Tutorial 2: Reading and Manipulating Files

Jason Pienaar and Tom Miller

Most of you want to use R to analyze data. However, while R does have a data editor, other programs such as excel are often better for entering and handling data before you start analysis with R. So, before we go any further, we want explain the best ways to efficiently get your data into R from files created by other programs.

Three Steps for Reading Files or Data Into R:

1. Preparing the data file in Excel (or other program)

We are going to focus on 4 simple ways to get files or data into R. But, before we do this, we are going to create a very simple data set in Excel to use for the rest of the exercise. R supports rectangular data sets (matrices) in the form of "data frames" where the rows represent observations or measurements and the columns represent variables. The data frame is the object that we will use most often to read in and store data to be analyzed. Open Excel and create a file with 3 columns labeled "color", "ear length", and "tail wags". For color enter: brown, brown, white, white, white. For ear length, enter: 10, 9, 5, 8, 2. For tails wags, enter: 23, . 5, 16, 3 (the second value is a period used to represent a missing value).

Now, we want to get this data into R. We intentionally started with a file that R cannot read. It has a couple of common problems that need to be fixed before any file can be brought into R.

-- Variable names (the column names) must follow the R rules.
-- Missing values must be converted to a form that R recognizes.

Remember the rules for naming variables given in Tutorial 1 (variable names are composed of letters, numerals, and periods (.) and spaces are not allowed). So, change the variable names from "ear length" to "ear.length" and from "tail wags" to "tail.wags".

Missing values are frequently a source of problems for different analysis and graphing programs. The simplest way to insure that R recognizes missing values is to convert them to "NA". Doing so, should now leave you with a file that looks something like:

color	ear.length	tail.wags
brown	10	23
brown	9	NA
white	5	5
white	8	16
white	2	3

2. Changing the Working Directory:
First however, we need to somehow tell R which directory we are using. When using R, it is much easier to have all the files you are going to use in one folder or directory.

If you are using a PC: click on *file* in the top left hand corner, select *change dir...* and *browse* in order to find the directory in which the data file was saved. (Note: it might be a good idea to create a folder to store data sets in, one that is easy to find).

If you are using a Mac, click on *Misc* on the top menu bar and pull down *Change Working Directory*. An even easier way is to find the folder with your data files, then grab the folder and move it over the R icon in your applications folder. If R is not running, it will start R with the appropriate default. If R is running, it will simply change the default folder. (A last option is to use the "setwd" function in R (see *help*(setwd))).

3. Reading the Data into R:
Here are the 4 ways to enter this data into R. In R, each data set will be referred to as a "data frame" and be given a name – in our example, we will create the data frame "dog".

1. The simplest type of file to read into R is a plain text file created in notepad. However, most of us don't use simple text files, but instead use something like Excel for entry and database management. Standard Excel files (".xls") have lots of extra hidden coding that is necessary for Excel, but makes it difficult for other programs to read these files. We could cut and paste the Excel file data into a text editor, but this would be involving a 3rd program as a translator between two others, which is getting a tad ridiculous. So, let's consider some more practical steps.
2. Most programs can write the output as text files. Then, the text files can be read straight into R. R simply requires that there be "white space" between the variables in each row of the data: this may be tabs or actual spaces. So, simply use "Save As . . ." to save the data file from Excel as "Text (tab delimited)" and name it "dog.txt". For some text editors, you may need to save the data as "Text Only with Line Breaks". Then, the file can be read into R using the following commands (we will call the file "dog"):

   ```
   > dog = read.table("dog.txt", header=T)
   > attach(dog)
   > dog
   ```

   The first command finds the file and recognizes that there are labels ("headers") for each variable. The "T" stands for TRUE and must be capitalized. The second "attach" command tells R to use the headers as local variable names, which is both useful and necessary. The third command ("dog") simply causes the file to be written out. The datafile will now be available for use in R.
3. In a database manager such as Excel, use "Save As . . ." and choose the CSV

(Comma delimited) option. This will save the file as “dog.csv”. Then, read dog.csv into R using:

```
> dog=read.csv(“dog.csv”,header=T)
> attach(dog)
> dog
```

Again, the datafile will now be available for use in R.

4. Finally, my favorite. You can simply copy data from Excel and read the memory buffer right into R. So, while in Excel, simply highlight all the columns and rows that you want to read into R and use the “Copy” command, usually under “Edit”. Then open R and use the following command (here it differs between PC and Mac)

   For PC:
   ```
   > dog = read.table(file("clipboard"), header = T)
   ```

   For Mac:
   ```
   > dog = read.table(pipe(“pbpaste”), header=T)
   ```

   then, for either PC or Mac:

   ```
   > attach(dog)
   > dog
   ```

Editing using the R data frame

R has a spreadsheet like editor into which we can type variable names and data values. This editor is useful primarily for viewing the contents of data frames or for modifying individual values. It is far faster and easier to read the data into R using the *read.table* function as described above and we will always do this but sometimes we may need to modify individual values in the data frame (e.g. if it was typed incorrectly in the first place, or is identified as a statistical outlier). An existing data frame can be viewed or edited by using the *fix* function.

```
>fix(data1)
```

where data1 is the name of your data frame. This function opens your data frame in the data editor and individual values can be modified by double clicking in the cell that contains the datum. You should then use *detach*(file1) to remove the existing data frame and then re-attach the data frame with *attach*(file1). The modified values are now the ones on which R operates.

Summary

Often, the most frustrating part of using any program is trying to figure out how to get your data files into the program so that you can start analyzing it. The point of the above was to provide a simple demonstration of how to do this in R. There are many other ways to do this, like reading in data directly from *Excel* or some other statistics program, but the procedures above works every time and is by far the simplest. Here is a quick summary of how to get your data into R and make the variables available for further analyses:

1. Get your data into a notepad text file, excel or word, making sure that it is in correct R format for dealing with variable names and missing values.
2. Open R and make sure you have the working directory set correctly.
3. Use something like:

   > myfile = *read.table*("filename.txt",header=T)
   > *attach*(myfile)

Some Exercises (again, some answers are provided at the end)

Some of these exercises go beyond what we have covered in the tutorials. We are intentionally trying to get you to figure out R on your own.

1. Use Excel to create a rectangular data set with 2 columns, the first which contains 200 random numbers under the heading *knock.knock* (e.g., in box A1, type "knock.knock", then in box A2 type "=Rand( )", hit return, then drag the function down across 200 cells). In the second column, under the heading *whos.there*, add the values in the first column to a second set of random numbers (e.g. in cell B2, type "=(a2 +rand( ))", then drag down across 200 cells matching those in column 1).
2. Save this as a text file (Text (tab delimited)) named *corrie.txt*.
3. Open R and create a data frame of the data in corrie.txt named *corrie.who*.
4. Let's use this data a bit. Before we can use the data, make sure you have used the *attach* function, because this allows us to now just use column names. To create a scatter plot of the two variables in your file, use the *plot* function.

   >plot(variable1,variable2)

5. I guess one could name this scatter plot corrie.lation (this is a very sad joke and should not be taken too seriously). One can label axes and the graph itself in a variety of ways. Try using the "help(plot)" function to figure out how to correctly label the axes and create a title for the graph.
6. Create a histogram of the *whos.there* variable using the function *hist*.

Answers or Hints for selected problems:

3. >corrie.who = read.table("corrie.txt", header=T)
   >attach(corrie.who)
4. >plot(knock.knock,whos.there)
5. >plot(knock.knock,whos.there, main="corrie.lation")
6. >hist(whos.there)