21BDS0340

Abhinav Dinesh Srivatsa

Programming for Data Science Lab

Digital Assignment – I

Code

```
# BASIC MATH
# sum
300.1 + 200.3 + 300.34

# square root
sqrt(256)

# log and trig methods
log10(100) * cos(pi)

# cumulative sum
cumsum(c(2, 3, 4, 5, 6))

# reverse cumulative sum
cumsum(rev(c(2, 3, 4, 5, 6)))
```

Output

```
> # BASIC MATH
> # sum
> 300.1 + 200.3 + 300.34
[1] 800.74
>
> # square root
> sqrt(256)
[1] 16
>
> # log and trig methods
> log10(100) * cos(pi)
[1] -2
>
> # cumulative sum
> cumsum(c(2, 3, 4, 5, 6))
[1]  2  5  9 14 20
>
> # reverse cumulative sum
> cumsum(rev(c(2, 3, 4, 5, 6)))
[1]  6 11 15 18 20
```

Code

```
# ALPHABETS
LETTERS[19]
```

```
letters[19]

# last letter
letters[length(letters)]
```

Output

```
> # ALPHABETS
> LETTERS[19]
[1] "S"
> letters[19]
[1] "s"
>
> # last letter
> letters[length(letters)]
[1] "z"
```

Code

```
# READING DATASET
Titanic
head(Titanic)
```

Output

```
> head(Titanic)
, , Age = Child, Survived = No

      Sex
Class  Male Female
  1st     0      0
  2nd     0      0
  3rd    35     17
  Crew    0      0

, , Age = Adult, Survived = No

      Sex
Class  Male Female
  1st   118      4
  2nd   154     13
  3rd   387     89
  Crew  670      3

, , Age = Child, Survived = Yes

      Sex
Class  Male Female
  1st     5      1
  2nd    11     13
  3rd    13     14
  Crew    0      0

, , Age = Adult, Survived = Yes
```

```
      Sex
Class  Male Female
  1st    57    140
  2nd    14     80
  3rd    75     76
  Crew  192     20
```

Code

```
# SIMPLE OBJECTS
x <- 10
y <- 20
z <- x + y
z
```

Output

```
> # SIMPLE OBJECTS
> x <- 10
> y <- 20
> z <- x + y
> z
[1] 30
```

Code

```
# VECTORS
myvec <- c(x, y, z)
max(myvec)
min(myvec)
length(myvec)
var(myvec)

rainfall_of_2020 <-c(0.1, 0.6, 0.8, 0.9, 0.1,
                     0.4, 0.5, 0.6, 0.9, 0.4,
                     0.4, 0.2)
mean(rainfall_of_2020)
sd(rainfall_of_2020)
cumsum(rainfall_of_2020)
which.max(rainfall_of_2020)
which.min(rainfall_of_2020)

x <- c(1, 2, 5, 9, 11)
y <- c(2, 5, 1, 0, 23)
intersect(x,y)
setdiff(x, y)
setdiff(y, x)
union(x, y)
c(x, y)
```

Output

```
> # VECTORS
```

```
> myvec <- c(x, y, z)
> max(myvec)
[1] 30
> min(myvec)
[1] 10
> length(myvec)
[1] 3
> var(myvec)
[1] 100
> rainfall_of_2020 <-c(0.1, 0.6, 0.8, 0.9, 0.1,
+                      0.4, 0.5, 0.6, 0.9, 0.4,
+                      0.4, 0.2)
> mean(rainfall_of_2020)
[1] 0.4916667
> sd(rainfall_of_2020)
[1] 0.2810963
> cumsum(rainfall_of_2020)
 [1] 0.1 0.7 1.5 2.4 2.5 2.9 3.4 4.0 4.9 5.3 5.7 5.9
> which.max(rainfall_of_2020)
[1] 4
> which.min(rainfall_of_2020)
[1] 1
> x <- c(1, 2, 5, 9, 11)
> y <- c(2, 5, 1, 0, 23)
> intersect(x,y)
[1] 1 2 5
> setdiff(x, y)
[1]  9 11
> setdiff(y, x)
[1]  0 23
> union(x, y)
[1]  1  2  5  9 11  0 23
> c(x, y)
 [1]  1  2  5  9 11  2  5  1  0 23
```

