University of Toronto

Coursework & Homework demo

YiFei Gu

1

(a) Since $\text{Var}(\boldsymbol{V}^T A\boldsymbol{V}) = \mathbb{E}[(\boldsymbol{V}^T A\boldsymbol{V})] - \text{tr}(A^2)$, note that:

$$\mathbb{E}[(\boldsymbol{V}^T A\boldsymbol{V})] = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} a_{ij}a_{kl}\,\mathbb{E}(V_iV_jV_kV_l) \tag{1}$$

It suffices to minimize the above equation. Note that since $\mathbb{E}(\boldsymbol{V_i}\boldsymbol{V_i^T})) = I$, then for $i, j, k, l \in n$, $\mathbb{E}(V_iV_jV_kV_l) = 0$ if $i = j = k = l$ with $V_i = V_j = V_k = V_l = 0$, and $\mathbb{E}(V_iV_jV_kV_l) = 1$ if $i = j = k = l$ with $V_i = V_j = V_k = V_l = 1$.
Thus the equation to be minimized becomes:

$$\mathbb{E}[(\boldsymbol{V}^T A\boldsymbol{V})] = \sum_{i}^{n} a_{ii}^2\,\mathbb{E}(V_i^4) + c$$

With c being other components of the summation in equation 1.
Note that since $\text{Var}(V_i^2) = \mathbb{E}((V_i^2)^2) - \mathbb{E}^2(V_i^2)$, then $\mathbb{E}(V_i^4) = \text{Var}(V_i^2) + 1$, it suffices to minimize $\text{Var}(V_i^2)$, or in other words, let $V_i = \pm 1$ with probability of $\frac{1}{2}$, thus $(V_i)^2 = 1$ as a constant minimizes $\text{Var}(\hat{\text{tr}}(A))$.

(b) Given that $H = X(X^TX)^{-1}X^T$, with

$$H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$$

Then for $H_{11}\boldsymbol{V}$, we have:

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

And for $H_{11}^k\boldsymbol{V}$:

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

(c) Modification for the function is needed, since we need both X_1 and X_2 as the parameter of function.

```
> leverage_mod <- function(x1,x2,w,r=10,m=100) {
+     qrx1 <- qr(x1)
+     qrx2 <- qr(x2)
+     n <- nrow(x1)
+     lev1 <- NULL
+     lev2 <- NULL
+     for (i in 1:m) {
+         v <- ifelse(runif(n)>0.5,1,-1)
+         v[-w] <- 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))
+         lev2 <- c(lev2,sum(v*f2))
+     }
+     se1 <- exp(-mean(lev1))*sd(lev1)/sqrt(m)
+     se2 <- exp(-mean(lev2))*sd(lev2)/sqrt(m)
+     lev1 <- 1 - exp(-mean(lev1))
+     lev2 <- 1 - exp(-mean(lev2))
+     r <- list(lev=c(lev1,lev2),std.err=c(se1,se2))
+     r
+ }
> 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)
> X2 <- cbind(1,bs(x,df=10))
> plot(x,X2[,2])
> for (i in 3:11) points(x,X2[,i])
> r1 = leverage_mod(X1, X2, c(1:50), r = 10, m= 100)
$lev
[1] 0.5391303 0.9685574

$std.err
[1] 0.04428045 0.01285608
> r2 = leverage_mod(X1, X2, c(51:100), r = 10, m= 100)
> r2
$lev
[1] 0.4345439 0.5354604

$std.err
[1] 0.03878436 0.04189221
> r3 = leverage_mod(X1, X2, c(101:150), r = 10, m= 100)
> r3
$lev
[1] 0.5744342 0.5550813

$std.err
[1] 0.04637881 0.04616616
> r4 = leverage_mod(X1, X2, c(151:200), r = 10, m= 100)
> r4
$lev
[1] 0.5405984 0.4701040
```

```
$std.err
[1] 0.05214043 0.05041175
> r5 = leverage_mod(X1, X2, c(201:250), r = 10, m= 100)
> r5
$lev
[1] 0.4366747 0.3374175

$std.err
[1] 0.03676599 0.03209490

> r6 = leverage_mod(X1, X2, c(251:300), r = 10, m= 100)
> r6
$lev
[1] 0.4752764 0.4030359

$std.err
[1] 0.04751170 0.04412447

> r7 = leverage_mod(X1, X2, c(301:350), r = 10, m= 100)
> r7
$lev
[1] 0.4132389 0.2692684

$std.err
[1] 0.04181562 0.03062999

> r8 = leverage_mod(X1, X2, c(351:400), r = 10, m= 100)
> r8
$lev
[1] 0.5442169 0.4810134

$std.err
[1] 0.04356693 0.04312811

> r9 = leverage_mod(X1, X2, c(401:450), r = 10, m= 100)
> r9
$lev
[1] 0.4972792 0.3303129

$std.err
[1] 0.04337605 0.03355239

> r10 = leverage_mod(X1, X2, c(451:500), r = 10, m= 100)
> r10
$lev
[1] 0.4254857 0.3245338

$std.err
[1] 0.04105524 0.03658856
```

```
> r11 = leverage_mod(X1, X2, c(501:550), r = 10, m= 100)
> r11
$lev
[1] 0.5567861 0.4411578

$std.err
[1] 0.04703956 0.04458231

> r12 = leverage_mod(X1, X2, c(551:600), r = 10, m= 100)
> r12
$lev
[1] 0.5012264 0.3334002

$std.err
[1] 0.04229100 0.03222697

> r13 = leverage_mod(X1, X2, c(601:650), r = 10, m= 100)
> r13
$lev
[1] 0.5699727 0.5063734

$std.err
[1] 0.05825033 0.05783727

> r14 = leverage_mod(X1, X2, c(651:700), r = 10, m= 100)
> r14
$lev
[1] 0.5137694 0.3432151

$std.err
[1] 0.04500656 0.03542896

> r15 = leverage_mod(X1, X2, c(701:750), r = 10, m= 100)
> r15
$lev
[1] 0.5686833 0.4875390

$std.err
[1] 0.05014784 0.04925307

> r16 = leverage_mod(X1, X2, c(751:800), r = 10, m= 100)
> r16
$lev
[1] 0.4189314 0.3240162

$std.err
[1] 0.03213981 0.02735649

> r17 = leverage_mod(X1, X2, c(801:850), r = 10, m= 100)
> r17
$lev
```

```
[1] 0.4838094 0.4117553

$std.err
[1] 0.04468499 0.04109876

> r18 = leverage_mod(X1, X2, c(851:900), r = 10, m= 100)
> r18
$lev
[1] 0.5568911 0.5351291

$std.err
[1] 0.03999362 0.03993935

> r19 = leverage_mod(X1, X2, c(901:950), r = 10, m= 100)
> r19
$lev
[1] 0.4879356 0.5837424

$std.err
[1] 0.04846789 0.05169848

> r20 = leverage_mod(X1, X2, c(951:1000), r = 10, m= 100)
> r20
$lev
[1] 0.5041636 0.9573154

$std.err
[1] 0.04788232 0.01589871
```

