Introduction to R

Saeed Saffari Saeed.Saffari@dal.ca

Winter 2025

Contents

1 Introduction **3**
1.1 Markdown: . . . 4

2 Basic of learning **5**
2.1 How to Print . . . 5
2.2 Data Types (Classes) . . . 5
2.3 Arithmetic Operators . . . 5
2.3.1 Practice: . . . 7
2.4 Relational Operators . . . 7
2.5 Practice: . . . 9
2.6 Loops . . . 9
2.6.1 if , elif . . . 10
2.6.2 for loops . . . 10
2.6.3 while loops . . . 11
2.6.4 Practice: . . . 11
2.7 function . . . 11
2.7.1 Practice I: . . . 12
2.7.2 Practice II: . . . 13
2.7.3 Practice III: . . . 13

3 Vetors **14**
3.1 Vector Indexing . . . 14
3.2 Matching Operator . . . 16
3.3 Vector Arithmetic's . . . 16
3.4 Vector Methods . . . 17
3.5 Logical Vector . . . 17
3.6 Factors . . . 18
3.7 Mathematical Function in R . . . 18
3.8 Random Number in R . . . 19
3.9 Practice: . . . 19

4 Matrix **20**
4.1 Creat Matrix . . . 20
4.2 Matrix diag . . . 20
4.3 Matrix Indexing . . . 21
4.4 Matrix Specific Functions . . . 22
4.5 Practice I . . . 23
4.6 Practice II . . . 24

5 Lists **25**
5.1 Creat list . . . 25

5.2 List subset Operator . . . 25

6 Dataframe **26**
6.1 Creating Dataframes . . . 26
6.2 Dataframes Indexing . . . 26
6.3 Dataframes `subset()` function for filtering . . . 27
6.4 Dataframes `rbine()` and `cbind()` . . . 27
6.5 Saving data in csv . . . 28
6.6 Missing Data . . . 28

7 Dplyr Package **30**
7.1 dplyr `select()` function . . . 32
7.2 dplyr `filter()` function . . . 37
7.3 dplyr `rename()` function . . . 40
7.4 dplyr `mutate()` function . . . 41
7.4.1 Practice: . . . 41
7.5 dplyr `group_by()` function . . . 42
7.6 dplyr Pipe Operator %>% . . . 43
7.6.1 Practice: . . . 46

8 Data Visualization with `dplyr` **47**
8.1 Bar Graphs . . . 47
8.2 Scatter Plots . . . 49
8.3 Line Graphs . . . 50
8.4 Box plots . . . 51
8.5 Multiple Plots in Layout . . . 52

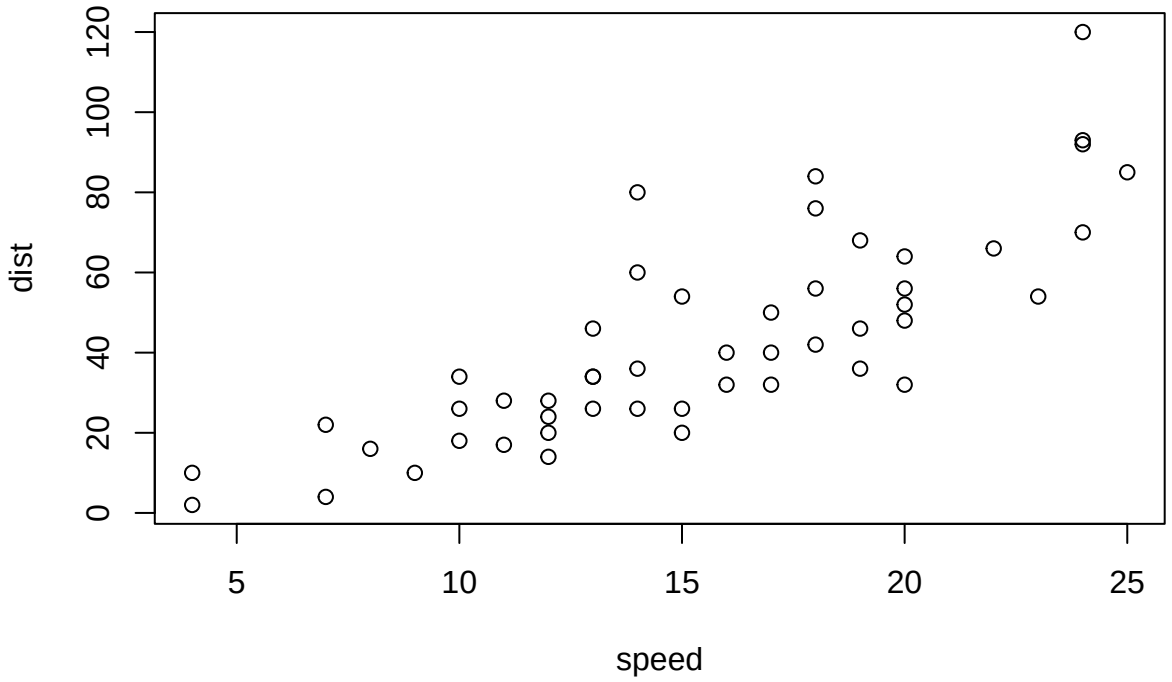
9 Regressions and Models **53**
9.1 Simple Linear Regression . . . 53
9.2 Multiple linear regression . . . 54
9.2.0.1 Practice: . . . 55
9.2.1 t-test (Comparing Means) . . . 55
9.2.2 Correlation Test . . . 56
9.2.2.1 Practice . . . 56
9.2.3 ANOVA (Analysis of Variance) . . . 56
9.2.4 Regression Plots . . . 56

1 Introduction

This is an R Markdown Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter* (on Mac) or *Ctrl+Shift+Enter* (on Windows).

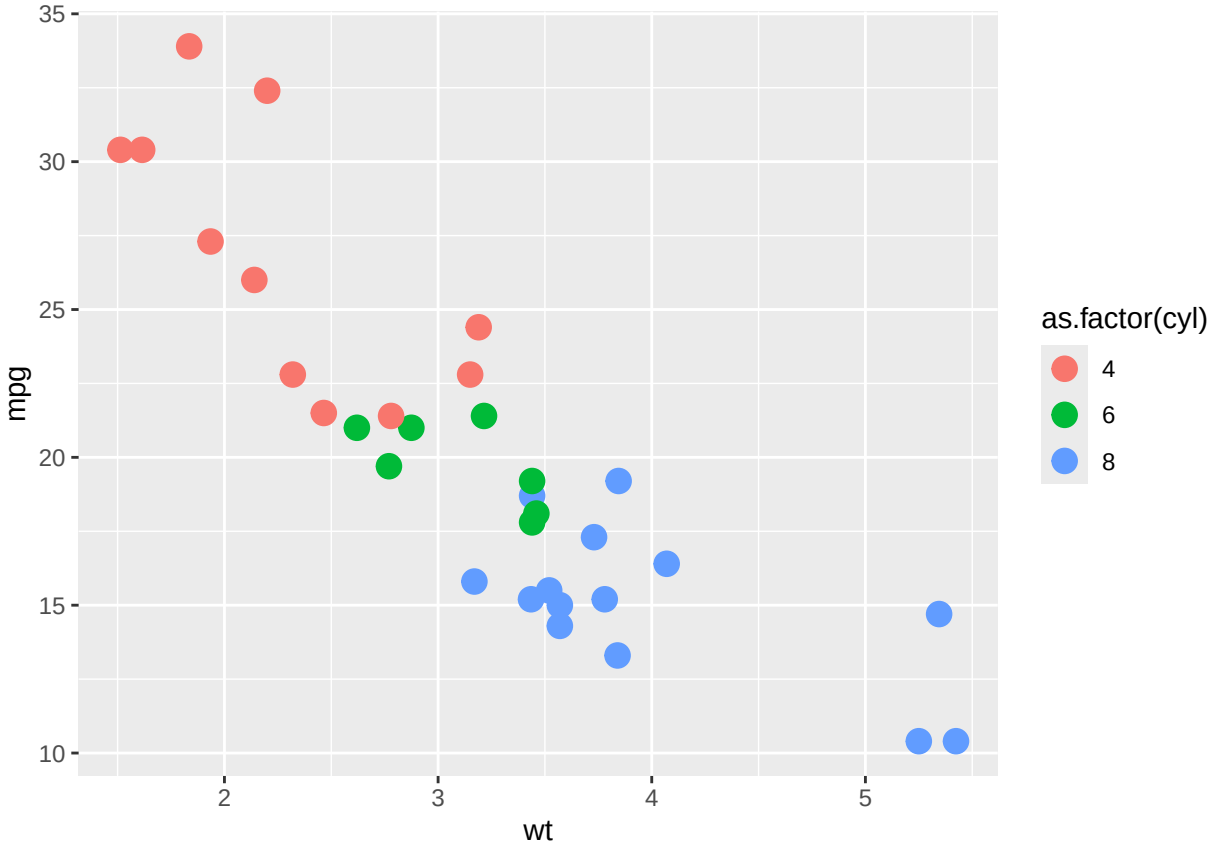
```
plot(cars)
```




```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
data("mtcars")
ggplot(data = mtcars, aes(x = wt, y= mpg)) +
  geom_point(aes(color=as.factor(cyl)), size = 4)
```



Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Cmd+Option+I* (on Mac) or *Ctrl+Alt+I* (on Windows).

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Cmd+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.

You can download and install packages with `install.packages("The name of package")`.

1.1 Markdown:

$$\alpha = \beta = \frac{-\alpha \cdot \gamma^2}{\sqrt{\sigma^3}}$$

2 Basic of learning

In this part we talk about basic elements of R programming.

2.1 How to Print

```
print('Hello world')
```

```
## [1] "Hello world"
```

```
print("This is R programming Workshop!")
```

```
## [1] "This is R programming Workshop!"
```

2.2 Data Types (Classes)

In R, data types (or classes) define the kind of data stored in variables. Here are the most common data types in R:

Class	Description	Example	Code Example
`numeric`	Represents decimal or whole numbers	3.14, 42, -7.8	`x <- 3.14; class(x)`
`character`	Represents text strings	"Hello", "R Programming"	`z <- "Hello"; class(z)`
`logical`	Represents Boolean values (TRUE or FALSE)	TRUE, FALSE	`is_valid <- TRUE; class(is_valid)`
`complex`	Represents complex numbers	2+3i, 1-4i	`c <- 2 + 3i; class(c)`
`list`	Represents a collection of different types	A list of numbers, text	`lst <- list(1, "Hello", TRUE); class(lst)`

```
print(class(3.14))
```

```
## [1] "numeric"
```

```
print(class('R programming'))
```

```
## [1] "character"
```

```
print(class(TRUE))
```

```
## [1] "logical"
```

2.3 Arithmetic Operators

Symbol	Task Performed
+	Addition
-	Subtraction
/	division
*	multiplication
**	to the power of
^	to the power of
%%	modulus
%/%	floor division

```r
18 + 4
```

```
## [1] 22
```

```r
18 - 4
```

```
## [1] 14
```

```r
18 * 4
```

```
## [1] 72
```

```r
18 / 4
```

```
## [1] 4.5
```

```r
2 ** 3
```

```
## [1] 8
```

```r
2 ^ 3
```

```
## [1] 8
```

```r
18 %% 4
```

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

```r
18 %/% 4
```

```
## [1] 4
```

```r
log(2)
```

```
## [1] 0.6931472
```

```r
log10(2)
```

```
## [1] 0.30103
```

```r
5 + (4 - 3 * 2)**3 + 1
```

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

We can save values in variables:

```r
x <- 18
y = 4
z <- x + y
print(z)
```

```
## [1] 22
```

```r
class(z)
```

```
## [1] "numeric"
```

In R programming, code runs line by line, with only the last assignment determining the final value of a variable.

```r
a = 5 + (4 - 3 * 2)**3 + 1
a = 10
a = a * 2
a = a - 5
a
```

```
## [1] 15
```

2.3.1 Practice:

Convert a given temperature X degrees Celsius to Fahrenheit.

$$F = C \cdot \frac{9}{5} + 32$$

```
temp = 20
fahrenheit <- (temp * 9 / 5) + 32
print(fahrenheit)
```

```
## [1] 68
```

2.4 Relational Operators

Symbol	Task Performed
<-	Assignment
=	Assignment
assign()	Assignment
==	True, if it is equal
!=	True, if not equal to
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to

```
z <- 10
y = 6
assign('x', 2)

x < y
```

```
## [1] TRUE
```

```
x >= y
```

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

```
x != y
```

```
## [1] TRUE
```

```
x == y
```

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

```
x > 2
```

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

```
x >= 2
```

```
## [1] TRUE
```

```
x > 1 & y < 10
```

```
## [1] TRUE
```

```
x > 1 & y > 10
```

