STENO Introductory R-Workshop: Loading a Data Set

Tommi Suvitaival, tsvv@steno.dk, Steno Diabetes Center

June 11, 2015

Contents

1 Introduction

Today's goals:

- Recap
- Data containers
- Functions
- Reading a data set into R
- First look at the data

2 Recap: Variables

- Numeric

```
a = 2
b = 3

a + b
```

```
## [1] 5
```

- String

```
a = "This"
b = "is a string"

paste( a, b )
```

```
## [1] "This is a string"
```

- Logical

```
a = TRUE
b = FALSE

a & b
```

```
## [1] FALSE
```

For the vectors and matrices, let's go through the Section 3 of the document *Introductory R-workshop: Variables, data types and containers* at https://github.com/leonjessen/introductoryR/tree/master/01_basic_introduction .

- Click the name of the `.pdf` file.
- Click `Raw`.

3 Data Containers

3.1 Vectors

```
nums = c( 5, 2, NA, 3, 1 )
nums
```

```
## [1]  5  2 NA  3  1
```

```r
# Show the 4th element of the vector.
nums[ 4 ]
```

```
## [1] 3
```

```r
# Show all elements of the vector, except the 4th element.
nums[ -4 ]
```

```
## [1]  5  2 NA  1
```

```r
# Create a sequence from 1 to 15.
sequence = seq( from=1, to=15 )

# Show the sequence.
sequence
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

3.2 Matrices

```r
# Define the number of rows
m = 3
# Define the number of columns
n = 5

# Define the elements of the matrix
A = matrix( data=sequence, nrow=m, ncol=n )

A
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    4    7   10   13
## [2,]    2    5    8   11   14
## [3,]    3    6    9   12   15
```

```r
# Take the item in the 3rd row and 2nd column.
A[ 3, 2 ]
```

```
## [1] 6
```

```r
# Take the 3rd row.
A[ 3, ]
```

```
## [1]  3  6  9 12 15
```

```r
# Take the 2nd column.
A[ , 2 ]
```

```
## [1] 4 5 6
```

```r
# Take the matrix without the second column.
A[ , -2 ]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    7   10   13
## [2,]    2    8   11   14
## [3,]    3    9   12   15
```

4 Functions

- Function combines a set of instructions into a single line of code.
  - Examples: `mean( x )` and `read.table( file )`
- Function may have *arguments*.
  - Arguments are the input to the function.
  - Examples:
    - `mean( x )` has argument `x`, which is a numeric vector.
    - `read.table( file )` has argument `file`, which is a name of the file (and of the string data type).
- Function may have a *return value*.
  - Return value is the output of the function.
  - Examples:
    - `mean( x )` returns a single number.
    - `read.table( file )` returns a data frame. (More of that later.)
- You can find the *documentation* of a function by typing ? and the name of the function.
  - Examples: `?mean`, `?read.table`
- How to *call* a function?
  - You have done it already for `mean( x )`.

```r
mean( x=c( 1, 2, 3 ) )
```

```
## [1] 2
```

5 Reading a Data Set

5.1 Introduction

5.1.1 Comma Separated Values

- Let's read a comma separated values (CSV) file to R.
  - CSV is a text file that contains a data table.
  - Values are organized into rows and columns and separated by a comma.
  - CSV can also be imported/exported to/from Excel.
  - An example with 3 columns (variables) and 10 rows (samples):

