CHAPTER 23

Outputting to a File

To save data to a text file, use the following codes:

```
> x<-1:5
> write.table(x,file="c:/test.txt",quote=F,row.names=F)
```

The second example shows how to save an R data set, the best format of which is in R. The speed of retrieval is one of the major advantages of using the R-data format (RData). For more details, see chapter 43.

```
> x<-1:100
> y<-rnorm(200)
> save(x,y,file="c:/test.RData")
```

To upload an R data set, we use the **load()** function.

```
> load("c:/test.RData")
```

23.1. Writing to a Text File

There are several advantages of saving our results to a text file. First, it is easy to read since we can use Notepad or Microsoft Word to open such a file. Second, we can use other softwares to process the data further. Third, we can exchange data or results with other non-R users. To write a text file, we can use the **write.table(), write.csv(), write(), or cat()** functions which would be discussed later in the chapter.

23.2 The Function write.table()

This function is the most used R function to write a text file, including a CSV file. In the following example, pay attention to the double quotations and the first column in the right panel below, which are the row names.

```
> x<-100:1
> write.table(x,'c:/temp/test.txt')
```

```
"x"
"1" 100
"2" 99
"3" 98
"4" 97
```

However usually we don't need double quotation marks and row names. In this case, we add **quote=F, row.names=F**, where **F** stands "false."

```
> x<-1:100
> write.table(x,'c:/temp/test.txt',quote=F,row.names=F)
```

```
x
1
2
3
4
```

23.3. Writing a CSV (Comma Separate Value) File

A CSV file is a special format among text files. The advantage of a CSV file is that it saves space since we don't need blanks to make our output files more readable. Second, reading or writing a CSV file is quite common for almost all softwares. Third, it is quite easy to use Excel to input a CSV file.

23.4. The write.csv() Function

Below, we generate a vector from 1 to 100 then save it to a file called **test.dat.**

```
> x<-1:100
> write.csv(x,'c:/test.dat')
```

```
"","x"
"1",1
"2",2
"99",99
"100",100
```

Again, double quotations and row names are usually not necessary.

```
x<-1:10
> y<-rev(x)
> z<-cbind(x,y)
> write.csv(z,"c:/t.csv",quote=F,row.names=F)
```

```
x,y
1,10
2,9
3,8
4,7
5,6
6,5
7,4
8,3
9,2
10,1
```

There are two ways to retrieve such data from Excel. The first is to start Excel then open the above output file called **t.csv**. For the second method, we save our data to a clipboard first.

```
x<-1:10
> y<-rev(x)
> z<-cbind(x,y)
> write.csv(z, "clipboard",quote=F,row.names=F)
```

```
x,y
1,10
2,9
3,8
4,7
5,6
6,5
7,4
8,3
9,2
10,1
```

Go to Excel and paste it. Highlight the output then click **Data** then **Text to Column**. Choose **Delimiter** then **comma**.

	A
1	x,y
2	1,10
3	2,9
4	3,8
5	4,7
6	5,6
7	6,5
8	7,4
9	8,3
10	9,2
11	10,1

It is better to show the result if we have a matrix (multiple columns). The output is shown in the right panel.

```
> x<-1:10
> y<-11:2
> z<-cbind(x,y)
> write.csv(z,'c:/temp/test.txt')
```

```
"","x","y"
"1",1,11
"2",2,10
"3",3,9
"4",4,8
"5",5,7
"6",6,6
"7",7,5
"8",8,4
"9",9,3
"10",10,2
```

23.5. The write() Function

Below, we generate two vectors, combine them, and save the final results by using the **write()** function.

```
> x<-1:5
> y<-11:7
> z<-cbind(x,y)
> z
x y
[1,] 1 11
[2,] 2 10
[3,] 3 9
[4,] 4 8
[5,] 5 7
> write(z,'c:/temp/test.txt',ncolumns=2)
```

```
1 2
3 4
5 6
7 8
9 10
11 10
```

```
> write(x, "c:/test.txt") # double quotations
```

```
> write(x, "c:\\test.txt") # use \\ instead of /
```

If we use the **write()** function instead of the **write.table()** function, we will get a different output (see the right panel below).

```
> x<-1:100
> write(x,'c:/test.dat')
```

```
1 2 3 4 5
6 7 8 9 10
91 92 93 94 95
96 97 98 99 100
```

We can save the column names as well.

```
> x<-c(19900102, 0.023, 19900102, 0.001)
> x2<-t(matrix(x,2,2))
> x2
     [,1]  [,2]
[1,] 19900102 0.023
[2,] 19900102 0.001
> colnames(x2)<-c("date","ret")
> write.table(x2,file="c:/test.dat",col.names=TRUE)
```

We can use the **read.table()** function to read the output saved by the above program.

```
> read.table("c:/test.dat")
  date ret
1 19900102 0.023
2 19900102 0.001
```

We can define a variable representing the name of the output file.

```
> outfile<-"c:/test.txt"
> write.table(x,outfile)
```

23.6. Writing and Loading an R Data Set

To save an RData set, we use the **save()** function.

```
> x<-1:1000
> save(x,file="c:/temp/test.RData")
```

