3701_Assignment3

JinYao

2019/7/19

howework3

Problem 1

i

```
sim.beta.hat = function(x.mat,beta,sigma,reps){
  n = nrow(x.mat);p = ncol(x.mat)
  beta.hat.mat = matrix(NA,nrow = reps,ncol = p)
  for(r in 1:reps){
  epsilon.vec = sigma*(rexp(n=n,rate=1)-1)
  y.vec = x.mat%*%beta+epsilon.vec
  beta.hat.mat[r,] = qr.coef(qr=qr(x=x.mat),y=y.vec)
  }
  beta.hat = apply(beta.hat.mat,2,mean)
  return(beta.hat)
}
```

ii.

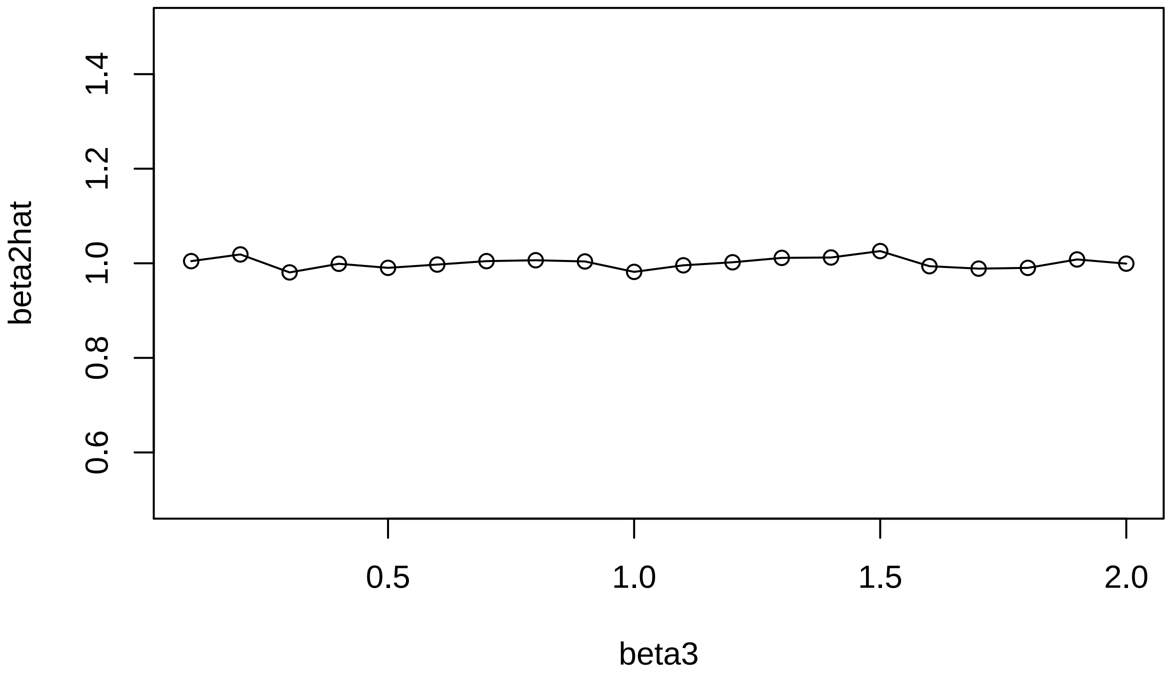
```
x.mat = read.csv('~/Desktop/3701/X.mat.csv',header=TRUE)
x.mat = as.matrix(x.mat)
beta = c(-1,1,0,1)
sigma = 1.5;reps = 5e3
betasim = sim.beta.hat(x.mat = x.mat, beta = beta, sigma = sigma, reps = reps)
cbind(betasim[2],beta[2])
```

```
##          [,1] [,2]
## [1,] 1.002454    1
```

```
# they are very close
```

iii.

```
x.mat = read.csv('~/Desktop/3701/X.mat.csv',header=TRUE)
x.mat = as.matrix(x.mat)
beta3 = seq(from = 0.1, to = 2, by = 0.1)
#beta = c(-1,1,beta3,0)
num = length(beta3)
sigma = 1.5; reps = 1e3
beta2hat = numeric(num)
for(r in 1:num){
  betasim = sim.beta.hat(x.mat = x.mat, beta = c(-1,1,beta3[r],0), sigma = sigma, reps = reps)
  beta2hat[r] = betasim[2]
}
plot(beta3,beta2hat,type = "o",ylim = c(0.5,1.5))
```



```
# we adjust the x axis to the sequence of beta3, as the line represents the beta2hat,
# then adjust the limit of the y axis, we can approximately find it is almost a line
# at around 1.0
#cbind(beta2hat,beta[3])
# this plot shows that the beta2hat is almost stay stable at around at 1.0. So we can say that it stays
```

