

# TP STATISTIQUE EN GRANDE DIMENSION

AGBLODOE Komi/ M2 SSD

7 janvier 2020

## TP 1

Soit  $X$  la matrice de données de taille  $n.p$ . Dans ce TP, il s'agit de montrer empiriquement qu'il y a plusieurs inconvénients à utiliser le maximum de vraisemblance lorsqu'on est en présence de données en grande dimension.

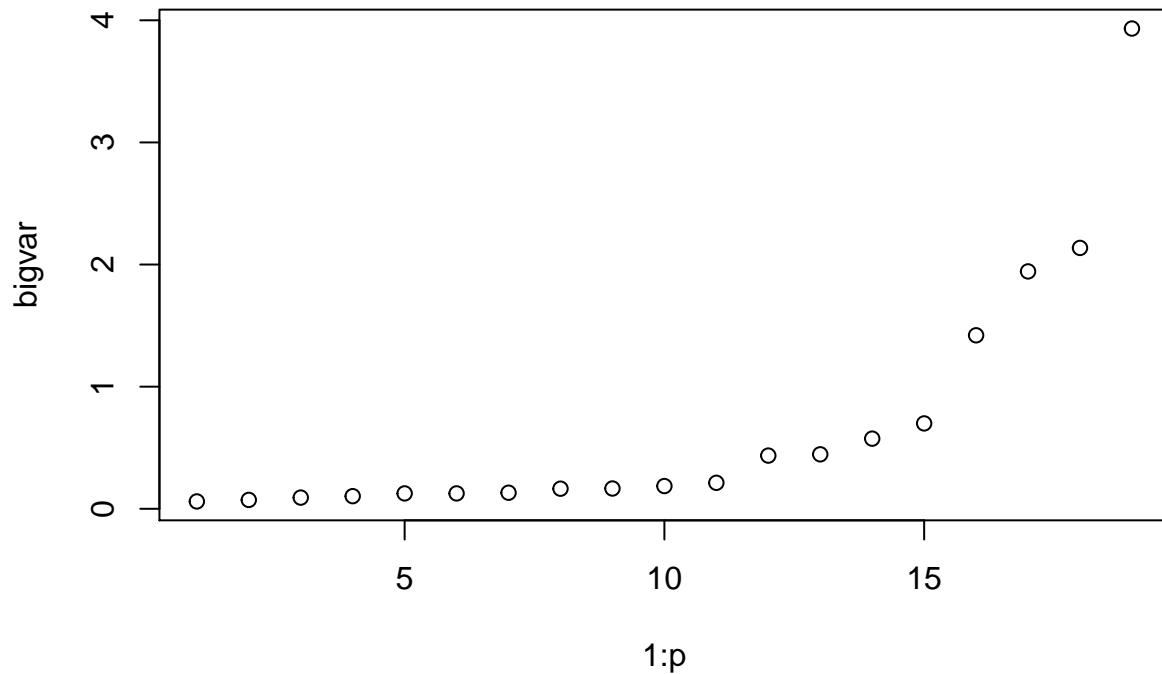
### 1

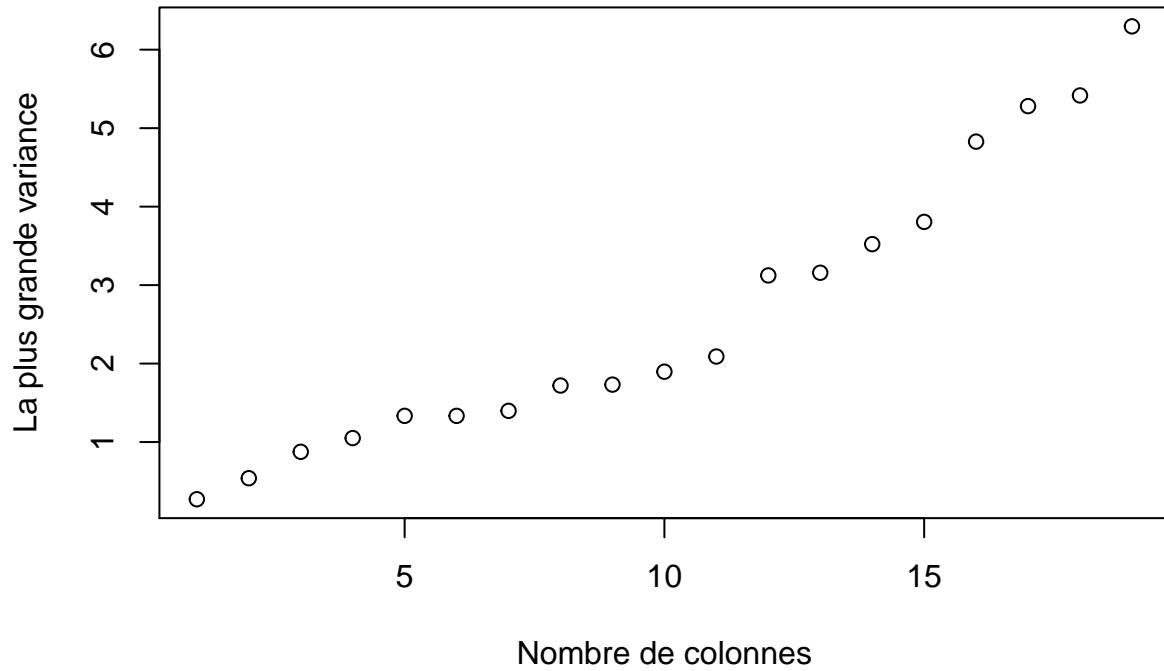
Ici on génère une matrice  $X$  obtenue à partir de la loi normale standard de taille  $n = 20$  pour  $p = 19$  co-variables.

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] -0.6264538  0.91897737 -0.1645236  2.40161776 -0.5686687 -0.62036668
## [2,]  0.1836433  0.78213630 -0.2533617 -0.03924000 -0.1351786  0.04211587
## [3,] -0.8356286  0.07456498  0.6969634  0.68973936  1.1780870 -0.91092165
## [4,]  1.5952808 -1.98935170  0.5566632  0.02800216 -1.5235668  0.15802877
## [5,]  0.3295078  0.61982575 -0.6887557 -0.74327321  0.5939462 -0.65458464
## [6,] -0.8204684 -0.05612874 -0.7074952  0.18879230  0.3329504  1.76728727
##           [,7]      [,8]      [,9]      [,10]     [,11]     [,12]
## [1,] -0.5059575 -1.9143594  0.4251004 -1.2313234  0.4094018 -1.73321841
