LINEAR ALGEBRA AND MATRIX COMPUTATION

Mohammad Wasiq

Matrix Using R Programming Language

Linear Algebra and Matrix Computation
DSM-1002

Table of Contents

1 Linear Algebra and Matrix Computation

Book :-: **Basic of Algebra for Statistics with R**
Teacher :-: **Prof. Ahmed Yusuf Adhami Sir**
Department of Statistics and Operation Research , Aligarh Muslim University , Aligarh

Writer :-: **Mohammad Wasiq**

(MS Data Science , AMU)

2 Matrix – Theory

Matrix : A matrix is a rectangular array (arrangement) of scalars, usually presented in the following form:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & a_{ij} & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}_{m\times n}$$

$$B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \cdots & \cdots & b_{ij} & \cdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix}_{m\times n}$$

2.1 Types of Matrix

2.1.1 Row Matrix

A matrix with only one row is called a row matrix or row vector.

$$\begin{bmatrix} a_{11} \\ a_{21} \\ \cdots \\ a_{m1} \end{bmatrix}_{m\times 1}$$

2.1.2 Column Matrix

A matrix with only one column is called a column matrix or column vector.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \end{bmatrix}_{1\times n},$$

2.1.3 Zero Matrix / Null Matrix

A matrix whose elements are all zero is called a zero matrix and is usually denoted by 0.

$$0 = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \cdots & \cdots & 0 & \cdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}_{m\times n}$$

2.1.4 Square Matrix

A matrix whose rows and columns are same.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}_{m\times m}$$

2.1.5 Diagonal Matrix

Let $A = [a_{ij}]$ be an n-square mtx. The diagonal of A consists of the elements with the same subscript , that is $a_{11}, a_{22}, \cdots, a_{ij}, \cdots, a_{mn} \quad ; m = n$

$$A = \begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & a_{ij} & \\ & & & a_{mn} \end{bmatrix}_{m\times n} \quad ; where \quad i = j/m = n$$

2.1.6 Trace of Matrix

The sum of all diagonal elements of a matrix A. It's denoted by **tr(A)** $a_{11} + a_{22} + \cdots + a_{ii} + \cdots + a_{mn} \quad ; i = 1,2..n = m \sum_{i=j=1}^{n} a_{ij} \quad ; i = j = 1,2,..,n$

Suppose $A = [a_{ij}]$ and $B = [b_{ij}]$ are n-square matrices and k is a scaler , then

- **tr(A + B) = tr(A) + tr(B)**
- **tr(kA) = ktr(A)**
- $tr(A^T) = tr(A)$
- **tr(AB) = tr(BA)**

2.1.7 Identity Matix / Unit Matrix

A square matrix which contain unit element diagonal matrix and all non-diagonal elements are zero .

$$I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdots & \cdots & 1 & \cdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{m\times n}$$

2.1.8 Triangular Matrix

If every element above or below the leading diagonal of a square matrix is zero , then the matrix is called a triangular matrix.

2.1.8.1 Lower Triangular Matrix

$$\begin{bmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{bmatrix}_{3\times 3}$$

2.1.8.2 upper Triangular Matrix

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix}_{3\times 3}$$

2.1.9 Symmetric Matrix

A square matrix is called symmetric , if for all values of i and j $a_{ij} = a_{ji} \Rightarrow A^T = A$

2.1.10 Skew Symmetric Matrix

A square matrix is called symmetric , if for all values of i and j $a_{ij} = -a_{ji} \Rightarrow A^T = -A$

2.1.11 Orthogonal Matrix

A aquare matrix **A** is called orthogonal matrix , if

$$AA^T = I$$

2.1.12 Conjugate of a Matrix

Let

$$A = \begin{bmatrix} -1+i & -2-3i & -4 \\ -7+2i & -i & -3-2i \end{bmatrix}_{2\times 3}$$

then Conjugate of Matrix A

$$\bar{A} = \begin{bmatrix} -1+i & -2+3i & -4 \\ -7-2i & -i & -3+2i \end{bmatrix}_{2\times 3}$$

2.1.13 Matrix A^θ

Transpose of the Conjugate Matrix A is A^θ .

$$A \rightarrow \bar{A} \rightarrow (\bar{A})^T = A^\theta$$

2.1.14 Unitary Matrix

A square matrix A is said to be unitary if $A^\theta A = I$

$$\bar{A} = \begin{bmatrix} \frac{1+i}{2} & \frac{-1+i}{2} \\ \frac{1+i}{2} & \frac{1-i}{2} \end{bmatrix}_{2\times 2}$$

