605HW2

Bin Lin

2017-2-18

1. Problem set 1

1. Show that ATA≠AAT in general. (Proof and demonstration.)

First, if A is not a square matrix, we can assume the dimension of A is m$\times$ n and m≠n Thereofore, A^T will have dimension of n $\times$ m. So that the resulting matrix we got for AA^T will be a m $\times$m matrix, while for A^TA, it will be n $\times$n matrix. Since m≠n, $AA^T \neq A^TA$.

Second, if A is a square matrix,

$$AA^T = \begin{bmatrix} a_{11} & a_{1n} \\ a_{n1} & a_{nn} \end{bmatrix} \times \begin{bmatrix} a_{11} & a_{n1} \\ a_{1n} & a_{nn} \end{bmatrix} = \begin{bmatrix} {a_{11}}^2 + {a_{1n}}^2 & a_{11} * a_{n1} \ + \ a_{1n} * a_{nn} \\ a_{11} * a_{n1} \ + \ a_{1n} * a_{nn} & {a_{n1}}^2 + {a_{nn}}^2 \end{bmatrix}$$

$$A^TA = \begin{bmatrix} a_{11} & a_{n1} \\ a_{1n} & a_{nn} \end{bmatrix} \times \begin{bmatrix} a_{11} & a_{1n} \\ a_{n1} & a_{nn} \end{bmatrix} = \begin{bmatrix} {a_{11}}^2 + {a_{n1}}^2 & a_{11} * a_{1n} \ + \ a_{n1} * a_{nn} \\ a_{11} * a_{1n} \ + \ a_{1n} * a_{nn} & {a_{1n}}^2 + {a_{nn}}^2 \end{bmatrix}$$

Obviously, $AA^T \neq A^TA$

2. For a special type of square matrix A, we get ATA != AAT . Under what conditions could this be true? (Hint: The Identity matrix I is an example of such a matrix). Please typeset your response using LaTeX mode in RStudio.

The first proof shown in the above tells us this special matrix has to be a square matrix. The second proof indicate the special matrix A has to be symmetric along the diagonal (where those pivot points are). According to the example that is provided, a_{1n} has to be equal a_{n1}. To generalize , a_{ij} has to be equal a_{ji}. Then we can get $AA^T = A^TA$.

2. Problem set 2 Write an R function to factorize a square matrix A into LU or LDU, whichever you prefer. You don't have to worry about permuting rows of A and you can assume that A is less than 5x5, if you need to hard-code any variables in your code. If you doing the entire assignment in R, then please submit only one markdown document for both the problems.

```
factorize <- function(A)
  {
  r <- dim(A)[1]
  c <- dim(A)[2]

  if (r!=c)
    {
    print ("Please enter a square matrix")
    }
  else
  {
    L <- diag(r)
    D <- diag(r)

    for (j in 1:(c-1))
      {
      for (i in (j+1):r)
        {
        multiplier <- (A[i,j]/A[j,j])
        A[i,] <- A[i,] - (multiplier * A[j,])
        L[i, j] <- multiplier

      }
    }

    }
  U <- A

  for (i in 1:r)
  {
    D[i,i] <- U[i,i]
    U[i,] <- U[i,] / U[i,i]
  }
  print ("Upper Triangular Matrix")
  print (U)
  print ("Lower Triangular Matrix")
  print (L)
  print ("Diagonal Matrix")
  print (D)
  print ("LDU")
  print (L %*% D %*% U)
}
A <- matrix(17:32, nrow = 4, ncol = 4, byrow = TRUE)
factorize(A)
```

```
## [1] "Upper Triangular Matrix"
##      [,1]     [,2]     [,3]     [,4]
## [1,]    1 1.058824 1.117647 1.176471
## [2,]    0 1.000000 2.000000 3.000000
## [3,]    0 0.000000 1.000000 1.500000
## [4,]    0 0.000000 0.000000 1.000000
## [1] "Lower Triangular Matrix"
##          [,1] [,2] [,3] [,4]
## [1,] 1.000000    0  0.0    0
## [2,] 1.235294    1  0.0    0
## [3,] 1.470588    2  1.0    0
## [4,] 1.705882    3  0.5    1
## [1] "Diagonal Matrix"
##      [,1]       [,2]         [,3]          [,4]
## [1,]   17  0.0000000 0.000000e+00  0.000000e+00
## [2,]    0 -0.2352941 0.000000e+00  0.000000e+00
## [3,]    0  0.0000000 7.105427e-15  0.000000e+00
## [4,]    0  0.0000000 0.000000e+00 -1.776357e-15
## [1] "LDU"
##      [,1] [,2] [,3] [,4]
## [1,]   17   18   19   20
## [2,]   21   22   23   24
## [3,]   25   26   27   28
## [4,]   29   30   31   32
```