SElshahawy_Assign4

Salma Elshahawy

2/5/2020

Problem_set1

In this problem, we'll verify using R that SVD and Eigenvalues are related as worked out in the weekly module. Given a 3 x 2 matrix A write code in R to 1. compute $X = AA^T$ and $Y = A^TA$. Then, 2. compute the eigenvalues and eigenvectors of X and Y using the built-in commands in R.

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

Then, 3. compute the left-singular, singular values, and right-singular vectors of A using the svd command. 4. Examine the two sets of singular vectors and show that they are indeed eigenvectors of X and Y. 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.

Your code should compute all these vectors and scalars and store them in variables. Please add enough comments in your code to show me how to interpret your steps.

Answer:-

Compute X and Y using built-in functions

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

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

```
#  getting X value using t() function
X <- A %*% t(A)
X
```

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

```
# doing the same for y
Y <- t(A) %*% A
Y
```

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

Getting eign values and eign vectors

```
eign_X <- eigen(X)
eign_X
```

```
## eigen() decomposition
## $values
## [1] 26.601802  4.398198
##
## $vectors
##           [,1]       [,2]
## [1,] 0.6576043 -0.7533635
## [2,] 0.7533635  0.6576043
```

```
eign_Y <- eigen(Y)
eign_Y
```

```
## eigen() decomposition
## $values
## [1] 2.660180e+01 4.398198e+00 1.058982e-16
##
## $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 left, right singular using svd() and compare

```
svd_A <- svd(A)
svd_A
```

```
## $d
## [1] 5.157693 2.097188
##
## $u
##            [,1]       [,2]
## [1,] -0.6576043 -0.7533635
## [2,] -0.7533635  0.6576043
##
## $v
##             [,1]       [,2]
## [1,]  0.01856629 -0.6727903
## [2,] -0.25499937 -0.7184510
## [3,] -0.96676296  0.1765824
```

```
#  merge A$u matrix with the eign vector of x
compare_X <- cbind(svd_A$u, eign_X$vectors)
# name the columns
```

```r
colnames(compare_X) <- c('SVDu1', 'SVDu2', 'EVX1=u1', 'EVX2=u2')

(compare_X)
```

```
##           SVDu1      SVDu2   EVX1=u1    EVX2=u2
## [1,] -0.6576043 -0.7533635 0.6576043 -0.7533635
## [2,] -0.7533635  0.6576043 0.7533635  0.6576043
```

```r
compare_Y <- cbind(svd_A$v, eign_Y$vectors)
compare_Y <- compare_Y[,1:4]
colnames(compare_Y) <- c('SVDv1', 'SVDv2', 'EVY1=v1', 'EVY2=v2')
(compare_Y)
```

```
##            SVDv1      SVDv2     EVY1=v1    EVY2=v2
## [1,]  0.01856629 -0.6727903 -0.01856629 -0.6727903
## [2,] -0.25499937 -0.7184510  0.25499937 -0.7184510
## [3,] -0.96676296  0.1765824  0.96676296  0.1765824
```

Problem_set2

Using the procedure outlined in section 1 of the weekly handout, write a function to compute the inverse of a well-conditioned full-rank square matrix using co-factors. In order to compute the co-factors, you may use built-in commands to compute the determinant.

Your function should have the following signature: $B = myinverse(A)$ where A is a matrix and B is its inverse and $A \times B = I$. The off-diagonal elements of I should be close to zero, if not zero. Likewise, the diagonal elements should be close to 1, if not 1. Small numerical precision errors are acceptable but the function myinverse should be correct and must use co-factors and determinant of A to compute the inverse.

Answer:-

```r
a <- matrix(c(2,4,-2,
              1,-1,5,
              3,3,5), nrow = 3, byrow = TRUE)
a
```

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

```r
getCofactors <- function(M) {
  stopifnot(length(unique(dim(M)))==1) # Check if Matrix = Square
  cf <- M # creating a Matrix that has the same Dimensions as M
  for(i in 1:dim(M)[1]){
    for(j in 1:dim(M)[2]){
      cf[i,j] <- (det(M[-i,-j])*(-1)^(i+j)) # overwriting the Values of cf Matrix with cofactors
    }
  }
  return(cf) # output of cofactors matrix
}
```

```r
a_cofactor <- getCofactors(a)
a_cofactor
```

```
##      [,1] [,2] [,3]
## [1,]  -20   10    6
## [2,]  -26   16    6
## [3,]   18  -12   -6
```

```r
myinverse <- function(A){
    det_A = det(A) # get the determinant of matrix  A
    inverse <- (1/det_A) * t(getCofactors(A)) # inverse should be determinant inverse multiplied by the
    return(inverse)
}

B <- myinverse(a)
B
```

```
##            [,1]      [,2] [,3]
## [1,]  1.6666667  2.166667 -1.5
## [2,] -0.8333333 -1.333333  1.0
## [3,] -0.5000000 -0.500000  0.5
```