STAT243 Problem Set 7

Name: Chih Hui Wang SID: 26955255

November 2, 2015

1.

- **What are the goals of their simulation study and what are the metrics that they consider in assessing their method?**

The goals of the simulation study are to evaluate the accuracy of their proposed asymptotic approximation in finite samples and to examine the power of their EM test.

- **What choices did the authors have to make in designing their simulation study? What are the key aspects of the data generating mechanism that likely affect the statistical power of the test?**

When they design their simulation study, they have to decide θ and σ for the mixture normal models. Also, they calculated the test statistic based on their recommendation for β and K as well as the two penalty functions $p(\beta)$ and $p(\sigma^2, \hat{\sigma}^2)$. If the choice of θ and σ was inappropriate, it would affect power of the test. For example, if we pick θ_1 and θ_2 which were too closed to each other, we cannot distinguish the two groups, which indicated the existence of dominance group.

- **Suggest some alternatives to how the authors designed their study. Are there data-generating scenarios that they did not consider that would be useful to consider?**

They can consider the scenarios that the null order is 1 and compare it with alternative order bigger than one.

- **Give some thoughts on how to set up a simulation study for their problem that uses principles of basic experimental design (see the Unit 10 notes) or if you think it would be difficult, say why.**

The part for generating random number from mixture normal distribution will not be too difficult. For two mixture normal, we can do by generating random number from two normal distributions. After that, we create another random number with value 0s and 1s to decide the ith observation is from which normal distribution. The proportion of 0s in the random number is α_1 while the proportion of 1s is α_2. The difficult part of the simulation study is to choose the appropriate sample size, replication number, parameters of normal distribution.

- **Do their figures/tables do a good job of presenting the simulation results and do you have any alternative suggestions for how to do this? Do the authors address the issue of simulation uncertainty/simulation standard errors and/or do they convince the reader they've done enough simulation replications?**

I do not think that their figures/tables give a very clear summary about their simulation study. They have set up 12 null models with order 2, used them to calculate the Type I error based on 5000 replications and summarized by boxplot. However, I wondered why there is only 1 boxplot for each scenario (different sample sizes and significant levels). Are there supposed to have boxplot to summarize each null model? The same things happened when computing powers. They give table to summarize the power under different scenarios (different sample size, numbers of iteration,

combinations of alternative θ and σ, weights) while I think that powers would be different among different null model, so perhaps there should be several tables to present the result. For the uncertainty, it seems that they did not reveal any information about it. For the choice of replications such as 5000 replications for calculating Type I error and 1000 replications for calculating power, they did not explain the reason why they chose the number too.

- **Interpret their tables on power (Tables 4 and 6) - do the results make sense in terms of how the power varies as a function of the data generating mechanism?**

The results make sense. The power increases as the sample size increases. Also, when the alternative models become far away from one another, the power increases. Finally, the number of iteration increases, the power increases while not dramatically.

- **Discuss the extent to which they follow JASA's guidelines on simulation studies (see the end of the Unit 10 class notes for the JASA guidelines).**

Overall, they follow the JASA's guidelines such as algorithm, programming language and major software components that were used. However, the part for estimated accuracy of results and descriptions of pseudorandom-number generators are not so clear when I browse through the paper. For example, it did not show that how they generate the data and some uncertain measures such as standard error.

2.

(a) Below are the steps for the Cholesky decomposition:

1. $U_{11} = \sqrt{A_{11}}$
2. For $j = 2, \ldots, n$, $U_{1j} = A_{1j}/U_{11}$ $\{n-1\}$
3. For $i = 2, \ldots, n$,

- $U_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1} U_{ki}^2}$ $\{(1 + 2 + \cdots + (n-1) = \frac{n(n-1)}{2}\}$
- for $j = i+1, \ldots, n$: $U_{ij} = (A_{ij} - \sum_{k=1}^{i-1} U_{ki}U_{kj})/U_{ii}$ $\{\sum_{i=2}^{n}(n-i)i = \sum_{i=2}^{n} ni - i^2 = \frac{n^3 - 7n + 6}{6}\}$

Hence, the total steps is $(n-1) + \frac{n(n-1)}{2} + \frac{n^3 - 7n + 6}{6} = \frac{n^3 + 3n^2 - 4n}{6}$.