## [2,]  1.3430388  1.1765833 -0.2386471  0.9838956  1.6888733  0.00213186
## [3,] -0.2145794 -1.6649724  1.0584830  0.2199248  1.5865884 -0.63030033
## [4,] -0.1795565 -0.4635304  0.8864227 -1.4672500 -0.3309078 -0.34096858
## [5,] -0.1001907 -1.1159201 -0.6192430  0.5210227 -2.2852355 -1.15657236
## [6,]  0.7126663 -0.7508190  2.2061025 -0.1587546  2.4976616  1.80314191
##           [,13]     [,14]     [,15]     [,16]     [,17]     [,18]
## [1,]  0.7073107  0.9510128  0.3981302  0.8936737  1.3079015  0.7395892
## [2,]  1.0341077 -0.3892372 -0.4075286 -1.0472981  1.4970410 -1.0634574
## [3,]  0.2234804 -0.2843307  1.3242586  1.9713374  0.8147027  0.2462108
## [4,] -0.8787076  0.8574098 -0.7012317 -0.3836321 -1.8697888 -0.2894994
## [5,]  1.1629646  1.7196273 -0.5806143  1.6541453  0.4820295 -2.2648894
## [6,] -2.0001649  0.2700549 -1.0010722  1.5122127  0.4561356 -1.4088505
##           [,19]
## [1,] -2.5923277
## [2,]  1.3140022
## [3,] -0.6355430
## [4,] -0.4299788
## [5,] -0.1693183
## [6,]  0.6122182
```

**2**

Ensuite on trace le maximum des variances des estimateurs des coefficients, du modèle linéaire en fonction du nombre de co-variables  $p$ .





