Northeastern University

CS6020: Collecting, Storing, and Retrieving Information

Data Import

Data Import

IMPORTING DATA FROM TEXT FILES

Lesson Objectives

- After completing this lesson, you are able to:
  - import data from CSV and Excel files
  - load data from tab delimited text files
  - load *.Rdata* object files
  - import data from SAS, SPSS, and Stata
  - specify import parameters

File Formats

- Data is often stored in different file formats.

File Format	Extension	Source	Format
Comma Separated	.csv	Excel; many programs export data in that format	Text
Tab Delimited	.txt	many programs export data in that format	Text
Excel	.xls	Excel files prior to version 2010	Encoded
Excel	.xlsx	Excel version 2010 and later	XML
R Data Object	.RData	R	Binary

Importing from Other Packages

- Other than text and R binary files, R can also import data from other statistical packages using the “`foreign`” library.
- Among many others, the “`foreign`” library supports importing from:
  - Stata
  - SPSS
  - SAS
- R also has support from reading and writing XML.

Directories and Folders

- Data is located in folders (also called directories).
- R must be directed to the folder in which a local text file is located or to a URL if the file is located on the web.

```
> ## absolute path to a folder
> setwd("C:/Users/Martin/Downloads")
> ## relative path to a folder
> setwd("../")
```

Path Separator in Windows

- R for Windows understands the forward slash but not a single back slash, i.e., this specification will **not** work:

```
> ## will not work
> setwd("C:\Users\Martin\Downloads")
```

But this will:

```
> ## will not work
> setwd("C:\\Users\\Martin\\Downloads")
```

Checking for File Existence

- Two important functions for checking directory or file existence:
  - `file.exists()` – checks if the file or directory exists and returns TRUE if it does, FALSE otherwise
  - `dir.create()` – creates the directory if it does not exist

```
> ## create a new directory (folder) if it
> ## does not exist
> if (!file.exists("data")) {
+   dir.create("data")
+ }
```

Downloading Files From Web

- Data files are often available through web sites.
- Data Scientists should distribute files through the web rather than sending.
- Use `download.file()` to load a file through a URL using HTTP or FTP.

```
> download.file(fileURL,
    destfile="c:/users/martin/downloads/web114.zip",
    method="curl")
```

Large Files

- When a file is loaded, its contents is stored in the computer's RAM (main memory) which is often limited to generally.
- The 64-bit version of R can hold more data than the 32-bit version of R.
- When "big data" files need to be analyzed, then a database should be used.
- Aside from in-memory storage, files can also take significant time to load.

Importing Data From Text Files

FILE IMPORT: CSV

CSV File Layout

- CSV files contain data records organized in rows with an optional header row containing the column labels.
- There is no standard CSV file format, although RFC 4180 is an attempt to standardize some aspects of CSV.
- One of the key issues is that data an be double quoted and may also be missing.

Example CSV File

```
"first_name","last_name","company_name","address","city","county","state","zip"
,"phone1","phone2","email","web"
"James","Butt","Benton, John B Jr","6649 N Blue Gum St","New
Orleans","Orleans","LA",70116,"504-621-8927","504-845-
1427","jbutt@gmail.com","http://www.bentonjohnbjr.com"
"Josephine","Darakjy","Chanay, Jeffrey A Esq","4 B Blue Ridge
Blvd","Brighton","Livingston","MI",48116,"810-292-9388","810-374-
9840","josephine_darakjy@darakjy.org","http://www.chanayjeffreyaesq.com"
```

Loading a CSV Files

- If the file contains a header row with column labels, specify `header=TRUE`.
- If the file starts anywhere but the first row, specify `skip=x`, where `x` is number of lines to be skipped.

```
> setwd("C:/Users/Martin/Downloads")
> people <- read.csv("us-500.csv",header=TRUE)
> str(people)
```

Alternative Separators

- While CSV files are expected to use comma (,) as a separator, some might not.
- The `sep="x"` allows you to specify any character as the field separator.

```
> setwd("C:/Users/Martin/Downloads")
> people <- read.csv("us-500.csv",header=TRUE,sep=",")
> str(people)
```

Quotes

- In both CSV and tab delimited flat files, it is not uncommon that data values are placed inside quotes: " or '
- Specifying `quote=""` causes the quotes to not be read.
- If data values are placed within non-standard quotes, then the data values must be processed as character strings and the quotes must be removed through programming.

Strings as Factors

- Strings are often converted to factors which may not be the desired result.
- Specify `stringsAsFactors=FALSE` to avoid having R automatically convert string values to factor.

Acquiring Data

FILE IMPORT: TAB DELIMITED

File Layout

- A tab delimited file is the same as a CSV file expect that the separator is a tab (the `\t` character in R).
- Reading a tab delimited file can be done with:
  - `read.csv()` using `sep="\t"`
  - `read.table()`

Loading a Tab Delimited File

- If the file contains a header row with column labels, specify `header=TRUE`.
- If the file starts anywhere but the first row, specify `skip=x`, where x is number of lines to be skipped.

```
> setwd("C:/Users/Martin/Downloads")
> people <- read.table("us-500.txt",header=TRUE)
> str(people)
```

Acquiring Data

FILE IMPORT: EXCEL

File Format

- This file format is the native format for Excel.
- Excel also exports data as CSV but sometimes data is only available in the native file format.
- Excel files have a `.xls` and `.xlsx` extension.
- Importing such file requires the **xlsx** or **openxslx** libraries.
- The **openxslx** library is preferable for large files.

Loading and Excel File

- Excel files can contain one or more worksheets, so you must specify which sheet is to be loaded using `sheetIndex=`*`x`* where *`x`* is the number of the worksheet.

```
> setwd("C:/Users/Martin/Downloads")
> library(xlsx)
> people <- read.xlsx("us-500.xlsx",sheetIndex=1)
> str(people)
```

- Note that this may require the use of the 32-bit version of R.

Writing Excel Files

- Data can be written to an Excel file using the function `write.xlsx()`.
- However, for compatibility, store data in CSV or tab-delimited text files rather than in Excel format.

Alternative Packages

- The `xlsx` package is stable but slow for large files.
- As an alternative, use the `xlsx2` or `XLConnect` packages.
- These packages are faster and have more options for writing Excel files.

Acquiring Data

FILE IMPORT: BINARY R DATA

The R Object File Format

- Saving R objects in .RData files is fast and convenient but not compatible with programs other than R.
- Use the `load()` function to load previously save objects.
- To determine which objects were loaded use the `objects()` or `ls()` commands.

Summary

- In this lesson, you learned that:
  - R supports importing from a variety of data file formats
  - CSV and tab delimited files are the most common file format for exchanging data between programs
  - R can read native Excel files

Summary, Review, & Questions…