hw5

Jared Yu

May 31, 2018

```
# STA 243 HW 5

#### Design density

#Generate data

x_i = (c(1:200)-.5)/200
phi = function(u){
       1/sqrt(2*pi)*exp(-u^2/2)}
##Noise level
f=function(x){
1.5*phi((x-0.35)/0.15)-phi((x-0.8)/0.04)}
sig=function(j){0.02+0.04*(j-1)^2}
yn=matrix(0,200,6)
for (j in 1:6){
se=sig(j)*e
yn[,j]=f(x_i)+se}

##Design density
set.seed(2)
u=runif(200)
sigma=0.1
finv=function(x,j){
    qbeta(x,(j+4)/5,(11-j)/5)}
yd=matrix(0,200,6)
e=rnorm(200)
for (j in 1:6){
Xji=finv(u,j)
yd[,j]=f(Xji)+sigma*e}

##Spatial variation
sigma=0.2
fs=function(x,j){
    sqrt(x*(1-x))*sin(2*pi*(1+2^((9-4*j)/5))/(x+2^((9-4*j)/5)))}
ys=matrix(0,200,6)
for (j in 1:6){
fsj=fs(x_i,j)
ys[,j]=fsj+sigma*e}

##Variance function
yv=matrix(0,200,6)
fv=function(x,j){
(0.15*(1+0.4*(2*j-7)*(x-0.5)))^2}
for (j in 1:6){
yv[,j]=f(x_i)+sqrt(fv(x_i,j))*e}


## Design matrix   ##  t : knot  data-->y
```

```
t<-(1:30)/31
designmatrix=function(data,t){
f=function(a){
b=a*rep(1,30)-t
pos=((abs(b)+b)/2)^3
f=c(1,a,a^2,a^3,pos)
}
designmatrix=t(sapply(data,f))
}

## Hat matrix
hatmatrix=function(designmatrix,lambda){
D=diag(c(rep(0,4),rep(1,30)))
hatmatrix=designmatrix%*%solve(t(designmatrix)%*%designmatrix+lambda*D)%*%t(designmatr
ix)
}

## CV selection criterion   ## lambda: smoothing parameter; response: y; X:design matr
ix
CV=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
fit=hat%*%y
correct=1-diag(hat)
s=(y-fit)/correct
CV=sum(s^2)
}

## GCV selection criterion
GCV=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
correct=1-sum(diag(hat))/n
fit=hat%*%y
GCV=sum((y-fit)/correct)^2
}
## corrected AIC selection criterion
AICC=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
fit=hat%*%y
resSum=sum((y-fit)^2)
tr=sum(diag(hat))
AICC=log(resSum)+2*(tr+1)/(n-tr-2)
}

## minimun risk selection criterion
```

```
Risk=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
fit=hat%*%y
resSum=sum((y-fit)^2)
tr=sum(diag(hat))
sigma=resSum/(n-tr)
Risk=resSum+sigma^2*(2*tr-n)
}


diff=function(lambda,obs){sum((f(x_i)-hatmatrix(design,lambda)*obs)^2)}

######### Noise

#Generate data

x_i = (c(1:200)-.5)/200
phi = function(u){
       1/sqrt(2*pi)*exp(-u^2/2)}
set.seed(1)
e = rnorm(200)

##Noise level
f=function(x){
1.5*phi((x-0.35)/0.15)-phi((x-0.8)/0.04)}
sig=function(j){0.02+0.04*(j-1)^2}
yn=matrix(0,200,6)
for (j in 1:6){
se=sig(j)*e
yn[,j]=f(x_i)+se}

## Design matrix   ##  t : knot  data-->y
t<-(1:30)/31
designmatrix=function(data,t){
f=function(a){
b=a*rep(1,30)-t
pos=((abs(b)+b)/2)^3
f=c(1,a,a^2,a^3,pos)
}
designmatrix=t(sapply(data,f))
}

## Hat matrix
hatmatrix=function(designmatrix,lambda){
D=diag(c(rep(0,4),rep(1,30)))
hatmatrix=designmatrix%*%solve(t(designmatrix)%*%designmatrix+lambda*D)%*%t(designmatr
ix)
```

```
}

## CV selection criterion   ## lambda: smoothing parameter; response: y; X:design matr
ix
CV=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
fit=hat%*%y
correct=1-diag(hat)
s=(y-fit)/correct
CV=sum(s^2)
}

## GCV selection criterion
GCV=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
correct=1-sum(diag(hat))/n
fit=hat%*%y
GCV=sum((y-fit)/correct)^2
}