2.1.15 Hermitian Matrix

A Square matrix $A = a_{ij}$ os said to be Hermitian matrix if every $i - j^{th}$ element of A is equal to conjugate complex of $j - i^{th}$ element of A or in other words $a_{ij} = -a_{ji}$ The necessary condition and sufficient condition for the matrix A to be Hermitian is that $A = A^\theta$

2.1.16 Skew Hermitian Matrix

$A^\theta = -A \Rightarrow (\bar{A})^T = -A$

2.1.17 Idempotent Matrix

$A^2 = A$

2.1.18 Involutory Matrix

$A^2 = I$

2.1.19 Nilpotent Matrix

$A^k = 0$

2.1.20 Periodic Matrix

$A^{k+1} = A$

2.1.21 Normal Matrix

A real matrix is normal if it commutes with its transpose A^T , that is, $AA^T = A^T A$ All *symmetric, skew-symmetric* and *orthogonal* matrices are normal matrices.

$$\begin{bmatrix} 6 & -3 \\ 3 & 6 \end{bmatrix}$$

2.2 Algebra of Matrix

2.2.1 Addition of Matrix

Consider matrices A , B , C (with same size) and any scalars k and k' , then ,

- **(A+B) + C = A + (B+C)**
- **A + 0 = 0 + A = 0**
- **A + (-A) = (-A) + A = 0**
- **A + B = B = A**
- **k(A + B) = kA + kB**
- **(k + k')A = kA + k'A**
- **(kk')A = k(k'A) = k'(kA)**
- **1.A = A**

2.2.2 Multiplication of Matrix

- **(AB)C = A(BC)** {Associative Law}
- **A(B + C) = AB + AC** {Left Distribution}
- **(B + C)A = BA + CA** {Right Distribution}
- **k(AB) = (kA)B = A(kB)**
- $AB \neq BA$
- **0.A = 0**

2.3 Transpose of a Matrix

The transpose of a Matrix A is the matrix obtained by writing the columns of A in order as row. * **Properties** * $(A+B)^T = A^T + B^T$ * $(A^T)^T = A$ * $(kA)^T = kA^T$ * $(AB)^T = B^T A^T$

2.4 Rank of a Matrix

Let A be any $m \times n$ matrix, It has square sub-matrices of different orders. The determinent of these square sub-matrices are called minors of A.

A mtx. A is said to be of rank **r** if ,

- It has atleast one non-zero minor of order **r**.
- All the minors of order or higher than are zero.
- It's denoted by $\rho(A)$ or $r(A)$.

Properties

- If is a null matrix, then $\rho(A) = 0$.
- If A is not a null matrix, then $\rho(A) \geq 1$
- If A is an $m \times n$ matrix, then $\rho(A) \leq min(m, n)$.
- If A is a non-singular matrix, then $\rho(A) = n$.
- Rank of I_n is n .
- $\rho(A) = \rho(A^T)$.

2.5 Echelon Form Matrix

A matrix is in echelon form, if it satisfies the following conditions:

- The first nonzero element in each row, called the leading element, is always strictly to the right of the leading element of the row above it.
- Rows with all zero elements, if any are below the rows having a nonzero element.

2.6 Reduced Row Echelon Form Matrix

A matrix is in echelon form, if it satisfies the following conditions:

- The first nonzero element in each row, called the leading element, is always strictly to the right of the leading element of the row above it.
- Rows with all zero elements, if any are below the rows having a nonzero element
- The leading element in each nonzero row should be *1*.

- Each leading is the only nonzero element in its column.

2.6.1 Pivot Position

A position of a leading element in an echelon form of a matrix is called pivot position.

2.6.2 Pivot Column

A column that contains a pivot position is called a pivot column.

2.7 Rank of a Matrix by Echelon Form

Once a matrix is transformed into an echelon form by using the *elementary row* operations, rank of the matrix is equal to the number of nonzero rows in its echelon form matrix.

