Contents

Chapter 1

Linear Algebra

1.1 With R

Reader is assumed to have installed R in the machine and have some Google knowledge on R. Aja captures the essence of Linear Algebra through R language, peculiar syntax of R is explained wherever needed. We knew you are more active and eager in trying out following code and we being lazy skipped the outputs!

1.1.1 Vector

To create a vector in R, there are several approaches. Lets create a vector called **"vec"**:

```
# We can use the concatenate function to make an
   ordered list of numbers, e.g.,
vec <- c( 1.9, -2.8, 5.6, 0, 3.4, -4.2 )
vec
# We can use the colon operator to make a sequence of
   integers, e.g.,
vec <- -5:5
vec
vec <- 5:-5
vec
# We can use the sequence function. We have to give a
     starting value, and ending value, and an increment
```

```
    value, e.g.,
vec <- seq(-3, 3, by=0.1)
vec
# We can use the replicate function to a get a vector
   whose elements are all the same, e.g.,
vec <- rep(1,10)
vec
#We can concatenate two or more vectors to make a
   larger vector, e.g.,
vec <- c( rep(1,3), 2:5, c(6,7,8))
vec
# We can take one of the columns from a data table and
    make it a vector, e.g.,
vec <- Data$ACT
```

Some vector functions available:

```
# Vector misc functions
# If we want to know how many elements are in the
   vector, we use the length function, e.g.,
length(vec)
# If we want to reference an element in the vector, we
    use square brackets, e.g.,
vec[5]
# If we want to reference a consecutive subset of the
   vector, we use the colon operator within the square
    brackets, e.g.,
vec[5:10]
```

1.1.2 Matrix

To create a matrix in R, we may use the matrix function. We need to provide a vector containing the elements of the matrix, and specify either the number of rows or the number of columns of the matrix. This number should divide evenly into the length of the vector, or we will get a warning.

```
# For example, to make a 4 x 5 matrix named mat
    consisting of the integers 1 through 20, we can do
    this:
mat <- array(1:20, dim=c(4,5))
mat

# For example, to make a 2 x 3 matrix named mat
    consisting of the integers 1 through 6, we can do
    this:
mat <- matrix( 1:6, nrow=2 )
#or this
mat <- matrix ( 1:6, ncol=3 )
#we get
mat
#Note that R places the row numbers on the left and
    the column numbers on top.  Also note that R filled
     in the matrix column-by-column.  If we prefer to
    fill in the matrix row-by-row, we must activate the
     byrow setting, e.g.,
mat <- matrix( 1:6, ncol=3, byrow=TRUE )
# In place of the vector 1:6  you would place any
    vector containing the desired elements of the
    matrix.

#Extracting a Row of a Matrix
mat[1,]
#Extracting a Column of a Matrix
mat[,1]

#Extracting several Rows and/or columns
mat <- matrix( 1:25, ncol=5, byrow=TRUE )
mat
mat[1:3,2:4]

#You can combine several matrices with the same number
     of
#columns by joining them as rows/cols, using the rbind
    ()/cbind()
```

```
#command
matA <- matrix(c (1 ,3 ,3 ,9 ,6 ,5) ,2 ,3)
matB <- matrix(c (9 ,8 ,8 ,2 ,9 ,0) ,2 ,3)
matA
matB
rbind(matA,matB)
rbind(matB,matA)
cbind(matA,matB)
cbind(matB,matA)

# Matrix Opeartions
# Transpose
matA
t( matA )

# Addition
matA <- matrix ( rep(2,6), nrow=2, ncol=3, byrow=FALSE
    )
matB <- matrix ( rep(1,6), nrow=2, ncol=3, byrow=FALSE
    )
matA
matB
matA + matB
matA - matB
```

1.1.2.1 Inverse of a Matrix

For a square matrix A, the inverse is written A^-1. When A is multiplied by A$^-$1 the result is the identity matrix I. Non-square matrices do not have inverses.