Problem 2

i.

```
mat = read.csv('~/Desktop/3701/X.df.csv')
ncol(mat)
```

```
## [1] 4
```

```
y.mat = as.matrix(mat[,1])
x.mat = as.matrix(mat[,2:4])
betahat = qr.coef(qr = qr(x=x.mat),y=y.mat)
betahat
```

```
##                         [,1]
## Intercept          5.1073949
## Father.Height      0.7661411
## Grandfather.height 0.1632550
```

```
beta1hat = betahat[1]
beta2hat = betahat[2]
beta3hat = betahat[3]
# Interpretation: Holding other variables constant, with one inch increase of father,
# the height of the son would increase at average of 0.7661411 inches in height. And holding other
# variables constant, with one inch increase in the height of the grandfather, the
# height of the son would increase at average of 0.1632550 inches. And the intercept is
# just on average the starting height of the son given the heights of father and
# grandfather are 0, while it is not realisitc.
```

ii.

```
set.seed(3701)
p = length(betahat)
n = nrow(x.mat)
beta = betahat
x.mat = x.mat
xtxinv = qr.solve(crossprod(x.mat))
sqrtxtxinv_22 = sqrt(xtxinv[2,2])
t.perc = qt(1-0.05/2,n-p)
captured.list = numeric(reps)
sE = sqrt(sum((y.mat - x.mat %*% beta)^2)/(n-p))
moe = t.perc*sE*sqrtxtxinv_22
left.pt = beta[2]-moe
right.pt = beta[2]+moe
ci = cbind(left.pt,right.pt)
ci
```

```
##                  left.pt right.pt
## Father.Height 0.4901989 1.042083
```

```
# we found 0 is not in this interval, which we have to say that beta2 is significantly
# different from 0, so Father Height is significant.
```

iii.

```
set.seed(3701)
p = length(betahat)
n = nrow(x.mat)
beta = betahat
x.mat = x.mat
xtxinv = qr.solve(crossprod(x.mat))
sqrtxtxinv_33 = sqrt(xtxinv[3,3])
t.perc = qt(1-0.05/2,n-p)
captured.list = numeric(reps)
sE = sqrt(sum((y.mat - x.mat %*% beta)^2)/(n-p))
moe = t.perc*sE*sqrtxtxinv_33
t = beta[3]/(sE*sqrtxtxinv_33)
2*pt(-abs(t), n - p)
```

```
## Grandfather.height
##         0.2301573
```

```
# we can see that the p value bigger than alpha, so we fail to reject the null hypo
# -thesis, meaning the grandfather's height is significant in this model, we cannot
# ignore this coefficient.
```

iv.

```
mat = read.csv('~/Desktop/3701/X.df.csv')
y.mat = as.matrix(mat[,1])
x.mat = as.matrix(mat[,2:4])
xnew = c(1,67,66)
est.mean = sum(betahat*xnew)
est.mean
```

```
## [1] 67.21368
```

```
cat("point estimate:", est.mean,'\n')
```

```
## point estimate: 67.21368
```

```
p = length(betahat)
n = nrow(x.mat)
x.mat = x.mat
t.perc = qt((1-0.04/2),n-p)
sE = sqrt(sum((y.mat - x.mat %*% betahat)^2)/(n-p))
sqrtquadform = sqrt(t(xnew)%*%qr.solve(crossprod(x.mat))%*%xnew)[1]
est.mean = sum(betahat*xnew)
moe = t.perc*sE*sqrtquadform
left.pt.ci = est.mean-moe
right.pt.ci = est.mean+moe
cat("left:",left.pt.ci,"right:", right.pt.ci,'\n')
```

```
## left: 66.53299 right: 67.89437
```

```
sqrtquadform_pi = sqrt(1+t(xnew)%*%qr.solve(crossprod(x.mat))%*%xnew)[1]
moe = t.perc*sE*sqrtquadform_pi
left.pt.pi = est.mean-moe
right.pt.pi = est.mean+moe
cat("left:",left.pt.pi,"right:",right.pt.pi,"\n")
```

```
## left: 65.6768 right: 68.75056
```

```
cat("ci:",left.pt.ci,right.pt.ci,"pi:",left.pt.pi,right.pt.pi)
```

```
## ci: 66.53299 67.89437 pi: 65.6768 68.75056
```

```
#we can clearly see that the interval of pi is larger than ci
```

