STA410 Assignment #3

student number: 1003942326

Yulin WANG

14/11/2019

Question 1

(a)

Since we have,

$$\hat{tr(A)} = \frac{1}{m}\sum_{i=1}^{m} V_i^T A V_i$$

Then:

$$Var(\hat{tr(A)}) = \frac{1}{m^2} Var(V^T AV) = \frac{1}{m^2}\ \{E[(V^T AV)^2] - tr(A)^2\}$$

, where $\frac{1}{m^2}$ and $tr(A)^2$ are constant.
So it is sufficient to minimize $E[(V^T AV)^2]$.
Since we are assuming that $V_1, ..., V_n$ are independent with mean 0 and variance 1,
it follows that $E(V_i V_j V_k V_l) = 0$ or 1 unless $i = j = k = l$,
so we have:

$$\begin{aligned} E[(V^T AV)^2] &= \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} a_{ij}a_{kl}E(V_i V_j V_k V_l) \\ &= a_{11}^2 E(V_1^4) + a_{22}^2 E(V_2^4) + ... + a_{nn}^2 E(V_n^4) + \text{constant} \\ &= \sum_{i=1}^{n} a_{ii}^2 E(V_i^4) + \text{constant} \end{aligned}$$

So it is sufficient to minimize $E(V_i^4)$.
Since

$$Var(V_i) = E(V_i^2) - [E(V_i)]^2$$

so

$$E(V_i^2) = Var(V_i) + [E(V_i)]^2 = 1 + 0 = 1$$

Then

$$E(V_i^4) = Var(V_i^2) + [E(V_i)^2]^2 = Var(V_i^2) + 1$$

So it is sufficient to minimize $Var(V_i^2)$.
$Var(V_i^2)$ is minimized if V_i^2 is constant (with probability 1).
Since $E(V_i)$ and $E(V_i^2)$ must equal to 0 and 1 respectively, so that $Var(\hat{tr(A)})$ is minimized by taking the elements of V_i to be ± 1, and each with probability $\frac{1}{2}$.

(b)

Since

$$H = X(X^TX)^{-1}X^T = \begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{pmatrix}$$

So

$$H\begin{pmatrix} V \\ 0 \end{pmatrix} = \begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{pmatrix}\begin{pmatrix} V \\ 0 \end{pmatrix} = \begin{pmatrix} H_{11}V \\ H_{21}V \end{pmatrix}$$

$$H\begin{pmatrix} H_{11}^{k-1}V \\ 0 \end{pmatrix} = \begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{pmatrix}\begin{pmatrix} H_{11}^{k-1}V \\ 0 \end{pmatrix} = \begin{pmatrix} H_{11}^{k}V \\ H_{21}H_{11}^{k-1}V \end{pmatrix}$$

(c)

Firstly, modify the given leverage function as follows:

```r
#modify the given leverage function
leverage <- function(x1,x2,w,r=10,m=100) {
qrx1 <- qr(x1)
qrx2 <- qr(x2)
n <- nrow(x1) #since X1 and X2 have the same #row=1000
lev1 <- NULL
lev2 <- NULL
for (i in 1:m) {
  v <- ifelse(runif(n)>0.5,1,-1)
  v[-w] <- 0 #set other points = 0
  v01 <- qr.fitted(qrx1,v)
  v02 <- qr.fitted(qrx2,v)
  f1 <- v01
  f2 <- v02
  for (j in 2:r) {
    v01[-w] <- 0
    v02[-w] <- 0
    v01 <- qr.fitted(qrx1,v01)
    v02 <- qr.fitted(qrx2,v02)
    f1 <- f1 + v01/j
    f2 <- f2 + v02/j
    }
  lev1 <- c(lev1,sum(v*f1)) #estimated leverage of X1
  lev2 <- c(lev2,sum(v*f2)) #estimated leverage of X2
  }
std.err1 <- exp(-mean(lev1))*sd(lev1)/sqrt(m) #standard error of leverage of X1
std.err2 <- exp(-mean(lev2))*sd(lev2)/sqrt(m) #standard error of leverage of X2
std.err.diff <- 0.5*(exp(-mean(lev1))+exp(-mean(lev2)))*sd(lev1-lev2)/sqrt(m)
lev1 <- 1 - exp(-mean(lev1))
lev2 <- 1 - exp(-mean(lev2))
r <- list(lev = c(lev1,lev2), lev.diff = lev1-lev2, std.err=c(std.err1,std.err2,std.err.diff))
r  #($std.err[3]: the third element of $std.err is an estimate of the standard error of the difference.
}
```