## corrected AIC selection criterion
AICC=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
fit=hat%*%y
resSum=sum((y-fit)^2)
tr=sum(diag(hat))
AICC=log(resSum)+2*(tr+1)/(n-tr-2)
}

## minimun risk selection criterion
Risk=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
fit=hat%*%y
resSum=sum((y-fit)^2)
tr=sum(diag(hat))
sigma=resSum/(n-tr)
Risk=resSum+sigma^2*(2*tr-n)
}


diff=function(lambda,obs){sum((f(x_i)-hatmatrix(design,lambda)*obs)^2)}

logr_n_cv=matrix(0,100,6)
```

```
for (j in 1:6){
for (k in 1:100){
yn=matrix(0,200,6)
e=rnorm(200)
yn[,j]=f(x_i)+sig(j)*e
design=designmatrix(yn[,j],t)
lambdamin=optimize(CV,interval=c(1e-5,1), response=yn[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-5,1),obs=yn[,j])$objective
r=diff(lambdamin,yn[,j])/mindiff
logr_n_cv[k,j]=log(r,base=exp(1))}
}

logr_n_gcv=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
yn=matrix(0,200,6)
e=rnorm(200)
yn[,j]=f(x_i)+sig(j)*e
design=designmatrix(yn[,j],t)
lambdamin=optimize(GCV,interval=c(1e-5,1), response=yn[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-5,1),obs=yn[,j])$objective
r=diff(lambdamin,yn[,j])/mindiff
logr_n_gcv[k,j]=log(r,base=exp(1))}
}

logr_n_aicc=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
yn=matrix(0,200,6)
e=rnorm(200)
yn[,j]=f(x_i)+sig(j)*e
design=designmatrix(yn[,j],t)
lambdamin=optimize(AICC,interval=c(1e-5,1), response=yn[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-5,1),obs=yn[,j])$objective
r=diff(lambdamin,yn[,j])/mindiff
logr_n_aicc[k,j]=log(r,base=exp(1))}
}

logr_n_risk=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
yn=matrix(0,200,6)
e=rnorm(200)
yn[,j]=f(x_i)+sig(j)*e
design=designmatrix(yn[,j],t)
lambdamin=optimize(Risk,interval=c(1e-5,1), response=yn[,j],X=design)$minimum
```

```
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-5,1),obs=yn[,j])$objective
r=diff(lambdamin,yn[,j])/mindiff
logr_n_risk[k,j]=log(r,base=exp(1))}


save(logr_n_cv,logr_n_gcv,logr_n_aicc,logr_n_risk,file="noiseresults.rda")

## Spatial density

fs=function(x,j){
    sqrt(x*(1-x))*sin(2*pi*(1+2^((9-4*j)/5))/(x+2^((9-4*j)/5)))}


## Design matrix   ##  t : knot
t<-(1:30)/31
designmatrix=function(data,t){
f=function(a){
b=a*rep(1,30)-t
pos=((abs(b)+b)/2)^3
f=c(1,a,a^2,a^3,pos)
}
designmatrix=t(sapply(data,f))
}

## Hat matrix
hatmatrix=function(designmatrix,lambda){
D=diag(c(rep(0,4),rep(1,30)))
hatmatrix=designmatrix%*%solve(t(designmatrix)%*%designmatrix+lambda*D)%*%t(designmatr
ix)
}

## CV selection criterion   ## lambda: smoothing parameter; response: y; X:design matr
ix
CV=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
fit=hat%*%y
correct=1-diag(hat)
s=(y-fit)/correct
CV=sum(s^2)
}

## GCV selection criterion
GCV=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
correct=1-sum(diag(hat))/n
```

```
fit=hat%*%y
GCV=sum((y-fit)/correct)^2
}
## corrected AIC selection criterion
AICC=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
fit=hat%*%y
resSum=sum((y-fit)^2)
tr=sum(diag(hat))
AICC=log(resSum)+2*(tr+1)/(n-tr-2)
}

## minimun risk selection criterion
Risk=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
fit=hat%*%y
resSum=sum((y-fit)^2)