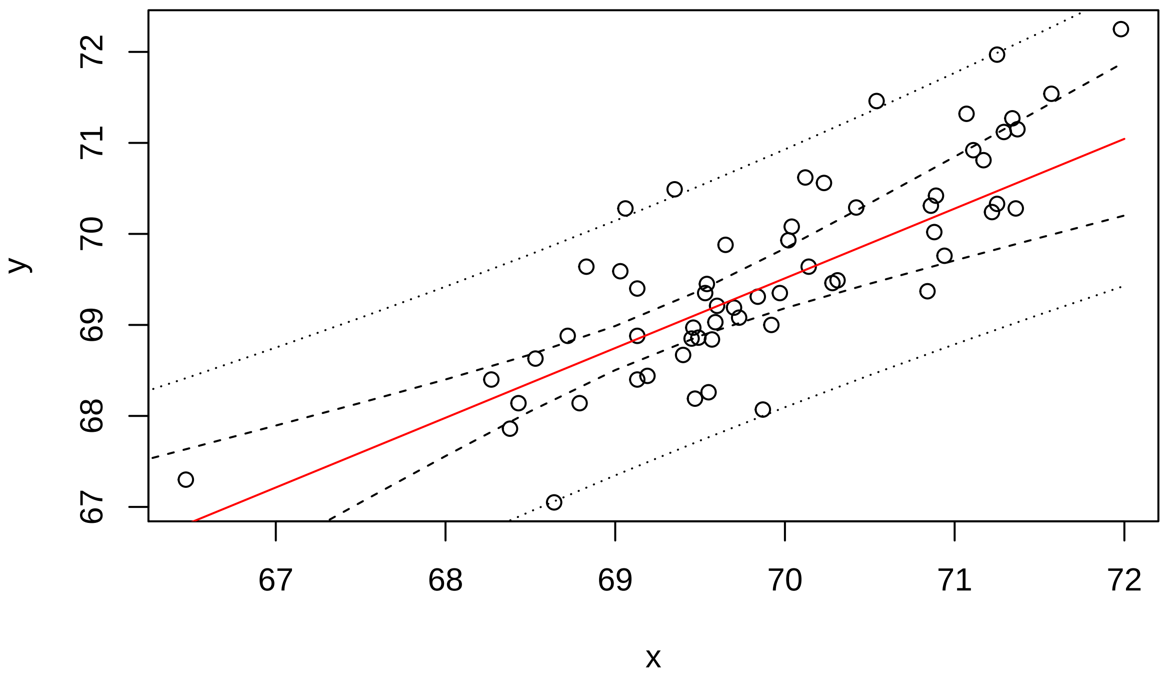
v.

```
x2 = seq(from = 66, to = 72, by = 0.5)
x3 = 66
num = length(x2)
LB.pi = numeric(num)
```

```r
UB.pi = numeric(num)
LB.ci = numeric(num)
UB.ci = numeric(num)
t.perc = qt((1-0.04/2),n-p)
est.mean = numeric(num)
for(r in 1:num){
  xnew = c(1,x2[r],x3)
  est.mean[r] = sum(betahat*xnew)
  sqrtquadform.ci = sqrt(t(xnew)%*%qr.solve(crossprod(x.mat))%*%xnew)[1]
  sqrtquadform.pi = sqrt(1 + t(xnew)%*%qr.solve(crossprod(x.mat))%*%xnew)[1]
  moe.ci = t.perc*sE*sqrtquadform.ci
  moe.pi = t.perc*sE*sqrtquadform.pi
  LB.pi[r] = est.mean[r] - moe.pi
  UB.pi[r] = est.mean[r] + moe.pi
  LB.ci[r] = est.mean[r] - moe.ci
  UB.ci[r] = est.mean[r] + moe.ci
}
cat("point estimate: ", mean(est.mean))
```

```
## point estimate:  68.74596
```

```r
plot(mat$Father.Height,mat$Son.Height,xlab = "x",ylab = "y")
lines(x2, est.mean, lty = 1,col = "red")
lines(x2, LB.ci, lty = 2)
lines(x2, UB.ci, lty = 2)
lines(x2, LB.pi, lty = 3)
lines(x2, UB.pi, lty = 3)
```



