DATA 605: Assignment 2

Semester: Spring 2021; Professor Doc Larry

Philip Tanofsky

07 February 2021

Problem Set 1

1. Show that $A^TA \neq AA^T$ in general. (Proof and demonstration.)
2. For a special type of square matrix A, we get $A^TA = AA^T$. Under what conditions could this be true? (Hint: The Identity matrix I is an example of such a matrix).

Part 1

Proof of $AA^T \neq A^TA$ in general.

In essence, prove matrix multiplication is not commutative. First, given a matrix A with size $m \times n$, if $m \neq n$, then AA^T would result in a $m \times m$ matrix while A^TA would result in a $n \times n$ matrix. The resulting matrices are different sizes, and thus not equal.

Proof

Now to prove $A^TA \neq AA^T$ in general for a square matrix.

Given a matrix

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

With a transpose as

$$A^T = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$$

Matrix multiply AA^T

$$AA^T = \begin{bmatrix} a^2 + b^2 & ac + bd \\ ac + bd & c^2 + d^2 \end{bmatrix}$$

Matrix multiply A^TA

$$A^TA = \begin{bmatrix} a^2 + c^2 & ab + cd \\ ab + cd & b^2 + d^2 \end{bmatrix}$$

Thus

$$\begin{bmatrix} a^2+b^2 & ac+bd \\ ac+bd & c^2+d^2 \end{bmatrix} \neq \begin{bmatrix} a^2+c^2 & ab+cd \\ ab+cd & b^2+d^2 \end{bmatrix}$$

Demonstration

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

With a transpose as

$$A^T = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Matrix multiply AA^T

$$AA^T = \begin{bmatrix} 10 & 14 \\ 14 & 20 \end{bmatrix}$$

Matrix multiply A^TA

$$A^TA = \begin{bmatrix} 5 & 11 \\ 11 & 25 \end{bmatrix}$$

Thus

$$\begin{bmatrix} 10 & 14 \\ 14 & 20 \end{bmatrix} \neq \begin{bmatrix} 5 & 11 \\ 11 & 25 \end{bmatrix}$$

Validation with R

```r
a = matrix(c(1,2,3,4), 2, 2)
at = matrix(c(1,3,2,4), 2, 2)

a
```

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

```r
at
```

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

```r
aat <- a %*% at

ata <- at %*% a

aat
```

```
##      [,1] [,2]
## [1,]   10   14
## [2,]   14   20
```

```r
ata
```

```
##      [,1] [,2]
## [1,]    5   11
## [2,]   11   25
```

```r
(aat == ata)
```

```
##       [,1]  [,2]
## [1,] FALSE FALSE
## [2,] FALSE FALSE
```

Part 2

Under what conditions is $AA^T = A^T A$?

Answer: When matrix A is **symmetric**. Thus $A = A^T$. And then, with substitution $AA = AA$.

$$A = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

With a transpose as

$$A^T = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

Matrix multiply AA^T

$$AA^T = \begin{bmatrix} 5 & 10 \\ 10 & 20 \end{bmatrix}$$

Matrix multiply $A^T A$

$$A^T A = \begin{bmatrix} 5 & 10 \\ 10 & 20 \end{bmatrix}$$

Thus

$$\begin{bmatrix} 5 & 10 \\ 10 & 20 \end{bmatrix} = \begin{bmatrix} 5 & 10 \\ 10 & 20 \end{bmatrix}$$

Validation with R

```r
a = matrix(c(1,2,2,4), 2, 2)
at = matrix(c(1,2,2,4), 2, 2)

a
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    2    4
```

```
at
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    2    4
```

```
aat <- a %*% at

ata <- at %*% a

aat
```

```
##      [,1] [,2]
## [1,]    5   10
## [2,]   10   20
```

```
ata
```

```
##      [,1] [,2]
## [1,]    5   10
## [2,]   10   20
```

```
(aat == ata)
```

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

Problem Set 2

Write an R function to factorize a square matrix A into LU or LDU, whichever you prefer. Please submit your response in an R Markdown document.