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

```
x > 1 | y > 10
```

```
## [1] TRUE
```

you can use below command to get special values:

```
x <- pi
x
```

```
## [1] 3.141593
```

```
e <- exp(1)
e
```

```
## [1] 2.718282
```

```
x <- letters
x
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s"
## [20] "t" "u" "v" "w" "x" "y" "z"
```

```
x <- LETTERS
x
```

```
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S"
## [20] "T" "U" "V" "W" "X" "Y" "Z"
```

```
x <- month.name
x
```

```
##  [1] "January"   "February"  "March"     "April"     "May"       "June"
##  [7] "July"      "August"    "September" "October"   "November"  "December"
```

```
x <- month.abb
x
```

```
##  [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"
```

you can write comment with # :

```
# This line is comment!
r = 0.2 # interest rate
```

you can create sequence numbers with below command:

This work like *arange* in numpy pakage in Python

```
x <- 1:10
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
x <- 1:10 * 2
x
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

```
x <- seq(5)
x
```

```
## [1] 1 2 3 4 5
```

```
x <- seq(from=1, to=9)
x
```

```
## [1] 1 2 3 4 5 6 7 8 9
```

```
x <- seq(from=1, to=9, by=3)
x
```

```
## [1] 1 4 7
```

```
x <- seq(1,10,2)
x
```

```
## [1] 1 3 5 7 9
```

This work like *linspace* in numpy pakage in Python

```
x <- seq(1,10,length = 5)
x
```

```
## [1]  1.00  3.25  5.50  7.75 10.00
```

Replicate function:

```
x <- 1:3
x
```

```
## [1] 1 2 3
```

```
y <- rep(x, time = 5)
y
```

```
##  [1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```

```
y <- rep(x, each = 5)
y
```

```
##  [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
```

2.5 Practice:

Simulate GDP growth from the year 2000 to 2025 with an annual growth rate of 3% starting from 1000 units.

$$GDP_t = 1000 \cdot (1 + r)^n$$

```
years = 2000:2025
growth_rate <- 0.03 # rate
GDP = 1000 * (1+growth_rate)**(years - 2000)
GDP
```

```
##  [1] 1000.000 1030.000 1060.900 1092.727 1125.509 1159.274 1194.052 1229.874
##  [9] 1266.770 1304.773 1343.916 1384.234 1425.761 1468.534 1512.590 1557.967
## [17] 1604.706 1652.848 1702.433 1753.506 1806.111 1860.295 1916.103 1973.587
## [25] 2032.794 2093.778
```

2.6 Loops

2.6.1 if , elif

```
#"R" + 2
```

```
age <- 15
if (age >= 18){
  print('You are old enough to vote!')
} else {
  print('You can NOT vote yet!')
  print(paste('You can will vote after ', 18 - age, ' years.'))
}
```

```
## [1] "You can NOT vote yet!"
## [1] "You can will vote after  3  years."
```

```
age <- 16
if (age <= 4){
  price = 0
} else if (age < 16){
  price = 50
} else {
#} else if (age >= 16){
  price = 100
}

print(paste("Your cost is $", price))
```

```
## [1] "Your cost is $ 100"
```

2.6.2 for loops

```
for (i in 1:5){
  print(i)
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

```
for (i in 1:5){
  print(i*i)
}
```

```
## [1] 1
## [1] 4
## [1] 9
## [1] 16
## [1] 25
```

```
z = 0
for (i in 1:10){
  z = z + i
  print(z)
}
```

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

```
## [1] 3
## [1] 6
## [1] 10
## [1] 15
## [1] 21
## [1] 28
## [1] 36
## [1] 45
## [1] 55
```

2.6.3 while loops

```
i <- 1
while (i < 10){
  print(i)
  i <- i + 1
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
```

2.6.4 Practice:

Write a conditional statement to check whether the variable is positive, negative, or zero and print the appropriate message.

```
"It's R"
```

```
## [1] "It's R"
```

```
x <- -5
if (x > 0) {
  print('Positive')
} else if (x < 0){
  print("Negative")
} else {
  print('Zero')
}
```

```
## [1] "Negative"
```

2.7 function

```
average = function(a,b,c){
  summ = a + b + c
  ave = summ / 3
  return(ave)
}
```

```r
average(34,12,-23) * 2
```

```
## [1] 15.33333
```

```r
average(23,56,98)
```

```
## [1] 59
```

```r
price_func = function(age){
if (age <= 4){
  price = 0
} else if (age < 16){
  price = 50
} else {
  price = 100
}
print(paste("Your cost is $", price))
}

price_func(18)
```

```
## [1] "Your cost is $ 100"
```

```r
for (i in seq(1:20)){
  #print(i)
  price_func(i)
}
```

```
## [1] "Your cost is $ 0"
## [1] "Your cost is $ 0"
## [1] "Your cost is $ 0"
## [1] "Your cost is $ 0"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 50"
## [1] "Your cost is $ 100"
## [1] "Your cost is $ 100"
## [1] "Your cost is $ 100"
## [1] "Your cost is $ 100"
## [1] "Your cost is $ 100"
```

2.7.1 Practice I:

Write a function `temp()` that converts a temperature in Celsius to Fahrenheit.

```r
temp <- function(temps){
  fahrenheit <- (temps * 9 / 5) + 32
  print(fahrenheit)
```

```
}
temp(90)
```

```
## [1] 194
```

2.7.2 Practice II:

Create a function to calculate the **future value** K_n of an investment after `n` years with a given principal K and interest rate r.

The formula for compound interest is:

$$K_n = K \times (1 + r)^n$$

```
future_value <- function(k, r, n){
  k_n <- k * (1+r)^n # future value formula
  return(k_n)
}
future_value(1000, 0.05, 10)
```

```
## [1] 1628.895
```

```
future_value(1000, 0.05, 10) - 1000
```

```
## [1] 628.8946
```

2.7.3 Practice III:

Write a function in R that takes a number as input and returns whether the number is even or odd.

3 Vetors

The most common way to create vectors is to use function `c()`.

```
x <- c(10.25, 3.5, 8.75, 23.15, 12)
x
```

```
## [1] 10.25  3.50  8.75 23.15 12.00
```

```
x <- c(10.25, 3.5, 8.75, 23.15, 12, 'a', 'b', "c")
x
```

```
## [1] "10.25" "3.5"   "8.75"  "23.15" "12"    "a"     "b"     "c"
```

```
class(x)
```

```
## [1] "character"
```

```
x <- c(1,2,3,4,5,6,7)
x
```

```
## [1] 1 2 3 4 5 6 7
```

```
y <- 1:7
y
```

```
## [1] 1 2 3 4 5 6 7
```

Join vectors

```
x <- c(10,20,30,40)
y <- c(3.5, 4.75)

z <- c(x,y)
z
```

```
## [1] 10.00 20.00 30.00 40.00  3.50  4.75
```

You can find *lenght* of vectors with *lenght()* function:

```
x <- c(1.5, 3.25, 8.75, 13.15)
x
```

```
## [1]  1.50  3.25  8.75 13.15
```

```
length(x)
```

```
## [1] 4
```

3.1 Vector Indexing

```
x <- c(10,45,30,50,35,50,80)
x
```

```
## [1] 10 45 30 50 35 50 80
```

```
x[1]
```

```
## [1] 10
```

```
x[3]
```

```
## [1] 30
```

```
x[-2]
```

```
## [1] 10 30 50 35 50 80
```

```
x[3:6]
```

```
## [1] 30 50 35 50
```

```
x[c(1,3,4)]
```

```
## [1] 10 30 50
```

```
length(x)
```

```
## [1] 7
```

```
x[10]
```

```
## [1] NA
```

```
x
```

```
## [1] 10 45 30 50 35 50 80
```

```
x[2]
```

```
## [1] 45
```

```
x[2] <- -8
x
```

```
## [1] 10 -8 30 50 35 50 80
```

```
x[10] = 20
x
```

```
##  [1] 10 -8 30 50 35 50 80 NA NA 20
```

```
x[-3] = 6
x
```

```
##  [1]  6  6 30  6  6  6  6  6  6  6
```

```
x
```

```
##  [1]  6  6 30  6  6  6  6  6  6  6
```

```
y <- c(TRUE, FALSE, FALSE, TRUE, TRUE, FALSE, TRUE)
y <- c(T, F, F, T, T, F, T)
x[y]
```

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

Use for loops for access to elemets of vectors

```
for (i in x){
  print(x*2)
}
```

```
##  [1] 12 12 60 12 12 12 12 12 12 12
##  [1] 12 12 60 12 12 12 12 12 12 12
##  [1] 12 12 60 12 12 12 12 12 12 12
##  [1] 12 12 60 12 12 12 12 12 12 12
##  [1] 12 12 60 12 12 12 12 12 12 12
##  [1] 12 12 60 12 12 12 12 12 12 12
```

```
##  [1] 12 12 60 12 12 12 12 12 12 12
##  [1] 12 12 60 12 12 12 12 12 12 12
##  [1] 12 12 60 12 12 12 12 12 12 12
##  [1] 12 12 60 12 12 12 12 12 12 12
```

3.2 Matching Operator

```
x <- c(10,45,30,50,35,50,80)
x
```

```
## [1] 10 45 30 50 35 50 80
```

```
35 %in% x
```

```
## [1] TRUE
```

```
37 %in% x
```

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

```
y <- c(30, 37, 45)
y %in% x
```

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

3.3 Vector Arithmetic's

```
x <- c(10,45,30,50,35,50,80)
x
```

```
## [1] 10 45 30 50 35 50 80
```

```
x + 2
```

```
## [1] 12 47 32 52 37 52 82
```

```
x * 2
```

```
## [1]  20  90  60 100  70 100 160
```

```
sqrt(x)
```

```
## [1] 3.162278 6.708204 5.477226 7.071068 5.916080 7.071068 8.944272
```

```
x <- c(10,45,30,50)
y <- c(5,1,2,4)

x + y
```

```
## [1] 15 46 32 54
```

```
z <- c(10,20,30)
x + z
```

```
## Warning in x + z: longer object length is not a multiple of shorter object
## length
```

```
## [1] 20 65 60 60
```

3.4 Vector Methods

```
x <- c(10,45,30,50)
x
```

```
## [1] 10 45 30 50
```

```
length(x)
```

```
## [1] 4
```

```
sum(x)
```

```
## [1] 135
```

```
mean(x)
```

```
## [1] 33.75
```

```
prod(x)
```

```
## [1] 675000
```

```
rev(x)
```

```
## [1] 50 30 45 10
```

```
sort(x)
```

```
## [1] 10 30 45 50
```

```
sort(x, decreasing = TRUE)
```

```
## [1] 50 45 30 10
```

3.5 Logical Vector

```
x <- c(10,45,30,50,35)
x
```

```
## [1] 10 45 30 50 35
```

```
y <- x > 30 & x < 50
y
```

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

```
x[y]
```

```
## [1] 45 35
```

```
x <- c(10,45,30,50,35)
x
```

```
## [1] 10 45 30 50 35
```

```
which(x>30)
```

```
## [1] 2 4 5
```

```
x[which(x>30)]
```

```
## [1] 45 50 35
```

3.6 Factors

- Used to represent categorical data
- Treated as integer vector, having a label
- Factors are self describing

```
x <- c('Male', "Female", "Male", 'Male', "Female")
x
```

```
## [1] "Male"   "Female" "Male"   "Male"   "Female"
```

```
x <- factor(x)
x
```

```
## [1] Male   Female Male   Male   Female
## Levels: Female Male
```

```
table(x)
```

```
## x
## Female   Male
##      2      3
```

3.7 Mathematical Function in R

```
x <- c(4.325, -3.453, 5.324, 7.844)
x
```

```
## [1]  4.325 -3.453  5.324  7.844
```

```
ceiling(x)  # next integer
```

```
## [1]  5 -3  6  8
```

```
floor(x)
```

```
## [1]  4 -4  5  7
```

```
round(x)
```

```
## [1]  4 -3  5  8
```

```
round(x, digits = 2)
```

```
## [1]  4.32 -3.45  5.32  7.84
```

```
x <- c(16,25,30,81,36)

sqrt(x)
```

```
## [1] 4.000000 5.000000 5.477226 9.000000 6.000000
```

```
log(x)
```

```
## [1] 2.772589 3.218876 3.401197 4.394449 3.583519
```

```
log(x, base = 2)
```

```
## [1] 4.000000 4.643856 4.906891 6.339850 5.169925
```

```
log10(x)
```

```
## [1] 1.204120 1.397940 1.477121 1.908485 1.556303
```

```r
x <- c(3,4,5,6)
factorial(x)
```

```
## [1]   6  24 120 720
```

3.8 Random Number in R

```r
x <- rnorm(10)
x
```

```
##  [1]  1.59097821 -0.37127685 -1.01699058 -0.09066686  1.00334107  2.36293603
##  [7]  1.08353789 -1.35457016 -0.51559349 -1.64843289
```

```r
x <- rnorm(10000, mean = 0, sd=1)
#x

mean(x)
```

```
## [1] 0.01357585
```

```r
sd(x)
```

```
## [1] 1.008174
```

3.9 Practice:

Given a list of students `("Alice", "Bob", "Charlie", "David", "Eve")` and their corresponding scores `(85, 92, 78, 55, 88)`, extract the names and scores of students who passed (`score >= 60`). Also, calculate the mean score of the students who passed.

