STATS230-Hw2

Mingyu Du

Problem 1

$$\begin{aligned} A &= LL^T \\ a_{11} &= l_{11}^2 = 2 \\ a_{12} &= l_{11}l_{21} = -2 \\ a_{22} &= l_{21}^2 + l_{22}^2 = 5 \end{aligned}$$

We can see that $l_{11} = \sqrt{2}$, $l_{21} = -\sqrt{2}$, and $l_{22} = \sqrt{3}$. Thus,

$$L = \begin{pmatrix} \sqrt{2} & 0 \\ -\sqrt{2} & \sqrt{3} \end{pmatrix}$$

Problem 2

Assume A is a positive definite matrix with band condition in the sense that $a_{ij} = 0$ when $|i-j| > d$, and B is its Cholesky decomposition matrix such that $A = BB^T$ and

$$B = \begin{pmatrix} b_{11} & 0^T \\ \bar{b} & B_{22} \end{pmatrix},$$

we can obtain $b_{11} = \sqrt{a_{11}}$ and

$$\bar{b} = b_{11}^{-1} \left(a_{21}, a_{31}, \ldots, a_{d+1,1}, 0, \ldots, 0\right)^T,$$

in which $\bar{b}_i = 0$ when $i > d$. Thus, we have $b_{i1} = 0$ when $|i-1| > d$.

Furthermore, B_{22} is the Cholesky decomposition of $A_{22} - \bar{b}\bar{b}^T$.

Denote the first column of A_{22} as $\bar{c}$ and denote the first column of $\bar{b}\bar{b}^T$ as $\bar{d}$.
$\bar{c} = \left(a_{22}, \ldots, a_{d+2,2}, 0, \ldots, 0\right)^T$, so $\bar{c}_i = 0$ when $i > d+1$. And for the matrix $\bar{b}\bar{b}^T$, we can find that $\bar{d}_i = 0$ when $i > d$. Denote B_{22} as

$$B_{22} = \begin{pmatrix} b_{22} & 0^T \\ \bar{e} & B_{33} \end{pmatrix},$$

After solving the equations, we get $b_{22} = \sqrt{a_{22} - \frac{a_{21}^2}{a_{11}}}$ and

$$\bar{e} = b_{22}^{-1} \left(\bar{c}_2 - \bar{d}_2, \bar{c}_3 - \bar{d}_3, \ldots, \bar{c}_{d+1} - \bar{d}_{d+1}, 0, \ldots, 0\right)^T$$

in which $\bar{e}_i = 0$ when $i > d$. Thus, we get $b_{i2} = 0$ when $|i-2| > d$.

We have already made the proof for $j = 1, 2$. Similarly, this condition also applies for other values of j, so $b_{ij} = 0$ when $|i-j| > d$.

Problem 3

$$\begin{aligned} X^T X &= (QR)^T QR \\ &= R^T Q^T QR \\ &= R^T R \end{aligned}$$

Thus, we get

$$\begin{aligned} X\left(X^T X\right)^{-1} X^T &= QR\left(R^T R\right)^{-1} R^T Q^T \\ &= QRR^{-1}\left(R^T\right)^{-1} R^T Q^T \\ &= QQ^T \end{aligned}$$

$$\begin{aligned} |det\,(X)\,| &= |det\,(QR)\,| \\ &= |det\,(Q)\,||det\,(R)\,| \\ &= |det\,(R)\,| \end{aligned}$$

in which $|det\,(Q)\,| = 1$ since Q is a matrix with orthonormal columns.

$$\begin{aligned} det\left(X^T X\right) &= det\left(X^T\right) det\,(X) \\ &= det\,(X)\,det\,(X) \\ &= [det(R)]^2 \end{aligned}$$

Problem 4

$$\begin{aligned} AA^T &= \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ \sin(\theta) & -\cos(\theta) \end{pmatrix} \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ \sin(\theta) & -\cos(\theta) \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{aligned}$$

Thus, A is orthogonal.

Solving $|det\,(A - \lambda I)\,| = 0$, we can get $\lambda_1 = 1$ and $\lambda_2 = -1$.

For $\lambda_1 = 1$, solving $(A - \lambda_1 I)v_1 = 0$, we can get $v_1 = \left(1, \frac{sin\theta}{cos\theta+1}\right)^T$.

For $\lambda_2 = -1$, solving $(A - \lambda_2 I)v_2 = 0$, we can get $v_2 = \left(1, \frac{sin\theta}{cos\theta-1}\right)^T$.

Problem 5

Since $Ov = \lambda v$, we have

$$\begin{aligned} |\lambda|^2 v^T v &= (Ov)^T (Ov) \\ &= v^T O^T Ov \\ &= v^T v \end{aligned}$$

Thus, $\lambda = \pm 1$

Problem 6

Based on singular value decomposition, $A^{-1} = U\Sigma^{-1}V^T$, where

$$\Sigma^{-1} = \begin{pmatrix} \frac{1}{\sigma_1} & & & \\ & \frac{1}{\sigma_2} & & \\ & & \ddots & \\ & & & \frac{1}{\sigma_m} \end{pmatrix}$$

Thus,

$$\begin{aligned} cond_2(A) &= \|A\|_2 \left\|A^{-1}\right\|_2 \\ &= max_i \sigma_i \times max_i(\frac{1}{\sigma_i}) \\ &= \frac{\max_i \sigma_i}{\min_i \sigma_i} \end{aligned}$$

Problem 7

Simulation from n-dimensional multivariate normal distribution via Cholesky decomposition can be summarized as follows:

- Compute the Cholesky decomposition matrix L such that $\Sigma = LL^T$.
- Generate $\bar{z} \sim N_n(0, I)$.
- Compute $\bar{x} = \bar{\mu} + L\bar{z}$.