tr=sum(diag(hat))
sigma=resSum/(n-tr)
Risk=resSum+sigma^2*(2*tr-n)
}

diff=function(lambda,obs){sum((f(x_i)-hatmatrix(design,lambda)*obs)^2)}


logr_s_cv=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
ys=matrix(0,200,6)
e=rnorm(200)
sigma=0.2
ys[,j]=fs(x_i,j)+sigma*e
design=designmatrix(ys[,j],t)
lambdamin=optimize(CV,interval=c(1e-5,1), response=ys[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-5,1),obs=ys[,j])$objective
r=diff(lambdamin,ys[,j])/mindiff
logr_s_cv[k,j]=log(r,base=exp(1))}
}

logr_s_gcv=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
ys=matrix(0,200,6)
e=rnorm(200)
```

```
sig=0.2
ys[,j]=fs(x_i,j)+sig*e
design=designmatrix(ys[,j],t)
lambdamin=optimize(GCV,interval=c(1e-5,1), response=ys[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-5,1),obs=ys[,j])$objective
r=diff(lambdamin,ys[,j])/mindiff
logr_s_gcv[k,j]=log(r,base=exp(1))}
}


logr_s_aicc=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
ys=matrix(0,200,6)
e=rnorm(200)
sigma=0.2
ys[,j]=fs(x_i,j)+sigma*e
design=designmatrix(ys[,j],t)
lambdamin=optimize(AICC,interval=c(1e-5,1), response=ys[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-5,1),obs=ys[,j])$objective
r=diff(lambdamin,ys[,j])/mindiff
logr_s_aicc[k,j]=log(r,base=exp(1))}
}

logr_s_risk=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
ys=matrix(0,200,6)
e=rnorm(200)
sig=0.2
ys[,j]=fs(x_i,j)+sig*e
design=designmatrix(ys[,j],t)
lambdamin=optimize(Risk,interval=c(1e-5,1), response=ys[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-5,1),obs=ys[,j])$objective
r=diff(lambdamin,ys[,j])/mindiff
logr_s_risk[k,j]=log(r,base=exp(1))}
}

save(logr_s_cv,logr_s_gcv,logr_s_aicc,logr_s_risk,file="svresult.rda")


## variation function

x_i = (c(1:200)-.5)/200
phi = function(u){
       1/sqrt(2*pi)*exp(-u^2/2)}
```

```
f=function(x){
1.5*phi((x-0.35)/0.15)-phi((x-0.8)/0.04)}
fv=function(x,j){
(0.15*(1+0.4*(2*j-7)*(x-0.5)))^2}

##Variance function
yv=matrix(0,200,6)
e=rnorm(200)
fv=function(x,j){
(0.15*(1+0.4*(2*j-7)*(x-0.5)))^2}
for (j in 1:6){
yv[,j]=f(x_i)+sqrt(fv(x_i,j))*e}



## Design matrix   ##  t : knot  data-->y
t<-(1:30)/31
designmatrix=function(data,t){
f=function(a){
b=a*rep(1,30)-t
pos=((abs(b)+b)/2)^3
f=c(1,a,a^2,a^3,pos)
}
designmatrix=t(sapply(data,f))
}

## Hat matrix
hatmatrix=function(designmatrix,lambda){
D=diag(c(rep(0,4),rep(1,30)))
hatmatrix=designmatrix%*%solve(t(designmatrix)%*%designmatrix+lambda*D)%*%t(designmatr
ix)
}

## CV selection criterion   ## lambda: smoothing parameter; response: y; X:design matr
ix
CV=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
fit=hat%*%y
correct=1-diag(hat)
s=(y-fit)/correct
CV=sum(s^2)
}

## GCV selection criterion
GCV=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
```

```
correct=1-sum(diag(hat))/n
fit=hat%*%y
GCV=sum((y-fit)/correct)^2
}
## corrected AIC selection criterion
AICC=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
fit=hat%*%y
resSum=sum((y-fit)^2)
tr=sum(diag(hat))
AICC=log(resSum)+2*(tr+1)/(n-tr-2)
}

## minimun risk selection criterion
Risk=function(lambda,response,X){
hat=hatmatrix(X,lambda)
y=response
n=length(y)
fit=hat%*%y
