Correction du TP2 : Estimation dans le modèle lineaire gaussien de grande dimension

M2 Statistique, 2019-2020

Statistique mathématique en grande dimension et applications, cours de M. Roquain

Exercice 1

1)

```
library(glmnet)
```

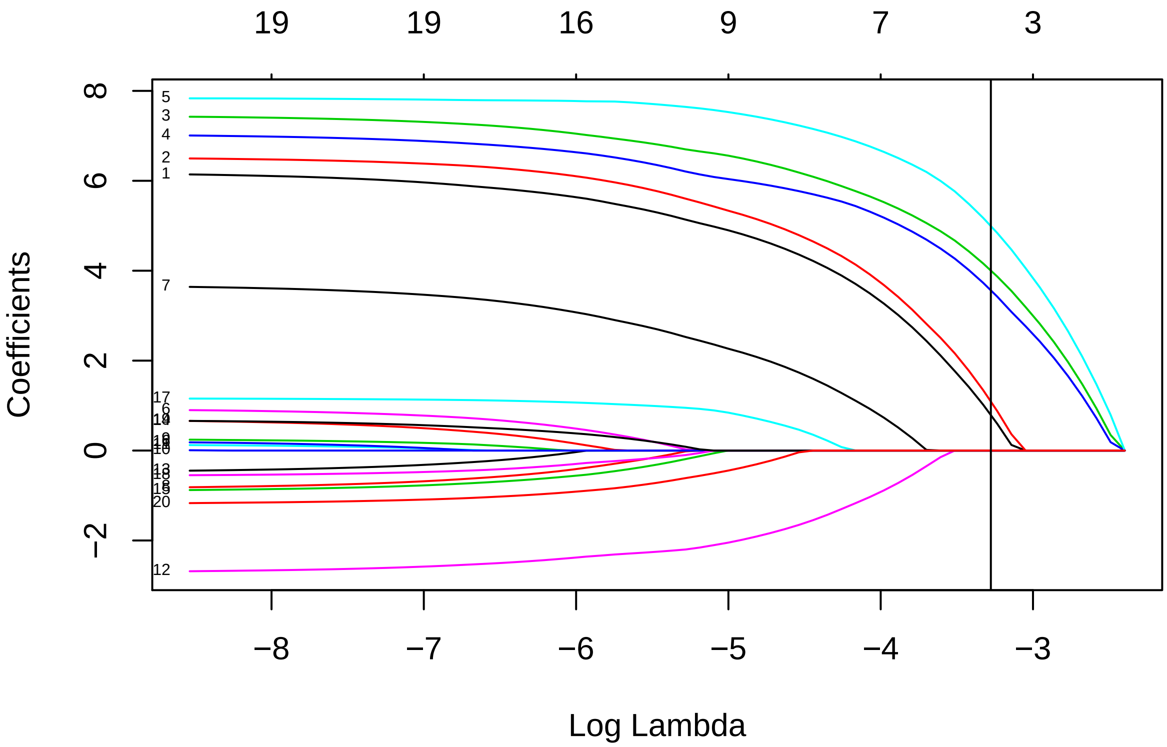
```
## Warning: package 'glmnet' was built under R version 3.3.2
## Loading required package: Matrix
## Warning: package 'Matrix' was built under R version 3.3.2
## Loading required package: foreach
## Loaded glmnet 2.0-10
```

```
n=102
p=20
X = matrix(rnorm(n*p),n,p)
normeX=apply(X,2,function(col) sum(col^2))
X = X%*%diag(1/sqrt(normeX))
apply(X,2,function(col) sum(col^2))[1:10]
```