```r
students <- c("Alice", "Bob", "Charlie", "David", "Eve")
scores <- c(85, 92, 78, 55, 88)
```

4 Matrix

4.1 Creat Matrix

Matrix are 2-dimensional vectors and dimensional attribute is of length 2 (rows and columns). We should to know that Matrix contain elements of same type.

```
m <- matrix(nrow = 2, ncol = 3)
m
```

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

```
dim(m)
```

```
## [1] 2 3
```

```
m <- matrix(c(1,2,3,4,5,6))

m <- matrix(c(1,2,3,4,5,6), nrow = 2, ncol = 3)

m <- matrix(c(1,2,3,4,5,6), nrow = 2, ncol = 3, byrow = TRUE)

m
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
```

```
dim(m)
```

```
## [1] 2 3
```

```
nrow(m)
```

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

```
ncol(m)
```

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

```
length(m)
```

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

4.2 Matrix diag

like `numpy.full` in python

```
m <- matrix(0, 3,3)
m
```

```
##      [,1] [,2] [,3]
## [1,]    0    0    0
## [2,]    0    0    0
## [3,]    0    0    0
```

like `numpy.diag` in python

```
m <- diag(1, 3,3)
```

```r
m <- diag(4)

m <- diag(1:5)
m
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    2    0    0    0
## [3,]    0    0    3    0    0
## [4,]    0    0    0    4    0
## [5,]    0    0    0    0    5
```

for find the elements of diagonal of matrix:

```r
m <- matrix(seq(1, 31, by=2), nrow = 4, ncol = 4, byrow = T)
m
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    5    7
## [2,]    9   11   13   15
## [3,]   17   19   21   23
## [4,]   25   27   29   31
```

```r
diag(m)
```

```
## [1]  1 11 21 31
```

4.3 Matrix Indexing

Indexing in R programming is similar to Python.

```r
m <- matrix(seq(1, 31, by=2), nrow = 4, ncol = 4, byrow = T)
m
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    5    7
## [2,]    9   11   13   15
## [3,]   17   19   21   23
## [4,]   25   27   29   31
```

```r
m[1, 2]
```

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

```r
m[1,]    # for get single row
```

```
## [1] 1 3 5 7
```

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

```
## [1]  3 11 19 27
```

```r
m[,1:2]
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    9   11
## [3,]   17   19
## [4,]   25   27
```

```
m[2:3,1:2]
```

```
##      [,1] [,2]
## [1,]    9   11
## [2,]   17   19
```

```
m[,c(1,3)]
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    9   13
## [3,]   17   21
## [4,]   25   29
```

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

```
##      [,1] [,2] [,3]
## [1,]    1    5    7
## [2,]    9   13   15
## [3,]   17   21   23
## [4,]   25   29   31
```

You can change values in matrix.

```
m
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    5    7
## [2,]    9   11   13   15
## [3,]   17   19   21   23
## [4,]   25   27   29   31
```

```
m[2,1] = 23
m
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    5    7
## [2,]   23   11   13   15
## [3,]   17   19   21   23
## [4,]   25   27   29   31
```

4.4 Matrix Specific Functions

```
m
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    5    7
## [2,]   23   11   13   15
## [3,]   17   19   21   23
## [4,]   25   27   29   31
```

```
rowSums(m)
```

```
## [1]  16  62  80 112
```

```
colSums(m)
```

```
## [1] 66 60 68 76
```

```r
rowMeans(m)
```

```
## [1]  4.0 15.5 20.0 28.0
```

```r
colMeans(m)
```

```
## [1] 16.5 15.0 17.0 19.0
```

```r
t(m)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1   23   17   25
## [2,]    3   11   19   27
## [3,]    5   13   21   29
## [4,]    7   15   23   31
```

```r
colSums(m)[3]
```

```
## [1] 68
```

4.5 Practice I

Create a 4x4 matrix of random integers between 1 and 100.

- Print the matrix.
- Calculate the row-wise sum and column-wise mean.
- Check if the matrix is symmetric by comparing it to its transpose.

```r
set.seed(123)
mat <- matrix(sample(1:100, 16), nrow = 4)
mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]   31   67   97   57
## [2,]   79   42   25    9
## [3,]   51   50   90   72
## [4,]   14   43   69   26
```

```r
rowMeans(mat)
```

```
## [1] 63.00 38.75 65.75 38.00
```

```r
colMeans(mat)
```

```
## [1] 43.75 50.50 70.25 41.00
```

```r
mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]   31   67   97   57
## [2,]   79   42   25    9
## [3,]   51   50   90   72
## [4,]   14   43   69   26
```

```r
t(mat)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]   31   79   51   14
## [2,]   67   42   50   43
## [3,]   97   25   90   69
## [4,]   57    9   72   26
```

```r
all(mat == t(mat))
```

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

4.6 Practice II

Given a 3x3 matrix A of integers and a vector b = (30, 20, 15), solve the linear equation $A \cdot X = b$.

```r
set.seed(123)
A <- matrix(sample(1:10, 9, replace = TRUE), nrow = 3)
b <- c(30, 20, 15)

A
```

```
##      [,1] [,2] [,3]
## [1,]    3    2    4
## [2,]    3    6    6
## [3,]   10    5    9
```

```r
b
```

```
## [1] 30 20 15
```

```r
x <- solve(A, b)   # solve for x
x
```

```
## [1] -13.80952 -15.23810  25.47619
```

```r
Ax = A %*% x
Ax
```

```
##      [,1]
## [1,]   30
## [2,]   20
## [3,]   15
```

5 Lists

5.1 Creat list

`Lists` are also collecting of data and another kind of data storage. Lists can contain elemnts of any type of R object and these elements of list don't need be same type. You can creat list by using `list()` function.

```
x <- list(10, 'Saeed', TRUE)
x
```

```
## [[1]]
## [1] 10
##
## [[2]]
## [1] "Saeed"
##
## [[3]]
## [1] TRUE
```

Create list with vectors

```
class
```

```
## function (x)  .Primitive("class")
```

5.2 List subset Operator

```
id <- c(101,102,103, 104, 105)
names <- c("Sanaz", "Saeed", "James", "Peter", "Emma")
scores <- c(98.45, 45.65, 78.79, 56.32, 87.23)

students <- list(id, names, scores)
students
```

```
## [[1]]
## [1] 101 102 103 104 105
##
## [[2]]
## [1] "Sanaz" "Saeed" "James" "Peter" "Emma"
##
## [[3]]
## [1] 98.45 45.65 78.79 56.32 87.23
```

```
students[2]
```

```
## [[1]]
## [1] "Sanaz" "Saeed" "James" "Peter" "Emma"
```

```
students[[2]][1]
```

```
## [1] "Sanaz"
```

6 Dataframe

Dataframes are objects in R and used to store tabular data. Unlike a matrix in data frame each column can contain different modes of data. The first column can be numeric while the second column can be character and third column can be logical. It is a list of vectors of equal length. Dataframe can be created using `data.frame()` function or imported from various file types.

- ‘read.table()“
- ‘read.csv()“

6.1 Creating Dataframes

```
id <- c(101,102,103, 104, 105)
names <- c("Sanaz", "Saeed", "James", "Peter", "Emma")
scores <- c(98.45, 45.65, 78.79, 56.32, 87.23)
students <- data.frame(id, names, scores)
students
```

```
##    id names scores
## 1 101 Sanaz  98.45
## 2 102 Saeed  45.65
## 3 103 James  78.79
## 4 104 Peter  56.32
## 5 105  Emma  87.23
```

6.2 Dataframes Indexing

```
students
```

```
##    id names scores
## 1 101 Sanaz  98.45
## 2 102 Saeed  45.65
## 3 103 James  78.79
## 4 104 Peter  56.32
## 5 105  Emma  87.23
```

```
students[1,]
```

```
##    id names scores
## 1 101 Sanaz  98.45
```

```
students[,2]
```

```
## [1] "Sanaz" "Saeed" "James" "Peter" "Emma"
```

```
# Same as Matrix

students$names
```

```
## [1] "Sanaz" "Saeed" "James" "Peter" "Emma"
```

```
students$scores
```

```
## [1] 98.45 45.65 78.79 56.32 87.23
```

```
students$names[2]
```

```
## [1] "Saeed"
```

6.3 Dataframes `subset()` function for filtering

```
students
```

```
##    id names scores
## 1 101 Sanaz  98.45
## 2 102 Saeed  45.65
## 3 103 James  78.79
## 4 104 Peter  56.32
## 5 105  Emma  87.23
```

```
report <- subset(students, scores < 80)
report
```

```
##    id names scores
## 2 102 Saeed  45.65
## 3 103 James  78.79
## 4 104 Peter  56.32
```

```
report <- subset(students, scores < 80 & id <=103)
report
```

```
##    id names scores
## 2 102 Saeed  45.65
## 3 103 James  78.79
```

```
report <- subset(students, scores < 80, select = c(names))
report
```

```
##   names
## 2 Saeed
## 3 James
## 4 Peter
```

```
report <- subset(students, scores < 80, select = c(names, scores))
report
```

```
##   names scores
## 2 Saeed  45.65
## 3 James  78.79
## 4 Peter  56.32
```

```
report <- subset(students, scores < 80, select = c(-names))
report
```

```
##    id scores
## 2 102  45.65
## 3 103  78.79
## 4 104  56.32
```

6.4 Dataframes `rbine()` and `cbind()`

```
students
```

```
##    id names scores
## 1 101 Sanaz  98.45
## 2 102 Saeed  45.65
## 3 103 James  78.79
```

```
## 4 104 Peter  56.32
## 5 105  Emma  87.23
```

```
students <- rbind(students, data.frame(id= 106, names= 'Sara', scores = 68.57))
students
```

```
##    id names scores
## 1 101 Sanaz  98.45
## 2 102 Saeed  45.65
## 3 103 James  78.79
## 4 104 Peter  56.32
## 5 105  Emma  87.23
## 6 106  Sara  68.57
```

add rows

add columns

```
students = cbind(students, age = c(18,24,19,26,34,23))
students
```

```
##    id names scores age
## 1 101 Sanaz  98.45  18
## 2 102 Saeed  45.65  24
## 3 103 James  78.79  19
## 4 104 Peter  56.32  26
## 5 105  Emma  87.23  34
## 6 106  Sara  68.57  23
```

6.5 Saving data in csv

```
students
```

```
##    id names scores age
## 1 101 Sanaz  98.45  18
## 2 102 Saeed  45.65  24
## 3 103 James  78.79  19
## 4 104 Peter  56.32  26
## 5 105  Emma  87.23  34
## 6 106  Sara  68.57  23
```

```
write.csv(students, file = "scoring.csv")
```

6.6 Missing Data

In this part we find out how handle a missing data like `NA`.

This function is like `.isnull()` in python programming.

```
x <- c(10,4,NA,7,15,NaN)
x
```

```
## [1]  10   4  NA   7  15 NaN
```

```
is.na(x)
```

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

```
is.nan(x)
```

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

Remove missing values

```
x <- c(10,4,NA,7,15,NaN)
x
```

```
## [1]  10   4  NA   7  15 NaN
```

```
y <- is.na(x)
y
```

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

```
!y
```

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

```
x[!y]
```

```
## [1] 10  4  7 15
```

```
weather <- data.frame(
  id = c(101, 102, 103, 104, 105),
  temperature = c(25.8, 34.2, NA, 27.4, 20.5),
  wind = c(78, 59, 63, 40, 68),
  humidity = c(25, 45, 85, NA, 61)
)
weather
```

```
##    id temperature wind humidity
## 1 101        25.8   78       25
## 2 102        34.2   59       45
## 3 103          NA   63       85
## 4 104        27.4   40       NA
## 5 105        20.5   68       61
```

```
weatherNA <- complete.cases(weather)
weatherNA
```

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

```
weather[weatherNA,]
```

```
##    id temperature wind humidity
## 1 101        25.8   78       25
## 2 102        34.2   59       45
## 5 105        20.5   68       61
```

7 Dplyr Package

You can download and install packages with install.packages("The name of package"). In this case, run `install.packages('dplyr')` to download and install `dplyr` package.

Also, you can import packages in R with `library()` function.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Data is imported in to dataframes using: `read.csv()`

`read.csv()` arguments:

- **file:** name of the file (mandatory argument)
- **header:** logical value (default is false)
- **sep:** separator (default is comma (,))

```
df <- read.csv("murders.csv")
df
```

```
##     X          state abb        region population PopulationDensity murders
## 1   0        Alabama  AL         South    4779736              94.65     199
## 2   1        Arizona  AZ          West    6392017              57.05     352
## 3   2     California  CA          West   37253956             244.20    1811
## 4   3       Colorado  CO          West    5029196              49.33     117
## 5   4    Connecticut  CT     Northeast    3574097             741.40     131
## 6   5        Florida  FL         South   19687653             360.20     987
## 7   6        Georgia  GA         South    9920000             172.50     527
## 8   7       Illinois  IL North Central   12830632             231.90     453
## 9   8        Indiana  IN North Central    6483802             182.50     198
## 10  9       Kentucky  KY         South    4339367             110.00     180
## 11 10      Louisiana  LA         South    4533372             105.00     437
## 12 11       Maryland  MD         South    5773552             606.20     424
## 13 12  Massachusetts  MA     Northeast    6547629             852.10     209
## 14 13       Michigan  MI North Central    9883640             174.80     558
## 15 14       Missouri  MO North Central    5988927              87.26     419
## 16 15     New Jersey  NJ     Northeast    8791894            1189.00     363
## 17 16       New York  NY     Northeast   19378102             415.30     860
## 18 17 North Carolina  NC         South    9535483             200.60     445
## 19 18           Ohio  OH North Central   11536504             282.50     460
## 20 19       Oklahoma  OK         South    3751351              55.22     188
## 21 20   Pennsylvania  PA     Northeast   12702379             285.30     646
## 22 21      Tennessee  TN         South    6346105             156.60     356
## 23 22          Texas  TX         South   25145561              98.07    1246
## 24 23       Virginia  VA         South    8001024             207.30     369
## 25 24      Wisconsin  WI North Central    5686986             105.20     151
##    gunmurders gunownership
## 1         135        0.517
```

```
## 2         232        0.311
## 3        1257        0.213
## 4          65        0.347
## 5          97        0.167
## 6         669        0.245
## 7         376        0.403
## 8         364        0.202
## 9         142        0.391
## 10        116        0.477
## 11        351        0.441
## 12        293        0.213
## 13        118        0.126
## 14        413        0.384
## 15        321        0.417
## 16        246        0.123
## 17        517        0.180
## 18        286        0.413
## 19        310        0.324
## 20        111        0.429
## 21        457        0.347
## 22        219        0.439
## 23        805        0.359
## 24        250        0.351
## 25         97        0.444
```

you can see head or tail of data with below function. It's work like `.head()` and `.tail()` in Pandas package in python.