2.7.1 Rank of a Matrix by Normal Form method

In Ais an $m \times n$ matrix and by a series of elementary (row or column or both) transformations, it can be put into one of the following forms (called normal or canonical forms):

$$\begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} I_r \\ 0 \end{bmatrix}, [I_r \quad 0], [I_r]$$

2.8 Characteristic Equation , Eigen Value & Eigen Vector

For every square matrix A of order n , we can form a matrix $A - \lambda I$, where λ is a scalar and I is the unit matrix of order n. The determinant of this matrix equated to zero i.e.,$|A - \lambda I| = 0$ is called the characteristic equation of . On expanding the determinant we can write this equation as

$$(-1)^n[\lambda^n + k_1\lambda^{n-1} + k_2\lambda^{n-2} + \cdots + k_n] = 0$$

The roots of this equation are called **characteristic roots** or **latent roots** or **eigen values** . Now consider $(A - \lambda I)X = 0$

$$i.e. \quad A = \begin{bmatrix} a_{11} - \lambda & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} - \lambda & \cdots & a_{2n} \\ \cdots & \cdots & a_{ii} - \lambda & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} - \lambda \end{bmatrix}_{n \times n} \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}_{n \times 1} = \begin{bmatrix} 0 \\ 0 \\ \cdots \\ 0 \end{bmatrix}_{n \times 1}$$

These equations will have a non-trivial solution, only if $\rho(A - \lambda I) < n$ (=no. of unknown) , which is possible when $(A - \lambda I)$is singular, i.e., if $|A - \lambda I| = 0$.

This is the characteristic equation of the matrix A and has roots, which are the eigen values of A . Corresponding to each root, the homogeneous system $(A - \lambda I)X = 0$, has a nonzero solution

$$\begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}_{n\times 1}$$

which is called the **eigen vector or latent vector**.

Properties

- Any mtx. A and it's A^T have the same eigen values.
- The sum of eigen values of a mtx. is equal to the sum of the elements on the principal diagonal of the mtx.
- The product of the eigen values of a mtx. A is equal to the determination of A .
- $\lambda_1, \lambda_2, \ldots, \lambda_m$, where $\lambda_i{}'s$are eigen values of A.

$kA \to k\lambda_1, k\lambda_2, \ldots, k\lambda_m$

$A^m \to \lambda_1^m, \lambda_2^m, \ldots, \lambda_1^m$

$A^{-1} = \frac{1}{\lambda_1}, \frac{1}{\lambda_2}, \ldots, \frac{1}{\lambda_m}$ for non-singular matrix.

2.9 Cayley-Hamilton (C-H) Theorem

$$|A - \lambda I| = (-1)^n[\lambda^n + k_1\lambda^{n-1} + k_2\lambda^{n-2} + \cdots + k_n] = 0$$

then ,

$$(-1)^n[A^n + k_1A^{n-1} + k_2A^{n-2} + \cdots + k_nI] = 0$$

3 Summary of Matrix Operators in R

The operations given below assume that the orders of the matrices involved allow the operation to be evaluated. More details of these operations will be given in the note later. **R** is case sensitive so A and a denote distinct objects.

- To create a vector x $\Rightarrow x = c(x_1, x_2, \ldots, x_p)$.
- To access an individual element in a vector x, the i^{th} , **x[i]**.
- To create a matrix A $\Rightarrow A = matrix(data, nrow = m, ncol = n, byrow = F)$.
- To access an individual element in a matrix A, the $(i,j)^th$ $\Rightarrow$ **A[i,j]**.

- To access an individual row in a matrix A, the i^{th} ⇒ **A[i ,]**
- To access an individual column in a matrix A, the j^{th} ⇒ **A[, j]**.
- To access a subset of rows in a matrix A⇒ **A[i1 : i2 ,]**.
- To access a subset of columns in a matrix A ⇒ **A[, j1 : j2]**.
- To access a sub-matrix of A ⇒ **A[i1 : i2 , j1 : j2]**.
- Addition A+B ⇒ $A + B$.
- Subtraction A−B ⇒ $A - B$.
- Multiplication AB ⇒ **A %*% B**
- Hadamard multiplication $A \odot B \Rightarrow A * B$.
- Kronecker multiplication $A \otimes B$ ⇒ **A %x% B**.
- Transpose $A' \Rightarrow t(A)$
- Matrix cross-product $A'B \Rightarrow crossprod(A, B)$.
- Inversion $A^{-1} \Rightarrow solve(A)$.
- Moore–Penrose generalized inverse $A^+ \Rightarrow ginv(A)$ (in *MASS library*) [or $MPinv(A)$ (in *gnm library*)].

