Problem Set 7

Yue Hu

Nov 2017

problem 1

We can calculate the mean of the estimation of standard error of the 1000 simulations, and then calculate the standard deviation of the estimated θ of the simulations. If these two are about the same, then the standard error is a good estimator.

problem 2

First do eigen dicompisition on A as $A = \Gamma\Lambda\Gamma^\top$. So that $\|A\|_2 = sup\sqrt{(Az)^\top Az} = sup\sqrt{z^\top A^\top Az} = sup\sqrt{z^\top\Gamma\Lambda\Gamma^\top\Gamma\Lambda\Gamma^\top z} = sup\sqrt{z^\top\Gamma\Lambda^2\Gamma^\top z}$. Set $y = \Gamma^\top z$, Then $\|y\|_2 = \sqrt{y\|y\|_2 y} = \sqrt{z^\top\Gamma\Gamma^\top z} = \sqrt{z^\top z} = \|z\|_2 = 1$.

Let $D = \Lambda^2$, and D is a diagnal matrix with entry $\sigma_1^2, \sigma_2^2, ..., \sigma_n^2$, where $\sigma_i$is the diagnal entries of Λ sorted in decent order, thus is the eigenvalues of A ordered. So $\|A\|_2 = \sqrt{y^\top Dy} = \sqrt{\sigma_1^2 y_1^2 + \sigma_2^2 y_2^2 + ... + \sigma_n^2 y_n^2}$. Since $\|y\|_2 = 1$, $y_1^2 + y_2^2 + ... + y_n^2 = 1$.

Next we prove $\sigma_1^2 y_1^2 + \sigma_2^2 y_2^2 + ... + \sigma_n^2 y_n^2 \leq \sigma_1^2$. Since $\sigma_1^2 - (\sigma_1^2 y_1^2 + \sigma_2^2 y_2^2 + ... + \sigma_n^2 y_n^2) = \sigma_1(y_1^2 + y_2^2 + ... + y_n^2) - (\sigma_1^2 y_1^2 + \sigma_2^2 y_2^2 + ... + \sigma_n^2 y_n^2) = (\sigma_1^2 - \sigma_2^2)y_2^2 + ... + (\sigma_1^2 - \sigma_n^2)y_n^2 \geq 0$ and only equals 0 when $y_1 = 1$ and for others $y_i = 0$.

So $\|A\|_2 = \sqrt{y^\top Dy} = \sqrt{\sigma_1^2 y_1^2 + \sigma_2^2 y_2^2 + ... + \sigma_n^2 y_n^2} \leq \sqrt{\sigma_1^2} =\mid \sigma_1 \mid$. Thus $\|A\|_2$ equals $\mid \sigma_1 \mid$, the largest of the absolute values of the eigenvalues of A.

problem 3

(a)

Suppose we have singular value decomposition $X = U\sum V^\top$, then $X^\top X = V\sum^\top U^\top U\sum V^\top = V\sum^\top\sum V^\top$. and $\sum^\top\sum$ is a square diagonal matrix whose first k diagonal entries are squares of X's sigular values σ_i^2, and the remaining diagonal entries equal to 0. Thus $X^\top X = V\sum^\top\sum V^\top$ is the eigen decomposition of the symetric matrix $X^\top X$, its eigenvectors are the right eigen singular vectors of X, and its eigenvalues are the squares of the singular values of X.

Since the eigenvalues of $X^\top X$ is either possitive or zero as proved above, it is possitive semi-definite.

(b)

Supose we have computed $\sum = \Gamma\Lambda\Gamma^\top$. Then $Z = \sum + cI = \Gamma\Lambda\Gamma^\top + c\Gamma\Gamma^\top = \Gamma\Lambda\Gamma^\top + \Gamma D\Gamma^\top$, where D is a diagnal matrix with all entries equal to c. So $Z = \Gamma(\Lambda + D)\Gamma^\top$. So the eigenvalues of $\sum$ plus c is the eigenvalues of Z, and it's O(n) operation.

Problem 4

(a)

Firstly do QR decomposition on X, $X = QR$, so $C = X^\top X = R^\top Q^\top QR = R^\top R$.

Then solve and store the value $x = C^{-1}d$. We can solve $Cx = d$ using backsolve, because C is decomposed to be product of two triangular matrices.

Then to decrease computation orders we compute from right to left, calculate $y = -Ax + b$.

Then to calcualte $AC^{-1}A^\top$, notice $C^{-1} = R^{-1}R^{-\top}$, so $AC^{-1}A^\top = AR^{-1}R^{-\top}A^\top$. Do QR on $R^{-\top}A^\top = Q_1R_1$, then $AR^{-1}R^{-\top}A^\top = (R^{-\top}A^\top)^\top R^{-\top}A^\top = R_1^\top R_1$. Use backsolve to get itsd inverse.