Note: Not all square matrices have inverses. A square matrix which has an inverse is called **invertible** or **nonsingular**, and a square matrix without an inverse is called **noninvertible** or **singular**.

$AA^{-1} = A^{-1}A = I$

Example :

For matrix $A_{2,2} = \begin{bmatrix} 4 & 3 \\ 3 & 2 \end{bmatrix}$, its inverse is $A^{-1} = \begin{bmatrix} -2 & 3 \\ 3 & -4 \end{bmatrix}$ since

$$AA^{-1} = \begin{bmatrix} 4 & 3 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} -2 & 3 \\ 3 & -4 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and}$$
$$A^{-1}A = \begin{bmatrix} -2 & 3 \\ 3 & -4 \end{bmatrix} \begin{bmatrix} 4 & 3 \\ 3 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Here are three ways to find the inverse of a matrix:

1.1.2.1.1 Shortcut for 2x2 matrices

For $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, the inverse can be found using this formula:
$A^{-1} = \frac{1}{detA} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$

1.1.2.1.2 Augmented matrix method

Before we find the inverse, we see how to do **Gauss-Jordan elimination**. Use Gauss-Jordan elimination to transform [A — I] into [I — A-1].

Example : The following steps result in $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^{-1}$

Finding the A^{-1}

$$\left[\begin{array}{cc|cc} 1 & 2 & 1 & 0 \\ 3 & 4 & 0 & 1 \end{array}\right] = \left[\begin{array}{cc|cc} 1 & 2 & 1 & 0 \\ 0 & -2 & -3 & 1 \end{array}\right] = \left[\begin{array}{cc|cc} 1 & 2 & 1 & 0 \\ 0 & 1 & 3/2 & -1/2 \end{array}\right]$$
$$= \left[\begin{array}{cc|cc} 1 & 0 & -2 & 1 \\ 0 & 1 & 3/2 & -1/2 \end{array}\right]$$

So we see that $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^{-1} = \left[\begin{array}{cc|cc} 1 & 0 & -2 & 1 \\ 0 & 1 & 3/2 & -1/2 \end{array}\right]$

1.1.2.1.2.1 Gauss-Jordan elimination The system of equations
$x + y + z = 3$
$2x + 3y + 7z = 0$
$x + 3y - 2z = 17$
$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 3 \\ 2 & 3 & 7 & 0 \\ 1 & 3 & -2 & 17 \end{array}\right]$
Row operations can be used to express the matrix in reduced row-echelon form.
Which has following properties

- Each row contains only zerosuntil the first nonzero element, which must be 1.
- As the rows are followed from top to bottom, the first nonzero number occurs further to the right than in the previous row.
- The entries above and below the first 1 in each row must all be 0.
- Matrix in the reduced row-echelon form : $\begin{bmatrix} 1 & 0 & -2 & 0 & 6 \\ 0 & 1 & 7 & 0 & 1 \\ 0 & 0 & 0 & 1 & 5 \end{bmatrix}$

Solving the system of equations

$$\begin{aligned} \left[\begin{array}{ccc|c} 1 & 1 & 1 & 3 \\ 2 & 3 & 7 & 0 \\ 1 & 3 & -2 & 17 \end{array}\right] &= \left[\begin{array}{ccc|c} 1 & 1 & 1 & 3 \\ 0 & 1 & 5 & -6 \\ 0 & 2 & -3 & 14 \end{array}\right] \\ &= \left[\begin{array}{ccc|c} 1 & 0 & -4 & 9 \\ 0 & 1 & 5 & -6 \\ 0 & 0 & -13 & 26 \end{array}\right] \\ &= \left[\begin{array}{ccc|c} 1 & 0 & -4 & 9 \\ 0 & 1 & 5 & -6 \\ 0 & 0 & 1 & -2 \end{array}\right] \\ &= \left[\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 1 & -2 \end{array}\right] \end{aligned}$$

The augmented matrix now says that x=1, y=4, and z=-2.

