Data Import

It is often necessary to import sample textbook data into R before you start working on your homework.

Excel File

Quite frequently, the sample data is in Excel format, and needs to be imported into R prior to use. For this, we can use the function read.xls from the gdata package. It reads from an Excel spreadsheet and returns a data frame. The following shows how to load an Excel spreadsheet named "mydata.xls". This method requires Perl runtime to be present in the system.

```
> library(gdata)                  # load gdata package
> help(read.xls)                  # documentation
> mydata = read.xls("mydata.xls")  # read from first sheet
```

Alternatively, we can use the function loadWorkbook from the XLConnect package to read the entire workbook, and then load the worksheets with readWorksheet. The XLConnect package requires Java to be pre-installed.

```
> library(XLConnect)              # load XLConnect package
> wk = loadWorkbook("mydata.xls")
> df = readWorksheet(wk, sheet="Sheet1")
```

Minitab File

If the data file is in Minitab Portable Worksheet format, it can be opened with the function read.mtp from the foreign package. It returns a list of components in the Minitab worksheet.

```
> library(foreign)                # load the foreign package
> help(read.mtp)                  # documentation
> mydata = read.mtp("mydata.mtp")  # read from .mtp file
```

SPSS File

For the data files in SPSS format, it can be opened with the function read.spss also from the foreign package. There is a "to.data.frame" option for choosing whether a data frame is to be returned. By default, it returns a list of components instead.

```
> library(foreign)                # load the foreign package
> help(read.spss)                 # documentation
> mydata = read.spss("myfile", to.data.frame=TRUE)
```

Table File

A data table can resides in a text file. The cells inside the table are separated by blank characters. Here is an example of a table with 4 rows and 3 columns.

```
100   a1   b1
200   a2   b2
300   a3   b3
400   a4   b4
```

Now copy and paste the table above in a file named "mydata.txt" with a text editor. Then load the data into the workspace with the function read.table.

```
> mydata = read.table("mydata.txt")  # read text file
> mydata                             # print data frame
   V1 V2 V3
1 100 a1 b1
2 200 a2 b2
3 300 a3 b3
4 400 a4 b4
```

For further detail of the function read.table, please consult the R documentation.

```
> help(read.table)
```

CSV File

The sample data can also be in comma separated values (CSV) format. Each cell inside such data file is separated by a special character, which usually is a comma, although other characters can be used as well.

The first row of the data file should contain the column names instead of the actual data. Here is a sample of the expected format.

```
Col1,Col2,Col3
100,a1,b1
200,a2,b2
300,a3,b3
```

After we copy and paste the data above in a file named "mydata.csv" with a text editor, we can read the data with the function read.csv.

```
> mydata = read.csv("mydata.csv")  # read csv file
> mydata
  Col1 Col2 Col3
1  100   a1   b1
```

```
2  200   a2   b2
3  300   a3   b3
```

In various European locales, as the comma character serves as the decimal point, the function read.csv2 should be used instead. For further detail of the read.csv and read.csv2 functions, please consult the R documentation.

```
> help(read.csv)
```

Working Directory

Finally, the code samples above assume the data files are located in the R working directory, which can be found with the function getwd.

```
> getwd()               # get current working directory
```

You can select a different working directory with the function setwd(), and thus avoid entering the full path of the data files.

```
> setwd("<new path>")   # set working directory
```

Note that the forward slash should be used as the path separator even on Windows platform.

```
> setwd("C:/MyDoc")
```

Exporting Data

There are numerous methods for exporting R objects into other formats . For SPSS, SAS and Stata, you will need to load the foreign packages. For Excel, you will need the xlsReadWrite package.

To A Tab Delimited Text File

```
write.table(mydata, "c:/mydata.txt", sep="\t")
```

To an Excel Spreadsheet

```
library(xlsx)
write.xlsx(mydata, "c:/mydata.xlsx")
```

To SPSS

```
# write out text datafile and
# an SPSS program to read it
library(foreign)
write.foreign(mydata, "c:/mydata.txt", "c:/mydata.sps",   package="SPSS")
```

To SAS

```
# write out text datafile and
# an SAS program to read it
library(foreign)
write.foreign(mydata, "c:/mydata.txt", "c:/mydata.sas",   package="SAS")
```

To Stata

```
# export data frame to Stata binary format
library(foreign)
write.dta(mydata, "c:/mydata.dta")
```