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

2)

```
s=5
delta=7
beta=c(rep(delta,s),rep(0,p-s))
Y=rnorm(n)+X%*%matrix(beta,p,1)
fit=glmnet(X,Y,standardize=FALSE,intercept=FALSE)
plot(fit,xvar="lambda",label=TRUE)
lambdath=3*sqrt(2*(1.1)*log(p))/(2*n)
abline(v=log(lambdath),lwd=1)
```




```r
betachap=as.vector(coef(fit,s=lambdath)[-1])
which(betachap!=0)
```

```
## [1] 1 2 3 4 5
```

```r
which(beta!=0)
```

```
## [1] 1 2 3 4 5
```

```r
beta
```

```
##  [1] 7 7 7 7 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

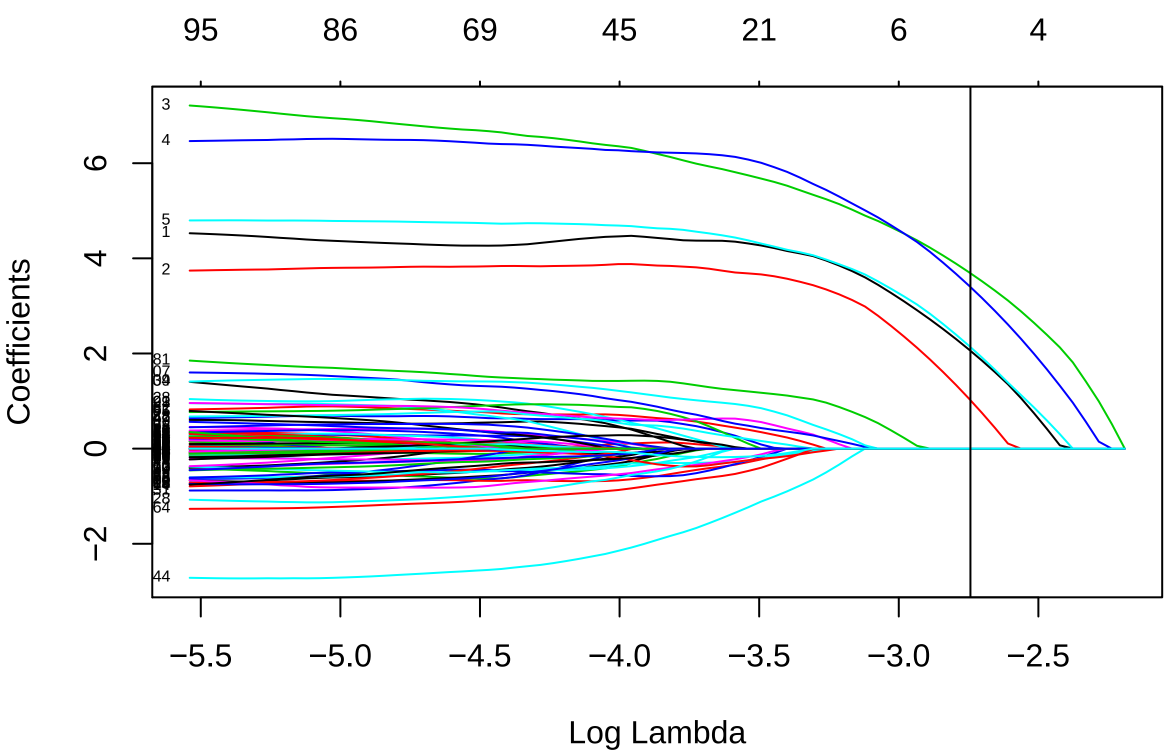
```r
betachap
```

```
##  [1] 0.7926171 1.0944123 4.0057249 3.5664047 4.9953394 0.0000000 0.0000000
##  [8] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
## [15] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
```

On voit qu'on sélectionne bien les 5 coordonnées de départ avec grande probabilité.

3)

```r
n=102
p=6033
X = matrix(rnorm(n*p),n,p)
normeX=apply(X,2,function(col) sum(col^2))
X = X%*%diag(1/sqrt(normeX))
delta=7
beta=c(rep(delta,s),rep(0,p-s))
Y=rnorm(n)+X%*%matrix(beta,p,1)
fit=glmnet(X,Y,standardize=FALSE,intercept=FALSE)
plot(fit,xvar="lambda",label=TRUE)
lambdath=3*sqrt(2*(1.1)*log(p))/(2*n)
abline(v=log(lambdath),lwd=1)
```




```
betachap=as.vector(coef(fit,s=lambdath)[-1])
which(betachap!=0)
```

```
## [1] 1 2 3 4 5
```

```
which(beta!=0)
```

```
## [1] 1 2 3 4 5
```

On voit que le LASSO s'en sort encore bien alors que la dimension est très grande. Notons que tout de même la force du signal ($\delta = 7$) est très grande. Le λ théorique est cependant un peu grand.

4)