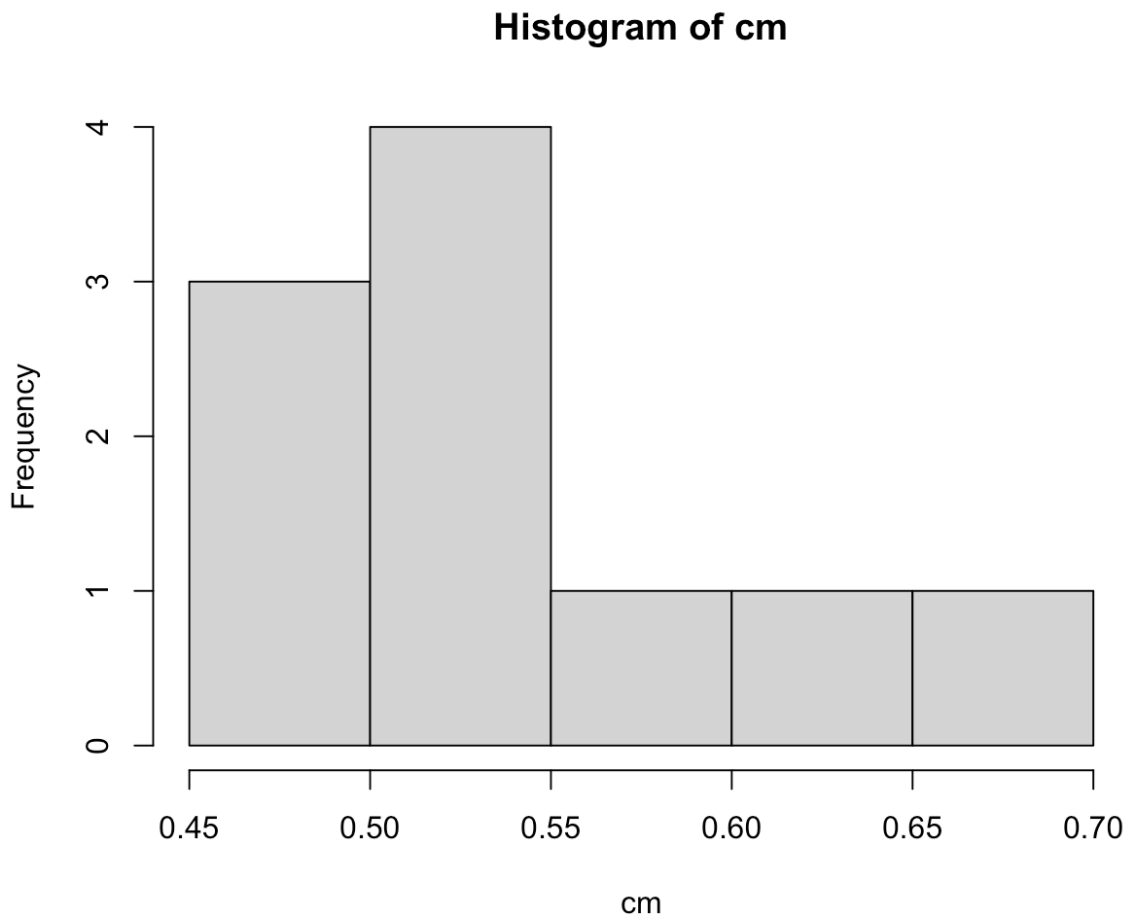
Code

```
# MATRICES
m <- matrix(runif(100), ncol = 10)
rowMeans(m)
sd(rowMeans(m))

m <-matrix(runif(100),ncol=10)
cm <-colMeans(m)
cm
hist(cm)
```

Output

```
> m <- matrix(runif(100), ncol = 10)
> rowMeans(m)
 [1] 0.4778937 0.5027204 0.4014370 0.4948259 0.5073903 0.6161583 0.4782482
0.5407035 0.4567420 0.5404460
```

```
> sd(rowMeans(m))
[1] 0.05714941
>
> m <-matrix(runif(100),ncol=10)
> cm <-colMeans(m)
> cm
 [1] 0.6056659 0.4703680 0.5182126 0.5468369 0.4621683 0.6947287 0.5383646
0.4717294 0.5380697 0.5741593
> hist(cm)
```



Code

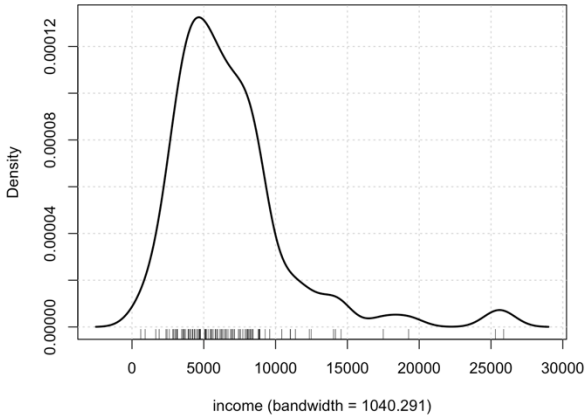
```
# PACKAGES
install.packages("car")
library(car)
# get help of density plot
??densityplot
example(densityPlot)
```