```
head(df, 5)
```

```
##   X       state abb    region population PopulationDensity murders gunmurders
## 1 0     Alabama  AL     South    4779736             94.65     199        135
## 2 1     Arizona  AZ      West    6392017             57.05     352        232
## 3 2  California  CA      West   37253956            244.20    1811       1257
## 4 3    Colorado  CO      West    5029196             49.33     117         65
## 5 4 Connecticut  CT Northeast    3574097            741.40     131         97
##   gunownership
## 1        0.517
## 2        0.311
## 3        0.213
## 4        0.347
## 5        0.167
```

```
tail(df, 5)
```

```
##     X        state abb        region population PopulationDensity murders
## 21 20 Pennsylvania  PA     Northeast   12702379            285.30     646
## 22 21    Tennessee  TN         South    6346105            156.60     356
## 23 22        Texas  TX         South   25145561             98.07    1246
## 24 23     Virginia  VA         South    8001024            207.30     369
## 25 24    Wisconsin  WI North Central    5686986            105.20     151
##    gunmurders gunownership
## 21        457        0.347
## 22        219        0.439
## 23        805        0.359
## 24        250        0.351
```

```
## 25         97        0.444
```

like `.shape` in pandas package in python

```
dim(df)
```

```
## [1] 25  9
```

like `.describe()` in pandas package in python for understand structure of data:

```
str(df)
```

```
## 'data.frame':    25 obs. of  9 variables:
##  $ X                 : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ state             : chr  "Alabama" "Arizona" "California" "Colorado" ...
##  $ abb               : chr  "AL" "AZ" "CA" "CO" ...
##  $ region            : chr  "South" "West" "West" "West" ...
##  $ population        : int  4779736 6392017 37253956 5029196 3574097 19687653 9920000 12830632 648380
##  $ PopulationDensity: num  94.7 57 244.2 49.3 741.4 ...
##  $ murders           : int  199 352 1811 117 131 987 527 453 198 180 ...
##  $ gunmurders        : int  135 232 1257 65 97 669 376 364 142 116 ...
##  $ gunownership      : num  0.517 0.311 0.213 0.347 0.167 0.245 0.403 0.202 0.391 0.477 ...
```

7.1 dplyr `select()` function

Select special columns

Select with number of columns:

```
df[c(2,4,5)]
```

```
##             state        region population
## 1         Alabama         South    4779736
## 2         Arizona          West    6392017
## 3      California          West   37253956
## 4        Colorado          West    5029196
## 5     Connecticut     Northeast    3574097
## 6         Florida         South   19687653
## 7         Georgia         South    9920000
## 8        Illinois North Central   12830632
## 9         Indiana North Central    6483802
## 10       Kentucky         South    4339367
## 11      Louisiana         South    4533372
## 12       Maryland         South    5773552
## 13  Massachusetts     Northeast    6547629
## 14       Michigan North Central    9883640
## 15       Missouri North Central    5988927
## 16     New Jersey     Northeast    8791894
## 17       New York     Northeast   19378102
## 18 North Carolina         South    9535483
## 19           Ohio North Central   11536504
## 20       Oklahoma         South    3751351
## 21   Pennsylvania     Northeast   12702379
## 22      Tennessee         South    6346105
## 23          Texas         South   25145561
## 24       Virginia         South    8001024
## 25      Wisconsin North Central    5686986
```

Select with name of columns:

```r
df[c('state', 'population', "murders")]
```

```
##             state population murders
## 1         Alabama    4779736     199
## 2         Arizona    6392017     352
## 3      California   37253956    1811
## 4        Colorado    5029196     117
## 5     Connecticut    3574097     131
## 6         Florida   19687653     987
## 7         Georgia    9920000     527
## 8        Illinois   12830632     453
## 9         Indiana    6483802     198
## 10       Kentucky    4339367     180
## 11      Louisiana    4533372     437
## 12       Maryland    5773552     424
## 13  Massachusetts    6547629     209
## 14       Michigan    9883640     558
## 15       Missouri    5988927     419
## 16     New Jersey    8791894     363
## 17       New York   19378102     860
## 18 North Carolina    9535483     445
## 19           Ohio   11536504     460
## 20       Oklahoma    3751351     188
## 21   Pennsylvania   12702379     646
## 22      Tennessee    6346105     356
## 23          Texas   25145561    1246
## 24       Virginia    8001024     369
## 25      Wisconsin    5686986     151
```

```r
dfprime = select(df, 'state', "region", 'murders', 'population')
dfprime
```

```
##             state        region murders population
## 1         Alabama         South     199    4779736
## 2         Arizona          West     352    6392017
## 3      California          West    1811   37253956
## 4        Colorado          West     117    5029196
## 5     Connecticut     Northeast     131    3574097
## 6         Florida         South     987   19687653
## 7         Georgia         South     527    9920000
## 8        Illinois North Central     453   12830632
## 9         Indiana North Central     198    6483802
## 10       Kentucky         South     180    4339367
## 11      Louisiana         South     437    4533372
## 12       Maryland         South     424    5773552
## 13  Massachusetts     Northeast     209    6547629
## 14       Michigan North Central     558    9883640
## 15       Missouri North Central     419    5988927
## 16     New Jersey     Northeast     363    8791894
## 17       New York     Northeast     860   19378102
## 18 North Carolina         South     445    9535483
## 19           Ohio North Central     460   11536504
## 20       Oklahoma         South     188    3751351
## 21   Pennsylvania     Northeast     646   12702379
```

```
## 22      Tennessee         South     356    6346105
## 23          Texas         South    1246   25145561
## 24       Virginia         South     369    8001024
## 25      Wisconsin North Central     151    5686986
```

for get names of columns, you can use below function. This work like `.columns` in pandas package in Python.

```
names(df)
```

```
## [1] "X"                 "state"             "abb"
## [4] "region"            "population"        "PopulationDensity"
## [7] "murders"           "gunmurders"        "gunownership"
```

```
#names(df)[3:5]
```

Also you can select range of columns:

```
dfprime <- select(df, state:population)
dfprime
```

```
##             state abb        region population
## 1         Alabama  AL         South    4779736
## 2         Arizona  AZ          West    6392017
## 3      California  CA          West   37253956
## 4        Colorado  CO          West    5029196
## 5     Connecticut  CT     Northeast    3574097
## 6         Florida  FL         South   19687653
## 7         Georgia  GA         South    9920000
## 8        Illinois  IL North Central   12830632
## 9         Indiana  IN North Central    6483802
## 10       Kentucky  KY         South    4339367
## 11      Louisiana  LA         South    4533372
## 12       Maryland  MD         South    5773552
## 13  Massachusetts  MA     Northeast    6547629
## 14       Michigan  MI North Central    9883640
## 15       Missouri  MO North Central    5988927
## 16     New Jersey  NJ     Northeast    8791894
## 17       New York  NY     Northeast   19378102
## 18 North Carolina  NC         South    9535483
## 19           Ohio  OH North Central   11536504
## 20       Oklahoma  OK         South    3751351
## 21   Pennsylvania  PA     Northeast   12702379
## 22      Tennessee  TN         South    6346105
## 23          Texas  TX         South   25145561
## 24       Virginia  VA         South    8001024
## 25      Wisconsin  WI North Central    5686986
```

You can drop columns with use minus sign (-) in `select` function.

```
dfprime <- select(df, -abb)
dfprime
```

```
##     X          state        region population PopulationDensity murders
## 1   0        Alabama         South    4779736             94.65     199
## 2   1        Arizona          West    6392017             57.05     352
## 3   2     California          West   37253956            244.20    1811
## 4   3       Colorado          West    5029196             49.33     117
## 5   4    Connecticut     Northeast    3574097            741.40     131
```

```
## 6   5        Florida         South   19687653            360.20     987
## 7   6        Georgia         South    9920000            172.50     527
## 8   7       Illinois North Central   12830632            231.90     453
## 9   8        Indiana North Central    6483802            182.50     198
## 10  9       Kentucky         South    4339367            110.00     180
## 11 10      Louisiana         South    4533372            105.00     437
## 12 11       Maryland         South    5773552            606.20     424
## 13 12  Massachusetts     Northeast    6547629            852.10     209
## 14 13       Michigan North Central    9883640            174.80     558
## 15 14       Missouri North Central    5988927             87.26     419
## 16 15     New Jersey     Northeast    8791894           1189.00     363
## 17 16       New York     Northeast   19378102            415.30     860
## 18 17 North Carolina         South    9535483            200.60     445
## 19 18           Ohio North Central   11536504            282.50     460
## 20 19       Oklahoma         South    3751351             55.22     188
## 21 20   Pennsylvania     Northeast   12702379            285.30     646
## 22 21      Tennessee         South    6346105            156.60     356
## 23 22          Texas         South   25145561             98.07    1246
## 24 23       Virginia         South    8001024            207.30     369
## 25 24      Wisconsin North Central    5686986            105.20     151
##    gunmurders gunownership
## 1         135        0.517
## 2         232        0.311
## 3        1257        0.213
## 4          65        0.347
## 5          97        0.167
## 6         669        0.245
## 7         376        0.403
## 8         364        0.202
## 9         142        0.391
## 10        116        0.477
## 11        351        0.441
## 12        293        0.213
## 13        118        0.126
## 14        413        0.384
## 15        321        0.417
## 16        246        0.123
## 17        517        0.180
## 18        286        0.413
## 19        310        0.324
## 20        111        0.429
## 21        457        0.347
## 22        219        0.439
## 23        805        0.359
## 24        250        0.351
## 25         97        0.444
```

```
dfprime <- select(df, - c(abb, murders, gunmurders))
dfprime
```

```
##     X          state        region population PopulationDensity gunownership
## 1   0        Alabama         South    4779736             94.65        0.517
## 2   1        Arizona          West    6392017             57.05        0.311
## 3   2     California          West   37253956            244.20        0.213
## 4   3       Colorado          West    5029196             49.33        0.347
```

```
## 5   4    Connecticut     Northeast    3574097            741.40        0.167
## 6   5        Florida         South   19687653            360.20        0.245
## 7   6        Georgia         South    9920000            172.50        0.403
## 8   7       Illinois North Central   12830632            231.90        0.202
## 9   8        Indiana North Central    6483802            182.50        0.391
## 10  9       Kentucky         South    4339367            110.00        0.477
## 11 10      Louisiana         South    4533372            105.00        0.441
## 12 11       Maryland         South    5773552            606.20        0.213
## 13 12  Massachusetts     Northeast    6547629            852.10        0.126
## 14 13       Michigan North Central    9883640            174.80        0.384
## 15 14       Missouri North Central    5988927             87.26        0.417
## 16 15     New Jersey     Northeast    8791894           1189.00        0.123
## 17 16       New York     Northeast   19378102            415.30        0.180
## 18 17 North Carolina         South    9535483            200.60        0.413
## 19 18           Ohio North Central   11536504            282.50        0.324
## 20 19       Oklahoma         South    3751351             55.22        0.429
## 21 20   Pennsylvania     Northeast   12702379            285.30        0.347
## 22 21      Tennessee         South    6346105            156.60        0.439
## 23 22          Texas         South   25145561             98.07        0.359
## 24 23       Virginia         South    8001024            207.30        0.351
## 25 24      Wisconsin North Central    5686986            105.20        0.444
```

