Matrix Algebra

Blake Pappas

December 17, 2023

Motor Trend Car Road Tests Data

```
data(mtcars)
vars <- which(names(mtcars) %in% c("mpg", "disp", "hp", "drat", "wt"))
cars <- mtcars[, vars]
```

Mean Vector and Covariance Matrix

```
(mean <- apply(cars, 2, mean)) #2 designates that the mean calculation will be by the column. A 1 would
```

```
##        mpg       disp         hp       drat         wt
##  20.090625 230.721875 146.687500   3.596563   3.217250
```

```
n <- dim(cars)[1]; p <- dim(cars)[2] # Pulls the size of the sample (n) and the number of variables (p)
X <- as.matrix(cars) # Converts the dataset cars to a matrix
ones <- rep(1, n) # Creates a vector of 1's with a length of n
(meanCal <- (1 / n) * t(X) %*% ones) # Calculates the mean using the matrix math formula from the lectu
```

```
##            [,1]
## mpg   20.090625
## disp 230.721875
## hp   146.687500
## drat   3.596563
## wt     3.217250
```

```
(S <- cov(cars)) # Calculates the covariance
```

```
##             mpg        disp         hp        drat          wt
## mpg    36.324103   -633.09721 -320.73206   2.1950635  -5.1166847
## disp -633.097208 15360.79983 6721.15867 -47.0640192 107.6842040
## hp   -320.732056  6721.15867 4700.86694 -16.4511089  44.1926613
## drat    2.195064    -47.06402  -16.45111   0.2858814  -0.3727207
## wt     -5.116685    107.68420   44.19266  -0.3727207   0.9573790
```

```r
(Scal <- (1 / (n - 1)) * t(X) %*% (diag(n) - (1 / n) * ones %*% t(ones)) %*% X) #Also calculates the co
```

```
##              mpg        disp         hp        drat          wt
## mpg    36.324103  -633.09721 -320.73206   2.1950635  -5.1166847
## disp -633.097208 15360.79983 6721.15867 -47.0640192 107.6842040
## hp   -320.732056  6721.15867 4700.86694 -16.4511089  44.1926613
## drat    2.195064   -47.06402  -16.45111   0.2858814  -0.3727207
## wt     -5.116685   107.68420   44.19266  -0.3727207   0.9573790
```

Inverse Matrix

```r
S_inv <- solve(S)
(S_inv %*% S)
```

```
##                mpg         disp            hp          drat            wt
## mpg   1.000000e+00 2.842171e-14  7.105427e-15 -1.110223e-16  2.220446e-16
## disp  2.775558e-17 1.000000e+00 -6.661338e-16 -1.734723e-18 -1.387779e-17
## hp   -2.775558e-17 4.440892e-16  1.000000e+00  0.000000e+00  0.000000e+00
## drat -7.549517e-15 1.350031e-13 -1.421085e-14  1.000000e+00  1.165734e-15
## wt    0.000000e+00 1.136868e-13  0.000000e+00  0.000000e+00  1.000000e+00
```

```r
# Values Are Not Exactly 1 or 0 In Actuality. However, They Are Approximately Equal to 1 or 0
```

Orthogonal Matrix Example

```r
Q <- matrix(c(2, 1, 2, -2, 2, 1, 1, 2, -2), ncol = 3) / 3

# Check
(Q %*% t(Q))
```

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

Eigenvalues and Eigenvectors

```r
eigen <- eigen(S)

(S %*% eigen$vectors[, 1] / eigen$vectors[, 1])
```

```
##          [,1]
## mpg  18636.79
## disp 18636.79
## hp   18636.79
## drat 18636.79
## wt   18636.79
```

```r
eigen$values[1]
```

```
## [1] 18636.79
```

```r
t(eigen$vectors[, 1]) %*% eigen$vectors[, 1]
```

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

Spectral Decomposition

```r
temp <- array(dim = c(5, 5, 5)) # Creates a placeholder for a 5x5 matrix

for (i in 1:5) {
  temp[i, , ] <- eigen$values[i] * eigen$vectors[, i] %*% t(eigen$vectors[, i])
}

# Check the Spectral Decomposition
(out <- apply(temp, 2:3, sum))
```

```
##             [,1]        [,2]       [,3]        [,4]        [,5]
## [1,]   36.324103  -633.09721 -320.73206   2.1950635  -5.1166847
## [2,] -633.097208 15360.79983 6721.15867 -47.0640192 107.6842040
## [3,] -320.732056  6721.15867 4700.86694 -16.4511089  44.1926613
## [4,]    2.195064   -47.06402  -16.45111   0.2858814  -0.3727207
## [5,]   -5.116685   107.68420   44.19266  -0.3727207   0.9573790
```

```r
S
```

```
##              mpg        disp         hp        drat          wt
## mpg    36.324103  -633.09721 -320.73206   2.1950635  -5.1166847
## disp -633.097208 15360.79983 6721.15867 -47.0640192 107.6842040
## hp   -320.732056  6721.15867 4700.86694 -16.4511089  44.1926613
## drat    2.195064   -47.06402  -16.45111   0.2858814  -0.3727207
## wt     -5.116685   107.68420   44.19266  -0.3727207   0.9573790
```

