Introductory Programming in R

By Asef Nazari

asef.nazari@monash.edu

Faculty of IT

Monash university

3. Data Tables

3.1 Matrices

A matrix is a rectangular array of numbers. From technical perspective, it is a vector, with two additional attributes, namely, the numbers of rows and columns. Vctors we considered so far were one-dimensional. Matrices are a special type of vetor. They have dimension attribute. in other words, matrices are a multi-dimensional vectors.

In [16]:

```
m <- matrix(nrow=2, ncol=3) #empty matrix with dimension
m
print(m)
```

NA	NA	NA
NA	NA	NA

```
     [,1] [,2] [,3]
[1,]   NA   NA   NA
[2,]   NA   NA   NA
```

In [2]:

```
attributes(m)
```

$dim =

```
2  3
```

In [3]:

```
dim(m)
```

```
2  3
```

In [4]:

```
print(paste(dim(m)[1], " + ", dim(m)[2]))
```

```
[1] "2  +  3"
```

In [18]:

```
m <- matrix(c(1,3,6,2,8,4), nrow=2, ncol=3 ) #matrices are build column-wise
print(m)
```

```
     [,1] [,2] [,3]
[1,]    1    6    8
[2,]    3    2    4
```

In [19]:

```
str(m) # one of the most important functions
```

```
 num [1:2, 1:3] 1 3 6 2 8 4
```

In [6]:

```
m[2,2]
```

```
2
```

Other commonly used approaches to create matrix are cbind() and rbind().

In [7]:

```
#two other methods to creat matrices
x <- c(1,11,111)
y <- c(2,22,222)
m1 <- cbind(x,y) #column-binding
print(m1)
print("****")
m2 <- rbind(x,y) #raw-binding
print(m2)
```

```
       x   y
[1,]   1   2
[2,]  11  22
[3,] 111 222
[1] "****"
  [,1] [,2] [,3]
x    1   11  111
y    2   22  222
```

3.2 Data Frames

Data frames are very important object in R. When you have m obsrvation with n attributes, you have a dataframe of size $m \times n$. As the attributes could be of any class, a data frame is technically a list, with each component being a vector corresponding to a column in our data "matrix." Therefore, dataframes are a special type of list, where every element of this list should have the same length. Dataframes can store different classes of object in each column. Matrices, should have the same class for every element.

In [8]:

```
# to create a dataframe
x <- c(1,2,3)
y <- c("a", "b", "c")
z <- c(TRUE, TRUE, FALSE)
df <- data.frame(x,y,z)
print(df)
```

```
  x y     z
1 1 a  TRUE
2 2 b  TRUE
3 3 c FALSE
```

In [9]:

```
attributes(df)
```

$names
'x' 'y' 'z'
$row.names
1 2 3
$class
'data.frame'

In [10]:

```
nrow(df)
```

3

In [11]:

```
ncol(df)
```

3

In [12]:

```
df[2,2]
```

b

In [10]:

```
z <- data.frame(c(1,2), c(3,4))
z
```

c.1..2.	c.3..4.
1	3
2	4

In [8]:

```
class(z)
```

'data.frame'

In [9]:

```
z1 <- data.frame(cbind(c(1,2), c(3,4)))
z1
class(z1)
```

X1	X2
1	3
2	4

'data.frame'

Names

In [1]:

```
x <- c(3,5,7)
names(x)
```

NULL

In [2]:

```
names(x) <- c("low", "med", "high")
print(x)
```

```
 low  med high 
   3    5    7 
```

In [3]:

```
names(x)
```

'low' 'med' 'high'

In [4]:

```
x
```

low
3
med
5
high
7

In [5]:

```
names(x) <- NULL
x
```

```
3 5 7
```

In [2]:

```
y <- list(low=3, med=5, high=7)
print(y)
```

```
$low
[1] 3

$med
[1] 5

$high
[1] 7
```

In [4]:

```
# Access list elements by their name
y$low
print(y$low)
```

```
3
[1] 3
```

In [17]:

```
m <- matrix(1:6, nrow=3, ncol=2)
dimnames(m)<- list(c("a", "b", "c"), c("d", "e"))
```

In [18]:

```
print(m)
```

```
  d e
a 1 4
b 2 5
c 3 6
```

In [19]:

```
colnames(m) <- c("male", "fmale")
rownames(m) <- c("ice-cream", "coffee", "cake")
print(m)
```

```
          male fmale
ice-cream    1     4
coffee       2     5
cake         3     6
```

In [20]:

```
print(df)
```

```
  x y     z
1 1 a  TRUE
2 2 b  TRUE
3 3 c FALSE
```

In [21]:

```
row.names(df) <- c("f1", "f2", "f3")
print(df)
```

```
   x y     z
f1 1 a  TRUE
f2 2 b  TRUE
f3 3 c FALSE
```

In [22]:

```
colnames(df) <- c("rank", "character", "value")
print(df)
```

```
   rank character value
f1    1         a  TRUE
f2    2         b  TRUE
f3    3         c FALSE
```

In [23]:

```
names(df) <- c("r1", "r2", "r3")
print(df)
```

```
   r1 r2    r3
f1  1  a  TRUE
f2  2  b  TRUE
f3  3  c FALSE
```