```
0.8409986,-1.607113,2.136502
-0.1525392,1.688292,1.646375
-0.908455,-1.029759,0.7707836
-1.597526,-1.678158,1.100898
-0.1857298,-0.4470996,-0.7282355
0.8096235,-1.731656,-0.8830574
0.6639507,0.1762918,-0.06044059
0.4281464,0.14222,-1.796507
1.352567,-0.2312621,0.2371557
0.7544238,-1.279464,-1.582303
```

5.1.2 Diabetes Data Set from UCI

- We are going to have a look at a small data set related to diabetes, the *Pima Indians Diabetes Database*, which consists of clinical data on a Native American group.
  - It is a widely-used test data set for computational analysis methods: "UCI Diabetes Database" has almost 10,000 hits on Google Scholar.
  - It is publicly available at the *UCI Repository Of Machine Learning Databases* at http://repository.seasr.org/Datasets/UCI/csv/diabetes.csv .
  - The original owner of the data is *National Institute of Diabetes and Digestive and Kidney Diseases*.
- We are going to download a CSV file from the URL https://goo.gl/W5hdik.
  - It is a cleaned version of the *Pima Indians Diabetes Database*.
  - We'll read it directly into R, but you can also click the link to access it on your browser.

5.2 Reading a File with the `read.table` Function

- The `read.table` function reads a data table from a text file.
  - *Arguments* of the function (*i.e.*, what it needs from the user as input):
    - `file`: File path or URL of the file to be read (a string).
    - `header`: Logical (`TRUE`/`FALSE`), deciding whether the first row of the file contains the names of the data columns.
    - `sep`: The character separating values (cells in Excel) in a row from each other.
    - `dec`: The decimal separator (typically "." in English; "," in Danish).
  - *Value* of the function (*i.e.*, what it returns as output):
    - A *data frame* containing values read from the file.
      - Data frame is a tightly coupled collection of variables.
      - We could say it is a data table which allows us to store multiple data types together (e.g., numeric, string, logical).
      - Each column on its own needs to have a specific data type.
      - It is "similar to SAS and SPSS datasets."

```
# Read the comma separated values (CSV) file from the web address (URL)
# using the "read.table" function,
# and assign ("=") the read data frame into the variable "data".
# When reading the file, assume/request that:
# - the file exists at the URL "https://goo.gl/W5hdik".
# - the first row of the file contains the names of the columns ('header=TRUE')
# - cells of the table are separated by comma ('sep=",")
# - the decimal point is marked by dot ('dec=".")
# The file downloaded is a cleaned version of the Diabetes data set
# from "http://repository.seasr.org/Datasets/UCI/csv/diabetes.csv":
# - the second row of the file has been removed
# - zeros in the columns 2 to 6 have been removed (and are now missing).
data = read.table( file="https://goo.gl/W5hdik", header=TRUE, sep=",", dec="." )
```

- Read without errors? Now we can have a look at the data!

5.3 First Look at the Data Set

5.3.1 The `View` Function

- In R Studio, we can explore a matrix or data frame interactively by typing `View( data )`
  - (Result not shown here.)

5.3.2 Backround: What is the Data Set?

- Each row is a vector of *observations* from one *subject*, related to the diagnosis of type 2 diabetes.
- Description of the variables is shown in the table below.
  - Adapted from http://cran.r-project.org/web/packages/mlbench/mlbench.pdf.

Variable	Description
preg	Number of times pregnant
plas	Plasma glucose concentration (glucose tolerance test)
pres	Diastolic blood pressure (mm Hg)
skin	Triceps skin fold thickness (mm)
insu	2-Hour serum insulin (mu U/ml)
mass	Body mass index (weight in kg/(height in m)^2)
pedi	Diabetes pedigree function
age	Age (years)
class	Class variable (test for diabetes)

5.3.3 Dimensions

- The `dim` function returns the dimensions of a matrix or data frame:
  - First element: the number of rows
  - Second element: the number of columns.

```
# Return the dimensions (rows and columns) of the data frame "data".
dim( data )
```

```
## [1] 768   9
```

5.3.4 Structure of the Data Container

- The `str` function returns an overview of any data container.
  - Applies also to other containers than matrices.

```
# View the structure of the data frame "data".
str( data )
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ preg : int  6 1 8 1 0 5 3 10 2 8 ...
##  $ plas : int  148 85 183 89 137 116 78 115 197 125 ...
```

```
##  $ pres : int  72 66 64 66 4 74 5 NA 7 96 ...
##  $ skin : int  35 29 NA 23 35 NA 32 NA 45 NA ...
##  $ insu : int  NA NA NA 94 168 NA 88 NA 543 NA ...
##  $ mass : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 3.5 NA ...
##  $ pedi : num  0.627 0.351 0.672 0.167 2.288 ...
##  $ age  : int  50 31 32 21 33 30 26 29 53 54 ...
##  $ class: Factor w/ 2 levels "tested_negative",..: 2 1 2 1 2 1 2 1 2 2 ...
```

- It is a *data frame*.
- The dimensions of the data frame are shown on the first row.
- Variables of the data frame are shown here as rows.
- The *data type* and few of the first elements of each variable and are shown in the print.

5.3.5 Subsetting the Matrix

- Printing an entire matrix will usually lead to an excess of information on the screen.
- Instead, we can print a subset of the entire data set, *e.g.*, the first 5 rows of the matrix.