resSum=sum((y-fit)^2)
tr=sum(diag(hat))
sigma=resSum/(n-tr)
Risk=resSum+sigma^2*(2*tr-n)
}

##diff
diff=function(lambda,obs){sum((f(x_i)-hatmatrix(design,lambda)*obs)^2)}


logr_v_cv=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
yv=matrix(0,200,6)
e=rnorm(200)
yv[,j]=f(x_i)+sqrt(fv(x_i,j))*e
design=designmatrix(yv[,j],t)
lambdamin=optimize(CV,interval=c(1e-10,1), response=yv[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-10,1),obs=yv[,j])$objective
r=diff(lambdamin,yv[,j])/mindiff
logr_v_cv[k,j]=log(r,base=exp(1))}
}

logr_v_gcv=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
yv=matrix(0,200,6)
```

```
e=rnorm(200)
yv[,j]=f(x_i)+sqrt(fv(x_i,j))*e
design=designmatrix(yv[,j],t)
lambdamin=optimize(GCV,interval=c(1e-10,1), response=yv[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-10,1),obs=yv[,j])$objective
r=diff(lambdamin,yv[,j])/mindiff
logr_v_gcv[k,j]=log(r,base=exp(1))}
}

logr_v_aicc=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
yv=matrix(0,200,6)
e=rnorm(200)
yv[,j]=f(x_i)+sqrt(fv(x_i,j))*e
design=designmatrix(yv[,j],t)
lambdamin=optimize(AICC,interval=c(1e-10,1), response=yv[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-10,1),obs=yv[,j])$objective
r=diff(lambdamin,yv[,j])/mindiff
logr_v_aicc[k,j]=log(r,base=exp(1))}
}


logr_v_risk=matrix(0,100,6)
for (j in 1:6){
for (k in 1:100){
yv=matrix(0,200,6)
e=rnorm(200)
yv[,j]=f(x_i)+sqrt(fv(x_i,j))*e
design=designmatrix(yv[,j],t)
lambdamin=optimize(Risk,interval=c(1e-10,1), response=yv[,j],X=design)$minimum
hat=hatmatrix(design,lambdamin)
mindiff=optimize(diff,interval=c(1e-10,1),obs=yv[,j])$objective
r=diff(lambdamin,yv[,j])/mindiff
logr_v_risk[k,j]=log(r,base=exp(1))}
}
save(logr_v_cv,logr_v_gcv,logr_v_aicc,logr_v_risk,file="vresult.rda")

# boxplot

load("noiseresults.rda")
par(mfrow=c(2,3))
for (i in 1:6){
boxplot(logr_n_cv[,i],logr_n_gcv[,i],logr_n_aicc[,i],logr_n_risk[,i],main=paste("j=",
i))}
#boxplot(logr_n_cv[,i],logr_n_gcv[,i],logr_n_aicc[,i],logr_n_risk[,i], ylim=c(0,0.000
2),main=paste("j=",i))}
```

```
load("svresult.rda")
par(mfrow=c(2,3))
for (i in 1:6){
boxplot(logr_s_cv[,i],logr_s_gcv[,i],logr_s_aicc[,i],logr_s_risk[,i],ylim=c(0,0.0000
3),main=paste("j=",i))}

load("vresult.rda")
par(mfrow=c(2,3))
for (i in 1:6){
boxplot(logr_v_cv[,i],logr_v_gcv[,i],logr_v_aicc[,i],logr_v_risk[,i],ylim=c(0,0.0000
3),main=paste("j=",i))}

## more details

# data generation
x = (c(1:200)-.5)/200
f = function(x) 1.5*0.15*dnorm(x,0.35,0.15)-0.04*dnorm(x,0.8,0.04)

gen.data = function(x,n,factor=1,f){
    f = match.fun(f)
    e = rnorm(n)
    if (factor == 1) {
        sigma = 0.02+0.04*(1:6-1)^2
        y = f(x) + outer(e,sigma)
        ty = f(x) + outer(e,sigma)*0
    }
    else if (factor == 2) {
        u = sort(runif(n))
        x = sapply(1:6, function(z) qbeta(u,(z+4)/5,(11-z)/5))
        y = f(x) + 0.1*e
        ty = f(x) + e*0
    }
    else if (factor == 3) {
        f = function(x,j) sqrt(x*(1-x))*sin(2*pi*(1+2^((9-4*j)/5))/(x+2^((9-4*j)/5)))
        y = sapply(1:6, function(j) f(x,j)) + 0.2*e
        ty = sapply(1:6, function(j) f(x,j)) + e*0
    }
    else if (factor == 4) {