We define the two design matrices X1 and X2 as follows:

```
#construct two design matrices X1 and X2
x <- c(1:1000)/1000
X1 <- 1
for (k in 1:5) X1 <- cbind(X1,cos(2*k*pi*x),sin(2*k*pi*x))
library(splines) # loads the library of functions to compute B-splines
X2 <- cbind(1,bs(x,df=10))
```

Then we can use the above modified leverage function to compute the 20 leverages for X1 and X2 and their differences.
In this example, we chose r = 20 and m = 100.

```
lev1 <- NULL #empty list of 20 estimated leverages for X1
lev2 <- NULL #empty list of 20 estimated leverages for X2
lev.diff <- NULL #empty list of 20 differences of two estimated leverages
std.err.diff <- NULL #empty list of estimates of the standard errors of the differences.
for (k in (1:20)) {
  w_k <- c((50*k-49):(50*k)) #(k-1)/20 < xi <= k/20 eg. when k=1, 0<xi<=1/20, w_k = c(1:50)
  r <- leverage(X1,X2,w_k,r=20,m=100)
  lev1 <- c(lev1, r$lev[1])
  lev2 <- c(lev2, r$lev[2])
  lev.diff <- c(lev.diff, r$lev.diff)
  std.err.diff <- c(std.err.diff, r$std.err[3])
}
```

Here is a list of 20 estimated leverages for X1:

```
lev1  #a list of 20 estimated leverages for X1
```

```
##  [1] 0.5305118 0.5599752 0.4236836 0.5238033 0.5026695 0.5138854 0.4797832
##  [8] 0.4662939 0.5213244 0.5392356 0.4990046 0.6080412 0.6029045 0.4995600
## [15] 0.5896916 0.5199767 0.5997865 0.4923020 0.5622733 0.4899432
```

Here is a list of 20 estimated leverages for X2:

```
lev2  #a list of 20 estimated leverages for X2
```

```
##  [1] 0.9785562 0.6697689 0.4146736 0.4563229 0.4009268 0.4432598 0.3156640
##  [8] 0.4048661 0.3479995 0.4293976 0.3880660 0.4149868 0.5384891 0.3320690
## [15] 0.5144641 0.4090392 0.5238404 0.4823393 0.6710516 0.9619090
```

Here is a list of 20 differences of two estimated leverages:

```
lev.diff  #a list of 20 differences of two estimated leverages
```

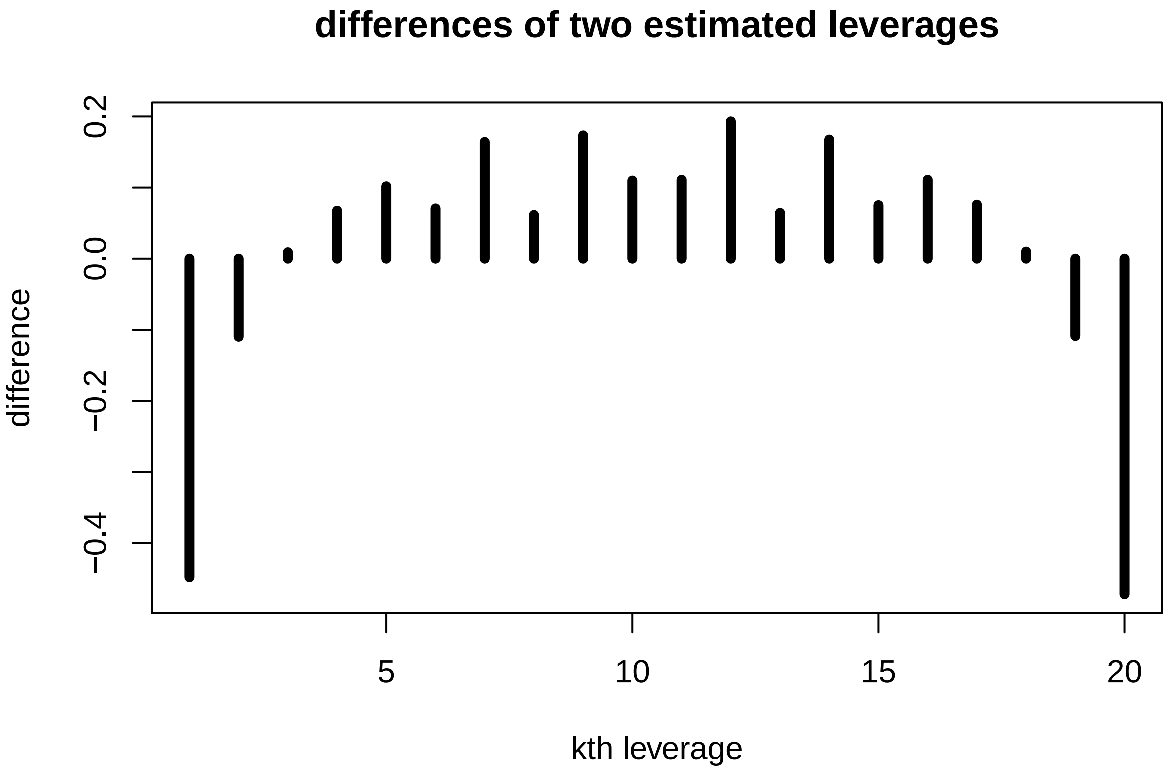
```
##  [1] -0.448044371 -0.109793724  0.009009995  0.067480367  0.101742641
##  [6]  0.070625601  0.164119261  0.061427770  0.173324886  0.109838041
## [11]  0.110938634  0.193054436  0.064415398  0.167490948  0.075227521
## [16]  0.110937470  0.075946109  0.009962668 -0.108778308 -0.471965779
```