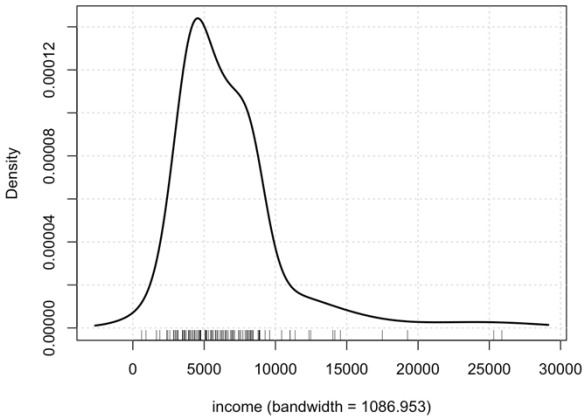
Output

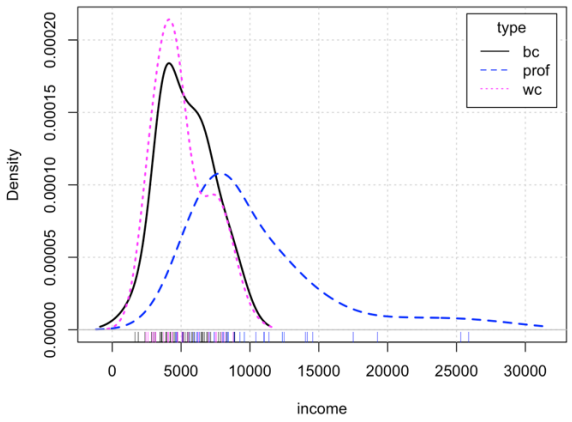
```
> # PACKAGES
> install.packages("car")
trying URL 'https://cran.rstudio.com/bin/macosx/big-sur-arm64/contrib/4.2/car_3.1-
2.tgz'
Content type 'application/x-gzip' length 1711787 bytes (1.6 MB)
```

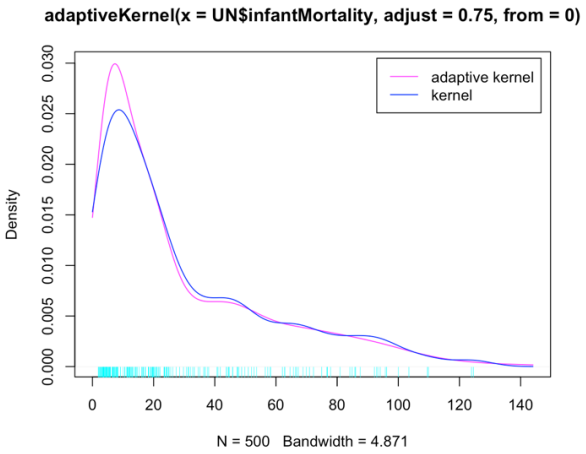
```
==================================================
downloaded 1.6 MB


The downloaded binary packages are in

/var/folders/2f/9fz2wbqj7vlcygt681kl2k0m0000gn/T//RtmpRS5Xox/downloaded_packages
> library(car)
Loading required package: carData
> # get help of density plot
> ??densityplot
> example(densityPlot)

dnstyP> densityPlot(~ income, show.bw=TRUE, method="kernel", data=Prestige)
Hit <Return> to see next plot:

dnstyP> densityPlot(~ income, show.bw=TRUE, data=Prestige)
Hit <Return> to see next plot:

dnstyP> densityPlot(~ income, from=0, normalize=TRUE, show.bw=TRUE, data=Prestige)
Hit <Return> to see next plot:

dnstyP> densityPlot(income ~ type, data=Prestige)
Hit <Return> to see next plot:

dnstyP> densityPlot(~ income, show.bw=TRUE, method="kernel", data=Prestige)
Hit <Return> to see next plot:

dnstyP> densityPlot(~ income, show.bw=TRUE, data=Prestige)
Hit <Return> to see next plot:

dnstyP> densityPlot(~ income, from=0, normalize=TRUE, show.bw=TRUE, data=Prestige)
Hit <Return> to see next plot:

dnstyP> densityPlot(income ~ type, kernel=depan, data=Prestige)
Hit <Return> to see next plot:

dnstyP> densityPlot(income ~ type, kernel=depan, legend=list(location="top"),
data=Prestige)
Hit <Return> to see next plot:

dnstyP> plot(adaptiveKernel(UN$infantMortality, from=0, adjust=0.75),
col="magenta")
Hit <Return> to see next plot:

dnstyP> lines(density(na.omit(UN$infantMortality), from=0, adjust=0.75),
col="blue")

dnstyP> rug(UN$infantMortality, col="cyan")

dnstyP> legend("topright", col=c("magenta", "blue"), lty=1,
```

```
dnstyP+         legend=c("adaptive kernel", "kernel"), inset=0.02)

dnstyP> plot(adaptiveKernel(UN$infantMortality, from=0, adjust=0.75),
col="magenta")
Hit <Return> to see next plot:

dnstyP> lines(density(na.omit(UN$infantMortality), from=0, adjust=0.75),
col="blue")

dnstyP> rug(UN$infantMortality, col="cyan")

dnstyP> legend("topright", col=c("magenta", "blue"), lty=1,
dnstyP+         legend=c("adaptive kernel", "kernel"), inset=0.02)
```