In [24]:

```
attributes(df)
```

$names

'r1' 'r2' 'r3'

$row.names

'f1' 'f2' 'f3'

$class

'data.frame'

In [25]:

```
class(df)
mode(df)
typeof(df)
```

'data.frame'

'list'

'list'

In [26]:

```
x<- 5
print(x)
names(x)
```

```
[1] 5
```

```
NULL
```

In [27]:

```
names(x) <- c("low")
print(x)
names(x)
```

```
low
  5
```

'low'

In [28]:

```
attributes(x)
```

$names = 'low'

Matrices and dataframes are very similar to each other as both are generally two-dimensional. However, matrices are extensions of vectors, and dataframes are extensions of lists. Matrices have all the data of te same type. Therefore, when your data has different data types, use dataframes.

In [20]:

```
m1<- matrix(1:25,5,5)
m1
```

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

In [21]:

```
str(m1)
```

```
 int [1:5, 1:5] 1 2 3 4 5 6 7 8 9 10 ...
```

In [22]:

```
is.matrix(m1)
```

TRUE

In [23]:

```
is.data.frame(m1)
```

FALSE

In [24]:

```
df1 <- as.data.frame(m1)
df1
```

V1	V2	V3	V4	V5
1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

In [25]:

```
str(df1)
```

```
'data.frame':   5 obs. of  5 variables:
 $ V1: int  1 2 3 4 5
 $ V2: int  6 7 8 9 10
 $ V3: int  11 12 13 14 15
 $ V4: int  16 17 18 19 20
 $ V5: int  21 22 23 24 25
```

In [27]:

```
#The object.size commands indicate how much memory of data take up in the computer
print(paste("the size of df1 is ", object.size(df1), " bytes and the size of m1 is
```

```
[1] "the size of df1 is  1264  bytes and the size of m1 is  328  byte
s"
```

3.3 Reading and Writing Data in R

Generally we read data from a file. In this unit we will focus on reading .txt (tab delimitted) and .csv (comma separated values) data files. In all cases, we will read a data file into a dataframe. That's why being able to manipulate a dataframe is very important. You need to make sure that either the data file exists in your current

working director, or you give a path to find the location of the file. Other than providing the name of the file, you would enter a sequence of parameters. please see ?read.table or ?read.csv to get an idea.

- **read.table()** to read a .txt data file, and read.csv() for .csv files
- source() to bring .r files and make the code inside the file available
- write.table(), write.csv() to export data into a file.

mydata <- read.table("c:/mydata.csv", header=TRUE, sep=",", row.names="id")

After working with a dataset, we might like to save it.

write.table(mydata, "c:/mydata.txt", sep="\t")

Important parameters

- hearder=TRUE the first row is the header
- sep="\t" tab delimitted
- sep=","
-

In [30]:

```
getwd() #gives you the current working directory
#pay attention to the way that a directory is represented in your OS
```

'/srv/home/asefn/ReDevFIT5197'

In [31]:

```
dir() # a list of files and folders
```

'airfoil_self_noise.txt' 'avgpm25.csv' 'GoogleTrends.csv' 'GoogleTrends.txt'
'Inro5.ipynb' 'Intro3.ipynb' 'Loops.ipynb' 'Lynda.ipynb' 'myplot.pdf'
'ProteinTertiary.csv' 'R0001.ipynb' 'R0002.ipynb' 'R0003.ipynb'
'R000-MarkDownPractice.ipynb' 'R_intro2.ipynb' 'R_intro.ipynb'
'Untitled.ipynb' 'workingDirectory'

In [10]:

```
data <- read.table(file='airfoil_self_noise.txt')
```

In [33]:

```
str(data)
```

```
'data.frame':   1503 obs. of  6 variables:
 $ V1: int  800 1000 1250 1600 2000 2500 3150 4000 5000 6300 ...
 $ V2: num  0 0 0 0 0 0 0 0 0 0 ...
 $ V3: num  0.305 0.305 0.305 0.305 0.305 ...
 $ V4: num  71.3 71.3 71.3 71.3 71.3 71.3 71.3 71.3 71.3 71.3 ...
 $ V5: num  0.00266 0.00266 0.00266 0.00266 0.00266 ...
 $ V6: num  126 125 126 128 127 ...
```

In [12]:

```
dim(data)
head(data)
```

1503 6

V1	V2	V3	V4	V5	V6
800	0	0.3048	71.3	0.00266337	126.201
1000	0	0.3048	71.3	0.00266337	125.201
1250	0	0.3048	71.3	0.00266337	125.951
1600	0	0.3048	71.3	0.00266337	127.591
2000	0	0.3048	71.3	0.00266337	127.461
2500	0	0.3048	71.3	0.00266337	125.571

