R Guideline
Basics

Useful R commands

Command	Functionality
getwd()	To obtain current working directory
setwd()	To set up the working directory
ls()	To list all variables
rm()	To remove all variables
fix(X)	To open another window to view or change values of X
#	To comment

Note that you can type R commands in script file (File -> New Script). To run R commands in the script file, highlight those lines and press Ctrl + R.

Data types

1. Character
   a. Example: “statistics”, “123”
2. Numeric: for real numbers
   a. Example: 1.4, pi
3. Integer
   a. Example: 1, 2, 3
4. logical: for Boolean values
   a. Two possible values: TRUE, FALSE

Useful functions

Name	Functionality
class	To check type
is.X	To check whether variable is type X or not. Return TRUE if it is type X.
as.X	To change variable to type X

Example:

```
> x <- 100
> y <- "cat"
> z <- TRUE
> class(x)
[1] "numeric"
> class(y)
[1] "character"
> class(z)
[1] "logical"
> is.integer(x)
```

```
[1] FALSE
> is.numeric(x)
[1] TRUE
> is.character(y)
[1] TRUE
> is.logical(z)
[1] TRUE
> xInt <- as.integer(x)
> xInt
[1] 100
> is.integer(xInt)
[1] TRUE
> xChar <- as.character(x)
> xChar
[1] "100"
> is.character(xChar)
[1] TRUE
```

Basic data structure

There are multiple data structures in R. For example: vector, matrix, factor, dataframe.

Vector: Array-like combination of elements. Use [*index*] to specify element where index starts from 1.

Example:

```
> v1 <- c(1,2,3)
> v1
[1] 1 2 3
> v2 <- c("Ant", "Bee", "Cat")
> v2
[1] "Ant" "Bee" "Cat"
> v3 <- c("Bob", 18, 3.91)
> v3
[1] "Bob"  "18"   "3.91"
> v2[1]
[1] "Ant"
> v3[3]
[1] "3.91"
> v2[2:3]
[1] "Bee" "Cat"
```

Matrix: multi-dimensional array of values, where all the values must be the same length and type. Use [index1, index2, …] to specify element where indices start from 1.

matrix: to construct a matrix with specified number of rows (*nrow*) and number of columns (*ncol*).

dim: to identify dimension of matrix.

cbind: to combine multiple vectors into matrix columnwise.

rbind: to combine multiple vectors into matrix rowwise.

Example:

```
> m <- matrix(c(1,2,3,4,5,6), nrow=2, ncol=3)
> m
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> dim(m)
[1] 2 3
> m[2,]
[1] 2 4 6
> m[,2]
[1] 3 4
> m2 <- matrix(c("Ant", "Bee", "Cat", "Dog"), nrow=2, ncol=2)
> m2
     [,1]  [,2] 
[1,] "Ant" "Cat"
[2,] "Bee" "Dog"
> v <- c(1,2,3)
> v2 <- c(4,5,6)
> m3 <- cbind(v,v2)
> m3
     v v2
[1,] 1  4
[2,] 2  5
[3,] 3  6
> m4 <- rbind(v,v2)
> m4
   [,1] [,2] [,3]
v     1    2    3
v2    4    5    6
```

Factor: used to represent categorical data.   Factor has integer corresponding with each unique label, where the integer represents frequency of such label.

factor: to construct factor from vector.

levels: to display the list of unique labels.

summary: to display integer values corresponding to the labels.

as.numeric: to convert factor labels to numeric values.

cut: to convert vector to factor based on the *breaks* parameter, where *breaks* can be number of intervals or vector containing intervals.

Example:

```
> v <- c("Ant", "Bee", "Cat", "Bee", "Bee", "Ant", "Cat", "Ant")
> f <- factor(v)
> f
[1] Ant Bee Cat Bee Bee Ant Cat Ant
Levels: Ant Bee Cat
> levels(f)
[1] "Ant" "Bee" "Cat"
```

```
> summary(f)
Ant Bee Cat 
  3   3   2 
> vn <- as.numeric(f)
> vn
[1] 1 2 3 2 2 1 3 1
> v2 <- c(1, 2, 9, 1, 2, 5, 6, 8, 8, 4, 5, 6)
> f2 <- cut(v2, breaks=3)
> summary(f2)
(0.992,3.67]  (3.67,6.33]  (6.33,9.01] 
           4            5            3 
> f3 <- cut(v2, breaks=c(0,2,4,6,8,10))
> summary(f3)
 (0,2]  (2,4]  (4,6]  (6,8] (8,10] 
     4      1      4      2      1 
```

