Introduction to R Software

Data Handling

::::

Importing Data Files of Other Software and Redirecting Output

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Importing Data Files

Spreadsheet (Excel) file data

The `xlsx` package has the function `read.xlsx()` for reading Excel files.

This will read the first sheet of an Excel spreadsheet.

To read Excel files, we first need to install the package

```
install.packages("xlsx")
library(xlsx)
data <- read.xlsx("datafile.xlsx", Sheet Index
or Sheet Name )
```

Importing Data Files

Spreadsheet (Excel) file data

To load other sheets with `read.xlsx()`, we specify a number for `sheetIndex` or a name for `sheetName`:

```
data <- read.xlsx("datafile.xlsx", sheetIndex=2)
```

```
data <- read.xlsx("datafile.xlsx",
sheetName="marks")
```

Importing Data Files

Spreadsheet (Excel) file data

For reading older Excel files in .xls format, use `gdata` package and function `read.xls()`

This will read the first sheet of an Excel spreadsheet.

To read Excel files, we first need to install the package

```
install.packages("gdata")
library(gdata)
data <- read.xls("datafile.xls", Sheet Index or
Sheet Name ))
```

Importing Data Files

SPSS data file

For reading SPSS data files, use `foreign` package and function `read.spss()`

To read SPSS files, we first need to install the package

```
install.packages(" foreign ")
library(foreign)
data <- read.spss("datafile.sav")
```

Importing Data Files

Other data files

The `foreign` package also includes functions to load from other formats, including:

- **`read.octave("<Path to file>"): Octave and MATLAB`**
- **`read.systat("<Path to file>"): SYSTAT`**
- **`read.xport("<Path to file>"): SAS XPORT`**
- **`read.dta("<Path to file>"): Stata`**

Importing Data Files

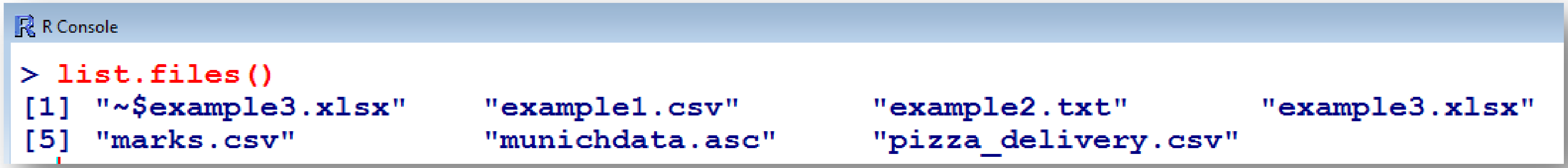
More description of data import and export can be found in the respective R manual at

`http://cran.r-project.org/doc/manuals/r-release/R-data.pdf`

Contents of working directory

The `list.files` function shows the contents of your working directory:

```
> list.files()


> setwd("C:/RCourse/")
> list.files()
[1] "~$example3.xlsx"    "example1.csv"       "example2.txt"
"example3.xlsx"
[5] "marks.csv"          "munichdata.asc"
"pizza_delivery.csv"
```



R Console

```
> list.files()
[1] "~$example3.xlsx"    "example1.csv"       "example2.txt"       "example3.xlsx"
[5] "marks.csv"          "munichdata.asc"     "pizza_delivery.csv"
```

Redirecting Output to a File

Issue:

We want to redirect the output from R into a file instead of your console.

Solution:

Redirect the output of the `cat` function by using its file argument:

```
> ans <- 6 + 8
> cat("The answer of 6 + 8 is", ans, "\n",
file="filename")
```

The output will be saved in the working directory with given filename

Redirecting Output to a File

Use the `sink` function to redirect all the output from both `print` and `cat`.

Call `sink` with a filename argument to begin redirecting console output to that file.

When we are done, `sink` with no argument to close the file and resume output to the console:

```
> sink("filename") #Begin writing output to file

. . . other session work . . .
> sink()
```

Redirecting Output to a File

The `print` and `cat` functions normally write the output to console.

The `cat` function writes to a file if we supply a file argument.

The `print` function cannot redirect its output.

The `sink` function can force all output to a file.

Redirecting Output to a File: Three steps

1.

```
> sink("output.txt") # Redirect output to file
```

2.

```
> source("script.R") # Run the script, capture
                       its output
```

3.