1.1.2.1.3 Adjoint method

$A^{-1} = \frac{1}{det A}(adjoint of A)$
or
$A^{-1} = \frac{1}{det A}(cofactor of A)^T$

Example:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 4 & 5 \\ 1 & 0 & 6 \end{bmatrix}$$

Cofactor matrix is

$$\begin{bmatrix} A_{1,1} & -A_{1,2} & A_{1,3} \\ -A_{2,1} & A_{2,2} & -A_{2,3} \\ A_{3,1} & -A_{3,2} & A_{3,3} \end{bmatrix}$$

where $A_{1,1} = \begin{bmatrix} 4 & 5 \\ 0 & 6 \end{bmatrix} = 24$ $A_{1,2} = \begin{bmatrix} 0 & 5 \\ 1 & 6 \end{bmatrix} = 5....$ $A_{3,3} = \begin{bmatrix} 1 & 2 \\ 0 & 4 \end{bmatrix} = 4$

The cofactor matrix for A is $\begin{bmatrix} 24 & 5 & -4 \\ -12 & 3 & 2 \\ -2 & -5 & 4 \end{bmatrix}$, so the adjoint is $\begin{bmatrix} 24 & -12 & -2 \\ 5 & 3 & -5 \\ -4 & 2 & 4 \end{bmatrix}$.
Since det A = 22, we get $A^{-1} = \frac{1}{22}\begin{bmatrix} 24 & -12 & -2 \\ 5 & 3 & -5 \\ -4 & 2 & 4 \end{bmatrix} = \begin{bmatrix} 12/11 & -6/11 & -1/11 \\ 5/22 & 3/22 & -5/22 \\ -2/11 & 1/11 & 2/11 \end{bmatrix}$

```
# Multiplication
#Scalar
3 * matA
# To multiply two matrices with compatible dimensions
    (i.e., the number of columns of the first matrix
    equals the number of rows of the second matrix), we
     use the matrix multiplication operator %*%.  For
    example,
```

```
matA <- matrix( rep(1,6), nrow=2, ncol=3 )
matB <- matrix( rep(1,6), nrow=3, ncol=2 )
matA %*% matB
#If we just use the multiplication operator *,  R will
    multiply the corresponding elements of the two
   matrices, provided they have the same dimensions.
   But this is not the way to multiply matrices.

#Likewise, to multiply two vectors to get their scalar
    (inner) product, we use the same operator, e.g.,
#Technically, we should use the transpose of a.  But R
    will transpose a for us rather than giving us an
   error message.
a %*% b

#To create the identity matrix for a desired dimension
   , we use the diagonal function, e.g.,
I <- diag( 5 ) #5 x 5
I

#To find the determinant of a square matrix N, use the
    determinant function, e.g.,
A <- matrix( c(1,2,3,0,4,5,1,0,6), nrow=3, byrow=TRUE)
det( A )

# To obtain the inverse N-1 of an invertible square
   matrix N, we use the solve function, e.g.,
# If the matrix is singular (not invertible), or
   almost singular, we get an error message.
solve( A )

# Lets test matrix inversion
#AA^-1 = I
A <- matrix (c(1,3,3,9,6,5,9,1,8), ncol=3)
solve(A)
A %*% solve(A)
#See the difference with following code
zapsmall(A %*% solve(A))
```

1.1.2.2 Solving Systems of Linear Equation

Let the system of equations be
$3x_1 + 2x_2 - 1x_3 = 1$
$2x_1 - 2x_2 + 4x_3 = -2$
$-x_1 + 0.5x_2 - x_3 = 0$

$$\begin{bmatrix} 3 & 2 & -1 \\ 2 & -2 & +4 \\ -1 & 0.5 & -1 \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ -2 \\ 0 \end{bmatrix}$$

$A * x = b$
$A^{-1} * b = x$

```
A <- array(c(3,2,-1,2,-2,0.5,-1,4,-1), dim=c(3,3))
b <- c(1,-2,0)
solve(A,b)

A %*% solve(A,b)
#A * x = b
zapsmall(A %*% solve(A,b))
# A^-1 * b = x
solve(A) %*% b
```