R Lab 7. Matrix Operations and Multivariate Linear Regression

 # Define a matrix by entering its elements manually.

```
> B = matrix(c(1,2,3,4,5,6,7,8,9),3,3)
> B
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
> B^2                    # This is an element-wise operation
     [,1] [,2] [,3]
[1,]    1   16   49
[2,]    4   25   64
[3,]    9   36   81
```

```
> B%*%B                  # This is matrix multiplication, B^2 = B*B
     [,1] [,2] [,3]
[1,]   30   66  102
[2,]   36   81  126
[3,]   42   96  150
```

```
> t(B)                   # Transposed matrix
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

```
> cbind(B,B)             # Joining two matrices side by side (as columns)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    4    7    1    4    7
[2,]    2    5    8    2    5    8
[3,]    3    6    9    3    6    9
```

```
> rbind(B,B)             # Joining two matrices below each other (as rows)
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
[4,]    1    4    7
[5,]    2    5    8
[6,]    3    6    9
```

```
> B[1:2,1:3]             # Sub-matrix, a part of matrix B
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
```

 # Inverting matrices is available in package "matlib"

```
> install.packages("matlib")
> library(matlib)
```

```
> inv(B)
Error in Inverse(X, tol = sqrt(.Machine$double.eps), ...) :
  X is numerically singular
```

 # This means there is a linear dependence among columns (and among rows) of matrix B. Such matrices are not invertible, and they have a determinant equal det(B)=0

```
> det(B)
[1] 0

> B[1,1]=100
> B
     [,1] [,2] [,3]
[1,]  100    4    7
[2,]    2    5    8
[3,]    3    6    9
> inv(B)
             [,1]        [,2]        [,3]
[1,]  0.01010101 -0.02020202  0.01010101
[2,] -0.02020202 -2.95959596  2.64646465
[3,]  0.01010101  1.97979798 -1.65656566
```

 # We changed the matrix by adding a "ridge", and now the inverse B^{-1} exists

 # Define a matrix from the "mtcars" data set and build a regression model that predicts miles per gallon based on the number of cylinders, horsepower, axel ratio, weight, and acceleration time.

```
> head(mtcars)
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1

> X = data.matrix(mtcars[,c(2,4:7)])
```

 # X-matrix of predictors

```
> head(X)
                  cyl  hp drat    wt  qsec
Mazda RX4           6 110 3.90 2.620 16.46
Mazda RX4 Wag       6 110 3.90 2.875 17.02
Datsun 710          4  93 3.85 2.320 18.61
Hornet 4 Drive      6 110 3.08 3.215 19.44
Hornet Sportabout   8 175 3.15 3.440 17.02
Valiant             6 105 2.76 3.460 20.22

> Y = data.matrix(mtcars[,1])
```

 # Vector of responses

```
> n = length(Y)
> one = matrix(1,n,1)
```

 # We also need a vector of 1s to include the intercept

```
> X = cbind(one,X)
```

```
> t(X) %*% X
```
 # This is matrix X'X

```
                     cyl        hp       drat         wt      qsec
       32.000   198.000   4694.00   115.0900   102.9520   571.160
cyl   198.000  1324.000  32204.00   691.4000   679.4040  3475.560
hp   4694.000 32204.000 834278.00 16372.2800 16471.7440 81092.160
drat  115.090   691.400  16372.28   422.7907   358.7190  2056.914
wt    102.952   679.404  16471.74   358.7190   360.9011  1828.095
qsec  571.160  3475.560  81092.16  2056.9140  1828.0946 10293.480
```

```
> slope = inv(t(X) %*% X) %*% t(X) %*% Y
```
 # Slope $\beta = (X'X)^{-1}X'Y$

```
> slope
            [,1]
[1,] 25.94553598
[2,] -0.48955421
[3,] -0.01505188
[4,]  1.13092435
[5,] -3.38272539
[6,]  0.34985343
```

 # We can certainly get the same slopes by the usual regression command "lm"

```
> lm( mpg ~ cyl + hp + drat + wt + qsec )

Call:
lm(formula = mpg ~ cyl + hp + drat + wt + qsec)

Coefficients:
(Intercept)          cyl           hp         drat           wt
qsec
   25.94521     -0.48967     -0.01539      1.13077     -3.38279
0.35011
```

 # Our estimated regression equation is
 # mpg = 25.95 – 0.49 cyl – 0.015 hp + 1.13 drat – 3.38 wt + 0.35 qsec + ε