Determinant and Trace

```r
# Trace: Equal to the Sum of the Diagonal Elements
(trace <- sum(diag(S)))
```

```
## [1] 20099.23
```

```r
sum(eigen$values)
```

```
## [1] 20099.23
```

```r
# Determinant: Equal to the Product of the Eigenvalues
det(S)
```

```
## [1] 3951786
```

```r
prod(eigen$values)
```

```
## [1] 3951786
```

Square-Root Matrices

```r
temp1 <- array(dim = c(5, 5, 5)) # Creates a placeholder for a 5x5 matrix

for (i in 1:5) {
  temp1[i, , ] <- (1 / eigen$values[i]) * eigen$vectors[, i] %*% t(eigen$vectors[, i])
}

# Check the Spectral Decomposition
(out1 <- apply(temp1, 2:3, sum))
```

```
##               [,1]          [,2]          [,3]        [,4]        [,5]
## [1,]  0.1695494031 -0.0006468718  0.0058975274 -0.29977161  0.58997555
## [2,] -0.0006468718  0.0005369064 -0.0003801427  0.02257595 -0.03751089
## [3,]  0.0058975274 -0.0003801427  0.0008208474 -0.02678451  0.02595898
## [4,] -0.2997716134  0.0225759526 -0.0267845083  8.50376340  0.40558365
## [5,]  0.5899755523 -0.0375108878  0.0259589804  0.40558365  7.37641228
```

```r
S_inv
```

```
##                mpg          disp            hp        drat          wt
## mpg   0.1695494031 -0.0006468718  0.0058975274 -0.29977161  0.58997555
## disp -0.0006468718  0.0005369064 -0.0003801427  0.02257595 -0.03751089
## hp    0.0058975274 -0.0003801427  0.0008208474 -0.02678451  0.02595898
## drat -0.2997716134  0.0225759526 -0.0267845083  8.50376340  0.40558365
## wt    0.5899755523 -0.0375108878  0.0259589804  0.40558365  7.37641228
```

```r
temp2 <- array(dim = c(5, 5, 5))

for (i in 1:5) {
  temp2[i, , ] <- sqrt(eigen$values[i]) * eigen$vectors[, i] %*% t(eigen$vectors[, i])
}

out2 <- apply(temp2, 2:3, sum)

(out2 %*% out2)
```

```
##             [,1]        [,2]       [,3]        [,4]        [,5]
## [1,]   36.324103   -633.09721 -320.73206   2.1950635  -5.1166847
## [2,] -633.097208 15360.79983 6721.15867 -47.0640192 107.6842040
```

```
## [3,] -320.732056  6721.15867 4700.86694 -16.4511089  44.1926613
## [4,]    2.195064   -47.06402  -16.45111   0.2858814  -0.3727207
## [5,]   -5.116685   107.68420   44.19266  -0.3727207   0.9573790
```

```r
S
```

```
##              mpg        disp         hp        drat          wt
## mpg    36.324103  -633.09721 -320.73206   2.1950635  -5.1166847
## disp -633.097208 15360.79983 6721.15867 -47.0640192 107.6842040
## hp   -320.732056  6721.15867 4700.86694 -16.4511089  44.1926613
## drat    2.195064   -47.06402  -16.45111   0.2858814  -0.3727207
## wt     -5.116685   107.68420   44.19266  -0.3727207   0.9573790
```

Partitioning Random Vectors

Let's partitioning the variables into two groups:

1. *disp*, *hp*, *wt*
2. *mpg*, *drat*

```r
vars1 <- which(names(mtcars) %in% c("disp", "hp", "wt"))
vars2 <- which(names(mtcars) %in% c("mpg", "drat"))

carPar <- mtcars[, c(vars1, vars2)]

(Sigma11 <- cov(carPar[1:3, 1:3]))
```

```
##          disp        hp       wt
## disp 901.3333 294.66667 7.410000
## hp   294.6667  96.33333 2.422500
## wt     7.4100   2.42250 0.077175
```

```r
(Sigma22 <- cov(carPar[4:5, 4:5]))
```

```
##          mpg     drat
## mpg   3.6450 -0.09450
## drat -0.0945  0.00245
```

```r
(Sigma12 <- cov(carPar)[1:3, 4:5])
```

```
##              mpg        drat
## disp -633.097208 -47.0640192
## hp   -320.732056 -16.4511089
## wt     -5.116685  -0.3727207
```