```
dfprime = select(df, -(abb:murders))
dfprime
```

```
##     X          state gunmurders gunownership
## 1   0        Alabama        135        0.517
## 2   1        Arizona        232        0.311
## 3   2     California       1257        0.213
## 4   3       Colorado         65        0.347
## 5   4    Connecticut         97        0.167
## 6   5        Florida        669        0.245
## 7   6        Georgia        376        0.403
## 8   7       Illinois        364        0.202
## 9   8        Indiana        142        0.391
## 10  9       Kentucky        116        0.477
## 11 10      Louisiana        351        0.441
## 12 11       Maryland        293        0.213
## 13 12  Massachusetts        118        0.126
## 14 13       Michigan        413        0.384
## 15 14       Missouri        321        0.417
## 16 15     New Jersey        246        0.123
## 17 16       New York        517        0.180
## 18 17 North Carolina        286        0.413
## 19 18           Ohio        310        0.324
## 20 19       Oklahoma        111        0.429
## 21 20   Pennsylvania        457        0.347
## 22 21      Tennessee        219        0.439
## 23 22          Texas        805        0.359
## 24 23       Virginia        250        0.351
## 25 24      Wisconsin         97        0.444
```

7.2 dplyr `filter()` function

```
head(df)
```

```
##   X       state abb    region population PopulationDensity murders gunmurders
## 1 0     Alabama  AL     South    4779736             94.65     199        135
## 2 1     Arizona  AZ      West    6392017             57.05     352        232
## 3 2  California  CA      West   37253956            244.20    1811       1257
## 4 3    Colorado  CO      West    5029196             49.33     117         65
## 5 4 Connecticut  CT Northeast    3574097            741.40     131         97
## 6 5     Florida  FL     South   19687653            360.20     987        669
##   gunownership
## 1        0.517
## 2        0.311
## 3        0.213
## 4        0.347
## 5        0.167
## 6        0.245
```

```
#names(df)

dfprime <- filter(df, murders > 500)
dfprime
```

```
##    X        state abb        region population PopulationDensity murders
## 1  2   California  CA          West   37253956            244.20    1811
## 2  5      Florida  FL         South   19687653            360.20     987
## 3  6      Georgia  GA         South    9920000            172.50     527
## 4 13     Michigan  MI North Central    9883640            174.80     558
## 5 16     New York  NY     Northeast   19378102            415.30     860
## 6 20 Pennsylvania  PA     Northeast   12702379            285.30     646
## 7 22        Texas  TX         South   25145561             98.07    1246
##   gunmurders gunownership
## 1       1257        0.213
## 2        669        0.245
## 3        376        0.403
## 4        413        0.384
## 5        517        0.180
## 6        457        0.347
## 7        805        0.359
```

```
dfprime <- filter(df, murders > 500 & population > 10^7)
dfprime
```

```
##    X        state abb    region population PopulationDensity murders gunmurders
## 1  2   California  CA      West   37253956            244.20    1811       1257
## 2  5      Florida  FL     South   19687653            360.20     987        669
## 3 16     New York  NY Northeast   19378102            415.30     860        517
## 4 20 Pennsylvania  PA Northeast   12702379            285.30     646        457
## 5 22        Texas  TX     South   25145561             98.07    1246        805
##   gunownership
## 1        0.213
## 2        0.245
## 3        0.180
## 4        0.347
## 5        0.359
```

```
df$population
```

```
##  [1]  4779736  6392017 37253956  5029196  3574097 19687653  9920000 12830632
##  [9]  6483802  4339367  4533372  5773552  6547629  9883640  5988927  8791894
## [17] 19378102  9535483 11536504  3751351 12702379  6346105 25145561  8001024
## [25]  5686986
```

```
mean(df$population)
```

```
## [1] 10155719
```

```
dfprime <- filter(df, population > mean(df$population))
dfprime
```

```
##    X        state abb        region population PopulationDensity murders
## 1  2   California  CA          West   37253956             244.20    1811
## 2  5      Florida  FL         South   19687653             360.20     987
## 3  7     Illinois  IL North Central   12830632             231.90     453
## 4 16     New York  NY     Northeast   19378102             415.30     860
## 5 18         Ohio  OH North Central   11536504             282.50     460
## 6 20 Pennsylvania  PA     Northeast   12702379             285.30     646
## 7 22        Texas  TX         South   25145561              98.07    1246
##   gunmurders gunownership
## 1       1257        0.213
## 2        669        0.245
## 3        364        0.202
## 4        517        0.180
## 5        310        0.324
## 6        457        0.347
## 7        805        0.359
```

dplyr `arrange()` function

```
head(df)
```

```
##   X       state abb    region population PopulationDensity murders gunmurders
## 1 0     Alabama  AL     South    4779736             94.65     199        135
## 2 1     Arizona  AZ      West    6392017             57.05     352        232
## 3 2  California  CA      West   37253956            244.20    1811       1257
## 4 3    Colorado  CO      West    5029196             49.33     117         65
## 5 4 Connecticut  CT Northeast    3574097            741.40     131         97
## 6 5     Florida  FL     South   19687653            360.20     987        669
##   gunownership
## 1        0.517
## 2        0.311
## 3        0.213
## 4        0.347
## 5        0.167
## 6        0.245
```

```
dfprime <- arrange(df, murders)
dfprime
```

```
##    X          state abb        region population PopulationDensity murders
## 1  3       Colorado  CO          West    5029196             49.33     117
## 2  4    Connecticut  CT     Northeast    3574097            741.40     131
## 3 24      Wisconsin  WI North Central    5686986            105.20     151
```

```
## 4   9       Kentucky  KY         South    4339367            110.00     180
## 5  19       Oklahoma  OK         South    3751351             55.22     188
## 6   8        Indiana  IN North Central    6483802            182.50     198
## 7   0        Alabama  AL         South    4779736             94.65     199
## 8  12  Massachusetts  MA     Northeast    6547629            852.10     209
## 9   1        Arizona  AZ          West    6392017             57.05     352
## 10 21      Tennessee  TN         South    6346105            156.60     356
## 11 15     New Jersey  NJ     Northeast    8791894           1189.00     363
## 12 23       Virginia  VA         South    8001024            207.30     369
## 13 14       Missouri  MO North Central    5988927             87.26     419
## 14 11       Maryland  MD         South    5773552            606.20     424
## 15 10      Louisiana  LA         South    4533372            105.00     437
## 16 17 North Carolina  NC         South    9535483            200.60     445
## 17  7       Illinois  IL North Central   12830632            231.90     453
## 18 18           Ohio  OH North Central   11536504            282.50     460
## 19  6        Georgia  GA         South    9920000            172.50     527
## 20 13       Michigan  MI North Central    9883640            174.80     558
## 21 20   Pennsylvania  PA     Northeast   12702379            285.30     646
## 22 16       New York  NY     Northeast   19378102            415.30     860
## 23  5        Florida  FL         South   19687653            360.20     987
## 24 22          Texas  TX         South   25145561             98.07    1246
## 25  2     California  CA          West   37253956            244.20    1811
##    gunmurders gunownership
## 1          65        0.347
## 2          97        0.167
## 3          97        0.444
## 4         116        0.477
## 5         111        0.429
## 6         142        0.391
## 7         135        0.517
## 8         118        0.126
## 9         232        0.311
## 10        219        0.439
## 11        246        0.123
## 12        250        0.351
## 13        321        0.417
## 14        293        0.213
## 15        351        0.441
## 16        286        0.413
## 17        364        0.202
## 18        310        0.324
## 19        376        0.403
## 20        413        0.384
## 21        457        0.347
## 22        517        0.180
## 23        669        0.245
## 24        805        0.359
## 25       1257        0.213
```

```
dfprime <- arrange(df, desc(murders))
dfprime
```

```
##     X          state abb        region population PopulationDensity murders
## 1   2     California  CA          West   37253956            244.20    1811
## 2  22          Texas  TX         South   25145561             98.07    1246
```

```
## 3   5        Florida  FL         South   19687653           360.20     987
## 4  16       New York  NY     Northeast   19378102           415.30     860
## 5  20   Pennsylvania  PA     Northeast   12702379           285.30     646
## 6  13       Michigan  MI North Central    9883640           174.80     558
## 7   6        Georgia  GA         South    9920000           172.50     527
## 8  18           Ohio  OH North Central   11536504           282.50     460
## 9   7       Illinois  IL North Central   12830632           231.90     453
## 10 17 North Carolina  NC         South    9535483           200.60     445
## 11 10      Louisiana  LA         South    4533372           105.00     437
## 12 11       Maryland  MD         South    5773552           606.20     424
## 13 14       Missouri  MO North Central    5988927            87.26     419
## 14 23       Virginia  VA         South    8001024           207.30     369
## 15 15     New Jersey  NJ     Northeast    8791894          1189.00     363
## 16 21      Tennessee  TN         South    6346105           156.60     356
## 17  1        Arizona  AZ          West    6392017            57.05     352
## 18 12  Massachusetts  MA     Northeast    6547629           852.10     209
## 19  0        Alabama  AL         South    4779736            94.65     199
## 20  8        Indiana  IN North Central    6483802           182.50     198
## 21 19       Oklahoma  OK         South    3751351            55.22     188
## 22  9       Kentucky  KY         South    4339367           110.00     180
## 23 24      Wisconsin  WI North Central    5686986           105.20     151
## 24  4    Connecticut  CT     Northeast    3574097           741.40     131
## 25  3       Colorado  CO          West    5029196            49.33     117
##    gunmurders gunownership
## 1        1257        0.213
## 2         805        0.359
## 3         669        0.245
## 4         517        0.180
## 5         457        0.347
## 6         413        0.384
## 7         376        0.403
## 8         310        0.324
## 9         364        0.202
## 10        286        0.413
## 11        351        0.441
## 12        293        0.213
## 13        321        0.417
## 14        250        0.351
## 15        246        0.123
## 16        219        0.439
## 17        232        0.311
## 18        118        0.126
## 19        135        0.517
## 20        142        0.391
## 21        111        0.429
## 22        116        0.477
## 23         97        0.444
## 24         97        0.167
## 25         65        0.347
```

7.3 dplyr `rename()` function

```
names(df)
```

```
## [1] "X"                 "state"             "abb"
## [4] "region"            "population"        "PopulationDensity"
## [7] "murders"           "gunmurders"        "gunownership"
```

```
df2 <- rename(df, abbreviation = abb)
names(df2)
```

```
## [1] "X"                 "state"             "abbreviation"
## [4] "region"            "population"        "PopulationDensity"
## [7] "murders"           "gunmurders"        "gunownership"
```

```
df2 <- rename(df, abbreviation = abb, homicide = murders)
names(df2)
```

```
## [1] "X"                 "state"             "abbreviation"
## [4] "region"            "population"        "PopulationDensity"
## [7] "homicide"          "gunmurders"        "gunownership"
```

7.4 dplyr `mutate()` function

```
dfprime <- mutate(df, ratio = murders / population)
head(dfprime)
```

```
##   X       state abb    region population PopulationDensity murders gunmurders
## 1 0     Alabama  AL     South    4779736             94.65     199        135
## 2 1     Arizona  AZ      West    6392017             57.05     352        232
## 3 2  California  CA      West   37253956            244.20    1811       1257
## 4 3    Colorado  CO      West    5029196             49.33     117         65
## 5 4 Connecticut  CT Northeast    3574097            741.40     131         97
## 6 5     Florida  FL     South   19687653            360.20     987        669
##   gunownership        ratio
## 1        0.517 4.163410e-05
## 2        0.311 5.506869e-05
## 3        0.213 4.861229e-05
## 4        0.347 2.326416e-05
## 5        0.167 3.665261e-05
## 6        0.245 5.013294e-05
```

7.4.1 Practice:

Import the data and Create a new column called murder_rate that shows the murder rate (murders per million people)

```
dfprime <- mutate(df, mpopulation = population / 10^6)
dfprime <- mutate(dfprime, murder_rate_m = murders / mpopulation)
dfprime
```

```
##    X          state abb        region population PopulationDensity murders
## 1  0        Alabama  AL         South    4779736             94.65     199
## 2  1        Arizona  AZ          West    6392017             57.05     352
## 3  2     California  CA          West   37253956            244.20    1811
## 4  3       Colorado  CO          West    5029196             49.33     117
## 5  4    Connecticut  CT     Northeast    3574097            741.40     131
## 6  5        Florida  FL         South   19687653            360.20     987
## 7  6        Georgia  GA         South    9920000            172.50     527
## 8  7       Illinois  IL North Central   12830632            231.90     453
## 9  8        Indiana  IN North Central    6483802            182.50     198
```