```
library(sda)
```

```
## Loading required package: entropy
```

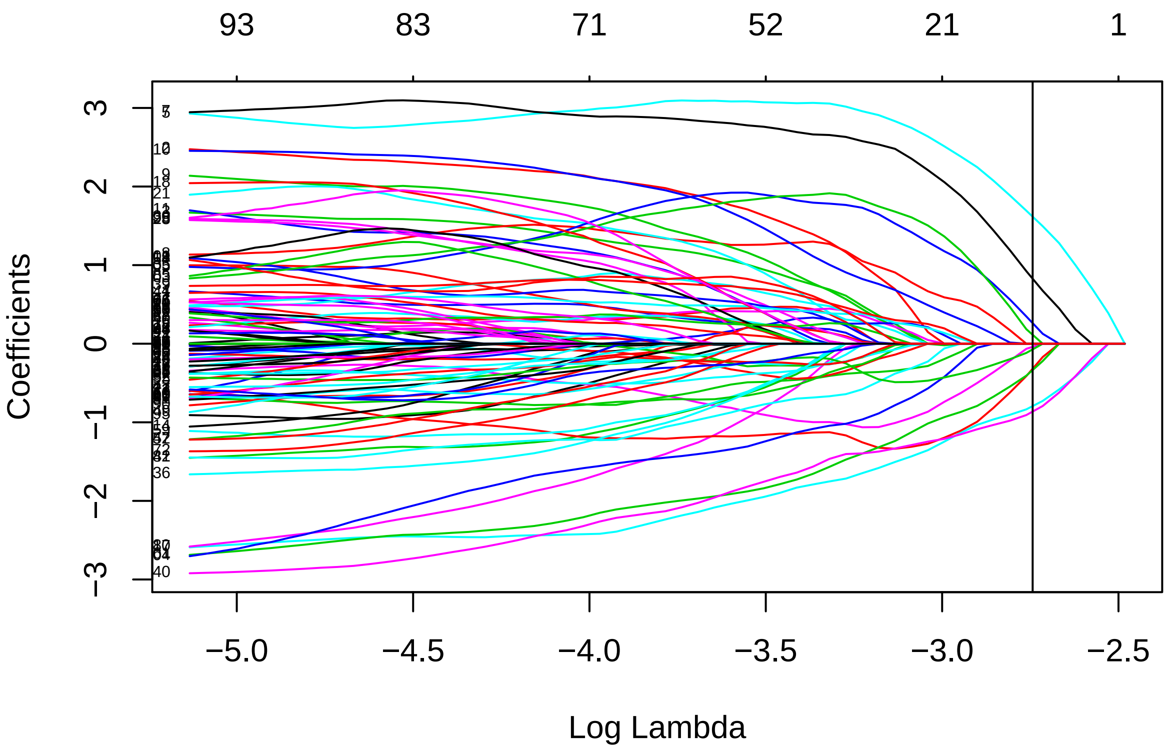
```
## Loading required package: corpcor
```

```
## Loading required package: fdrtool
```

```
data(singh2002)
X=singh2002$x
n=dim(X)[1]
p=dim(X)[2]
normeX=apply(X,2,function(col) sum(col^2))
X = X%*%diag(1/sqrt(normeX))
delta=7
s=10
beta=c(rep(delta,s),rep(0,p-s))
Y=rnorm(n)+X%*%matrix(beta,p,1)
fit=glmnet(X,Y,standardize=FALSE,intercept=FALSE)
```

```
plot(fit,xvar="lambda",label=TRUE)

lambdath=3*sqrt(2*(1.1)*log(p))/(2*n)
abline(v=log(lambdath),lwd=1)
```




```
betachap=as.vector(coef(fit,s=lambdath)[-1])
which(betachap!=0)
```

```
##  [1]    5    7    8   17 1232 1411 1905 2361 2440 2903 3752
```

```
which(beta!=0)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

On observe un comportement similaire à ce qui est au dessus. Notons que c'est assez fort car X n'est pas du tout orthogonale ici. Le λ théorique est cependant un peu grand.