You don't have to worry about permuting rows of A and you can assume that A is less than 5×5, if you need to hard-code any variables in your code.

Solution

Based on approach from "LU Decomposition Shortcut Method" video.

```
# Using the algorithm from the LU Decomposition Shortcut Method video
# http://www.youtube.com/watch?v=UlWcofkUDDU

lu_decomposition <- function(matrix_in) {

  # Get row count of matrix
```

```
  n <- dim(matrix_in)[1]

  # Get column count of matrix
  m <- dim(matrix_in)[2]

  # Return error message if matrix not a square
  if (n != m) {
    return("Error: Matrix not a square")
  }

  # Return error message if matrix too big
  if (n > 4) {
    return("Matrix greater than 4x4")
  }

  # Set upper matrix to input matrix
  upper <- matrix_in
  # Define lower matrix with diagonal (identity matrix)
  lower <- diag(n)

  # Outer loop to traverse the matrices top to bottom by row
  for (i in 2:n) {

    # Inner loop to traverse the matrices left to right by column
    for (j in 1:(i-1)) {

      # Calculate multiplier
      # Really, just divide the values and set lower matrix value per i,j
      lower[i,j] <- upper[i,j] / upper[j,j]

      # Calculate the row values for the upper matrix
      # Opposite of multiplier (lower value just set) mutiplied by above row in upper matrix
      # then added to row under consideration in upper matrix
      upper[i, ] <- -lower[i,j] * upper[j, ] + upper[i, ]
      }
  }

  # Outputs
  print("Input Matrix")
  print(matrix_in)
  print("Upper Matrix")
  print(upper)
  print("Lower Matrix")
  print(lower)

  # Return resulting upper and lower matrices
  return(list(upper,lower))
}
```

Validations

```r
# Validations
#Not Square
#mat <- matrix(c(1, 2, 1, 2, 4, 3), nrow=2, ncol=3)
#Too big
#mat <- matrix(c(rep(2,36)), nrow=6, ncol=6)
#Invalid
#mat <- matrix(c(1, 2, 1, 2, 4, 3, 3, 5, 4), nrow=3, ncol=3)
#Valid
#mat <- matrix(c(2, -4, -4, -1, 6, -2, -2, 3, 8), nrow=3, ncol=3)
#Valid
mat <- matrix(c(-3, -12, 6, 0, -3, -15, 12, -15, 2, 6, 4, 6, 2, 12, -10, 29), nrow=4, ncol=4)

result <- lu_decomposition(mat)
```

```
## [1] "Input Matrix"
##      [,1] [,2] [,3] [,4]
## [1,]   -3   -3    2    2
## [2,]  -12  -15    6   12
## [3,]    6   12    4  -10
## [4,]    0  -15    6   29
## [1] "Upper Matrix"
##      [,1] [,2] [,3] [,4]
## [1,]   -3   -3    2    2
## [2,]    0   -3   -2    4
## [3,]    0    0    4    2
## [4,]    0    0    0    1
## [1] "Lower Matrix"
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    4    1    0    0
## [3,]   -2   -2    1    0
## [4,]    0    5    4    1
```

```r
# Output result
result
```

```
## [[1]]
##      [,1] [,2] [,3] [,4]
## [1,]   -3   -3    2    2
## [2,]    0   -3   -2    4
## [3,]    0    0    4    2
## [4,]    0    0    0    1
##
## [[2]]
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    4    1    0    0
## [3,]   -2   -2    1    0
## [4,]    0    5    4    1
```

```r
# If result was not an error, then compare results to original matrix
if (is.list(result)) {
  l_times_u <- result[[2]] %*% result[[1]]
  (mat == l_times_u)
}
```

```
##      [,1] [,2] [,3] [,4]
## [1,] TRUE TRUE TRUE TRUE
## [2,] TRUE TRUE TRUE TRUE
## [3,] TRUE TRUE TRUE TRUE
## [4,] TRUE TRUE TRUE TRUE
```