HMW 4- Data 605

OMER OZEREN

Table of Contents

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

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

write code in R to compute $X = AA^T$ and $Y = A^TA$. Then, compute the eigenvalues and eigenvectors of X and Y using the built-in commands 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.

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

Write code in R to compute X = AAT and Y = ATA.

```
# Initial matrix A
A <- matrix(c(1,-1,2,0,3,4), nrow=2)
# Calculate X and Y
X <- A %*% t(A)
Y <- t(A) %*% A
```

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

```
eigenX <- eigen(X)
eigenY <- eigen(Y)
```

Eigenvalues of X and Y:

```
round(eigenX$values, digits=10)

## [1] 26.601802  4.398198

round(eigenY$values, digits=10)

## [1] 26.601802  4.398198  0.000000
```

Eigenvectors of X and Y

```
eigenX$vectors

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

eigenY$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 left-singular, singular values, and right-singular vectors of A using the svd command

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

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

svd_A$d

## [1] 5.157693 2.097188

svd_A$v
```

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

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

```r
# First, let us compare eigenvectors of $X$ and singular vectors $U$.
eigenX$vectors
```

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

```r
svd_A$u
```

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

I see that first vectors only differ by their sign. I will invert the sign of the first eigenvector (multiply it by -1).

```r
eigenX$vectors[,1] <- eigenX$vectors[,1] * (-1)
round(eigenX$vectors[,1:2], digits=10) == round(svd_A$u, digits=10)
```

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

```r
eigenY$vectors[,1] <- eigenY$vectors[,1] * (-1)
round(eigenY$vectors[,1:2], digits=10) == round(svd_A$v, digits=10)
```

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

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 $A \times B = I$. 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

Function

```r
myinverse <- function(A) {
  # Check that A is square
  n <- dim(A)[1]
  if (n!=dim(A)[2]) {
    return(NA)
  }

  # Check that A is inversible
  # Determinant should not be zero
  if (det(A)==0) {
    return(NA)
  }

  # Loop through all elements and compute co-factor matrix C
  C <- matrix(0, n, n)
  for(i in 1:n) {
    for(j in 1:n) {
      # The submatrix A[-i,-j] is forced into matrix type in case of 2x2
matrix
      # Otherwise, if 2x2 matrix, then A[-i,-j] would produce a single
element
      C[i,j] <- (-1)^(i+j)*det(matrix(A[-i,-j], n-1))
    }
  }

  # Inverse of A is equal to transpose of C divided by determinant of A
  B <- t(C)/det(A)

  return(B)
}
```

Example

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

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

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

# Calculate inverse using 'myinverse' function
B <- myinverse(A)
B

##             [,1]       [,2]       [,3]
## [1,] -0.02857143  0.2857143  0.1142857
## [2,] -0.05714286 -0.4285714  0.2285714
## [3,]  0.28571429  0.1428571 -0.1428571

# Double-check using 'solve' function
solve(A)

##             [,1]       [,2]       [,3]
## [1,] -0.02857143  0.2857143  0.1142857
## [2,] -0.05714286 -0.4285714  0.2285714
## [3,]  0.28571429  0.1428571 -0.1428571

round(solve(A), digits=10) == round(B, digits=10)

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

# Check that AB=I
round(A %*% B, digits=10)

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