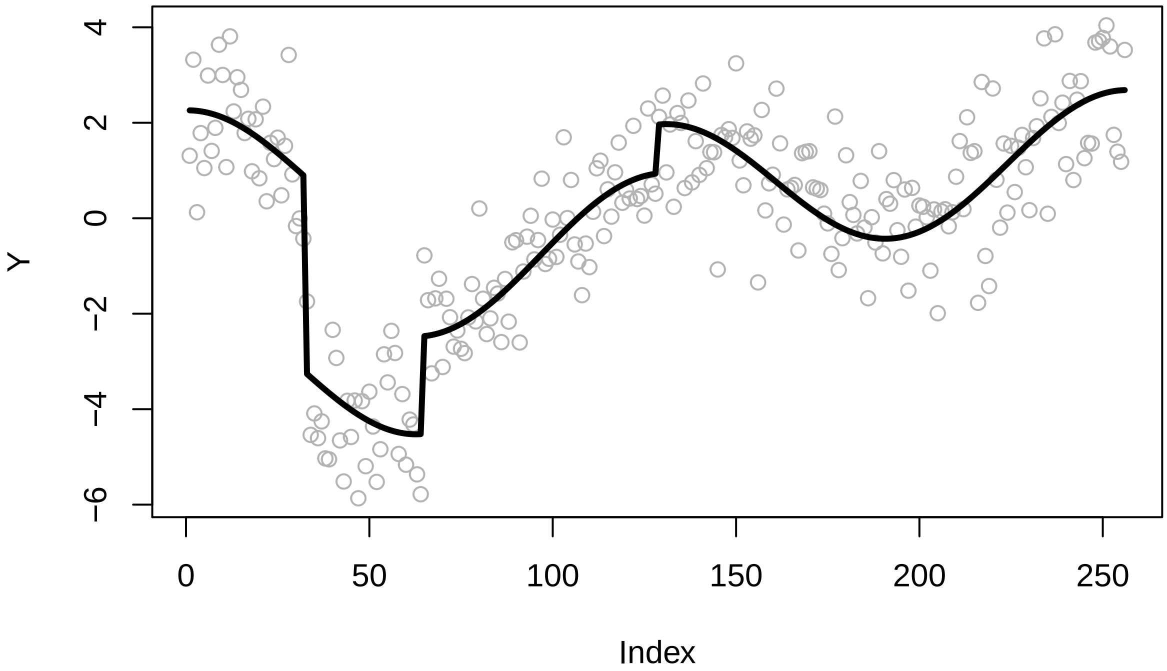
Exercice 2

```
load("dataforTP2")
n=2^8
C=t(ComputeCosMat(n))
H=t(ComputeHaarMat(n))
X=cbind(H[,2:n],C[,2:n])
dim(X)
```

```
## [1] 256 510
```

```
f=2*fmixte
Y=f + rnorm(n)
```

```r
plot(Y,col=gray(0.70))
lines(f,lwd=3)
```