```
## 10  9       Kentucky  KY         South    4339367           110.00     180
## 11 10      Louisiana  LA         South    4533372           105.00     437
## 12 11       Maryland  MD         South    5773552           606.20     424
## 13 12  Massachusetts  MA     Northeast    6547629           852.10     209
## 14 13       Michigan  MI North Central    9883640           174.80     558
## 15 14       Missouri  MO North Central    5988927            87.26     419
## 16 15     New Jersey  NJ     Northeast    8791894          1189.00     363
## 17 16       New York  NY     Northeast   19378102           415.30     860
## 18 17 North Carolina  NC         South    9535483           200.60     445
## 19 18           Ohio  OH North Central   11536504           282.50     460
## 20 19       Oklahoma  OK         South    3751351            55.22     188
## 21 20   Pennsylvania  PA     Northeast   12702379           285.30     646
## 22 21      Tennessee  TN         South    6346105           156.60     356
## 23 22          Texas  TX         South   25145561            98.07    1246
## 24 23       Virginia  VA         South    8001024           207.30     369
## 25 24      Wisconsin  WI North Central    5686986           105.20     151
##    gunmurders gunownership mpopulation murder_rate_m
## 1         135        0.517    4.779736      41.63410
## 2         232        0.311    6.392017      55.06869
## 3        1257        0.213   37.253956      48.61229
## 4          65        0.347    5.029196      23.26416
## 5          97        0.167    3.574097      36.65261
## 6         669        0.245   19.687653      50.13294
## 7         376        0.403    9.920000      53.12500
## 8         364        0.202   12.830632      35.30613
## 9         142        0.391    6.483802      30.53764
## 10        116        0.477    4.339367      41.48070
## 11        351        0.441    4.533372      96.39624
## 12        293        0.213    5.773552      73.43833
## 13        118        0.126    6.547629      31.91995
## 14        413        0.384    9.883640      56.45693
## 15        321        0.417    5.988927      69.96245
## 16        246        0.123    8.791894      41.28803
## 17        517        0.180   19.378102      44.37999
## 18        286        0.413    9.535483      46.66780
## 19        310        0.324   11.536504      39.87343
## 20        111        0.429    3.751351      50.11528
## 21        457        0.347   12.702379      50.85662
## 22        219        0.439    6.346105      56.09740
## 23        805        0.359   25.145561      49.55149
## 24        250        0.351    8.001024      46.11910
## 25         97        0.444    5.686986      26.55185
```

7.5 dplyr group_by() function

```
names(df)
```

```
## [1] "X"                 "state"             "abb"
## [4] "region"            "population"        "PopulationDensity"
## [7] "murders"           "gunmurders"        "gunownership"
```

```
dfprime <- group_by(df, region)
dfprime
```

```
## # A tibble: 25 x 9
```

```
## # Groups:   region [4]
##       X state      abb   region population PopulationDensity murders gunmurders
##   <int> <chr>      <chr> <chr>       <int>             <dbl>   <int>      <int>
##  1     0 Alabama    AL    South     4779736              94.6     199        135
##  2     1 Arizona    AZ    West      6392017              57.0     352        232
##  3     2 California CA    West     37253956             244.     1811       1257
##  4     3 Colorado   CO    West      5029196              49.3     117         65
##  5     4 Connectic~ CT    North~    3574097             741.      131         97
##  6     5 Florida    FL    South    19687653             360.      987        669
##  7     6 Georgia    GA    South     9920000             172.      527        376
##  8     7 Illinois   IL    North~   12830632             232.      453        364
##  9     8 Indiana    IN    North~    6483802             182.      198        142
## 10     9 Kentucky   KY    South     4339367             110       180        116
## # i 15 more rows
## # i 1 more variable: gunownership <dbl>
```

```
summarise(dfprime, sum(murders))
```

```
## # A tibble: 4 x 2
##   region        `sum(murders)`
##   <chr>                  <int>
## 1 North Central           2239
## 2 Northeast               2209
## 3 South                   5358
## 4 West                    2280
```

```
summarise(dfprime, sum(murders), mean(gunownership), median(population))
```

```
## # A tibble: 4 x 4
##   region        `sum(murders)` `mean(gunownership)` `median(population)`
##   <chr>                  <int>                <dbl>                  <dbl>
## 1 North Central           2239                0.360                8183721
## 2 Northeast               2209                0.189                8791894
## 3 South                   5358                0.390                6346105
## 4 West                    2280                0.290                6392017
```

7.6 dplyr Pipe Operator %>%

```
names(df)
```

```
## [1] "X"                "state"            "abb"
## [4] "region"           "population"       "PopulationDensity"
## [7] "murders"          "gunmurders"       "gunownership"
```

```
dfprime <- arrange(df, murders)
dfprime
```

```
##     X          state abb        region population PopulationDensity murders
## 1   3       Colorado  CO          West    5029196             49.33     117
## 2   4    Connecticut  CT     Northeast    3574097            741.40     131
## 3  24      Wisconsin  WI North Central    5686986            105.20     151
## 4   9       Kentucky  KY         South    4339367            110.00     180
## 5  19       Oklahoma  OK         South    3751351             55.22     188
## 6   8        Indiana  IN North Central    6483802            182.50     198
## 7   0        Alabama  AL         South    4779736             94.65     199
## 8  12  Massachusetts  MA     Northeast    6547629            852.10     209
```

```
## 9   1        Arizona  AZ          West    6392017             57.05     352
## 10 21      Tennessee  TN         South    6346105            156.60     356
## 11 15     New Jersey  NJ     Northeast    8791894           1189.00     363
## 12 23       Virginia  VA         South    8001024            207.30     369
## 13 14       Missouri  MO North Central    5988927             87.26     419
## 14 11       Maryland  MD         South    5773552            606.20     424
## 15 10      Louisiana  LA         South    4533372            105.00     437
## 16 17 North Carolina  NC         South    9535483            200.60     445
## 17  7       Illinois  IL North Central   12830632            231.90     453
## 18 18           Ohio  OH North Central   11536504            282.50     460
## 19  6        Georgia  GA         South    9920000            172.50     527
## 20 13       Michigan  MI North Central    9883640            174.80     558
## 21 20   Pennsylvania  PA     Northeast   12702379            285.30     646
## 22 16       New York  NY     Northeast   19378102            415.30     860
## 23  5        Florida  FL         South   19687653            360.20     987
## 24 22          Texas  TX         South   25145561             98.07    1246
## 25  2     California  CA          West   37253956            244.20    1811
##    gunmurders gunownership
## 1          65        0.347
## 2          97        0.167
## 3          97        0.444
## 4         116        0.477
## 5         111        0.429
## 6         142        0.391
## 7         135        0.517
## 8         118        0.126
## 9         232        0.311
## 10        219        0.439
## 11        246        0.123
## 12        250        0.351
## 13        321        0.417
## 14        293        0.213
## 15        351        0.441
## 16        286        0.413
## 17        364        0.202
## 18        310        0.324
## 19        376        0.403
## 20        413        0.384
## 21        457        0.347
## 22        517        0.180
## 23        669        0.245
## 24        805        0.359
## 25       1257        0.213
```

```
dfprime2 <- select(dfprime, state, murders)
dfprime2
```

```
##            state murders
## 1       Colorado     117
## 2    Connecticut     131
## 3      Wisconsin     151
## 4       Kentucky     180
## 5       Oklahoma     188
## 6        Indiana     198
## 7        Alabama     199
```

```
## 8   Massachusetts     209
## 9         Arizona     352
## 10      Tennessee     356
## 11     New Jersey     363
## 12       Virginia     369
## 13       Missouri     419
## 14       Maryland     424
## 15      Louisiana     437
## 16 North Carolina     445
## 17       Illinois     453
## 18           Ohio     460
## 19        Georgia     527
## 20       Michigan     558
## 21   Pennsylvania     646
## 22       New York     860
## 23        Florida     987
## 24          Texas    1246
## 25     California    1811
```

```
head(dfprime2)
```

```
##         state murders
## 1    Colorado     117
## 2 Connecticut     131
## 3   Wisconsin     151
## 4    Kentucky     180
## 5    Oklahoma     188
## 6     Indiana     198
```

```
arrange(df, murders) %>% select(state, murders) %>% head()
```

```
##         state murders
## 1    Colorado     117
## 2 Connecticut     131
## 3   Wisconsin     151
## 4    Kentucky     180
## 5    Oklahoma     188
## 6     Indiana     198
```

```
# Practice -1
df %>% mutate(murder_rate = (murders / population) * 10^6) %>% arrange(desc(murder_rate)) %>% head(10)
```

```
##     X        state abb        region population PopulationDensity murders
## 1  10    Louisiana  LA         South    4533372            105.00     437
## 2  11     Maryland  MD         South    5773552            606.20     424
## 3  14     Missouri  MO North Central    5988927             87.26     419
## 4  13     Michigan  MI North Central    9883640            174.80     558
## 5  21    Tennessee  TN         South    6346105            156.60     356
## 6   1      Arizona  AZ          West    6392017             57.05     352
## 7   6      Georgia  GA         South    9920000            172.50     527
## 8  20 Pennsylvania  PA     Northeast   12702379            285.30     646
## 9   5      Florida  FL         South   19687653            360.20     987
## 10 19     Oklahoma  OK         South    3751351             55.22     188
##    gunmurders gunownership murder_rate
## 1         351        0.441    96.39624
## 2         293        0.213    73.43833
```

```
## 3         321        0.417    69.96245
## 4         413        0.384    56.45693
## 5         219        0.439    56.09740
## 6         232        0.311    55.06869
## 7         376        0.403    53.12500
## 8         457        0.347    50.85662
## 9         669        0.245    50.13294
## 10        111        0.429    50.11528
```

7.6.1 Practice:

Find the states with a population greater than 10 million and murders greater than 500. Display only the state, population, and murder columns.

```
df %>% filter(population > 10^7 & murders > 500) %>% select(state, population, murders) %>% arrange(murd
```

```
##          state population murders
## 1 Pennsylvania   12702379     646
## 2     New York   19378102     860
## 3      Florida   19687653     987
## 4        Texas   25145561    1246
## 5   California   37253956    1811
```

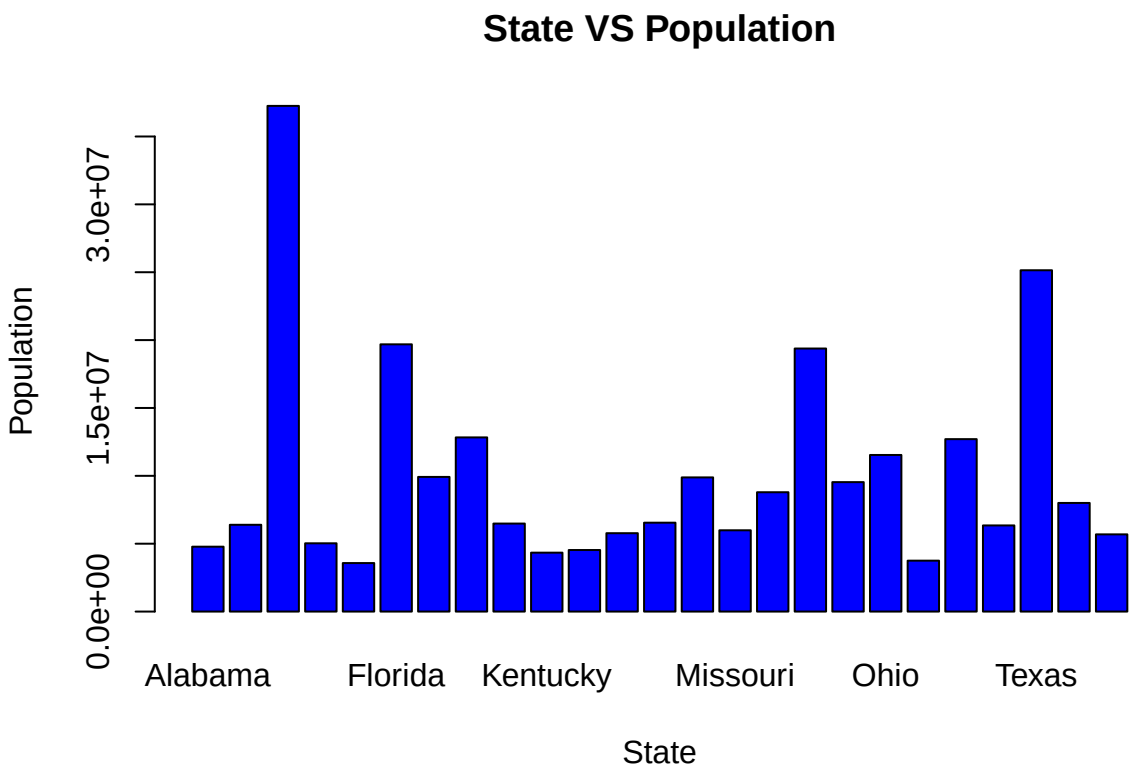
8 Data Visualization with `dplyr`

8.1 Bar Graphs