Here is a list of estimates of the standard errors of the differences.

```
std.err.diff  #a list of estimates of the standard errors of the differences.
```

```
##  [1] 0.097816078 0.018897767 0.003763207 0.010334948 0.013176147
##  [6] 0.007289563 0.020473768 0.006070282 0.025450391 0.013948671
## [11] 0.014781839 0.022027997 0.007904531 0.023221664 0.007347576
## [16] 0.012058281 0.012519934 0.003739175 0.015854084 0.096327879
```

```r
plot(c(1:20), lev.diff, type="h", lwd=5,
     main="differences of two estimated leverages",
     xlab="kth leverage",ylab="difference")
```



From the above plot, we find that the first(k=1) and last(k=20) differences are the two largest, and other differences are small, with absolute values less than 0.2
Additionally, the differences of the first 10 leverages($k = 1, ..., 10$) are almost symmetric with differences of the other 10 leverages($k = 11, ..., 20$).

Question 2

(a)

Let $\bar{x}$ be the sample mean and s^2 be the sample variance of the data $x_1, ..., x_n$
Since for Gamma Distribution, we have

$$E(x) = \frac{\alpha}{\lambda} \quad \text{and} \quad Var(x) = \frac{\alpha}{\lambda^2}$$

Then

$$\alpha = \frac{[E(x)]^2}{Var(x)} \Rightarrow \hat{\alpha} = \frac{\bar{x}^2}{s^2}$$
$$\lambda = \frac{E(x)}{Var(x)} \Rightarrow \hat{\lambda} = \frac{\bar{x}}{s^2}$$

(b)

Since we have:

$$f(x; \alpha, \lambda) = \frac{\lambda^\alpha x^{\alpha-1} exp(-\lambda x)}{\Gamma(\alpha)} \quad \text{for} \quad x > 0$$

Then, likelihood function for α and λ is:

$$L(\alpha, \lambda) = \prod_{i=1}^{n} \frac{\lambda^\alpha x_i^{\alpha-1} exp(-\lambda x_i)}{\Gamma(\alpha)}$$

log-likelihood function for α and λ is:

$$\begin{aligned} l(\alpha, \lambda) &= \sum_{i=1}^{n} ln\{\frac{\lambda^\alpha x_i^{\alpha-1} exp(-\lambda x_i)}{\Gamma(\alpha)}\} \\ &= \sum_{i=1}^{n} \alpha ln(\lambda) + (\alpha-1) ln(x_i) - \lambda x_i - ln(\Gamma(\alpha)) \\ &= n\alpha ln(\lambda) + (\alpha-1)\sum_{i=1}^{n} ln(x_i) - \lambda \sum_{i=1}^{n} x_i - n ln(\Gamma(\alpha)) \end{aligned}$$