As one can obtain that the leverage of X_1 fluctuates and remains approximately the same for the 20 leverage output, and X_2 decreased from the start, and increased to the original level in the last output.

2

(a) Note that since $\mathbb{E}(X) = \frac{\alpha}{\lambda^2}$ and $\text{Var}(X) = \frac{\alpha}{\lambda}$, then it is easy to obtain that:

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

Thus, let $\bar{x}$ to be the sample mean and s^2 to be the sample variance, then $\hat{\alpha} = \frac{\bar{x}^2}{s^2}$ and $\hat{\lambda} = \frac{\bar{x}^2}{s^2}$

(b)

$$\mathcal{L} = \prod_i^n \frac{\lambda^\alpha x_i^{\alpha-1}\exp(-\lambda x_i)}{\tau(\alpha)} = \frac{\lambda^{n\alpha}\prod_i^n (x_i)^{\alpha-1}\exp^{-\lambda\sum_i^n X_i}}{[\tau(\alpha)]^n}$$

$$l = \ln\mathcal{L} = n\alpha\ln\lambda + (\alpha-1)\ln\prod_i^n x_i - \lambda\sum_i^n x_i - n\ln(\tau(\alpha))$$

with fisher score for α and λ being:

$$\begin{bmatrix} \frac{dl}{d\alpha} \\ \frac{dl}{d\lambda} \end{bmatrix} = \begin{bmatrix} n\ln\lambda + \sum_i^n \ln x_i - n\frac{\tau'(\alpha)}{\tau(\alpha)}) \\ n\frac{\alpha}{\lambda} - \sum_i^n x_i \end{bmatrix}$$

And the firsher information with:

$$F = -\begin{bmatrix} \frac{\partial^2 l}{\partial \alpha^2} & \frac{\partial^2 l}{\partial \alpha \partial \lambda} \\ \frac{\partial^2 l}{\partial \alpha \partial \lambda} & \frac{\partial^2 l}{\partial \lambda^2} \end{bmatrix} = -\begin{bmatrix} -n(\frac{\tau'(\alpha)}{\tau(\alpha)})' & \frac{n}{\lambda} \\ \frac{n}{\lambda} & -\frac{n\alpha}{\lambda^2} \end{bmatrix} = \begin{bmatrix} n(\frac{\tau'(\alpha)}{\tau(\alpha)})' & -\frac{n}{\lambda} \\ -\frac{n}{\lambda} & \frac{n\alpha}{\lambda^2} \end{bmatrix}$$

The e Newton-Raphson algorithm to compute the MLE:

```
> NRMLE <- function(x,eps=1.e-8,max.iter=50) {
+     n <- length(x)
+     alpha <- mean(x)^2/var(x)
+     lambda <- mean(x)/var(x)
+     theta <- c(alpha,lambda)
+     score1 <- sum(log(x)) + n*(log(lambda) - digamma(alpha))
+     score2 <- n*alpha/lambda - sum(x)
+     score <- c(score1,score2)
+     iter <- 1
+     while (max(abs(score))>eps && iter<=max.iter) {
+         info11 <- n*trigamma(alpha)
+         info12 <- -n/lambda
+         info21 <- info12
+         info22 <- n*alpha/lambda^2
+         info <- matrix(c(info11, info12, info21, info22), ncol = 2)
+         theta <- theta + solve(info,score)
+         alpha <- theta[1]
+         lambda <- theta[2]
+         iter <- iter + 1
+         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))
+         info11 <- n*trigamma(alpha)
+         info12 <- -n/lambda
+         info21 <- info12
+         info22 <- n*alpha/lambda^2
+         info <- matrix(c(info11, info12, info21, info22), ncol = 2)
+         r <- list(alpha = alpha, lambda = lambda, info = info, vcmat =
↪  solve(info))
+         r
+     }
+ }
>
> x <- rgamma(100, shape = 0.5)
> r <- NRMLE(x)
[1] "Number of iterations = 5"
> r
$alpha
[1] 0.4387973

$lambda
```

```
[1] 0.8229495

$info
          [,1]       [,2]
[1,]  618.1949 -121.51413
[2,] -121.5141   64.79142

$vcmat
            [,1]        [,2]
[1,] 0.002562137 0.004805201
[2,] 0.004805201 0.024446135
```