<u>Data frame:</u> used to store table of data.   Data frame contains data with the same length, but they can be different types (In addition, type can be factor).   To specify each column, use "$" (see Example below.)

data.frame: to contruct data frame with specified *row.names*. If characters are used as values, column will be converted to factor automatically, unless setting *stringsAsFactors* to FALSE.

names: to display column headers.

sapply(X, class): to display type of each column of X.

dim: to display dimension of data frame.

head: to display the first n rows. Default value of n is 6.

tail: to display the last n rows. Default value of n is 6.

summary: to display 6 descriptive statistics: min, max, mean, median, first and third quartile.

Note that in R, there is an inherited data frame structure called table. This table structure is almost similar to data frame but works more efficiently.

<u>Example</u>:

```
> name <- c("Ann", "Bob", "Cat", "David")
> year <- c(1,2,3,4)
> gpa <- c(2.4,3.2,2.8,3.4)
> df <- data.frame(name,year,gpa,
+                  row.names=c("S1", "S2", "S3", "S4"),
+                  stringsAsFactors = FALSE)
> df
    name year gpa
S1   Ann    1 2.4
S2   Bob    2 3.2
S3   Cat    3 2.8
S4 David    4 3.4
> df$name
[1] "Ann"   "Bob"   "Cat"   "David"
> df$gpa
```

```
[1] 2.4 3.2 2.8 3.4
> dim(df)
[1] 4 3
> names(df)
[1] "name" "year" "gpa"
> sapply(df, class)
       name        year         gpa
"character"   "numeric"   "numeric"
> head(df)
    name year gpa
S1   Ann    1 2.4
S2   Bob    2 3.2
S3   Cat    3 2.8
S4 David    4 3.4
> tail(df, n=2)
    name year gpa
S3   Cat    3 2.8
S4 David    4 3.4
> summary(df)
     name                year           gpa
 Length:4           Min.   :1.00   Min.   :2.40
 Class :character   1st Qu.:1.75   1st Qu.:2.70
 Mode  :character   Median :2.50   Median :3.00
                    Mean   :2.50   Mean   :2.95
                    3rd Qu.:3.25   3rd Qu.:3.25
                    Max.   :4.00   Max.   :3.40
```

Indexing data frame:

- The first and the second indices of data frame respectively are row and column indices.
- Index can be a vector containing specific row(or column) indices to be displayed.
- Boolean indexing can be applied as a simple search for data frame values.
- which: can be used to find row indices that satisfy Boolean condition.

```
> name <- c("Ann", "Bob", "Cat", "David")
> year <- c(1,2,3,4)
> gpa <- c(2.4,3.2,2.8,3.4)
> df <- data.frame(name,year,gpa,
+                  row.names=c("S1", "S2", "S3", "S4"),
+                  stringsAsFactors = FALSE)
> df
    name year gpa
S1   Ann    1 2.4
S2   Bob    2 3.2
S3   Cat    3 2.8
S4 David    4 3.4
> df[1,]
   name year gpa
S1  Ann    1 2.4
> df[,1]
[1] "Ann"   "Bob"   "Cat"   "David"
> df[2,3]
[1] 3.2
```

```
> df[c(1,4),]
    name year gpa
S1   Ann    1 2.4
S4 David    4 3.4
> df[,c("name","gpa")]
    name gpa
S1   Ann 2.4
S2   Bob 3.2
S3   Cat 2.8
S4 David 3.4
> df$gpa > 3
[1] FALSE  TRUE FALSE  TRUE
> df[df$gpa > 3,]
    name year gpa
S2   Bob    2 3.2
S4 David    4 3.4
> I <- which(df$gpa > 3)
> I
[1] 2 4
> df[I,]
    name year gpa
S2   Bob    2 3.2
S4 David    4 3.4
```

Descriptive Statistics

R provides function to compute basic descriptive statistics as shown in the table:

Name	**Functionality**
mean(X)	Find mean (average) of X
median(X)	Find median of X
min(X)	Find minimum of X
max(X)	Find maximum of X
sum(X)	Find summation of X
cumsum(X)	Find cumulative summation of X
range(X)	Find range (difference between min and max) of X
var(X)	Find variance of X
sd(X)	Find standard deviation of X
quantile(X)	Find 5 quantile statistics (0%, 25%, 50%, 75%, 100%) of X
quantile(X, p)	Find pth percentile of X (Example of p: 0.25, 0.7)
IQR(X)	Find inter-quartile range
length(X)	Find length of X

Note that there is no mode function provided in R. We can use factor and print summary of such factor to find mode.   In the example below, since 1 has the highest frequency, mode of vector x is 1.

