BLin_Assign4

Bin Lin

2017-3-4

1. Problem Set 1 In this problem, we'll verify using R that SVD and Eigenvalues are related as worked out in the weekly module. Given a 3 * 2 matrix A. Write code in R to compute X = AAT and Y = ATA. Then, compute the eigenvalues and eigenvectors of X and Y using the built-in commans in R. Then, compute the left-singular, singular values, and right-singular vectors of A using the svd command. 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.

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

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

```
# Calculate the X and its eigentvalues and eigenvectors.
X <- A%*%t(A)
X
```

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

```
eigen(X)$values
```

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

```
eigen(X)$vectors
```

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

```
# Calculate the Y and its eigentvalues and eigenvectors.
Y <- t(A)%*%A
Y
```

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

```
eigen(Y)$values
```

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

```
eigen(Y)$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-singular, singular values and right-singular vectors of A
svd(A)$d
```

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

```
svd(A)$u
```

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

```
svd(A)$v
```

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

As shown above, eigenvectors of X is identical to orthonormal matrix U, eigenvector of Y is identical to orthonomal matrix of V, except the sign on the first column. The sign does not change the orthogonality and normalization of the vector X and Y as proved in the following code.

```
# Prove orthogonal and normalization for X
round(t(svd(A)$u[1,]) %*% svd(A)$u[2,])
```

```
##      [,1]
## [1,]    0
```

```
round(t(eigen(X)$vectors[1,]) %*% eigen(X)$vectors[2,])
```

```
##      [,1]
## [1,]    0
```

```
xlength <- sqrt(t(eigen(X)$vectors) %*% eigen(X)$vectors)
xlength
```

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

```
# Prove orthogonal and normalization for Y
round(t(svd(A)$v[1,]) %*% svd(A)$v[2,])
```

```
##      [,1]
## [1,]    0
```

```
round(t(eigen(Y)$vectors[1,]) %*% eigen(Y)$vectors[2,])
```

```
##      [,1]
## [1,]    0
```

```
ylength <- sqrt(t(eigen(Y)$vectors) %*% eigen(Y)$vectors)
```

```
## Warning in sqrt(t(eigen(Y)$vectors) %*% eigen(Y)$vectors): NaNs produced
```

```
round(ylength)
```

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

The following code shows the eigenvalues of X and Y is identical to the square of non-zero eisingular values of A.

```
svd(A)$d
```

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

```
eigen(X)$values
```

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

```
eigen(Y)$values[1:2]
```

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

```
all.equal((svd(A)$d) ^ 2, eigen(X)$values, eigen(Y)$values[1:2])
```

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

2. Problem Set 2 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*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.

```
myinverse <- function (A)
{
  if (nrow(A) != ncol(A))
  {
    print ("Matrix is not a square matrix. It has no inverse")
  }
  else
  {
    if (det(A) == 0)
      {
      print ("Determinant of matrix can not be zero")
    }
    else
    {
      C <- matrix(nrow = nrow(A), ncol = ncol(A))
      for (i in 1:nrow(A))
        {
        for (j in 1:ncol(A))
          {
          sub_matrix <- A[-i,-j]
          C[i,j] <- ((-1) ^ (i + j)) * det(sub_matrix)
        }
        }
      inverse_A <- t(C)/det(A)
      return (inverse_A)
    }
    }
}

set.seed(888)
A <- matrix(rnorm(100), 4, 4, byrow = TRUE)
det(A)
```

```
## [1] 2.709542
```

```
B <- myinverse(A)
B
```

```
##            [,1]        [,2]       [,3]       [,4]
## [1,] -1.6264098 -0.37842787 -1.6970709 -1.8096009
## [2,]  1.9404792  0.37995528  2.6541812  2.8678110
## [3,]  0.7481699 -0.23734260  0.7694800  0.6082485
## [4,] -0.9982202 -0.07769876 -0.8136986 -1.6351143
```