```
# Return the first 5 rows and all columns of the data frame "data".
data[ 1:5, ]
```

```
##   preg plas pres skin insu mass  pedi age          class
## 1    6  148   72   35   NA 33.6 0.627  50 tested_positive
## 2    1   85   66   29   NA 26.6 0.351  31 tested_negative
## 3    8  183   64   NA   NA 23.3 0.672  32 tested_positive
## 4    1   89   66   23   94 28.1 0.167  21 tested_negative
## 5    0  137    4   35  168 43.1 2.288  33 tested_positive
```

- Subsetting a column is useful for variable-specific computations.

```
# Compute the mean of the 1st column of the data frame "data",
# omitting the missing ("NA") values.
mean( x=data[ , 1 ], na.rm=TRUE )
```

```
## [1] 3.845052
```

```
# Compute the standard deviation of the 1st column of the data frame "data",
# omitting the missing ("NA") values.
sd( x=data[ , 1 ], na.rm=TRUE )
```

```
## [1] 3.369578
```

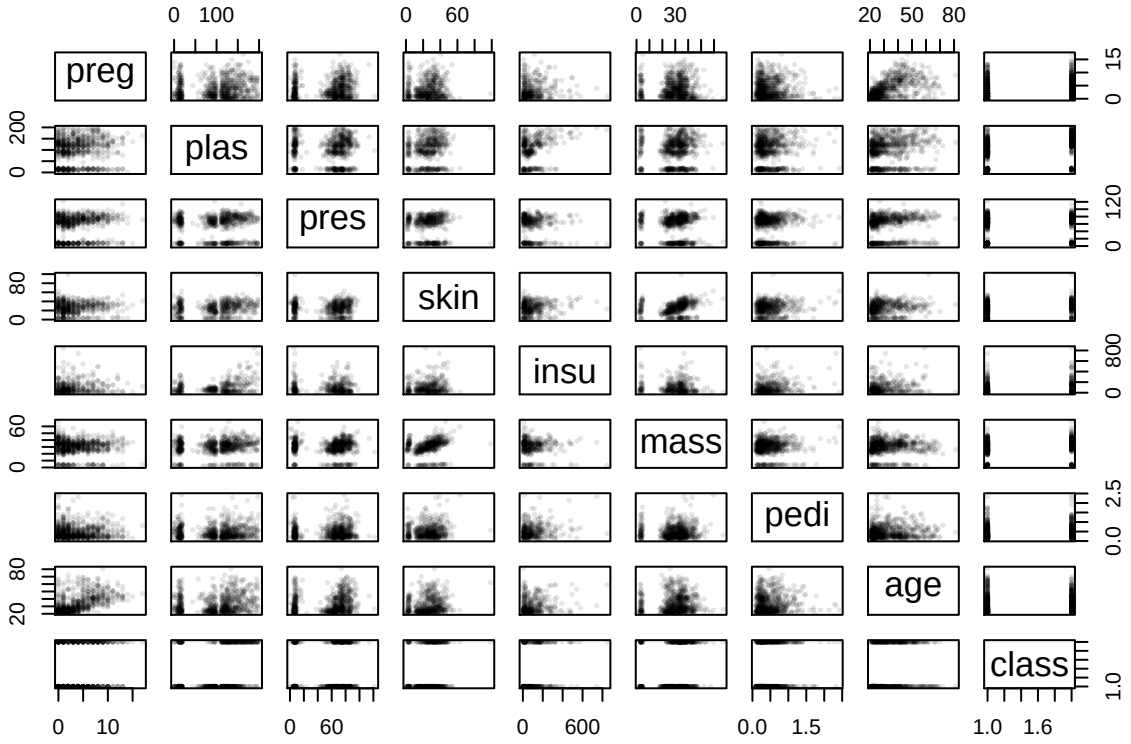
```
# Compute the standard deviation of the 1st column of the data frame "data",
# omitting the missing ("NA") values.
median( x=data[ , 1 ], na.rm=TRUE )
```

```
## [1] 3
```

6 Visualization with Scatter Plot

- Inspect the linear and non-linear statistical dependencies between individual variable pairs.