(b) No, we will not overwrite the value we need for calculation when we do the calculation and store the elements of U at the same time. As above steps indicated, we easily tell that the step 1 and 2 can be calculated and stored at the same time. For the step 3, when we calculate the term U_{ii}, we are using the elements above U_{ii} which will be calculated first. When we calculate the term U_{ij}, we are using U_{ii} and the elements above U_{ij} and U_{ii}. All of them will be calculated before we compute U_{ij}. Therefore, we can do the calculation and store the outcomes in the original matrix without using additional memory.

(c) From the results of **mem_used** and **gc**, we can see that there is 7.6 MB memory used temporarily when R do the Cholesky decomposition. It will be clean soon after R finished the computation. Therefore, when doing the Cholesky decomposition, R make a copy of the original matrix and do the Cholesky decomposition.

```
n <- 1000
X <- crossprod(matrix(rnorm(n^2), n))
gc(reset=TRUE)
```

```
##           used (Mb) gc trigger (Mb) max used (Mb)
## Ncells  287881 15.4     592000 31.7   287881 15.4
## Vcells 1489720 11.4    3987438 30.5  1489720 11.4

library(pryr)

## Warning:  package 'pryr' was built under R version 3.2.2
```

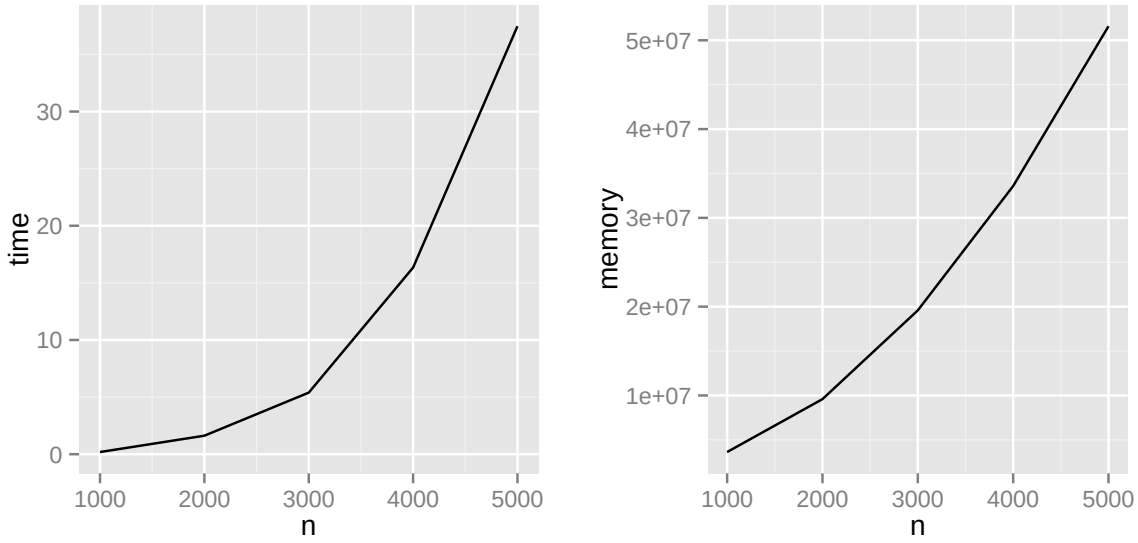
```
mem_used()

## 32.7 MB

gc()

##           used (Mb) gc trigger (Mb) max used (Mb)
## Ncells  361398 19.4     750400 40.1   592000 31.7
## Vcells 1570829 12.0    3987438 30.5  2015523 15.4

invisible(chol(X))
```

```
gc()

##           used (Mb) gc trigger (Mb) max used (Mb)
## Ncells  361498 19.4     750400 40.1   592000 31.7
## Vcells 1570972 12.0    3987438 30.5  2639266 20.2
```