```
head(df)
```

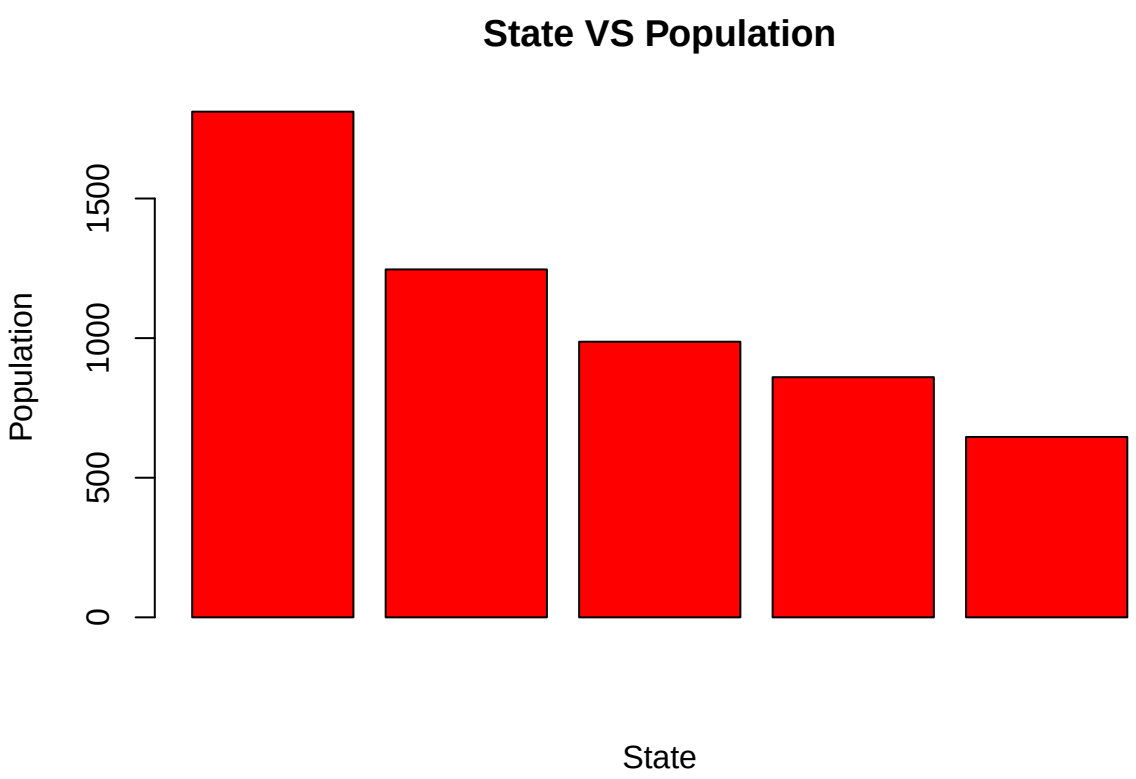
```
##   X       state abb    region population PopulationDensity murders gunmurders
## 1 0     Alabama  AL     South    4779736             94.65     199         135
## 2 1     Arizona  AZ      West    6392017             57.05     352         232
## 3 2  California  CA      West   37253956            244.20    1811        1257
## 4 3    Colorado  CO      West    5029196             49.33     117          65
## 5 4 Connecticut  CT Northeast    3574097            741.40     131          97
## 6 5     Florida  FL     South   19687653            360.20     987         669
##   gunownership
## 1        0.517
## 2        0.311
## 3        0.213
## 4        0.347
## 5        0.167
## 6        0.245
```

```
barplot(df$population,
        col = 'blue',
        xlab = 'State',
        ylab = 'Population',
        main = 'State VS Population',
        names.arg = df$state
        )
```



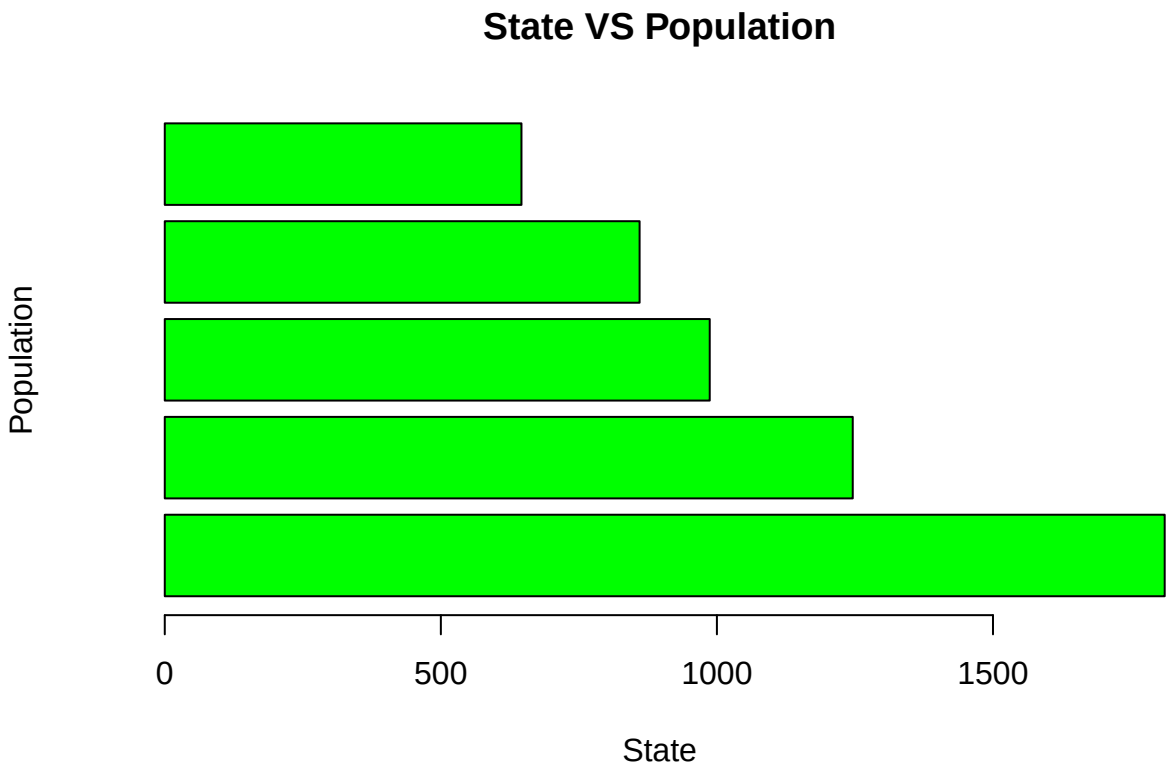

```
dfprime <- arrange(df, desc(murders)) %>% head(5)
#dfprime
```

```r
barplot(dfprime$murders,
        col = 'red',
        xlab = 'State',
        ylab = 'Population',
        main = 'State VS Population',
        )
```



Horizontsl Bar Graphs

```r
dfprime <- arrange(df, desc(murders)) %>% head(5)
#dfprime
barplot(dfprime$murders,
        col = 'green',
        xlab = 'State',
        ylab = 'Population',
        main = 'State VS Population',
        horiz = TRUE
        )
```



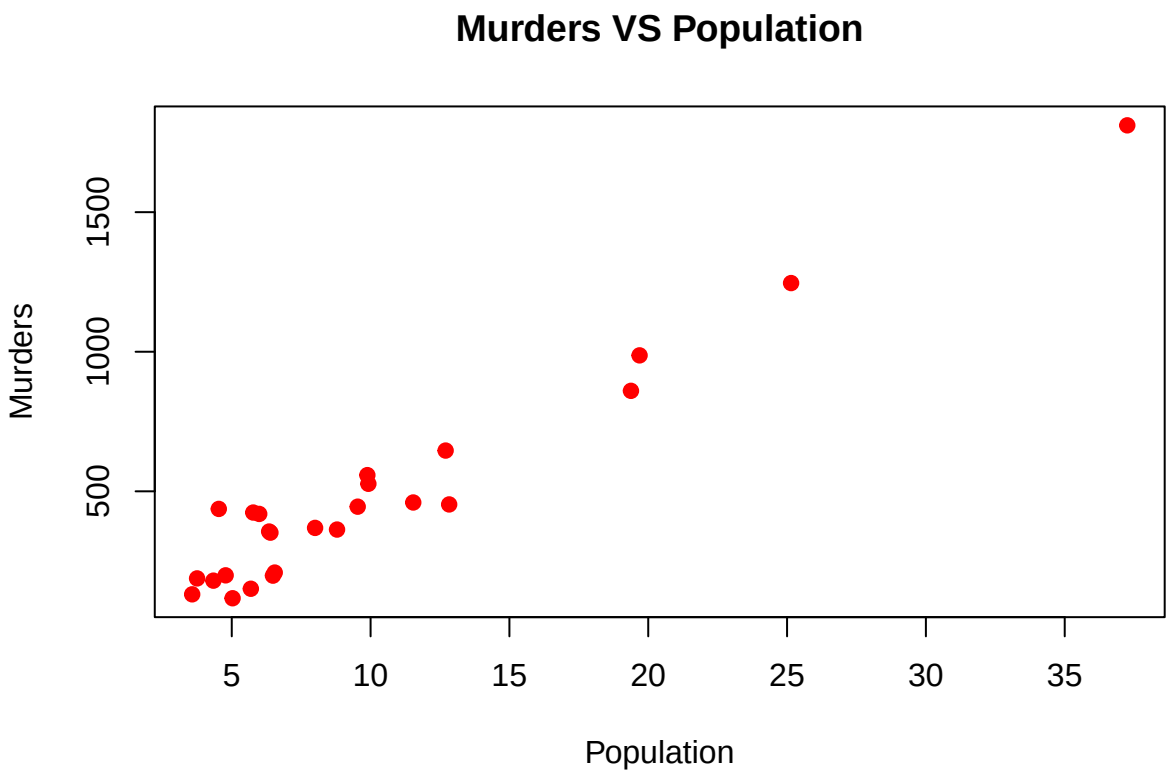
8.2 Scatter Plots

The default of `plot` functions is Scatter plot.

```
head(df)
```

```
##   X       state abb    region population PopulationDensity murders gunmurders
## 1 0     Alabama  AL     South    4779736             94.65     199        135
## 2 1     Arizona  AZ      West    6392017             57.05     352        232
## 3 2  California  CA      West   37253956            244.20    1811       1257
## 4 3    Colorado  CO      West    5029196             49.33     117         65
## 5 4 Connecticut  CT Northeast    3574097            741.40     131         97
## 6 5     Florida  FL     South   19687653            360.20     987        669
##   gunownership
## 1        0.517
## 2        0.311
## 3        0.213
## 4        0.347
## 5        0.167
## 6        0.245
```

```
plot(df$population / 10^6, df$murders,
        col = 'red',
        xlab = 'Population',
        ylab = 'Murders',
        main = 'Murders VS Population',
        pch = 19
    )
```



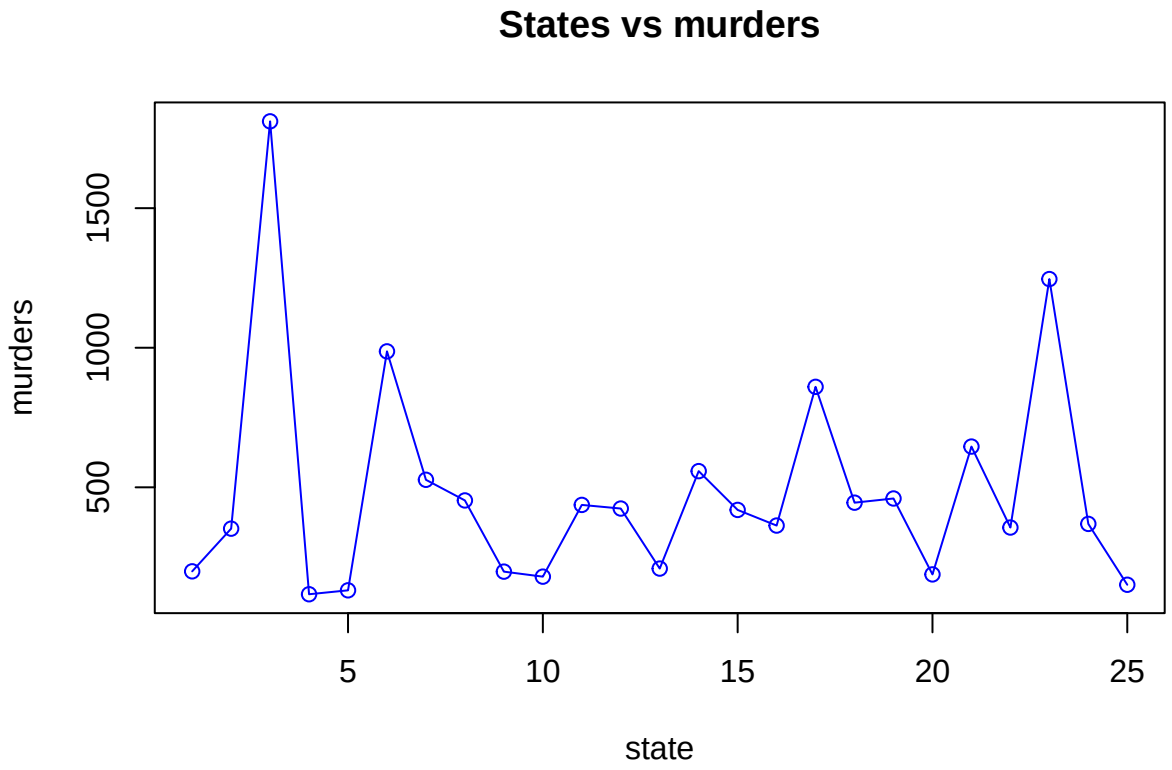
pch values

Values of pch are stored internally as integers.

Figure 1: Some values of pch

8.3 Line Graphs

