Getting Data into R

The most common source of "real" data is a csv file, for "comma separated values". The simplest kind of csv file is composed of data in columns, each of which has a header or name. If you open a csv file in Excel, it will look like a series of columns with headers. If you open a csv file with a simple text editor, each row will be composed of number entries separated by, you guessed it, commas (Excel overrides this simple display and formats it spreadsheet style). For now, let's assume someone has provided you with an appropriate csv file.

```
mydata <- read.csv("file name.csv")
str(mydata)
```

The first command above will look for the file `file.name.csv` in your working directory, and create an object called `mydata` in the process (the contents of the file, after parsing, are assigned to the variable `mydata`). The second (optional) command gives you the structure of `mydata`. To understand what `str()` tells you, you'll need to understand data types or modes. I mention it here because it is a good way to verify your data was read in correctly.

What if you need to create your own csv file, or you were given an xls or xlsx file that doesn't work with R? There are some important things to watch out for, some arising from Microsoft's ego. Here are some pointers:

1. Create or edit your Excel spreadsheet to organize your data in columns. Include a header row which has short, descriptive names that don't contain mathematical operators. For example, use `net.income` not `net-income`. This doesn't affect the reading of the data in the file, but the column headers become the variable name for later use. `net-income` will eventually be misinterpreted by R as subtract `income` from `net` and of course you intend no such thing! Make certain there is no other information in your file, only the header row and the data.
2. Now save your file using Excel's "Save as..." command. Select the format to be csv. Be careful: when you go to close the file, even though you just saved it, you will be asked if you want to save it. Say "no" and just close the file. Excel is thinking you would be an illogical twit to leave it as csv because this format doesn't have all the "features" Microsoft intends for you to use. Microsoft is really just asking you if you would rather save it as the "much superior" xls or xlsx format. If you save it when asked, the file will be converted back to xls or xlsx format, which is not what you want.
3. Now the file should be ready to be read into R as described above using `read.csv`.
4. If you will be doing a lot of interaction with Excel, particularly large data sets provided by others, there are packages in R which will read native Excel format correctly, but you will have to invest time to learn how to do it.

R also has some built in data sets. These are accessed by the `data` function:

```
data(name.of.data.set)
```

after which there will be an object of that name available to you in your workspace. You can see what's in it, and the corresponding data types (modes), with `str()`. Try this:

```
data(Puromycin) # loads the data set
str(Puromycin) # give the structure of Puroymcin
```

```
## 'data.frame': 23 obs. of  3 variables:
##  $ conc : num  0.02 0.02 0.06 0.06 0.11 0.11 0.22 0.22 0.56 0.56 ...
##  $ rate : num  76 47 97 107 123 139 159 152 191 201 ...
##  $ state: Factor w/ 2 levels "treated","untreated": 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "reference")= chr "A1.3, p. 269"
```

```
# ?Puromycin # calls the help page, which gives background
```

Finally, you can always enter or create the data by hand, if not too much is needed. See *Baby Steps with R* for ways to create random data or sequences. If you must enter data by hand, you could use the `> my.data <- c(...)` method. For quite a few values, you can use `> my.input <- scan()` which prompts you to put in one number per line, followed by a return. When you are done, just input a blank line and the data will be in `my.input`.