Note: ginv(. will work with almost any matrix but it is safer to use solve(.) if you expect the matrix to be non-singular since solve(.) will give an error message if the matrix is singular or non-square but ginv(.) will not.

- Determinant **det(A)** or $|A| \Rightarrow det(A)$.
- Eigenanalysis ⇒ $eigen(A)$.
- To extract a diagonal of a matrix A as a vector ⇒ $diag(A)$.
- Trace of a matrix A ⇒ **sum(diag(A))**.
- To create a diagonal matrix ⇒ $diag(c(x_{11}, x_{22}, \ldots, x_{pp}))$.
- To create a diagonal matrix from another matrix ⇒ $diag(diag(A))$.
- To change a dataframe into a matrix ⇒ **data.matrix(dataframe)**.
- To change some other object into a matrix ⇒ **as.matrix(object)**.
- To join vectors into a matrix as columns ⇒ $cbind(vec_1, vec_2, \ldots, vec_n)$.
- To join vectors into a matrix as rows ⇒ $rbind(vec_1, vec_2, \ldots, vec_n)$.
- To join matrices A and B together side by side ⇒ **cbind(A,B)**.
- To stack A and B together on top of each other ⇒ **rbind(A,B)** .
- To find the length of a vector x ⇒ **length(x)**.
- To find the dimensions of a matrix A ⇒ **dim(A)**

3.1 Creating Matrix

$$matix(c(x_1, x_2, \ldots, x_n)), nrow, ncol, byrow$$

```
# Create a Matrix A Column-wise (by default)
A <- matrix(c(1,2,3,4,5,6) , nrow = 2 , ncol = 3) ; A

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

```
# Create a Matrix B row-wise
B <- matrix(c(1,2,3,4,5,6) , nrow = 2 , ncol = 3 , byrow = T) ; B

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

3.2 Extraction of Matrix

```
# Extract the full Row
A[2 ,]

## [1] 2 4 6

# Extract the full Column
A[, 2]

## [1] 3 4

# Extract the element located at row 2 , col 2
A[2 , 2]

## [1] 4
```

3.3 Mathematical Operation on Matrix

```
# Addition of Matrices
A + B

##      [,1] [,2] [,3]
## [1,]    2    5    8
## [2,]    6    9   12

# Subtraction of Matrices
A - B

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




# Multiply the matrix by 2
2 * A

##      [,1] [,2] [,3]
## [1,]    2    6   10
## [2,]    4    8   12

A * 2
```

```
##      [,1] [,2] [,3]
## [1,]    2    6   10
## [2,]    4    8   12
```

```
# Multiplication of Matrices
#  A*B gives element-by-element multiplication (the Hadamard or Schur
product) which is rarely required
A*B
```

```
##      [,1] [,2] [,3]
## [1,]    1    6   15
## [2,]    8   20   36
```

3.4 Transpose of Matix

```
A
```

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

```
# Transpose of Matix
t(A)
```

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

```
# Multiplication of t(A) and B by %*%
t(A)%*%B
```

```
##      [,1] [,2] [,3]
## [1,]    9   12   15
## [2,]   19   26   33
## [3,]   29   40   51
```

3.5 Kronecker Products of Matrices

```
# Kronecker Products of A and B
A%x%B
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
## [1,]    1    2    3    3    6    9    5   10   15
## [2,]    4    5    6   12   15   18   20   25   30
## [3,]    2    4    6    4    8   12    6   12   18
## [4,]    8   10   12   16   20   24   24   30   36
```

```r
# Kronecker Products of A and B
B%x%A

##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
## [1,]    1    3    5    2    6   10    3    9   15
## [2,]    2    4    6    4    8   12    6   12   18
## [3,]    4   12   20    5   15   25    6   18   30
## [4,]    8   16   24   10   20   30   12   24   36
```

3.6 Length & Dimension of Matrix

```r
# No. of Elements in Matrix
length(A)

## [1] 6

# Dimension of Matrix
dim(A)

## [1] 2 3

dim(t(A))

## [1] 3 2

# Class of Matrix
class(A)

## [1] "matrix" "array"
```

3.7 Joining of Matrices

$$rbind() \quad cbind()$$

```r
# Joining Row-wise
rbind(A , B)

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

# Joining Column-wise
cbind(A , B)
```

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

```r
# Transpose of Joined Matrix
t(rbind(t(A) , t(B)))
```