```
plot(df$murders,
     type = "o",
     xlab = 'state',
     ylab = 'murders',
     main = 'States vs murders',
     col = "blue"
     )
```

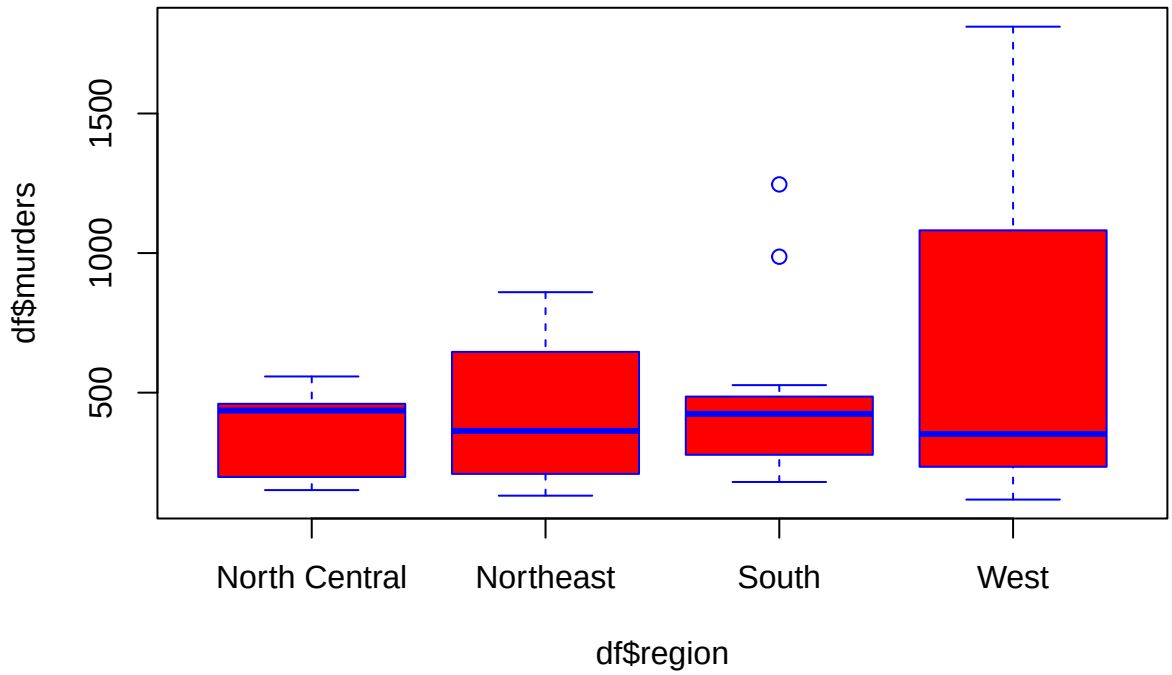


type in plot function

- **"p"** for points,
- **"l"** for lines,
- **"b"** for both points and lines,
- **"c"** for empty points joined by lines,
- **"o"** for overplotted points and lines,
- **"s"** and **"S"** for stair steps,
- **"h"** for histogram-like vertical lines,
- **"n"** does not produce any points or lines.

8.4 Box plots

```
boxplot(df$murders ~ df$region,
        col = 'red',
        border = 'blue'
        )
```

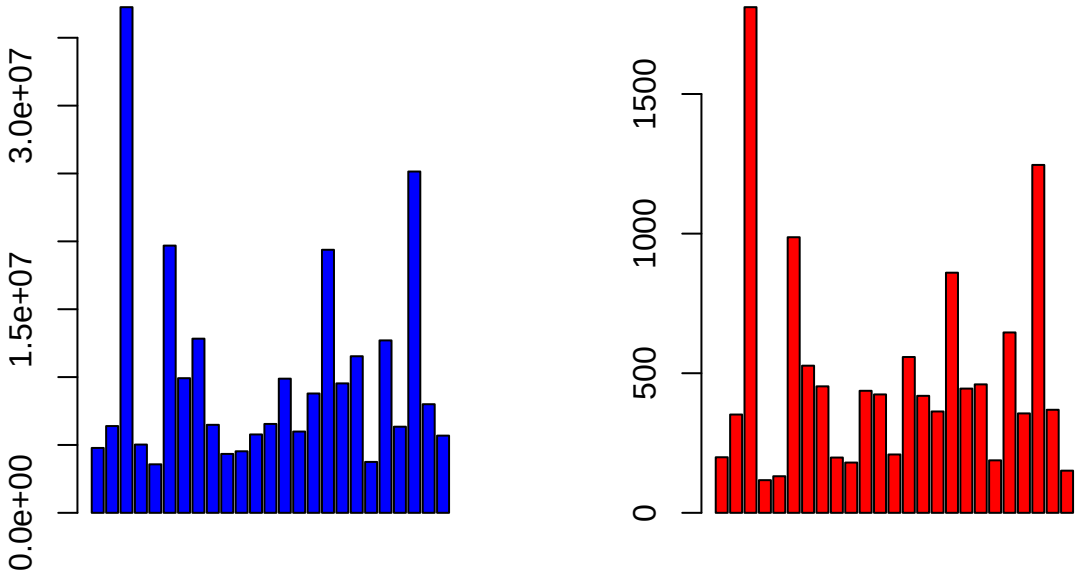


8.5 Multiple Plots in Layout

```
par(mfrow = c(1,2))

barplot(df$population, col = 'blue')

barplot(df$murders, col= 'red')
```




```
# add 4 plot
```

9 Regressions and Models

9.1 Simple Linear Regression

Linear regression is used to model the relationship between a dependent variable and one or more independent variables by fitting a line through the data.

```
df
```

```
##     X          state abb        region population PopulationDensity murders
## 1   0        Alabama  AL         South    4779736              94.65     199
## 2   1        Arizona  AZ          West    6392017              57.05     352
## 3   2     California  CA          West   37253956             244.20    1811
## 4   3       Colorado  CO          West    5029196              49.33     117
## 5   4    Connecticut  CT     Northeast    3574097             741.40     131
## 6   5        Florida  FL         South   19687653             360.20     987
## 7   6        Georgia  GA         South    9920000             172.50     527
## 8   7       Illinois  IL North Central   12830632             231.90     453
## 9   8        Indiana  IN North Central    6483802             182.50     198
## 10  9       Kentucky  KY         South    4339367             110.00     180
## 11 10      Louisiana  LA         South    4533372             105.00     437
## 12 11       Maryland  MD         South    5773552             606.20     424
## 13 12  Massachusetts  MA     Northeast    6547629             852.10     209
## 14 13       Michigan  MI North Central    9883640             174.80     558
## 15 14       Missouri  MO North Central    5988927              87.26     419
## 16 15     New Jersey  NJ     Northeast    8791894            1189.00     363
## 17 16       New York  NY     Northeast   19378102             415.30     860
## 18 17 North Carolina  NC         South    9535483             200.60     445
## 19 18           Ohio  OH North Central   11536504             282.50     460
## 20 19       Oklahoma  OK         South    3751351              55.22     188
## 21 20   Pennsylvania  PA     Northeast   12702379             285.30     646
## 22 21      Tennessee  TN         South    6346105             156.60     356
## 23 22          Texas  TX         South   25145561              98.07    1246
## 24 23       Virginia  VA         South    8001024             207.30     369
## 25 24      Wisconsin  WI North Central    5686986             105.20     151
##    gunmurders gunownership
## 1         135        0.517
## 2         232        0.311
## 3        1257        0.213
## 4          65        0.347
## 5          97        0.167
## 6         669        0.245
## 7         376        0.403
## 8         364        0.202
## 9         142        0.391
## 10        116        0.477
## 11        351        0.441
## 12        293        0.213
## 13        118        0.126
## 14        413        0.384
## 15        321        0.417
## 16        246        0.123
## 17        517        0.180
## 18        286        0.413
## 19        310        0.324
```

```
## 20        111        0.429
## 21        457        0.347
## 22        219        0.439
## 23        805        0.359
## 24        250        0.351
## 25         97        0.444
```

```
model <- lm(murders ~ population, data = df)
summary(model)
```

```
##
## Call:
## lm(formula = murders ~ population, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -159.382  -67.999   -8.542   49.987  224.582
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.110e+00  3.151e+01  -0.194    0.848
## population   4.820e-05  2.475e-06  19.476 8.62e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 95.03 on 23 degrees of freedom
## Multiple R-squared:  0.9428, Adjusted R-squared:  0.9403
## F-statistic: 379.3 on 1 and 23 DF,  p-value: 8.623e-16
```

9.2 Multiple linear regression

Multiple regression extends linear regression by adding more independent variables to better predict the dependent variable.

```
names(df)
```

```
## [1] "X"                 "state"             "abb"
## [4] "region"            "population"        "PopulationDensity"
## [7] "murders"           "gunmurders"        "gunownership"
```

```
model <- lm(murders ~ population + gunownership, data = df)
summary(model)
```

```
##
## Call:
## lm(formula = murders ~ population + gunownership, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -139.017  -51.392   -2.738   37.700  204.717
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -9.828e+01  7.408e+01  -1.327    0.198
## population    4.942e-05  2.586e-06  19.110 3.44e-15 ***
## gunownership  2.416e+02  1.764e+02   1.369    0.185
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 93.27 on 22 degrees of freedom
## Multiple R-squared:  0.9473, Adjusted R-squared:  0.9425
## F-statistic: 197.8 on 2 and 22 DF,  p-value: 8.674e-15
```

9.2.0.1 Practice: Add region (as a categorical variable) to the multiple regression model.

```
# as.factor(region)
model <- lm(murders ~ population + gunownership + as.factor(region), data = df)
summary(model)
```

```
##
## Call:
## lm(formula = murders ~ population + gunownership + as.factor(region),
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -105.79  -57.93  -15.43   29.63  176.13
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -8.751e+01  1.065e+02  -0.821   0.4216
## population                  4.891e-05  2.727e-06  17.938 2.28e-13 ***
## gunownership                9.279e+01  2.443e+02   0.380   0.7082
## as.factor(region)Northeast  1.298e+01  6.897e+01   0.188   0.8528
## as.factor(region)South      8.573e+01  4.740e+01   1.809   0.0864 .
## as.factor(region)West       2.699e+01  6.842e+01   0.394   0.6977
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 92.12 on 19 degrees of freedom
## Multiple R-squared:  0.9556, Adjusted R-squared:  0.9439
## F-statistic: 81.83 on 5 and 19 DF,  p-value: 3.503e-12
```

9.2.1 t-test (Comparing Means)

A t-test compares the means of two groups to determine if they are statistically different.

```
region1 <- df[df$region=='Northeast','murders']
region2 <- df[df$region=='South','murders']

t.test(region1, region2)
```

```
##
##  Welch Two Sample t-test
##
## data:  region1 and region2
## t = -0.26603, df = 8.5857, p-value = 0.7965
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -433.2737  342.6918
## sample estimates:
## mean of x mean of y
```

```
##  441.8000  487.0909
```

9.2.2 Correlation Test

A correlation test checks the strength and direction of the relationship between two variables.

```
cor.test(df$population, df$murders)
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$population and df$murders
## t = 19.476, df = 23, p-value = 8.623e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9343406 0.9873198
## sample estimates:
##       cor
## 0.9709935
```

```
# core gunownership and murders
```

9.2.2.1 Practice

9.2.3 ANOVA (Analysis of Variance)

ANOVA tests if there are significant differences between the means of three or more groups.

```
anova = aov(murders ~ region, data = df)
summary(anova)
```

```
##             Df  Sum Sq Mean Sq F value Pr(>F)
## region       3  311234  103745   0.656  0.588
## Residuals   21 3321917  158187
```

9.2.4 Regression Plots

```
plot(df$population, df$murders, col= 'blue', pch=19)
model <- lm(murders ~ population + gunownership, data = df)
summary(model)
```

```
##
## Call:
## lm(formula = murders ~ population + gunownership, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -139.017  -51.392   -2.738   37.700  204.717
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -9.828e+01  7.408e+01  -1.327    0.198
## population    4.942e-05  2.586e-06  19.110 3.44e-15 ***
## gunownership  2.416e+02  1.764e+02   1.369    0.185
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 93.27 on 22 degrees of freedom
## Multiple R-squared:  0.9473, Adjusted R-squared:  0.9425
## F-statistic: 197.8 on 2 and 22 DF,  p-value: 8.674e-15
```

```
abline(model, col='red', lwd=2)
```

```
## Warning in abline(model, col = "red", lwd = 2): only using the first two of 3
## regression coefficients
```