The graph indicated that both processing time and memory use are cubic to n.

```
#Reset
gc(reset=TRUE)

##           used (Mb) gc trigger (Mb) max used (Mb)
## Ncells  367309 19.7     750400 40.1   367309 19.7
## Vcells 1583002 12.1    3987438 30.5  1583002 12.1

n <- seq(1000, 5000, by=1000)
record <- function(n){
  X <- crossprod(matrix(rnorm(n^2), n))
  time <- rep(0, 2)

  #Get the processing time(elapsed)
  time[1] <- system.time(U <- chol(X))[3]

  #Get the maximum memory for Vcells
  time[2] <- gc()[2, 5]

  time
}

result <- t(sapply(n, record))
result <- as.data.frame(cbind(n, result))

#Change the variable name
```

```
names(result) <- c("n", "time", "memory")

library(ggplot2); library(gridExtra)

## Loading required package:  methods
## Warning:  package 'gridExtra' was built under R version 3.2.2

#Plot
grid.arrange(
  ggplot(result, aes(x=n, y=time)) + geom_line(),
  ggplot(result, aes(x=n, y=memory)) + geom_line(),
  ncol=2
)
```



3.

(a) **solve** in R use the LU decomposition and then backsolve to get the value. The order of computations for full inversion is n^3. Therefore, in (a), the order should be $n^3 + O(n^2)$ where n^2 is the time for multiplication. In (b), the order of computations for LU decomposition is $\frac{n^3}{3} + O(n^2)$. In (c), we use Cholesky decomposition and backsolve which should take around $\frac{n^3}{6} + O(n^2)$ computation order.

```
set.seed(0)

#Set-up
n <- 5000
X <- crossprod(matrix(rnorm(n^2), n))
y <- as.matrix(rnorm(n))
```

```
#First approach
system.time(b1 <- solve(X) %*% y)

##    user  system elapsed
##  188.78    0.36  193.77
```

```r
#Second approach
system.time(b2 <- solve(X, y))

##    user  system elapsed
##   47.03    0.08   48.50
```

```r
#Third approach
approach3 <- function(X, y){
  U <- chol(X)
  b <- backsolve(U, backsolve(U, y, transpose=TRUE))
  b
}

system.time(b3 <- approach3(X, y))

##    user  system elapsed
##   36.88    0.09   38.13
```

(b) We can find that the results of these three methods are 7 digits in agree.

```r
#print out 22 number after decimal point
options(digits=22)

#Transpose the result for comparison
t(cbind(head(b1, n=3), head(b2, n=3), head(b3, n=3)))

##                       [,1]                [,2]                 [,3]
## [1,] -4.8038330372858233 17.758187424775905 -19.910537802573568
## [2,] -4.8038330372856155 17.758187424775944 -19.910537802573721
## [3,] -4.8038330455629641 17.758187419905809 -19.910537821884134
```

The conditional number of matrix X is around 10^7 which implies that we will have accuracy of order 10^9. It is the same as the results above indicated.

```r
#Get back to default
options(digits=7)

#Eigenvalues
v <- eigen(X)$values

#condition number
v[1]/v[length(v)]

## [1] 40290488
```

4.

My strategy is to try make the form back to $X^TX\beta = X^TY$. By doing so, we can apply the QR decomposition and backsolve to get the solution, which takes $2np^2 - \frac{2}{3}p^3$. Therefore, I first start with

decomposition for Σ^{-1}. I use Cholesky decomposition, $\Sigma = U^T U$, which takes $\frac{n^3}{6} + O(n^2)$.

$$\begin{aligned} X^T\Sigma^{-1}X^{-1}\beta &= X^T\Sigma^{-1}Y \\ \Rightarrow X^T(U^TU)^{-1}X\beta &= X^T(U^TU)^{-1}Y \\ \Rightarrow X^TU^{-1}(U^T)^{-1}X\beta &= X^TU^{-1}(U^T)^{-1}Y \end{aligned}$$

Let $(U^T)^{-1}X = X^*$ (takes n^3)and $(U^T)^{-1}Y = Y^*$(takes n^2), then the above equation can be rewritten into

$$(X^*)^T X^*\beta = (X^*)^T Y^*$$

whick is back to the form we familiar with. Now we can use QR decomposition for X^* and solve the β by