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

```r
t(cbind(t(A) , t(B)))
```

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

3.8 Naming of Matrix

```r
dim(A)
```

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

```r
# Columns Names
colnames(A) <- c("Joya", "Amoha" , "UP")

# Rows Names
rownames(A) <- c("A" , "B")
A
```

```
##   Joya Amoha UP
## A    1     3  5
## B    2     4  6
```

```r
# Another Method for Naming Matrix
dimnames(A) <- list(c("A" , "B") , c("C" , "D" , "E"))
A
```

```
##   C D E
## A 1 3 5
## B 2 4 6
```

```r
# Another Method
rn <- c("a" , "b")
cn <-c("c" , "d" , "e")
rownames(A) <- rn
colnames(A) <- cn
A
```

```
##   c d e
## a 1 3 5
## b 2 4 6
```

3.9 Diagonal & Trace of Matrix

```
E <- matrix(c(1,2,3,4,5,6,7,8,9) , 3, 3 , byrow = T) ; E

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

# Diagonal of Matrix
diag(E)

## [1] 1 5 9

# Diagonal of Diagonal Matrix , it show only diagonal matrix
diag(diag(E))

##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    5    0
## [3,]    0    0    9

# Same as above
diag(c(1 , 5 , 9))

##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    5    0
## [3,]    0    0    9

# Trace of Matrix
sum(diag(E))

## [1] 15

# Trace of Product
F <- matrix(1:9 , 3,3)

sum(diag(E %*% F))

## [1] 285

sum(diag(F %*% E))

## [1] 285
```

3.10 Transepose of Product

```
t(E %*% F) # Correct way

##      [,1] [,2] [,3]
## [1,]   14   32   50
## [2,]   32   77  122
## [3,]   50  122  194

t(E) %*% t(F)

##      [,1] [,2] [,3]
## [1,]   66   78   90
## [2,]   78   93  108
## [3,]   90  108  126
```

3.11 Determinant of Matrix

```
G <- matrix(c(1,-2,2,2,0,1,1,1,-2) , 3,3, byrow = T) ; G

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

# Determinent of Matrix
det(G)

## [1] -7
```

3.12 Inverse of Matrix

Make sure that the Matrix should be **Square** and **Non-Singular**

```
A <- matrix(c(1,-2,2,2,0,1,1,1,-2) , nrow = 3 , ncol = 3) ; A

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

# Inverse of Matrix
solve(A)

##           [,1]       [,2]       [,3]
## [1,] 0.1428571 -0.7142857 -0.2857143
## [2,] 0.2857143  0.5714286  0.4285714
## [3,] 0.2857143 -0.4285714 -0.5714286

# Create a Matrix B row-wise
B <- matrix(c(4,6,5,-7,6,4,0,9,3) , nrow = 3 , ncol = 3 , byrow = T) ; B

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

```r
## [2,]   -7    6    4
## [3,]    0    9    3

# Inverse of Multiplication of Matix
solve(A %*% B)

##              [,1]        [,2]       [,3]
## [1,] -0.013136289 -0.11822660 -0.0771757
## [2,]  0.031198686 -0.05254516 -0.1083744
## [3,]  0.001642036  0.01477833  0.1346470

# Inverse by MASS Libray
library(MASS)
ginv(A %*% B)

##              [,1]        [,2]       [,3]
## [1,] -0.013136289 -0.11822660 -0.0771757
## [2,]  0.031198686 -0.05254516 -0.1083744
## [3,]  0.001642036  0.01477833  0.1346470
```

3.13 Eigen Values & Eigen Vector of Matrix

```r
e <- eigen(A) ; e

## eigen() decomposition
## $values
## [1] -2.509755+0.000000i  0.754878+1.489724i  0.754878-1.489724i
##
## $vectors
##               [,1]                   [,2]                   [,3]
## [1,] -0.0365345+0i  0.6969814+0.0000000i  0.6969814+0.0000000i
## [2,] -0.3948878+0i -0.2817990+0.5292839i -0.2817990-0.5292839i
## [3,]  0.9180027+0i  0.3927524-0.0202581i  0.3927524+0.0202581i