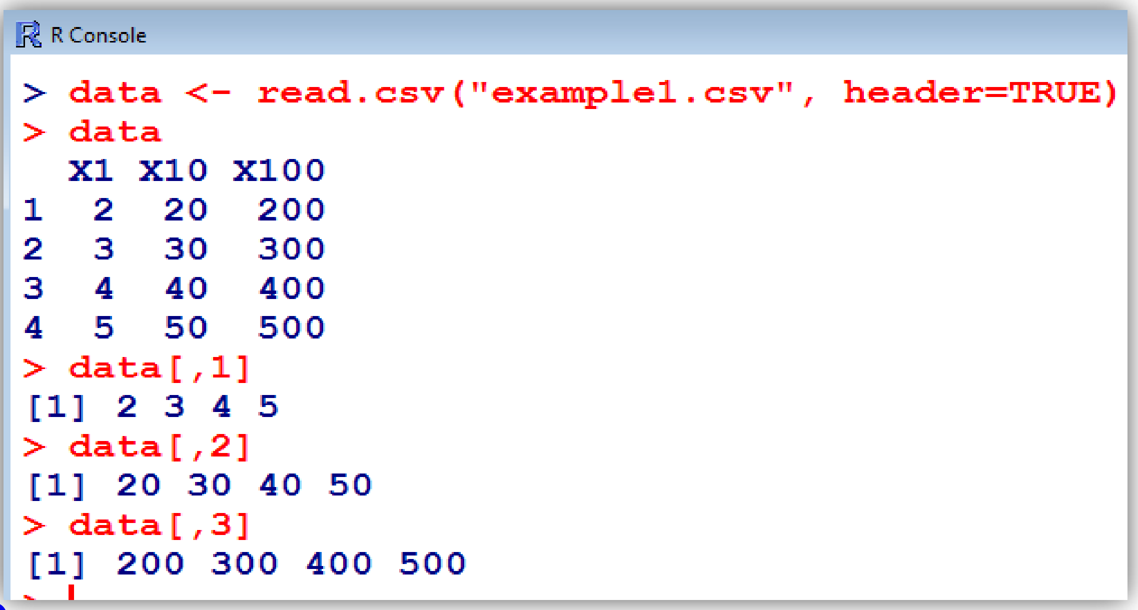
```
> sink() # Resume writing output to console
```

Other options like `append=TRUE/FALSE, split=TRUE/FALSE` are available.

Example:

Find the mean of all the three variables in the data set

```
example1.csv


setwd("C:/RCourse/")

data <- read.csv("example1.csv", header=TRUE)

> data[,1]
[1] 2 3 4 5

> data[,2]
[1] 20 30 40 50

> data[,3]
[1] 200 300 400 500
```



```
> data <- read.csv("example1.csv", header=TRUE)
> data
  X1 X10 X100
1  2  20  200
2  3  30  300
3  4  40  400
4  5  50  500
> data[,1]
[1] 2 3 4 5
> data[,2]
[1] 20 30 40 50
> data[,3]
[1] 200 300 400 500
>
```

Example:

Programme:

```
meanxyz <- function(data)
{
  meanofdata <- 0
  for (i in 1:3)
  {
    meanofdata[i] <- mean(data[,i])
    cat("The mean of X",i, "is", meanofdata[i], ".", "\n")
  }
}
meanxyz(data)
```

Save it as script, say `meanxyz.R`

Example:

```
> sink("output_meanxyz.txt")  # Creates a blank file
```

(Open the file and check it – a blank file will be there)

```
> source("meanxyz.R")  # Writes output inside the file
```

Or run the programme as

```
> meanxyz(data)
```

(Open the file and check it – a file with the output will be there)

```
> sink() # Resume writing output to console
```

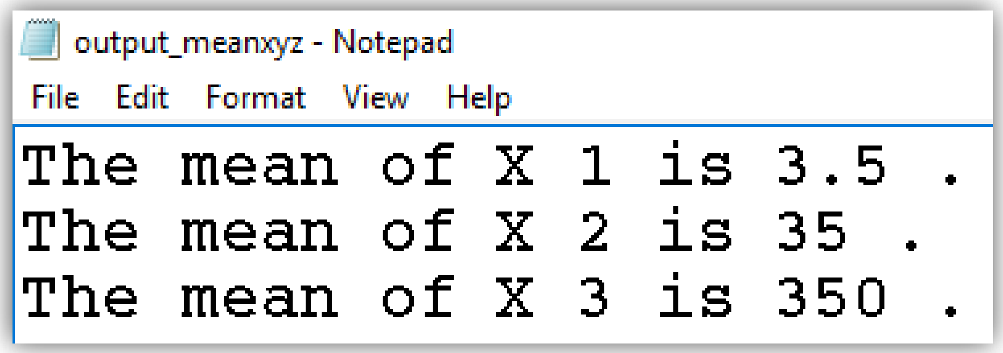
Output:

Open the directory `"C:/RCourse/".`

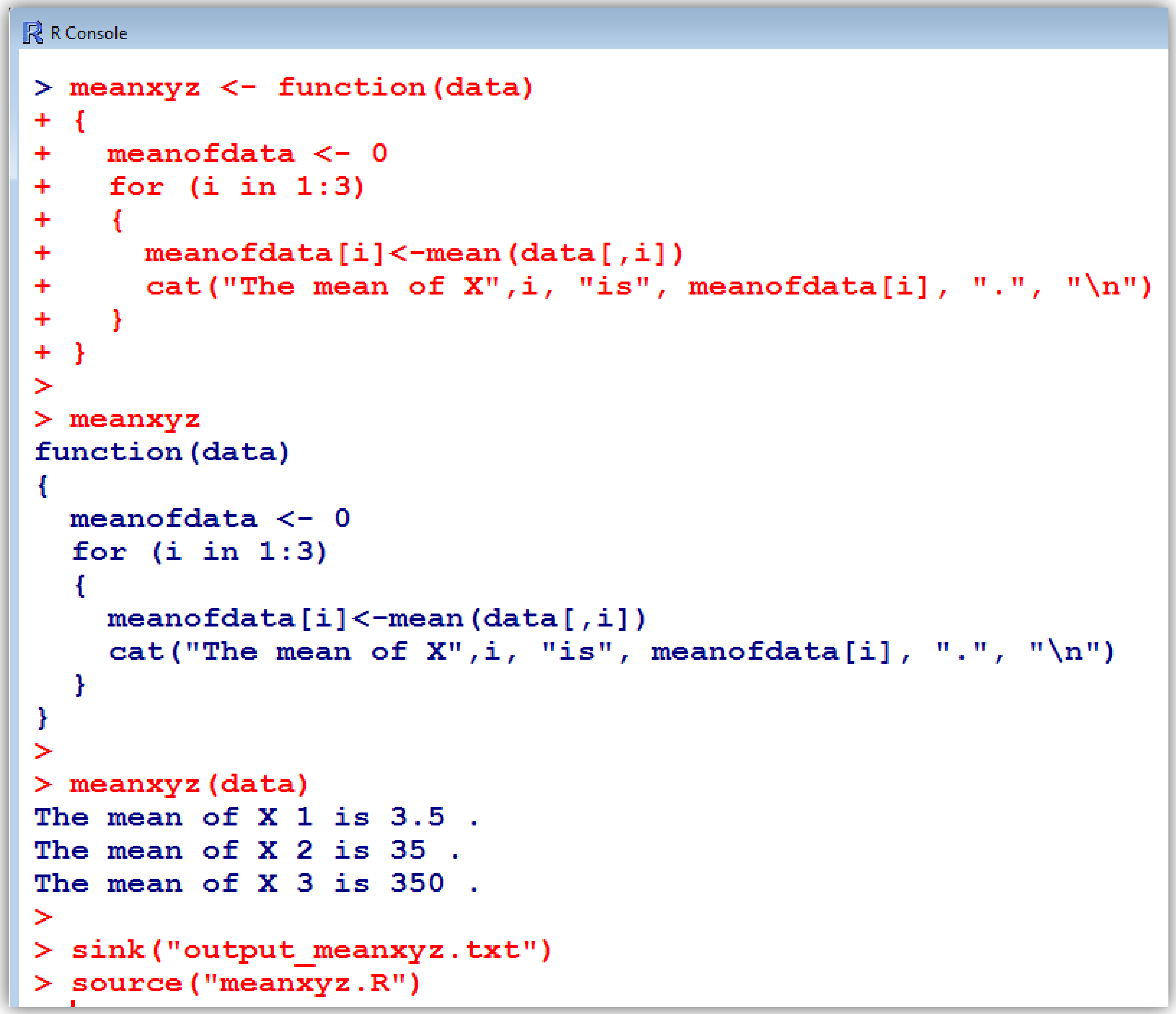
Find a file `output_meanxyz.txt`

Open it and we find the following output

```
The mean of X 1 is 3.5 .
The mean of X 2 is 35 .
The mean of X 3 is 350 .
```



```
The mean of X 1 is 3.5 .
The mean of X 2 is 35 .
The mean of X 3 is 350 .
```


R Console

```
> meanxyz <- function(data)
+ {
+    meanofdata <- 0
+    for (i in 1:3)
+    {
+      meanofdata[i]<-mean(data[,i])
+      cat("The mean of X",i, "is", meanofdata[i], ".", "\n")
+    }
+ }
>
> meanxyz
function(data)
{
   meanofdata <- 0
   for (i in 1:3)
   {
     meanofdata[i]<-mean(data[,i])
     cat("The mean of X",i, "is", meanofdata[i], ".", "\n")
   }
}
>
> meanxyz(data)
The mean of X 1 is 3.5 .
The mean of X 2 is 35 .
The mean of X 3 is 350 .
>
> sink("output_meanxyz.txt")
> source("meanxyz.R")
```

Writing to CSV files

Suppose we want to save a matrix or data frame in a file using the comma-separated values format.

The `write.csv` function writes tabular data to an ASCII file in CSV format.

Each row of data creates one line in the file, with data items separated by commas (,):

```
> write.csv(x, file="filename", row.names=FALSE)
```

Example:

```
> write.csv( meanxyz(data),
file="output_meanxyz.csv", row.names=FALSE )
```

Check working directory, file `output_meanxyz.csv` is created.