Data Import and Export

Feng Li
feng.li@cufe.edu.cn

School of Statistics and Mathematics
Central University of Finance and Economics

Revised on March 12, 2019

Today we are going to learn...

1. Data import
2. Read Database
3. Read images

Plain Text Format

- Reads a file in table format and creates a data frame from it.
  - `read.table()`
  - `read.csv()`
  - `read.csv2()`
- Write a file in text format.
  - `write.table()`
  - `write.csv()`
  - `write.csv2()`

Micro$oft Excel

- Install and load the package

```
install.packages("openxlsx")
library("openxlsx")
```

- Read XLS Files

```
require("openxlsx")
mydata <- read.xlsx("MyFile.xlsx",
                    sheet = 1,
                    startRow = 1,
                    colNames = TRUE)
```

Minitab, S-PLUS, SAS, SPSS, Stata, Systat

The recommended package foreign provides import facilities for files produced by these statistical systems.

- Function `read.xport()` reads a file in SAS Transport (XPORT) format and return a list of data frames.
- Function `read.mtp()` imports a 'Minitab Portable Worksheet'. This returns the components of the worksheet as an R list
- Function `read.spss()` can read files created by the 'save' and 'export' commands in SPSS
- Files from Stata can be read and written by functions `read.dta()` and `write.dta()`.

Read and Write Matlab format

R.matlab package: Read and Write MAT Files and Call MATLAB from Within R (No Matlab installation required.)

- Methods `readMat()` and `writeMat()` for reading and writing MAT files. For user with MATLAB v6 or newer installed (either locally or on a remote host), the package also provides methods for controlling MATLAB (trademark) via R and sending and retrieving data between R and MATLAB.

R and Database I

- SQL has limited numerical and statistical features. For example, it has no least squares fitting procedures, and to find quantiles requires a sophisticated query.
- Not only are basic statistical functions missing from SQL, but in many cases the numerical algorithms used in the basic aggregate functions are not implemented to safeguard numerical accuracy.
- For these reasons, it may be desireable or even necessary to perform a statistical analysis in a statistical package rather than in the database. One way to do this, is to extract the data from the database and import it into statistical software.
- The `RODBC` package provides access to databases (including Microsoft Access and Microsoft SQL Server) through an ODBC interface.
  - `odbcConnect(dsn, uid="", pwd="")`
    Open a connection to an ODBC database
  - `sqlFetch(channel, sqtable)`
    Read a table from an ODBC database into a data frame
  - `sqlQuery(channel, query)`
    Submit a query to an ODBC database and return the results

R and Database II

- `sqlSave(channel, mydf, tablename = sqtable, append = FALSE)`
  Write or update (append=True) a data frame to a table in the ODBC database
- `sqlDrop(channel, sqtable)`
  Remove a table from the ODBC database close(channel) Close the connection

- Use RODBC to read Oracle Database
  `https://blogs.oracle.com/R/entry/r_to_oracle_database_connectivit`
- The **DBI** package in R provides a uniform, client- side interface to different database management systems, such as MySQL, PostgreSQL, and Oracle. The basic model breaks the interface between the client and the server into three main elements: the driver facilitates the communication between the R session and a particular type of database management system (e.g. MySQL); the connection encapsulates the actual connection (with the aid of the driver) to a particular database management system and carries out the requested queries; and the result which tracks the status of a query, such as the number of rows that have been fetched and whether or not the query has completed
  - We provide a simple example here of how to extract data from a MySQL database in an R session. The first step: load a driver for a MySQL-type database

```
drv = dbDriver("MySQL")
```

R and Database III

- The next step is to make a connection to the database management server of interes

```
con = dbConnect(drv, user="yourusername", dbname="DataBaseName",
                host="HostName")
```

- Once the connection is established, queries can be sent to the database

```
dbListTables(con)
```

- Other queries can be executed by supplying the SQL statement.

```
dbGetQuery(con,"SELECT COUNT(*) FROM Allstar;")
```

- The **RMySQL** package provides an interface to MySQL.
- The **ROracle** package provides an interface for Oracle.
- The **RJDBC** package provides access to databases through a JDBC interface.

Lab exercise

- Read and query data from database.

Images

- JPEG Format

```
install.packages("jpeg")
require("jpeg")
readJPEG()
rasterImage()
```

- Packages **bmp**, **jpeg** and **png** read the formats after which they are named. See also packages **biOps** and **Momocs**, and Bioconductor package **EBImage**.

Images: Example

```
library("jpeg")
myprofile <- readJPEG("FengLi.jpg")
dim(myprofile)
image(1:1539, 1:1154, myprofile[, , 1])
image(1:1539, 1:1154, myprofile[, , 2])
image(1:1539, 1:1154, myprofile[, , 3])

plot(c(100, 250), c(300, 450), type = "n", xlab = "", ylab = "")
rasterImage(myprofile, 100, 300, 150, 350, interpolate = FALSE)
rasterImage(myprofile, 100, 400, 150, 450)
rasterImage(myprofile, 200, 300, 200 + xinch(.5),
            300 + yinch(.3), interpolate = FALSE)
rasterImage(myprofile, 200, 400, 250, 450,
            angle = 15, interpolate = FALSE)
```

Further Suggested Read

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