Example:

```
> x <- c(1, 2, 3, 1, 2, 3, 1, 2, 3, 1)
> f <- factor(x)
> summary(f)
1 2 3
4 3 3
```

Basic Input

R can read input file from many file formats. For example: text, Excel, SAS, SPSS, Stata, etc.

In this guideline, the csv file format is focused here.

read.csv: read csv input file and return result as data frame.

```
> x <- read.csv("week3_ex1_data.csv", sep=",", header=T, fileEncoding="UTF-
8-BOM")
> class(x)
[1] "data.frame"
> dim(x)
[1] 1000    7
> head(x)
                   Title     Genre             Director Year Runtime Rating
1 Guardians of the Galaxy    Action           James Gunn 2014     121    8.1
2              Prometheus Adventure         Ridley Scott 2012     124    7.0
3                   Split    Horror   M. Night Shyamalan 2016     117    7.3
4                    Sing Animation Christophe Lourdelet 2016     108    7.2
5           Suicide Squad    Action           David Ayer 2016     123    6.2
6          The Great Wall    Action          Yimou Zhang 2016     103    6.1
   Votes
1 757074
2 485820
3 157606
4  60545
5 393727
6  56036
```

Basic vector commands

To handle vector, here are basic commands.

Sequence: order of elements. Return output as vector

Example: seq(from, to, incremental value)

```
> x1 <- seq(0,1,0.1)
> x1
 [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
> is.vector(x1)
 [1] TRUE
> seq(100, 85, by=-1)
 [1] 100  99  98  97  96  95  94  93  92  91  90  89  88  87  86  85
```

Replicate: replication of elements   Return output as vector.

Example: rep(X, times = n1, length=n2, each = n3) where X can be a single value or a vector. n1 is number of times for X to be repeated, n2 is length of output vector, and n3 is number of times that each element in X is repeated.

```
> x1 <- rep(10,5)
> x1
 [1] 10 10 10 10 10
> is.vector(x1)
 [1] TRUE
> rep(c(1,2,3), times=3)
 [1] 1 2 3 1 2 3 1 2 3
> rep(c(1,2,3), times=3, length=10)
 [1] 1 2 3 1 2 3 1 2 3 1
> rep(c(1,2,3), each=2)
 [1] 1 1 2 2 3 3
> rep(c(1,2,3), times=3, each=2)
 [1] 1 1 2 2 3 3 1 1 2 2 3 3 1 1 2 2 3 3
```

Distribution

R provides distributions that we can use.   They are available in the following formats:

dDIST: for probability mass or density distribution (pdf or pmf)
pDIST: for cumulative distribution (cdf)
qDIST: for inverse cdf
rDIST: for random variable with DIST distribution

Replace DIST with name used in R as shown in the following table:

Distribution	**DIST name used in R**	**Arguments**
Binomial	binom	size, prob
Chi-squared	Chisq	df, ncp
Exponential	exp	rate
F	f	df1, df2, ncp
Geometric	geom	prob
Normal (Gaussian)	norm	mean, sd
Poisson	pois	lambda
Student’s t	t	df, ncp*
Uniform	unif	min, max

*ncp = non-centrality parameter. Mostly, we set to zero.

Example: Try with uniform distribution with minimum = 0 and maximum = 5.

```
> xmin <- 0
> xmax <- 10
> xinterval <- 1
> x <- seq (xmin, xmax, by=xinterval)
> x
 [1]  0  1  2  3  4  5  6  7  8  9 10
```

```
> min <- 0
> max <- 5
> pdf <- dunif(x, min, max)
> pdf
 [1] 0.2 0.2 0.2 0.2 0.2 0.2 0.0 0.0 0.0 0.0 0.0
> cdf <- punif(x, min, max)
> cdf
 [1] 0.0 0.2 0.4 0.6 0.8 1.0 1.0 1.0 1.0 1.0 1.0
> temp_x <- qunif(cdf, min, max)
> temp_x
 [1] 0 1 2 3 4 5 5 5 5 5 5
> n <- 5
> random_x <- runif(n, min, max)
> random_x
[1] 0.09836707 3.98996503 1.37159448 0.83304552 0.85075859
```

We can also use pDist and qDist to help look up values from the tables.