# Eigen Values of Matrix
round(e$values , 2)

## [1] -2.51+0.00i  0.75+1.49i  0.75-1.49i

# Eigen Vector of Matrix
round(e$vectors , 2)

##          [,1]        [,2]        [,3]
## [1,] -0.04+0i  0.70+0.00i  0.70+0.00i
```

```
## [2,] -0.39+0i -0.28+0.53i -0.28-0.53i
## [3,]  0.92+0i  0.39-0.02i  0.39+0.02i
```

```r
# Eigen of two Matries
e2 <- eigen(A %*% B)
round(e2$values, 2)
```

```
## [1] -6.48+13.68i -6.48-13.68i  7.97+ 0.00i
```

```r
round(e2$vectors , 2)
```

```
##              [,1]         [,2]     [,3]
## [1,] -0.88+0.00i -0.88+0.00i -0.03+0i
## [2,] -0.13-0.46i -0.13+0.46i -0.52+0i
## [3,]  0.03+0.03i  0.03-0.03i  0.85+0i
```

3.14 Singular Value Decomposition

```r
X <- matrix(c(1,2,3,4,5,6,7,8,9),3,3,byrow=T); X
```

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

```r
# Singular Value Decomposition
s <- svd(X)

# d
round(s$d , 2)
```

```
## [1] 16.85  1.07  0.00
```

```r
# u
round(s$u , 2)
```

```
##       [,1]  [,2]  [,3]
## [1,] -0.21  0.89  0.41
## [2,] -0.52  0.25 -0.82
## [3,] -0.83 -0.39  0.41
```

```r
# v
round(s$v , 2)
```

```
##       [,1]  [,2]  [,3]
## [1,] -0.48 -0.78 -0.41
## [2,] -0.57 -0.08  0.82
## [3,] -0.67  0.63 -0.41
```

3.15 Cross-Product of Matrix

It's same as $X^T X$

```
A <- matrix(c(1,2,3,4,5,6), nrow=2, ncol=3, byrow=T)
B <- matrix(c(1,2,3,4,5,6), nrow=2, ncol=3, byrow=T)

# Cross-Product of single matrix
crossprod(A)

##      [,1] [,2] [,3]
## [1,]   17   22   27
## [2,]   22   29   36
## [3,]   27   36   45

# Cross-Product of two matrices
crossprod(A , B)

##      [,1] [,2] [,3]
## [1,]   17   22   27
## [2,]   22   29   36
## [3,]   27   36   45
```

3.16 Statistics on Matrix

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

# Mean of complete matrix
mean(X)

## [1] 3.888889

# Standard Deviation of Matrix
sd(X)

## [1] 1.364225

# Column wise Sum of Matrix
colSums(X)

## [1] 10 12 13

# Column-wise Mean of Matrix
colMeans(X)

## [1] 3.333333 4.000000 4.333333

# Column-wise Summary of Matrix
summary(X)

##        V1              V2          V3
##  Min.   :2.000   Min.   :2   Min.   :4.000
##  1st Qu.:2.500   1st Qu.:3   1st Qu.:4.000
```

```
## Median :3.000   Median :4   Median :4.000
## Mean   :3.333   Mean   :4   Mean   :4.333
## 3rd Qu.:4.000   3rd Qu.:5   3rd Qu.:4.500
## Max.   :5.000   Max.   :6   Max.   :5.000
```

```
# Row-wise Sum of Matrix
rowSums(X)
```

```
## [1] 12 12 11
```

```
# Row-wise Mean
rowMeans(X)
```

```
## [1] 4.000000 4.000000 3.666667
```

Other Statistical Functions can be find by using $apply()$

3.17 Rank of Matrix

```
library(Matrix)
a <- matrix(1:9 , 3,3) ; a
```

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

```
# Rank of a Matrix
rankMatrix(a)
```

```
## [1] 2
## attr(,"method")
## [1] "tolNorm2"
## attr(,"useGrad")
## [1] FALSE
## attr(,"tol")
## [1] 6.661338e-16
```

3.18 Solving the Linear Equations

Task : Solve the following Equations

$$x + 2y + 3z = 20$$

$$2x + 2y + 3z = 100$$

$$3x + 2y + 8z = 200$$

```r
# create matrix A and B using given equations
A <- rbind(c(1, 2, 3),
           c(2, 2, 3),
           c(3, 2, 8))
A
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    2    2    3
## [3,]    3    2    8
```

```r
B <- c(20, 100, 200) ; B
```

```
## [1]  20 100 200
```

```r
# Solve them using solve function in R
solve(A, B)
```

```
## [1]  80 -36   4
```