Code

```
# STRINGS
str <- "Hello, Data Visualization!"
print(str)
```

Output

```
> # STRINGS
> str <- "Hello, Data Visualization!"
> print(str)
```

```
[1] "Hello, Data Visualization!"
```

Code

```
# COMMENTS
# My first  R Programming
help() # give help regarding a command, e.g. help(hist)

dim()  # gives the number of rows and the number of columns of a matrix, or a data
frame
head() # gives the first 6 rows of a large matrix, or data frame
tail() # gives the last 6 rows of a large matrix, or data frame
m[ ,3] # gives the 3rd column of the matrix m
m[2, ] # gives the 2nd row of the matrix m

sum()     # get the sum of the values in x by sum(x)
mean()    # get the mean of the values in x by mean(x)
median()  # get the median of the values in x by median(x)
sd()      # get the standard deviation of the values in x
var()     # get the variance of the values in x
IQR()     # get the IQR of the values in x
summary() # get the summary statistics of a single variable, or of all variables in
a data frame
round()   # round values in x to 3 decimal places by round(x,3)
sort()    # sort the values in x by giving sort(x)
unique()  # get the non-duplicate values from a list, e.g. x = c(3,5,7,2,3,5,9,3)
and then
unique(x) # gives 3 5 7 2 9
length(x) # gives the length of the vector x, which is 8
hist()    # create a histogram of the values in x by hist(x)
stem()    # create a stem and leaf plot of the values in x by stem(x)
boxplot() # create a boxplot of the values in x by boxplot(x)
plot()    # scatterplot of x vs. y by plot(x,y); for more parameters see
help(plot.default)
cor()     # gives the linear correlation coefficient
lm()      # fit a least squares regression of y (response) on x (predictor) by fit
= lm(y~x)
names()   # get or set the names of elements in a R object. E.g. names(fit) will
give the names of the R object named “fit”, or get or set the names of variables in
a data frame.

fit$coef      #gives the least squares coefficients from the fit above, i.e.
intercept and slope
fit$fitted    # gives the fitted values for the regression fitted above
fit$residuals # gives the residuals for the regression fitted above
lines()       # add a (regression) line to a plot by lines(x,fit$fitted)
abline()      # add a straight line to a scatter plot
points()      # add additional points (different plotting character) to a plot by
points(x, y2, pch = 5)
scan()        # read data for one variable from a text file, e.g. y =
scan("ping.dat") Don't forget to change to the appropriate directory first
```

```
read.table()  # read spreadsheet data (i.e. more than one variable) from a text
file
table()       # frequency counts of entries, ideally the entries are
factors(although it works with integers or even reals)
write()       # write the values of a variable y in a file data.txt by
write(y,file="data.txt")
log()         # natural logarithm (i.e. base e)
log10()       # logarithm to base 10
seq()         # create a sequence of integers from 2 to 11 by increment 3 with
seq(2,11,by=3)
rep()         # repeat n times the value x, e.g. rep(2,5) gives 2 2 2 2 2
getwd()       # get the current working directory.
setwd()       # change the directory to. E.g. setwd("c:/Ritchie/RRR.doc")
dir()         # list files in the current working directory
search()      # searching through reachable datasets and packages
library()     # link to a downloaded R package to the current R session. E.g.
library(Biostrings) link to the
```