Then use the result and crossproduct with $A^\top$, getting a vector z, solve $C^{-1}z$ and add the stored value x to get $\hat{\beta}$.

(b)

```
X.qr <- qr(X)
R <- qr.R(X.qr)
x <- backsolve(R, backsolve(R, d, transpose = TRUE))
y <- b- A%*%d

X1.qr <- qr(backsolve(R, tr(A), transpose = TRUE))
R1 <- qr.R(X1.qr)
V <- backsolve(R1, backsolve(R1, A, transpose = TRUE))
U <- tcrossprod(V, A)

Z<- backsolve(R, backsolve(R, corssprod(A,U), transpose = TRUE))
betaHat <- x + z
```

Problem 5

(a)

Because computing $Z(Z^\top Z)^{-1}Z^\top$ would result in a 60 million×60 million matrix, and $\hat{X}$ would be 60 million×600, it would be too large to store.

(b)

First calculate $W = (Z^\top Z)^{-1}$, which would be a 630 × 630 matrix, solve and get its inverse (you can use choleskey decompostion or LU decompositions).

Then calculate $\hat{X}^\top = X^\top ZW^\top Z^\top$, and insert it into the second equation getting $\hat{X}^\top\hat{X} = X^\top ZW^\top Z^\top X$. $Z^\top X$ is 630 × 600 and $\hat{X}^\top\hat{X} = tcrossprod(Z^\top X, crossprod(W, Z^\top X))$ would be 600 × 630 matrix, name it V.

At this time we are solveing $\hat{\beta} = V^{-1}X^\top ZW^\top Z^\top$. agian $X^\top Z$ is 600 × 630 matrix and now all components in $\hat{\beta} = V^{-1}(X^\top Z)W^\top Z^\top$ are matrices in hundres of lines and coloumns and we can use sparse matrix calculation and also OLS techniques to solve it.

Problem 6

From the sesult we can see that when condition number is in range of 1e18 matrix is not numerically positive definite. When condition number increases, the difference between estimated and true values increases.

```
# generate eigenvector
n <- 100
set.seed(1)
A <- crossprod(matrix(rnorm(n^2), n))
vecotrs <- eigen(A, symmetric = T)$vectors
```

```r
# input lower and upper bond to generate a range of
# some inaccuracy will accour but the numbers will be in the desired range
create_eigenVal <- function(MinVal, MaxVal) {
  step <- (log2(MaxVal)-log2(MinVal))* 0.01
  seq1<- seq(log2(MinVal),log2(MaxVal), step)[1:100] #generate 101 items and select 100
  2^seq1
}

# input eigenvalues and check if computed eigenvalue is the same
check_eigenVal <- function(eigenVal) {
  # show the condition Number
  ConNum <- eigenVal[100]/eigenVal[1]
  cat("Conditon Num:", ConNum, "\n")

  # construct new matrix with eigenvalue and eigenvector
  D <- diag(eigenVal)
  X <-tcrossprod(vecotrs %*% D, vecotrs)

  # calculate new eigenvalue and chack if they are the same
  NewEigenVal <- eigen(X)$values
  cat("smalles 3 eigenvalues;",NewEigenVal[98:100], "\n")
  all.equal(eigenVal,NewEigenVal)
}


#set eigenvalue to be the same
eigenVal1 <- rep(10,100)
check_eigenVal(eigenVal1)

## Conditon Num: 1
## smalles 3 eigenvalues; 10 10 10
## [1] TRUE

# 100 times range
eigenVal2 <- create_eigenVal(1,1e2)
check_eigenVal(eigenVal2)

## Conditon Num: 95.49926
## smalles 3 eigenvalues; 1.096478 1.047129 1
## [1] "Mean relative difference: 1.636364"

# 1e14 times range
eigenVal3 <- create_eigenVal(1e-7,1e7)
check_eigenVal(eigenVal3)

## Conditon Num: 7.24436e+13
## smalles 3 eigenvalues; 1.904992e-07 1.379728e-07 1.000352e-07
## [1] "Mean relative difference: 2"

# 1e18 times range
eigenVal4 <- create_eigenVal(1e-7,1e11)
check_eigenVal(eigenVal4)
```

```
## Conditon Num: 6.606934e+17
## smalles 3 eigenvalues; 9.81931e-07 8.808994e-07 7.071911e-07
## [1] "Mean relative difference: 2"

# 1e19 times range
eigenVal5 <- create_eigenVal(1e-7,1e12)
check_eigenVal(eigenVal5)

## Conditon Num: 6.456542e+18
## smalles 3 eigenvalues; -2.501662e-06 -3.442197e-06 -1.989659e-05
## [1] "Mean relative difference: 2"
```