        v = function(x,j) (0.15*(1+0.4*(2*j-7)*(x-0.5)))^2
        y = f(x) + sqrt(sapply(1:6, function(j) v(x,j)))*e
        ty = f(x) + sqrt(sapply(1:6, function(j) v(x,j)))*e*0
    } else stop('incorrect specify the factor')
    data = list(x=x,y=y,ty=ty)
    return(data)
}

# load and plot
```

```
load('nl.RData')

data = gen.data(x,200,1,f) # generate data with different settings
png('nl.png',width=980,height=780)
par(mfrow=c(3,4))
if (is.matrix(data$x)) {
    for (i in 1:6) {
        colnames(nl.list[[i]]) = c('CV','GCV','AICC','risk1','risk2')
        plot(data$x[,i],data$y[,i],cex=0.1,xlab='',ylab='')
        lines(data$x[,i],data$ty[,i])
        boxplot(nl.list[[i]],ylim=c(0,1.5),main=paste('j=',i,sep=''))
    }
} else {
    for (i in 1:6) {
        colnames(nl.list[[i]]) = c('CV','GCV','AICC','risk1','risk2')
        plot(data$x,data$y[,i],cex=0.1,xlab='',ylab='')
        lines(data$x,data$ty[,i])
        boxplot(nl.list[[i]],ylim=c(0,1.5),main=paste('j=',i,sep=''))
    }
}
dev.off()

load('dd.RData')

data = gen.data(x,200,2,f) # generate data with different settings
png('dd.png',width=980,height=780)
par(mfrow=c(3,4))
if (is.matrix(data$x)) {
    for (i in 1:6) {
        colnames(dd.list[[i]]) = c('CV','GCV','AICC','risk1','risk2')
        plot(data$x[,i],data$y[,i],cex=0.1,xlab='',ylab='')
        lines(data$x[,i],data$ty[,i])
        boxplot(dd.list[[i]],ylim=c(0,1.5),main=paste('j=',i,sep=''))
    }
} else {
    for (i in 1:6) {
        colnames(dd.list[[i]]) = c('CV','GCV','AICC','risk1','risk2')
        plot(data$x,data$y[,i],cex=0.1,xlab='',ylab='')
        lines(data$x,data$ty[,i])
        boxplot(dd.list[[i]],ylim=c(0,1.5),main=paste('j=',i,sep=''))
    }
}
dev.off()

load('sv.RData')

data = gen.data(x,200,3,f) # generate data with different settings
png('sv.png',width=980,height=780)
par(mfrow=c(3,4))
```

```
if (is.matrix(data$x)) {
    for (i in 1:6) {
        colnames(sv.list[[i]]) = c('CV','GCV','AICC','risk1','risk2')
        plot(data$x[,i],data$y[,i],cex=0.1,xlab='',ylab='')
        lines(data$x[,i],data$ty[,i])
        boxplot(sv.list[[i]],ylim=c(0,1.5),main=paste('j=',i,sep=''))
    }
} else {
    for (i in 1:6) {
        colnames(sv.list[[i]]) = c('CV','GCV','AICC','risk1','risk2')
        plot(data$x,data$y[,i],cex=0.1,xlab='',ylab='')
        lines(data$x,data$ty[,i])
        boxplot(sv.list[[i]],ylim=c(0,1.5),main=paste('j=',i,sep=''))
    }
}
dev.off()

load('vf.RData')

data = gen.data(x,200,4,f) # generate data with different settings
png('vf.png',width=980,height=780)
par(mfrow=c(3,4))
if (is.matrix(data$x)) {
    for (i in 1:6) {
        colnames(vf.list[[i]]) = c('CV','GCV','AICC','risk1','risk2')
        plot(data$x[,i],data$y[,i],cex=0.1,xlab='',ylab='')
        lines(data$x[,i],data$ty[,i])
        boxplot(vf.list[[i]],ylim=c(0,1.5),main=paste('j=',i,sep=''))
    }
} else {
    for (i in 1:6) {
        colnames(vf.list[[i]]) = c('CV','GCV','AICC','risk1','risk2')
        plot(data$x,data$y[,i],cex=0.1,xlab='',ylab='')
        lines(data$x,data$ty[,i])
        boxplot(vf.list[[i]],ylim=c(0,1.5),main=paste('j=',i,sep=''))
    }
}
dev.off()
```