Data 605 - Assignment 4

Gabe Abreu

9/20/2020

Data 605 - Assignment 4

###Problem Set 1

Write code to in R to compute X = A(A^T) and Y = (A^T)A

```
#Create the matrix A
A <- matrix(c(1,2,3,-1,0,4), byrow = T, nrow = 2)
A
```

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

```
#Compute X by multiplying A by its transpose
X <- A%*%t(A)
X
```

```
##      [,1] [,2]
## [1,]   14   11
## [2,]   11   17
```

```
#Compute Y by multiplying the transpose by A
Y <- t(A)%*%A
Y
```

```
##      [,1] [,2] [,3]
## [1,]    2    2   -1
## [2,]    2    4    6
## [3,]   -1    6   25
```

Compute the eigenvalues and eigenvectors of X and Y using the built-in commands in R.

```
eigX <- eigen(X)
eigY <- eigen(Y)
```

Eigvenvalues:

```r
#Eigenvalues for X
eigX$values
```

```
## [1] 26.601802  4.398198
```

```r
#Eigenvalues for Y
eigY$values
```

```
## [1] 2.660180e+01 4.398198e+00 1.058982e-16
```

Eigenvectors:

```r
#Eigenvectors for x
eigX$vectors
```

```
##           [,1]       [,2]
## [1,] 0.6576043 -0.7533635
## [2,] 0.7533635  0.6576043
```

```r
#Eigenvectors for Y
eigY$vectors
```

```
##             [,1]       [,2]       [,3]
## [1,] -0.01856629 -0.6727903  0.7396003
## [2,]  0.25499937 -0.7184510 -0.6471502
## [3,]  0.96676296  0.1765824  0.1849001
```

Compute the the left-singular/singular values/right-singular vectors using svd command

```r
SVD_A <- svd(A)

sing_val <- SVD_A$d #singular values of the given matrix

left_val <- SVD_A$u #left singular vectors of matrix

right_val <- SVD_A$v #right singular vectors of matrix
```

Examine the two sets of singular vectors and show that they are indeed eigenvectors of X and Y.

```r
#Compare Eigenvectors of X and left values of A
list(eigX$vectors, left_val)
```

```
## [[1]]
##           [,1]       [,2]
## [1,] 0.6576043 -0.7533635
## [2,] 0.7533635  0.6576043
##
## [[2]]
##            [,1]       [,2]
## [1,] -0.6576043 -0.7533635
## [2,] -0.7533635  0.6576043
```

```r
#Compare Eigenvectors of Y and right values of A
list(eigY$vectors, right_val)
```

```
## [[1]]
##             [,1]       [,2]       [,3]
## [1,] -0.01856629 -0.6727903  0.7396003
## [2,]  0.25499937 -0.7184510 -0.6471502
## [3,]  0.96676296  0.1765824  0.1849001
##
## [[2]]
##             [,1]       [,2]
## [1,]  0.01856629 -0.6727903
## [2,] -0.25499937 -0.7184510
## [3,] -0.96676296  0.1765824
```

In addition, the two non-zero eigenvalues (the 3rd value will be very close to zero, if not zero) of both X and Y are the same and are squares of the non-zero singular values of A.

```r
#Let's look at the single values without them being squared
sing_val
```

```
## [1] 5.157693 2.097188
```

```r
#Single values squared
sq_values <- round(sing_val^2)
sq_values
```

```
## [1] 27  4
```

```r
round(eigX$values)
```

```
## [1] 27  4
```

```r
round(eigY$values)
```

```
## [1] 27  4  0
```

Problem Set 2

```r
myinverse <- function(A) {
    #Check if matrix is square
    if(dim(A)[1] != dim(A)[2]){
        return("Not a square matrix")
    }

    #If the matrix is 2x2, this will return the inverse
    if(dim(A)[1] == 2)
    {
```

```
        det_1 <- (1/((A[1,1]*A[2,2])-(A[1,2]*A[2,1])))

        A_2 <- matrix(c(A[2,2], -A[1,2], -A[2,1], A[1,1]), byrow = T, nrow = 2)

        return (det_1 * A_2)
    }


    #Create Co-factor Matrix
    C_Matrix <- matrix(rep(0,length(A)),ncol=ncol(A))


    for (i in 1:ncol(A)) {
        for (j in 1:nrow(A)) {
            C_Matrix[i,j] <- (-1)^(i+j)*det(A[-i,-j])
        }
    }

    #Inverse of A is equal to transpose of Co-Factor Matrix divided by the determinant of A
    B <- t(C_Matrix)/det(A)

    return(B)

}
```

Let's run some test scenarios

```
#Test matrix was taken from https://www.mathsisfun.com/algebra/matrix-inverse.html
test_matrix <- matrix(c(3, 3.5, 3.2, 3.6), byrow = T, nrow = 2)
test_matrix
```

```
##      [,1] [,2]
## [1,]  3.0  3.5
## [2,]  3.2  3.6
```

```
test_matrix2 <- matrix(c(6,1,1,4,-2,5,2,8,7), byrow = T, nrow = 3)
test_matrix2
```

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

```
#Use the built-in solve function (also the website verifies the answer)
solve(test_matrix)
```

```
##      [,1]  [,2]
## [1,]   -9  8.75
## [2,]    8 -7.50
```

```r
solve(test_matrix2)
```

```
##             [,1]         [,2]        [,3]
## [1,]  0.17647059 -0.003267974 -0.02287582
## [2,]  0.05882353 -0.130718954  0.08496732
## [3,] -0.11764706  0.150326797  0.05228758
```

```r
#Test function
myinverse(test_matrix)
```

```
##      [,1]  [,2]
## [1,]   -9  8.75
## [2,]    8 -7.50
```

```r
myinverse(test_matrix2)
```

```
##             [,1]         [,2]        [,3]
## [1,]  0.17647059 -0.003267974 -0.02287582
## [2,]  0.05882353 -0.130718954  0.08496732
## [3,] -0.11764706  0.150326797  0.05228758
```