To retrieve the data set, we use the **load()** function.

```
> load("c:/temp/test.RData")
```

The extension of an R data set is not critical. You can try the following codes.

```
> x<-1:1000
> save(x,file="c:/temp/test")
> rm(x)
> load("c:/temp/test")
> x
> help(write.table) #get complete information about write.table()
write.table(x, file = "", append = FALSE, quote = TRUE, sep =
" ",eol = "\n", na = "NA", dec = ".", row.names = TRUE,col.
names = TRUE, qmethod = c("escape", "double"))
```

One of the advantages of using the **save()** function is that we can save multiple data sets simultaneously.

```
> x<-1:10
> y<-rnorm(100)
> save(x,y,file="c:/temp/test.RData")
```

23.7. Appending Data to an Existing Text File

It is quite often that we need to append data to an existing file.

```
> write.table(x,"c:/temp/test.txt")
> write.table(y,"c:/temp/test.txt",append=TRUE)
# For some reason "append=T" is not working for write.csv()
```

23.8. The dot-Rdata File (.Rdata)

The R data set called .RData under your working directory would save the data after you answer yes when quitting.

```
> # when quit answer yes to save the current settings (variables)
```

23.9. Using cat()

Here is another way to save data to a text file.

```
> cat("1 3 4 5","7 17 6 1",file="c:/test.txt",seq="\n")
>cat(file="c:/temp/test.txt","123456","987654",sep="\n")
```

```
123456
987654
```

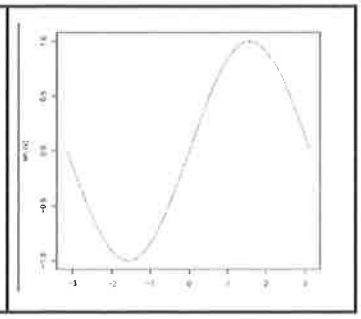
23.10. Writing a Binary File

There are many ways to write a binary file. Using the **save()** function is one of them. The advantage of writing a binary file is the speed. It is extremely efficient to retrieve a binary data set compared to retrieving data from a text file. For more details, see chapter 32.

23.11. Saving a PDF File

In the following program, we generate a PDF file.

```
> pdf("c:/temp/t.pdf")
> plot(sin,-pi,pi)
> dev.off()
```



23.12. Writing Data to a Clipboard

We all know that we can high copy when we use Excel, MS Word, or Notepad. When we are doing so, we copy the highlighted contents to a clipboard. We can do so in R.

```
> x<-1:10
> y<-rev(x)
> z<-cbind(x,y)
> write.csv(z,"clipboard")
# you can paste it to MS Word or Excel
```

```
"","x","y"
"1",1,10
"2",2,9
"3",3,8
"4",4,7
"5",5,6
"6",6,5
"7",7,4
"8",8,3
"9",9,2
"10",10,1
```

Again, if we don't like quotation marks and row numbers, we can remove them by specifying those conditions. The corresponding output is given in the right panel.

```
> write.csv(z,"clipboard",quote=F,row.names=F)
```

```
x,y
1,10
2,9
3,8
4,7
```

23.13. Row Names and Column Names

Assume that we have the following input saved as a text file **c:/temp/test.txt**.

IBM 19900101 0.1
IBM 19990102 0.2

The following program retrieves and saves the data. The saved data is shown in the right panel.

```
x<-read.table("c:/temp/test.txt")
> x
V1 V2 V3
1 IBM 19900101 0.1
2 IBM 19990102 0.2
> colnames(x)<-c("ticker","date","ret")
> write.table("c:/temp/test1.txt")
```

```
 "ticker" "date" "ret"
"1" "IBM" 19900101 0.1
"2" "IBM" 19990102 0.2
```

If we don't want the double quotation marks, we can specify **quote=F**. On the other hand, the row numbers seem redundant. To do without printing them, we specify **row.names=F**. The new output is shown in the right panel.

```
> write.table(x, "c:/temp/test1.txt",quote=F,row.names=F)
```

```
ticker date ret
IBM 19900101 0.1
IBM 19990102 0.2
```

23.14. The sink() Function

First, we should use the **sink()** function in pair, **sink("output_file_name")**, to initiate the operation and **sink()** to end the operation. After we issue **sink("output_file_name")**, all output will be saved to the output file the function has specified. The last **sink()** will finish this operation.

```
> x<-1:100
> sink("test.txt") # save subsequent print to this file
> x # nothing shown on the screen
> print(x) # again nothing shown
> sink() # end of the sink operation
> unsink("test.txt") # close the file
```

23.15. Temporary File tempfile()

The good thing about a temporary file is that after you quit R, the temporary file will vanish as well.

```
> help(tempfile)
> ff<-tempfile()
> ff
[1] "C:\\Users\\yyan\\AppData\\Local\\Temp\\RtmpYwK05b\\file46f33db4"> x<-10
> write.table(x,ff)
```

Exercises

23.1. Generate 100 random numbers from a standard normal distribution and save them to a text file by using **write.table()**.

```
> x<-rnom(100)
```

23.2. Generate a matrix with 10*4 and save them to an Rdata file.

23.3. Go to finance.yahoo.com to download IBM's monthly price data and save them to a text file.