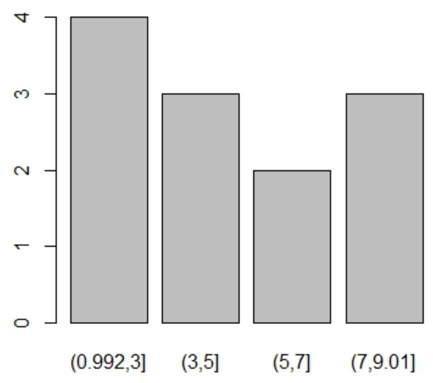
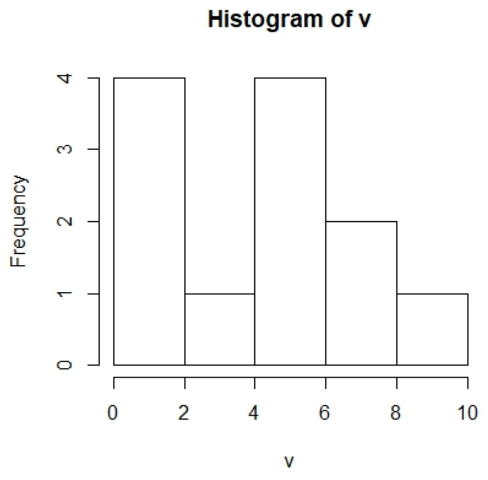
Example:

```
> # z-table  (normal CDF)
> # get normal CDF (area under normal curve from -infinity to x)
> x <- c(1.2816, 1.6449, 1.96, 2.3263, 2.5758)
> pnorm(x, mean = 0, sd = 1)
[1] 0.9000085 0.9500048 0.9750021 0.9899987 0.9949996
>
> # inverse z-table
> # get value of x from given area (or alpha)
> alpha <- c(0.005, 0.01, 0.025, 0.05, 0.1)
> qnorm(alpha, mean = 0, sd = 1)
[1] -2.575829 -2.326348 -1.959964 -1.644854 -1.281552
> alpha2 <- c(0.995, 0.99, 0.975, 0.95, 0.9)
> qnorm(alpha2, mean = 0, sd = 1)
[1] 2.575829 2.326348 1.959964 1.644854 1.281552
>
> # t-table
> # get value of t from given area (or alpha)
> alpha <- c(0.9, 0.95, 0.975, 0.99, 0.995)
> qt(alpha, df=10)
[1] 1.372184 1.812461 2.228139 2.763769 3.169273
> qt(alpha, df=20)
[1] 1.325341 1.724718 2.085963 2.527977 2.845340
>
> # chisq table
> # value of x from given area (or alpha)
> alpha <- c(0.005, 0.01, 0.025, 0.05, 0.1)
> qchisq(alpha, df=10)
[1] 2.155856 2.558212 3.246973 3.940299 4.865182
> alpha2 <- c(0.9, 0.95, 0.975, 0.99, 0.995)
> qchisq(alpha2, df=10)
[1] 15.98718 18.30704 20.48318 23.20925 25.18818
> alpha3 <- c(0.025, 0.975)
> qchisq(alpha3, df=24)
[1] 12.40115 39.36408
>
> # binomial
> # get binomial CDF (area under normal curve from -infinity to x)
> pbinom(15, size=20, prob=0.9)
[1] 0.0431745
> pbinom(16, size=20, prob=0.9)
[1] 0.1329533
> pbinom(16, size=20, prob=0.8)
[1] 0.5885511
>
> # get value of x from given area (or alpha)
> alpha <- c(0.043,0.133)
> qbinom(alpha, size=20, prob=0.9)
[1] 15 17
```

Basic histogram plot

barplot(X): to plot bar graph from frequency vector X
hist(X): to create histogram plot for sample vector X
windows(): to open new graph window. On unix, use x11(). On mac, use quartz()

```
> v <- c(1, 2, 9, 1, 2, 5, 6, 8, 8, 4, 5, 6)
> f <- cut(v, breaks=4)
> windows()
> barplot(summary(f))
> windows()
> hist(v)
```

Result graph from barplot	Result graph from hist
 0 1 2 3 4 (0.992,3] (3,5] (5,7] (7,9.01]	 Histogram of v Frequency 0 1 2 3 4 0 2 4 6 8 10 v