Problem 3

i.

```
set.seed(3701)
mat = read.csv('~/Desktop/3701/X.mat.3.csv')
x.mat = as.matrix(mat)
xnew = c(1,25,20,15,10)
beta = c(1,2,3,4,5)
reps = 1e3
n = nrow(x.mat)
p = ncol(x.mat)
qrt.mat = qr(x = x.mat)
sqrtquadform = sqrt(1+t(xnew)%*%qr.solve(crossprod(x.mat))%*%xnew)[1]
t.perc = qt((1-0.1/2),n-p)
captured.list = numeric(reps)

sigma_exp = 1.5
for(r in 1:reps){
  epsilon.vec = sigma_exp*(rexp(n, rate=1)-1)
  y.vec = x.mat%*%beta+epsilon.vec
  (beta.hat = qr.coef(qr = qrt.mat,y=y.vec))
  sE = sqrt(sum((y.vec - x.mat%*%beta.hat)^2)/(n-p))
  est.mean = sum(beta.hat*xnew)
  moe = t.perc*sE*sqrtquadform
  left.pt = est.mean-moe
  right.pt = est.mean+moe
  ynew = sum(beta*xnew)+rnorm(1,mean=0,sd=sigma_exp)
  captured.list[r] = 1*(left.pt<=ynew)&(ynew<=right.pt)
}
mean(ynew)
```

```
## [1] 220.8132
```

```
mean(captured.list)
```

```
## [1] 0.889
```

```
prop.test(x=sum(captured.list),n=reps,conf.level = 0.99,correct = FALSE)$conf.int[1:2]
```

```
## [1] 0.8608043 0.9120677
```

ii.

```
mat = read.csv('~/Desktop/3701/X.mat.3.csv')
x.mat = as.matrix(mat)
xnew = c(1,25,20,15,10)
beta = c(1,2,3,4,5)
reps = 1e3
n = nrow(x.mat)
p = ncol(x.mat)
qrt.mat = qr(x = x.mat)
sqrtquadform = sqrt(1+t(xnew)%*%qr.solve(crossprod(x.mat))%*%xnew)[1]
```

```r
t.perc = qt((1-0.1/2),n-p)
captured.list = numeric(reps)

sigma_unif = 1.5
for(r in 1:reps){
  u = runif(n)
  epsilon.vec = sigma_unif*sqrt(12)*(u-0.5)
  y.vec = x.mat%*%beta+epsilon.vec
  (beta.hat = qr.coef(qr = qrt.mat,y=y.vec))
  sE = sqrt(sum((y.vec - x.mat%*%beta.hat)^2)/(n-p))
  est.mean = sum(beta.hat*xnew)
  moe = t.perc*sE*sqrtquadform
  left.pt = est.mean-moe
  right.pt = est.mean+moe
  ynew = sum(beta*xnew)+rnorm(1,mean=0,sd=sigma_unif)
  captured.list[r] = 1*(left.pt<=ynew)&(ynew<=right.pt)
}
mean(ynew)
```

```
## [1] 220.9543
```

```r
mean(captured.list)
```

```
## [1] 0.899
```

```r
prop.test(x=sum(captured.list),n=reps,conf.level = 0.99,correct = FALSE)$conf.int[1:2]
```

```
## [1] 0.8717655 0.9209747
```