Then take the first partial derivatives:

$$\frac{\partial l(\alpha, \lambda)}{\partial \alpha} = n ln(\lambda) + \sum_{i=1}^{n} ln(x_i) - n\frac{\Gamma'(\alpha)}{\Gamma(\alpha)}$$
$$\frac{\partial l(\alpha, \lambda)}{\partial \lambda} = \frac{n\alpha}{\lambda} - \sum_{i=1}^{n} x_i$$

If we let $\psi(\alpha)$ and $\psi'(\alpha)$ be the first and second derivatives of $ln(\Gamma(\alpha))$,
then we can write the score vector as follows:

$$\begin{pmatrix} n ln(\lambda) + \sum_{i=1}^{n} ln(x_i) - n\psi(\alpha) \\ \frac{n\alpha}{\lambda} - \sum_{i=1}^{n} x_i \end{pmatrix}$$

Then we can get the Hessian matrix:

$$\begin{pmatrix} -n\psi'(\alpha) & \frac{n}{\lambda} \\ \frac{n}{\lambda} & -\frac{n\alpha}{\lambda^2} \end{pmatrix}$$

So the Fisher information matrix is:

$$\begin{pmatrix} n\psi'(\alpha) & -\frac{n}{\lambda} \\ -\frac{n}{\lambda} & \frac{n\alpha}{\lambda^2} \end{pmatrix}$$

The following function "newton_mle" implements the Newton-Raphson algorithm to compute the MLEs of α and λ based on $x_1, ..., x_n$.
It also outputs an estimate of the variance-covariance matrix of the MLEs. ($cov_matrix)

```
newton_mle <- function(x,eps=1.e-8,max.iter=50) {
n <- length(x)
alpha <- mean(x)^2/var(x) #estimate of alpha
lambda <- mean(x)/var(x) #estimate of lambda
theta <- c(alpha,lambda)
# compute the scores based on the initial estimates
score1 <- sum(log(x)) + n*(log(lambda) - digamma(alpha))
score2 <- n*alpha/lambda-sum(x)
score <- c(score1,score2) # score vector
iter <- 1
while (max(abs(score))>eps && iter<=max.iter) {
  # compute observed Fisher information
  info.11 <- n*trigamma(alpha)
  info.12 <- -n/lambda #same as the info.21
  info.22 <- n*alpha/lambda^2
  info <- matrix(c(info.11,info.12,info.12,info.22),ncol=2) # Fisher information matrix
  # Newton-Raphson iteration
  theta <- theta + solve(info,score)
  # update alpha and lambda
  alpha <- theta[1]
  lambda <- theta[2]
  iter <- iter + 1
  # update score vector
  score1 <- sum(log(x)) + n*(log(lambda) - digamma(alpha))
  score2 <- n*alpha/lambda-sum(x)
  score <- c(score1,score2)
}
if (max(abs(score))>eps) print("No convergence")
else {
  print(paste("Number of iterations =",iter-1))
  loglik <- n*alpha*log(lambda) + (alpha -1)*sum(log(x)) - lambda*sum(x) - n*log(gamma(alpha))
  # log-likelihood of alpha and lambda
  info.11 <- n*trigamma(alpha)
  info.12 <- -n/lambda
  info.22 <- n*alpha/lambda^2
  info <- matrix(c(info.11,info.12,info.12,info.22),ncol=2) # Fisher information matrix
  cov_matrix <- solve(info)  # estimated variance-covariance matrix
  r <- list(alpha=alpha,lambda=lambda,loglik=loglik,info=info,cov_matrix=cov_matrix)
  r
}
}
```

Then we can test the above function on data generated from a Gamma distribution.

```r
x <- rgamma(n=100, shape=1) # 100 observations from a gamma distribution with shape=1
r <- newton_mle(x) # use "newton_mle" estimates of alpha and lambda
```

```
## [1] "Number of iterations = 5"
```

```r
r
```

```
## $alpha
## [1] 1.009051
##
## $lambda
## [1] 0.8832763
##
## $loglik
## [1] -113.3101
##
## $info
##           [,1]      [,2]
## [1,]  162.3438 -113.2149
## [2,] -113.2149  129.3361
##
## $cov_matrix
##            [,1]       [,2]
## [1,] 0.01581259 0.01384161
## [2,] 0.01384161 0.01984810
```