```
pairs( x=data, col=gray( level=0, alpha=0.1 ), pch=20, cex=0.5 )
```



7 Extra: Logistic Regression Model

- $\mathbf{y} = \frac{1}{1+e^{-\boldsymbol{\beta}\mathbf{x}}}$
  - Interpretation: Binary dependent variable $\mathbf{y}$ is explained by the continuous independent variables $\mathbf{x}$ (through the regression coefficients $\boldsymbol{\beta}$ and the logistic link function).
  - Components of the model:
    - $\mathbf{y}$: Dependent variable called `class`
      - A factor coding the diabetes-negative and diabetes-positive subjects, respectively.
    - $\mathbf{x}$: Independent variables called preg, plas, pres, skin, insu, mass, pedi, age.

7.1 Fitting a Model with the `glm` Function

```
# Fit a logistic regression model by calling the "glm" function and
# assign the model objec to the variable "logistic.regression.fit".
# Formula: the regression equation
# Dependent variable: "class"
# Independent variables: "." giving "the rest" of the variables in the data set.
# Other arguments:
# - family: the type of the regression model
# - data: the data frame where the variables specified in the "formula" are found

logistic.regression.fit = glm( formula=class ~ .,
                               family=binomial( link="logit" ),
                               data=data )
```

7.2 Inspecting a Model with the `summary` Function

```
# Return a summary of the model.
summary( logistic.regression.fit )
```

```
##
## Call:
## glm(formula = class ~ ., family = binomial(link = "logit"), data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8329  -0.7235  -0.4495   0.7860   2.3819
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.053308   0.730602  -8.285  < 2e-16 ***
## preg         0.068907   0.050878   1.354  0.17562
## plas         0.010843   0.002617   4.144 3.41e-05 ***
## pres        -0.002260   0.004214  -0.536  0.59178
## skin         0.028474   0.010897   2.613  0.00897 **
## insu         0.001772   0.001158   1.530  0.12612
## mass         0.032477   0.012662   2.565  0.01032 *
## pedi         1.108575   0.389777   2.844  0.00445 **
```

```
## age          0.046759   0.016958   2.757  0.00583 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 498.10  on 391  degrees of freedom
## Residual deviance: 383.65  on 383  degrees of freedom
##   (376 observations deleted due to missingness)
## AIC: 401.65
##
## Number of Fisher Scoring iterations: 4
```

8 Your Own Practice Project

1. Save your data set as a CSV file. Pay attention to:
   - How the cells of the table are separated ("separating character")?
   - What is the decimal character? ("." or ","? Must not be the same as the separating character.)
   - How the missing values are symbolized? (Empty cell or a special string such as `NA`?)
2. Load the CSV file using the `read.table` function. Pay attention to
   - Does the first row contain the names of the columns?
   - Are there columns whose data type is other than numeric?
   - Are all the values of a column of the same data type? (Should be.)
3. Inspect the data matrix in R. Try:
   - Subsetting the data.
   - Computing the mean of a data column.
   - Changing values in the data.
   - Making logical tests to the data.
4. Make a scatter plot of the data.

9 Conclusion

- R is a powerful tool for statistical data analysis.
  - . . . once you have managed to load the data!
- There are vast libraries of functions and methods available for free.
- Knowledge of programming basics is useful

10 Future Topics?

- Programming basics:
  - conditional statements (`if`)
  - loops (`for` and `while`)
  - functions
  - objects
- Loading "raw" data sets, cleaning and pre-processing them in R
- Visualizations
- Statistical tests & p-value corrections
- How to find help for problems; how to find new packages
- Using the `knitr` package for writing reproducible reports (like this one)
- Widely-used packages for clustering and classification
- How to cross-validate or bootstrap a model