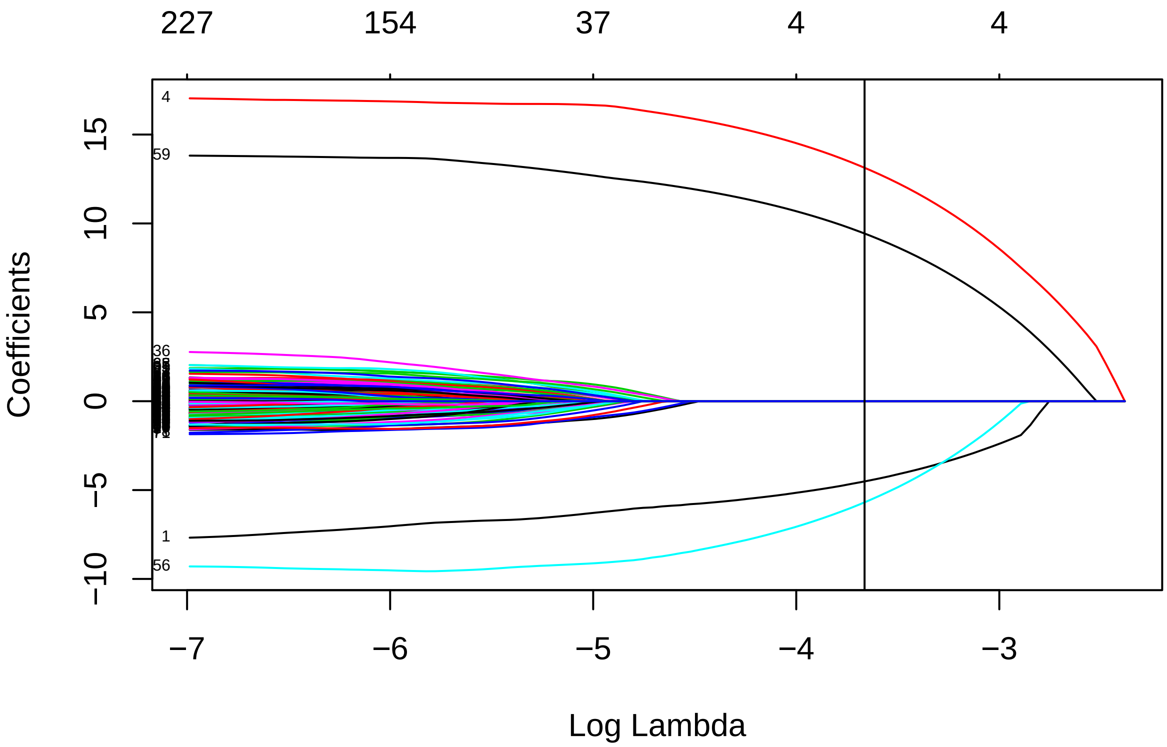
1) La matrice X n'a pas ses colonnes orthogonales puisque $p > n$. On peut s'en convaincre numériquement:

```r
sum(C[,2]*H[,2])
```

```
## [1] 0.900322
```

2)

```r
fit=glmnet(X,Y,standardize=FALSE,intercept=FALSE)
plot(fit,xvar="lambda",label=TRUE)
lambdath=3*sqrt(2*1.1*log(p))/(2*n)
abline(v=log(lambdath),lwd=1)
```

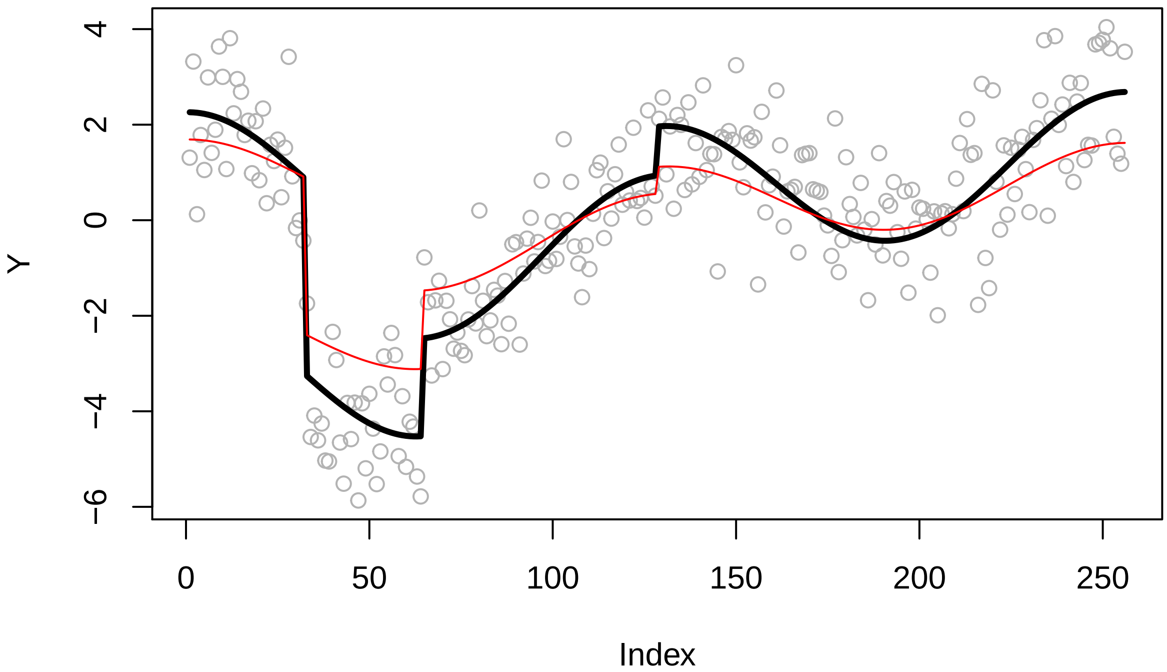



```
betachap=as.vector(coef(fit,s=lambdath)[-1])
which(betachap!=0)
```

```
## [1]   1   4 256 259
```