$$R^*\beta = (Q^*)^T Y$$

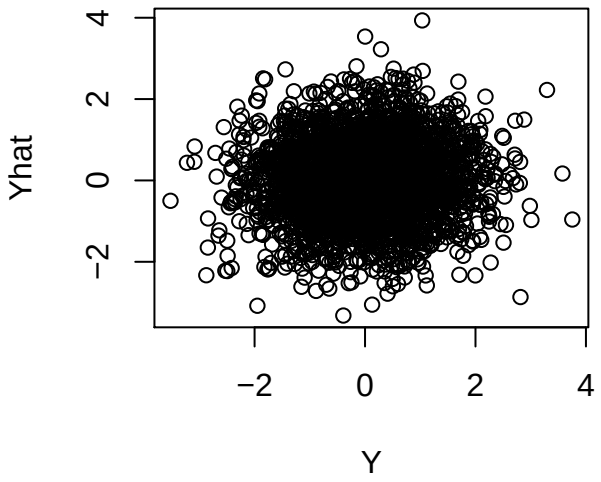
where $X^* = Q^*R^*$. The psesudo-cdoe and the order of computations for doing this are:

```
gls <- function(X, Sigma, y)
  U <- cholesky(Sigma)

  U_inv <- inverse(U)

  newX <- transpose(U_inv) %*% X
  newY <- transpose(U_inv) %*% Y

  Q <- QR(newX)
  R <- QR(newX)$R

  b <- backsolve(R, transpose(Q) %*% newY)
  b
}
```

The following is the R code and a example to run the function.

```
gls <- function(X, Sigma, y){
  #Cholesky decomposition of Sigma
  U <- chol(Sigma)

  #Compute the new X matrix and y vector
  UT_inv <- t(solve(U))
  newX <- UT_inv %*% X
  newy <- UT_inv %*% y

  #Do the QR decomposition
  qrX <- qr(newX)

  #Q and R matrix
  Q <- qr.Q(qrX); R <- qr.R(qrX)

  #Solve the equation
  b <- backsolve(R, t(Q) %*% newy)
  b
}
```

```r
#Setup
n <- 3000
p <- 100

X <- matrix(rnorm(n*p), ncol=p)
Sigma <- crossprod(matrix(runif(n^2), n))
Y <- rnorm(n)

system.time(beta <- gls(X, Sigma, Y))

##    user  system elapsed
##   22.49    0.13   22.94

#Predicted value
Yhat <- X %*% beta
plot(Y, Yhat)
```



5.

(a) 1° Right singular vectors of X are the eigenvectors of the matrix X^TX.
2° The eigenvalues of X^TX are the squares of the singular values of X.
By Singular value decomposition, we can rewrite X as UDV^T where U and V are matrices with orthonormal columns and D is diagonal matrix with non-negative diagonal elements. Let $X = UDV^T$ and plug it into X^TX

$$\begin{aligned} X^TX &= (UDV^T)^TUDV^T \\ &= VD^TU^TUDV^T \\ \text{(orthonormal)} &= VD^TDV^T \\ &= VD'V^T \\ \text{(Compare with eigenvalue decomposition)} &= \Gamma\Lambda\Gamma^T \end{aligned}$$

where D' is the diagonal matrix which its diagonal elements are the square of the diagonal elements in D. By compared the last two equation, we can get the conclusion that right singular vectors of X are the eigenvectors of the matrix X^TX and the eigenvalues of X^TX are the squares of the singular values of X.

3° X^TX is positive semi-definite.

By the definition, if a matrix M is positive semi-definite, then $a^TMa \geq 0$ for non-zero vector a.

$$\begin{aligned} a^TX^TXa &= (Xa)^TXa \\ (b = Xa) &= b^Tb \geq 0 \end{aligned}$$

Since $b = Xa$ is also a vector, the $(Xa)^TXa$ calculates its length, which cannot be negative. Therefore, X^TX is positive semi-definite.

(b) If λ_1 and v_1 are the eigenvalue and corresponding eigenvector of X, then $Xv_1 = \lambda_1 v_1$. Now we want to compute the eigenvalue of $Z = X + cI$. We have computed the eigendecomposition of X. Suppose λ_i and v_i are one pair of eigenvalue and eigenvector of X.

$$\begin{aligned} Zv_i &= Xv_i + cIv_i \\ &= \lambda_i v_i + cv_i \\ &= (\lambda_i + c)v_i \\ &= \lambda_i' v_i \end{aligned}$$

To compute the eigenvalues of Z, we only have to add the scalar c to each eigenvalue λ_i, which takes $O(n)$ additions.