I create a function `mvn_chol_sim` for the simulation.

```r
devtools::load_all("./package/")
library(bench)
set.seed(144535)
```

```r
n <- 4
# specify the mean vector and covariance matrix (positive definite)
mu <- runif(n)
A <- matrix(runif(n^2)*2-1, ncol = n)
sigma <- t(A) %*% A
```

```r
N <- 100
# simulation
sim_x <- mvn_chol_sim(N, mu, sigma)
# validation
sample_mean <- apply(sim_x, 1, mean)
sample_cov <- cov(t(sim_x))
mu
```

```
## [1] 0.1162259 0.1428832 0.9152608 0.9933460
```

```r
sample_mean
```

```
## [1]  0.17703580 -0.01884946  0.94982391  0.98804302
```

```r
sigma
```

```
##            [,1]       [,2]       [,3]       [,4]
## [1,]  0.8781829 -0.1182697 -0.8736224 -0.7256127
## [2,] -0.1182697  0.5274449 -0.0198087  0.3459312
## [3,] -0.8736224 -0.0198087  1.8731876  0.8248404
## [4,] -0.7256127  0.3459312  0.8248404  1.1350984
```

```r
sample_cov
```

```
##             [,1]        [,2]        [,3]        [,4]
## [1,]  1.90443915 -0.07852884 -0.88373831 -0.33856251
## [2,] -0.07852884  0.67294089 -0.25301518  0.18547312
## [3,] -0.88373831 -0.25301518  1.12572936 -0.06367799
## [4,] -0.33856251  0.18547312 -0.06367799  0.42608510
```

I also do the simulation when the number of realization $N = 100000$:

```r
N <- 100000
# simulation
sim_x <- mvn_chol_sim(N, mu, sigma)
# validation
sample_mean <- apply(sim_x, 1, mean)
sample_cov <- cov(t(sim_x))
mu
```

```
## [1] 0.1162259 0.1428832 0.9152608 0.9933460
```

```r
sample_mean
```

```
## [1] 0.1183631 0.1426343 0.9124371 0.9936996
```

```r
sigma
```

```
##            [,1]       [,2]       [,3]       [,4]
## [1,]  0.8781829 -0.1182697 -0.8736224 -0.7256127
## [2,] -0.1182697  0.5274449 -0.0198087  0.3459312
## [3,] -0.8736224 -0.0198087  1.8731876  0.8248404
## [4,] -0.7256127  0.3459312  0.8248404  1.1350984
```

```r
sample_cov
```

```
##            [,1]       [,2]       [,3]       [,4]
## [1,]  2.3645333 -0.1818670 -1.0499407 -0.4814929
## [2,] -0.1818670  0.6699434 -0.1306740  0.2177915
## [3,] -1.0499407 -0.1306740  0.9973594  0.1047758
## [4,] -0.4814929  0.2177915  0.1047758  0.3876923
```

The difference between ground truth and simulated statistics is much lower.

Problem 8

```
data <- read.csv("homework2_regression.csv")
y <- data[,1]
x <- data[,-1]
```

The method to obtain OLS estimates of coefficients using QR decomposition can be summarized as follows:

$$
\begin{aligned}
\hat{\beta} &= \left(X^T X\right)^{-1} X^T y \\
&= \left((QR)^T QR\right)^{-1} (QR)^T y \\
&= \left(R^T Q^T QR\right)^{-1} R^T Q^T y \\
&= \left(R^T R\right)^{-1} R^T Q^T y \\
&= R^{-1} Q^T y
\end{aligned}
$$

The method to obtain OLS estimates of coefficients using SVD can be summarized as follows: Assume $X = U\Sigma V^T$, we can get $\hat{\beta} = V\Sigma^{-1}U^T y$.

I create two functions: `ols_qr` and `ols_svd` for QR decomposition and SVD respectively:

```
ols_qr(x,y)
```

```
##           [,1]
## x1 -0.05006337
## x2 -2.05873307
## x3 -0.88978361
## x4  0.86985616
## x5  3.10461263
```

```
ols_svd(x,y)
```

```
##             [,1]
## [1,] -0.05006337
## [2,] -2.05873307
## [3,] -0.88978361
## [4,]  0.86985616
## [5,]  3.10461263
```

```
bench::mark(
  ols_qr(x,y),
  ols_svd(x,y), check = FALSE
)
```

```
## # A tibble: 2 x 6
##   expression         min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>    <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 ols_qr(x, y)     176us    222us     3988.   118.8KB     8.54
## 2 ols_svd(x, y)    128us    139us     6003.    74.3KB    12.9
```

Based on the benchmark result, SVD is more computationally efficient than QR decomposition.