3)

```
fchap=X%*%betachap
plot(Y,col=gray(0.70))
lines(f,lwd=3)
lines(fchap,lwd=1,col="red")
```

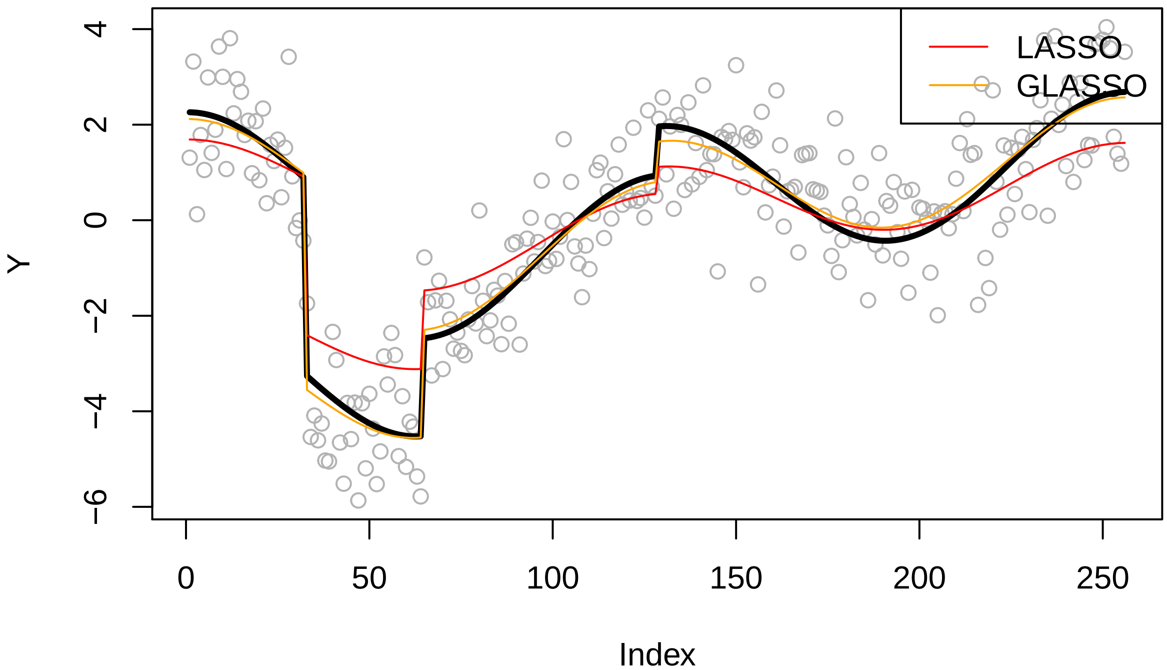


On voit que l'estimateur LASSO à l'air de sélectionner le bon support ici, mais il shrinke les coefficients. C'est la nature même de la norme ℓ^1, qui a utilisé le shrinkage pour mettre à zero les coefficients pas loin de 0.

4)

```
mhat=which(betachap!=0)
betaOLSmhat=lm(Y~X[,mhat]-1)$coefficients
fchapGLASSO=X[,mhat]%*%as.matrix(betaOLSmhat,length(mhat),1)
plot(Y,col=gray(0.70))
lines(f,lwd=3)
lines(fchap,lwd=1,col="red")
lines(fchapGLASSO,col="orange")

legend("topright",c("LASSO","GLASSO"),col=c("red","orange"),lwd=c(1,1))
```



Ici Gauss-LASSO marche très bien, car le support sélectionné par le LASSO est le bon, donc il s'agit de l'estimateur des moindres carrés sur le bon modèle, qui a des performances idéales.