortunately, R and most of its documentation is written for people who already know probability and statistics so you may find it difficult to penetrate. Please ask me if you have questions that you can't readily solve on your own. I want R to be a useful tool to aid your understanding rather than another obstacle to conquer. Links to pages with useful R resources are on my web page.