In [1]:

```
data2 <- read.csv('ProteinTertiary.csv')
```

In [2]:

```
str(data2)
```

```
'data.frame':   45730 obs. of  11 variables:
 $ X  : int  1 2 3 4 5 6 7 8 9 10 ...
 $ V1 : num  17.28 6.02 9.28 15.85 7.96 ...
 $ V2 : num  13558 6192 7726 8425 7461 ...
 $ V3 : num  4305 1623 1726 2368 1737 ...
 $ V4 : num  0.318 0.262 0.223 0.281 0.233 ...
 $ V5 : num  162.2 53.4 67.3 67.8 52.4 ...
 $ V6 : num  1872791 803447 1075648 1210472 1021020 ...
 $ V7 : num  215.4 87.2 81.8 109.4 94.5 ...
 $ V8 : num  4288 3329 2981 3248 2814 ...
 $ V9 : int  102 39 29 70 41 15 70 74 39 26 ...
 $ V10: num  27 38.5 38.8 39.1 39.9 ...
```

In [3]:

```
a <- data2[1:5,1:5]
```

In [4]:

```
a
```

X	V1	V2	V3	V4
1	17.284	13558.30	4305.35	0.31754
2	6.021	6191.96	1623.16	0.26213
3	9.275	7725.98	1726.28	0.22343
4	15.851	8424.58	2368.25	0.28111
5	7.962	7460.84	1736.94	0.23280

In [5]:

```
str(a)
```

```
'data.frame':   5 obs. of  5 variables:
 $ X : int  1 2 3 4 5
 $ V1: num  17.28 6.02 9.28 15.85 7.96
 $ V2: num  13558 6192 7726 8425 7461
 $ V3: num  4305 1623 1726 2368 1737
 $ V4: num  0.318 0.262 0.223 0.281 0.233
```

In [6]:

```
write.csv(a, file="mydata.csv")
```

In [7]:

```
dir()
```

'airfoil_self_noise.txt' 'avgpm25.csv' 'Data8' 'GoogleTrends.csv'
'GoogleTrends.txt' 'Inro5.ipynb' 'Intro3.ipynb' 'Loops.ipynb'
'LYNDA1-DataSetsR.ipynb' 'LYNDA2-HT.ipynb' 'LYNDA2.ipynb' 'Lynda.ipynb'
'mydata.csv' 'myplot.pdf' 'ProteinTertiary.csv' 'R0001.ipynb' 'R0002.ipynb'
'R0003.ipynb' 'R0004.ipynb' 'R0005.ipynb' 'R0006.ipynb' 'R0007.ipynb'
'R000-MarkDownPractice.ipynb' 'R_intro2.ipynb' 'R_intro.ipynb'
'Untitled.ipynb' 'workingDirectory'

In [8]:

```
write.table(a, file="mydata222.txt")
```

In [9]:

```
dir()
```

'airfoil_self_noise.txt' 'avgpm25.csv' 'Data8' 'GoogleTrends.csv'
'GoogleTrends.txt' 'Inro5.ipynb' 'Intro3.ipynb' 'Loops.ipynb'
'LYNDA1-DataSetsR.ipynb' 'LYNDA2-HT.ipynb' 'LYNDA2.ipynb' 'Lynda.ipynb'
'mydata222.txt' 'mydata.csv' 'myplot.pdf' 'ProteinTertiary.csv' 'R0001.ipynb'
'R0002.ipynb' 'R0003.ipynb' 'R0004.ipynb' 'R0005.ipynb' 'R0006.ipynb'
'R0007.ipynb' 'R000-MarkDownPractice.ipynb' 'R_intro2.ipynb' 'R_intro.ipynb'
'Untitled.ipynb' 'workingDirectory'

In [ ]:

In [11]:

```
str(longley)
```

```
'data.frame':   16 obs. of  7 variables:
 $ GNP.deflator: num  83 88.5 88.2 89.5 96.2 ...
 $ GNP         : num  234 259 258 285 329 ...
 $ Unemployed  : num  236 232 368 335 210 ...
 $ Armed.Forces: num  159 146 162 165 310 ...
 $ Population  : num  108 109 110 111 112 ...
 $ Year        : int  1947 1948 1949 1950 1951 1952 1953 1954 1955 195
6 ...
 $ Employed    : num  60.3 61.1 60.2 61.2 63.2 ...
```

In [ ]:

```
# Split up longley
a1 <- longley[1:14, 1:6]  # Starting data
a2 <- longley[1:14, 6:7]  # New column to add (with "Year" to match)
b <- longley[15:16, ]     # New rows to add
write.table(a1, "~/Desktop/R/longley.a1.txt", sep="\t")
write.table(a2, "~/Desktop/R/longley.a2.txt", sep="\t")
write.table(b, "~/Desktop/R/longley.b.txt", sep="\t")
rm(list=ls()) # Clear out everything to start fresh

# Import data
a1t <- read.table("~/Desktop/R/longley.a1.txt", sep="\t")
a2t <- read.table("~/Desktop/R/longley.a2.txt", sep="\t")
```

3.4 Manageing your files

- getwd(): to get the current working directory, inessence where you are now
- setwd(): to change the working directory
- dir(): gives youa list of all files and folders
- ls(): list a exisiting variables

In [41]:

```
#options() # gives you the setting of R. Most of its parameters are not changeable .
```

```
function (...)
.Internal(options(...))
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

3.5 Built-in Datasets