iii.

```r
set.seed(3701)
mat = read.csv('~/Desktop/3701/X.mat.3.csv')
x.mat = as.matrix(mat)
sigma_rnorm = 1.5
xnew = c(1,25,20,15,10)
beta = c(1,2,3,4,5)
reps = 1e3
n = nrow(x.mat)
p = ncol(x.mat)
qrt.mat = qr(x = x.mat)
sqrtquadform = sqrt(1+t(xnew)%*%qr.solve(crossprod(x.mat))%*%xnew)[1]
t.perc = qt((1-0.1/2),n-p)
captured.list = numeric(reps)
for(r in 1:reps){
  epsilon.vec = rnorm(n, sd = (sigma_rnorm)*sqrt((p^(-1)*apply(x.mat, 1, sum))))
  y.vec = x.mat%*%beta+epsilon.vec
  beta.hat = qr.coef(qr = qrt.mat,y=y.vec)
  sE = sqrt(sum((y.vec - x.mat%*%beta.hat)^2)/(n-p))
  est.mean = sum(beta.hat*xnew)
  moe = t.perc*sE*sqrtquadform
  left.pt = est.mean-moe
  right.pt = est.mean+moe
  sd2 = (sigma_rnorm)*sqrt((p^(-1)*sum(xnew)))
  ynew = sum(beta*xnew)+rnorm(1, mean = 0, sd = sd2)
  captured.list[r] = 1*(left.pt<=ynew)&(ynew<=right.pt)
}
mean(ynew)
```

```
## [1] 227.6563
```

```
mean(captured.list)
```

```
## [1] 0.864
```

```
prop.test(x=sum(captured.list),n=reps,conf.level = 0.99,correct = FALSE)$conf.int[1:2]
```

```
## [1] 0.8336680 0.8895337
```

we can find that the 90% coverage probability is only covered in the first two conditions. I think for the third condition, the error is not independent, that it is dependent on the observed x. which makes the iid not satisfying.

Problem 4

i.

$\hat{a} = argmin_{a \in R^+}(E[a\bar{X} - \mu])^2 + Var[a\bar{X}]$ $(E[a\bar{X} - \mu])^2 + Var[a\bar{X}] = (aE[\bar{X}] - \mu)^2 - a^2Var[\bar{X}] = (a\mu - \mu)^2 + a^2\frac{\sigma^2 a^2}{n} = \mu^2(a-1)^2 + a^2\frac{\sigma^2}{n}$ $\nabla(h(a)) = 2(\mu)^2(a-1) + 2a\frac{\sigma^2 a}{n}$ $\nabla^2 h(a) = 2(\mu)^2 + 2\frac{\sigma^2}{n} > 0$, so it is strictly convex because $a \in R^+$

ii.

$\nabla h(a) = 0 \Rightarrow \mu^2(a-1) + \frac{\sigma^2}{n}a = 0$ $a(\mu^2 + \frac{\sigma^2}{n}) = \mu^2$ so $a = \frac{\mu^2}{\mu^2 + \frac{\sigma^2}{n}}$ Due to the second derivative is higher than 0, we can say that all $a \in R^+$ and $f : R^+ \to R^+$ is differentiable, and $\nabla f(\bar{a}) = \nabla f(\hat{a}) = 0$, so abar = ahat is the global minimizer.

iii.

Because μ and σ are all parameters, we just have the realization of x, so we cannot get the real value of these parameters. So $\hat{a}$ is just the theorical value, we need to know the value of each parameter.

Problem 5

i.

$\nabla g(x) = \frac{sinx' \times x - sinx \times x'}{x^2} = \frac{xcosx - sinx}{x^2}, x \in (0, 6\pi]$

ii.

```r
bsearch = function(df,a,b,L=1e-7,quiet = FALSE, ...){
  k = 0
  while((b-a)>L)
  {
    k = k+1
    m = (a+b)/2
    df.at.m=df(m, ...)
    if(df.at.m<0){
      a = m
    }
    else if(df.at.m>0){
      b = m
    }else{
      a = m
      b = m
    }
    if(!quiet)
    {
      cat("after iteration k = ",k,
          "the interval is ",a,b,"nn")
    }
  }
  return((a+b)/2)
}

deriv.f = function(x.list){
  vals = (x.list * cos(x.list)-sin(x.list))/(x.list^2)
  return(vals)
}
n = 9
x.list = seq(from = 0.1, to = 6*pi,by = 1e-7)
xbar = bsearch(df=deriv.f, a = min(x.list), b= max(x.list), quiet = TRUE)
xbar
```

```
## [1] 17.22076
```

```r
sin(xbar)/xbar
```

```
## [1] -0.0579718
```

