PShaji_Assignment2

Priya Shaji

9/4/2019

Problem set 1

(1) Show that $A^T A$ not equal to AA^T in general. (Proof and demonstration.)

Answer 1)

Demonstration

Let $A =$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

And, $A^T =$

$$\begin{bmatrix} e & f \\ g & h \end{bmatrix}$$

Therefore,

$A^T A =$

$$\begin{bmatrix} ea + fc & eb + fd \\ ga + hc & gb + hd \end{bmatrix}$$

Now, as we can see that the products of the A and A^t is not equal to each other.

Testing

```
A <- matrix(c(10,11,12,13),ncol = 2)
t<- 2
ATA <- A^t %*% A
AAT <- A %*% A^t

identical(ATA, AAT)
```

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

Hence, using R function 'identical()' , it is proved that product of transpose of a matrix(AT) with the original matrix(A) i.e. $A^T A$,is not equal to the product of original matrix(A) with its transpose(AT), i.e. AA^T

(2) For a special type of square matrix A, we get $A^T A = AA^T$. Under what conditions could this be true? (Hint: The Identity matrix I is an example of such a matrix).

Answer 2)

For a special type of square matrix A, we get $A^T A = AA^T$.

The above statement is true provided the given matrices are symmetrical.

Let us see the demonstration in R:

1) Matrix A

```
A <- matrix(c(1, 2, 1, 2, 1, 2, 1, 2, 1), nrow = 3, byrow = T)
A
```

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

2) Transpose of Matrix A: A^T

```
t(A)
```

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

3) Test $A^T A = AA^T$

```
ATA <- t(A) %*% A
AAT <- A %*% t(A)

AAT == ATA
```

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

Hence verified, the conditions required for a special type of square matrix A, we get $A^T A = AA^T$, is that the Matrix should be Symmetrical.

Problem set 2

Write an R function to factorize a square matrix A into LU or LDU, whichever you prefer.

Answer)

1) Let's generate a sequence of random numbers using `set.seed`
2) set the size of matrix as n^2 , where n is number of elements in a row or column

```r
set.seed(605)
n <- 4
A <- matrix(sample.int(8,size = n^2, replace = TRUE), ncol = n)
A
```

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

3) Create a funcntion LU_matrix

4) for the L matrix , we will set the diagonal elements as 1.

5) Now we will perform row operations in `L` and `U` matrix and iterate through the matrix via `for()` loop

```r
LU_matrix <- function(A){

  U = A
  L = diag(x = 1, ncol = ncol(A), nrow = nrow(A))

  for (i in 1:(nrow(U) -1)) {
    for (j in (i+1):nrow(U)){
      if (U[i, i] != 0){
        multiplier = U[j, i] / U[i, i]
        L[j, i] = multiplier
        U[j, ] = U[j, ] - multiplier * U[i, ]
      }

    }

  }
  return(list('L' = L, 'U' = U))

}

x <- LU_matrix(A)

## Print L matrix
x$L
```

```
##           [,1]      [,2]       [,3] [,4]
## [1,] 1.0000000 0.0000000  0.0000000    0
## [2,] 1.0000000 1.0000000  0.0000000    0
## [3,] 0.3333333 0.8666667  1.0000000    0
## [4,] 0.6666667 1.3333333 -0.4761905    1
```

```r
## Print U matrix

x$U
```

```
##      [,1]          [,2] [,3]     [,4]
## [1,]    3  2.000000e+00  8.0  8.00000
## [2,]    0  5.000000e+00 -1.0 -4.00000
## [3,]    0  0.000000e+00  4.2  8.80000
## [4,]    0 -8.881784e-16  0.0 11.19048
```

6) Perform matrix multiplication

```
x$L %*% x$U
```

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

As we can see that the above matrix is A matrix, therefore our L and U matrices produces verified results.

Now let us test our `LU_matrix()` function

```
matrix_test <- matrix(c(1:9), ncol = 3)
test <- LU_matrix(matrix_test)
## Print matrix L
test$L
```

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

```
## Print matrix U
test$U
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    0   -3   -6
## [3,]    0    0    0
```

Perform matrix multiplication to verify the result

```
all.equal(test$L %*% test$U, matrix_test)
```

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

Therefore, product of matrix L and matrix U is equal to matrix `matrix_test`, which verifies the result.