knitr::opts\_chunk$set(echo = TRUE)

**Importing Data from csvfile**

df<-read.csv("xy.csv") #read data in csv file.

**x** “independent variable”  
**y** “dependent variable”

**Data Set**

df #dataframe

## x y  
## 1 46.75 92.64  
## 2 42.18 88.81  
## 3 41.86 86.44  
## 4 43.29 88.80  
## 5 42.12 86.38  
## 6 41.78 89.87  
## 7 41.47 88.53  
## 8 42.21 91.11  
## 9 41.03 81.22  
## 10 39.84 83.72  
## 11 39.15 84.54  
## 12 39.20 85.66  
## 13 39.52 85.87  
## 14 38.05 85.23  
## 15 39.16 87.75  
## 16 38.59 92.62  
## 17 36.54 91.56  
## 18 37.03 84.12  
## 19 36.60 81.22  
## 20 37.58 83.35  
## 21 36.48 82.29  
## 22 38.25 80.92  
## 23 37.26 76.92  
## 24 38.59 78.35  
## 25 40.89 74.57  
## 26 37.66 71.60  
## 27 38.79 65.64  
## 28 38.78 62.09  
## 29 36.70 61.66  
## 30 35.10 77.14  
## 31 33.75 75.47  
## 32 34.29 70.37  
## 33 32.26 66.71  
## 34 30.97 64.37  
## 35 28.20 56.09  
## 36 24.58 50.25  
## 37 20.25 43.65  
## 38 17.09 38.01  
## 39 14.35 31.40  
## 40 13.11 29.45  
## 41 9.50 29.02  
## 42 9.74 19.05  
## 43 9.34 20.36  
## 44 7.51 17.68  
## 45 8.35 19.23  
## 46 6.25 14.92  
## 47 5.45 11.44  
## 48 3.79 12.69

**48 Observation** Two Variables y is response variable and x is input variable.

**Question 1**

A simple least squares model y = B0+B1\*x from first principles and obtain B0 and B1.

Y<-df$y #dependent Variable  
intercept<-rep(1,length(Y)) #replicates the values.  
#intercept

**Design Matrix**

X<-as.matrix(cbind(intercept,df$x))  
X #Design Matrix

## intercept   
## [1,] 1 46.75  
## [2,] 1 42.18  
## [3,] 1 41.86  
## [4,] 1 43.29  
## [5,] 1 42.12  
## [6,] 1 41.78  
## [7,] 1 41.47  
## [8,] 1 42.21  
## [9,] 1 41.03  
## [10,] 1 39.84  
## [11,] 1 39.15  
## [12,] 1 39.20  
## [13,] 1 39.52  
## [14,] 1 38.05  
## [15,] 1 39.16  
## [16,] 1 38.59  
## [17,] 1 36.54  
## [18,] 1 37.03  
## [19,] 1 36.60  
## [20,] 1 37.58  
## [21,] 1 36.48  
## [22,] 1 38.25  
## [23,] 1 37.26  
## [24,] 1 38.59  
## [25,] 1 40.89  
## [26,] 1 37.66  
## [27,] 1 38.79  
## [28,] 1 38.78  
## [29,] 1 36.70  
## [30,] 1 35.10  
## [31,] 1 33.75  
## [32,] 1 34.29  
## [33,] 1 32.26  
## [34,] 1 30.97  
## [35,] 1 28.20  
## [36,] 1 24.58  
## [37,] 1 20.25  
## [38,] 1 17.09  
## [39,] 1 14.35  
## [40,] 1 13.11  
## [41,] 1 9.50  
## [42,] 1 9.74  
## [43,] 1 9.34  
## [44,] 1 7.51  
## [45,] 1 8.35  
## [46,] 1 6.25  
## [47,] 1 5.45  
## [48,] 1 3.79

**Estmiating Coefficient**

XtX\_matrix<-solve(crossprod(X)) #crossprod ->X'X (inverse of X matrix)  
#d\_matrix  
b=XtX\_matrix %\*% crossprod(X,Y) #Matrix Multiplication(%\*%)  
b

## [,1]  
## intercept 3.128201  
## 2.005476

So **intercept** is: 3.128201  
and **slope** is : 2.005476

**Find B0 and B1 using the linear model package lm.**

lm.fit<-lm(df$y~df$x,data=df) #lm function  
coef(lm.fit)

## (Intercept) df$x   
## 3.128201 2.005476

So result is same .

**Generate 100 values of x between [3:00; 50:00] and find the value of y**

random\_no<-runif(100,3,50) #Generate random no  
random\_no

## [1] 34.447733 22.797305 39.362707 44.719534 26.909360 13.163292 14.937898  
## [8] 24.550025 25.609542 42.829304 49.758748 30.840784 20.449185 19.521042  
## [15] 37.500965 49.479186 7.752977 35.345433 12.699086 49.580991 5.967161  
## [22] 29.304383 42.936910 44.927123 39.790763 49.546199 39.285874 45.149933  
## [29] 18.092168 23.927391 14.016245 14.926577 40.604194 48.861921 44.678658  
## [36] 21.203806 29.438488 31.450029 41.973803 38.783425 32.005493 39.598565  
## [43] 45.124435 35.692898 32.373517 31.644682 35.182664 42.478657 24.765243  
## [50] 47.667849 28.799449 13.600406 40.521654 44.947873 49.543408 6.663093  
## [57] 4.771184 10.506045 23.645520 27.888281 45.842459 20.583291 7.087307  
## [64] 49.622080 11.513606 27.597197 41.222688 18.857575 47.128986 49.663565  
## [71] 34.347505 49.161756 16.039788 38.322296 39.145275 25.516727 6.110292  
## [78] 33.803111 7.219098 39.382012 18.348904 47.243538 41.129679 41.542925  
## [85] 32.987849 30.660064 14.128644 44.845606 35.836990 47.705126 23.948817  
## [92] 26.076668 44.375738 9.726786 8.136368 14.944861 4.888177 28.895659  
## [99] 6.425720 46.863431

y=3.128+2.005\*random\_no  
y #yvalues

## [1] 72.19570 48.83660 82.05023 92.79067 57.08127 29.52040 33.07849  
## [8] 52.35080 54.47513 89.00075 102.89429 64.96377 44.12862 42.26769  
## [15] 78.31744 102.33377 18.67272 73.99559 28.58967 102.53789 15.09216  
## [22] 61.88329 89.21650 93.20688 82.90848 102.46813 81.89618 93.65362  
## [29] 39.40280 51.10242 31.23057 33.05579 84.53941 101.09615 92.70871  
## [36] 45.64163 62.15217 66.18531 87.28547 80.88877 67.29901 82.52312  
## [43] 93.60249 74.69226 68.03690 66.57559 73.66924 88.29771 52.78231  
## [50] 98.70204 60.87090 30.39681 84.37392 93.24849 102.46253 16.48750  
## [57] 12.69422 24.19262 50.53727 59.04400 95.04213 44.39750 17.33805  
## [64] 102.62027 26.21278 58.46038 85.77949 40.93744 97.62162 102.70345  
## [71] 71.99475 101.69732 35.28777 79.96420 81.61428 54.28904 15.37914  
## [78] 70.90324 17.60229 82.08894 39.91755 97.85129 85.59301 86.42156  
## [85] 69.26864 64.60143 31.45593 93.04344 74.98116 98.77678 51.14538  
## [92] 55.41172 92.10135 22.63021 19.44142 33.09245 12.92880 61.06380  
## [99] 16.01157 97.08918