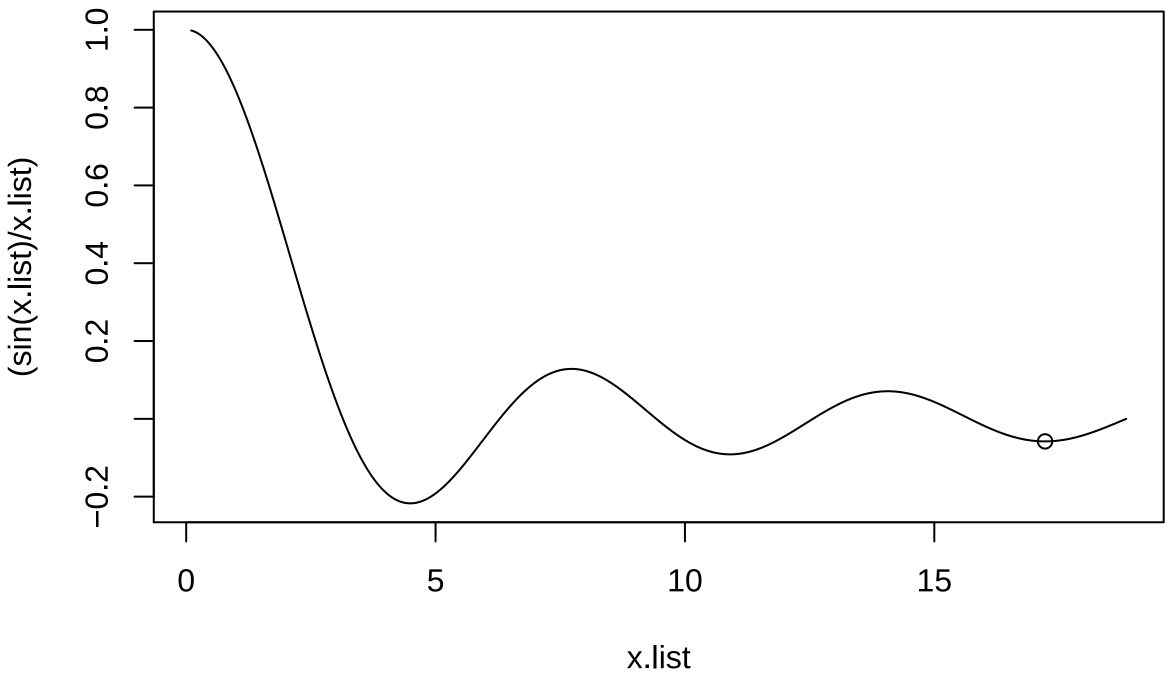
iii.

```r
bsearch = function(df,a,b,L=1e-7,quiet = FALSE, ...){
  k = 0
  while((b-a)>L)
  {
    k = k+1
    m = (a+b)/2
    df.at.m=df(m, ...)
    if(df.at.m<0){
      a = m
    }
    else if(df.at.m>0){
      b = m
    }else{
      a = m
      b = m
    }
    if(!quiet)
    {
      cat("after iteration k = ",k,
          "the interval is ",a,b,"nn")
    }
  }
  return((a+b)/2)
}

deriv.f = function(x.list){
  vals = (x.list * cos(x.list)-sin(x.list))/(x.list^2)
  return(vals)
}
n = 9
x.list = seq(from = 0.1, to = 6*pi,by = 0.001)
xbar = bsearch(df=deriv.f, a = min(x.list), b= max(x.list), quiet = TRUE)
xbar
```

```
## [1] 17.22076
```

```r
plot(x = x.list, y = (sin(x.list)/x.list),type = "l")
points(xbar,(sin(xbar)/xbar))
```



```r
# it is not a global minimizer, it is a local minimizer.
```