```
## [1] 11.58674
```

On a utilisé le log pour mieux voir l'augmentation des variances.

### 3

On remarque que quand  $p$  s'approche de  $n$ , la variance augmente largement de valeur. On est donc en présence d'une non convergence.

### 4

$MSE = Variance + (Bias)^2$  Pour un bon modèle, il faut que le MSE tende vers 0. On a vu que lorsque le nombre de co-variables devient grand, la variance augmente. Ce qui fait que le MSE ne tend pas vers 0. Ceci explique les limites de ces modèles en grande dimension.

## Reprendons l'exo avec un $p$ plus grand que $n$

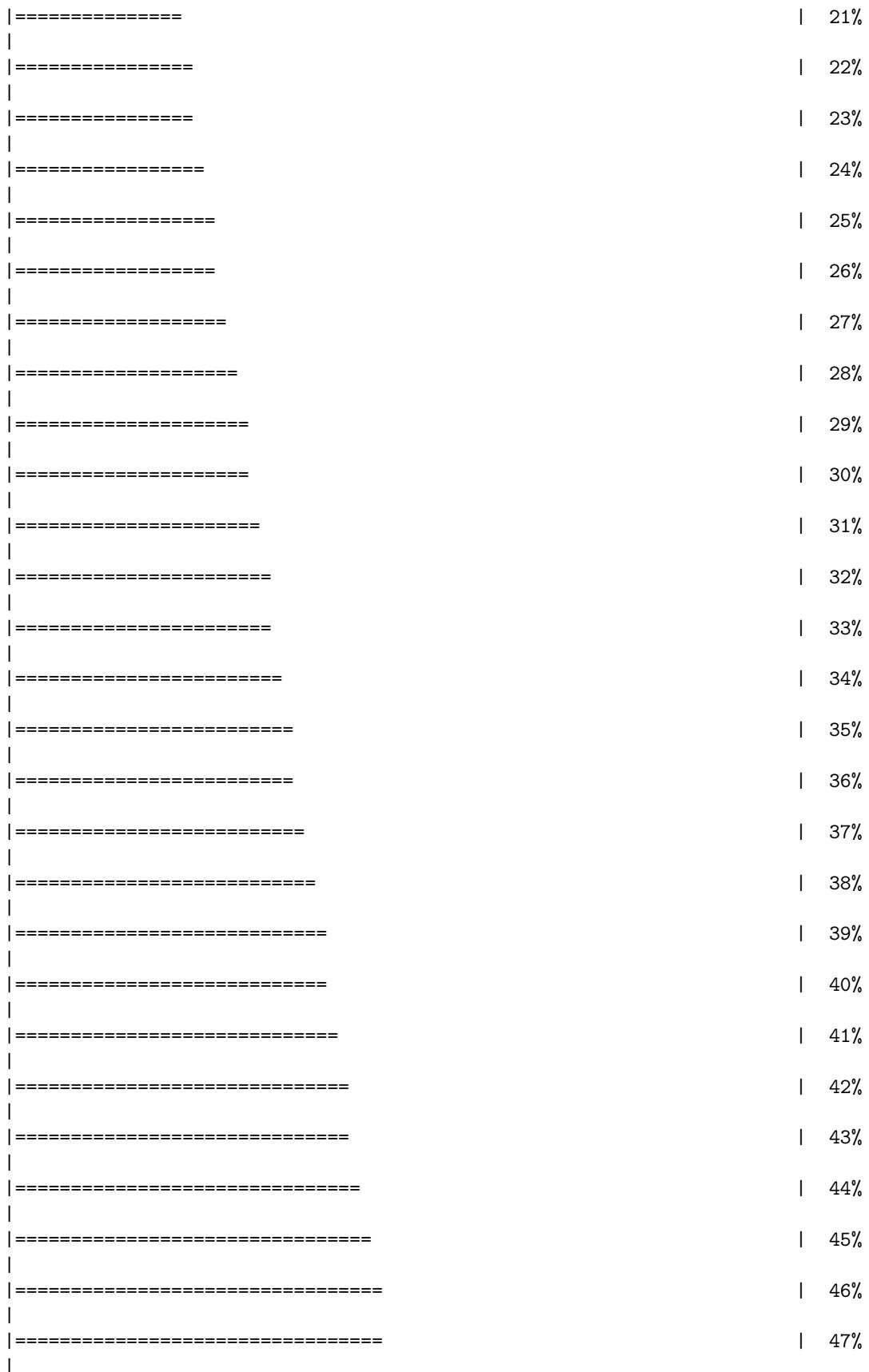
Pour  $p$  supérieur à  $n$ , cela ne marche pas car la matrice n'est plus inversible. Il y a une colinéarité entre certaines colonnes.

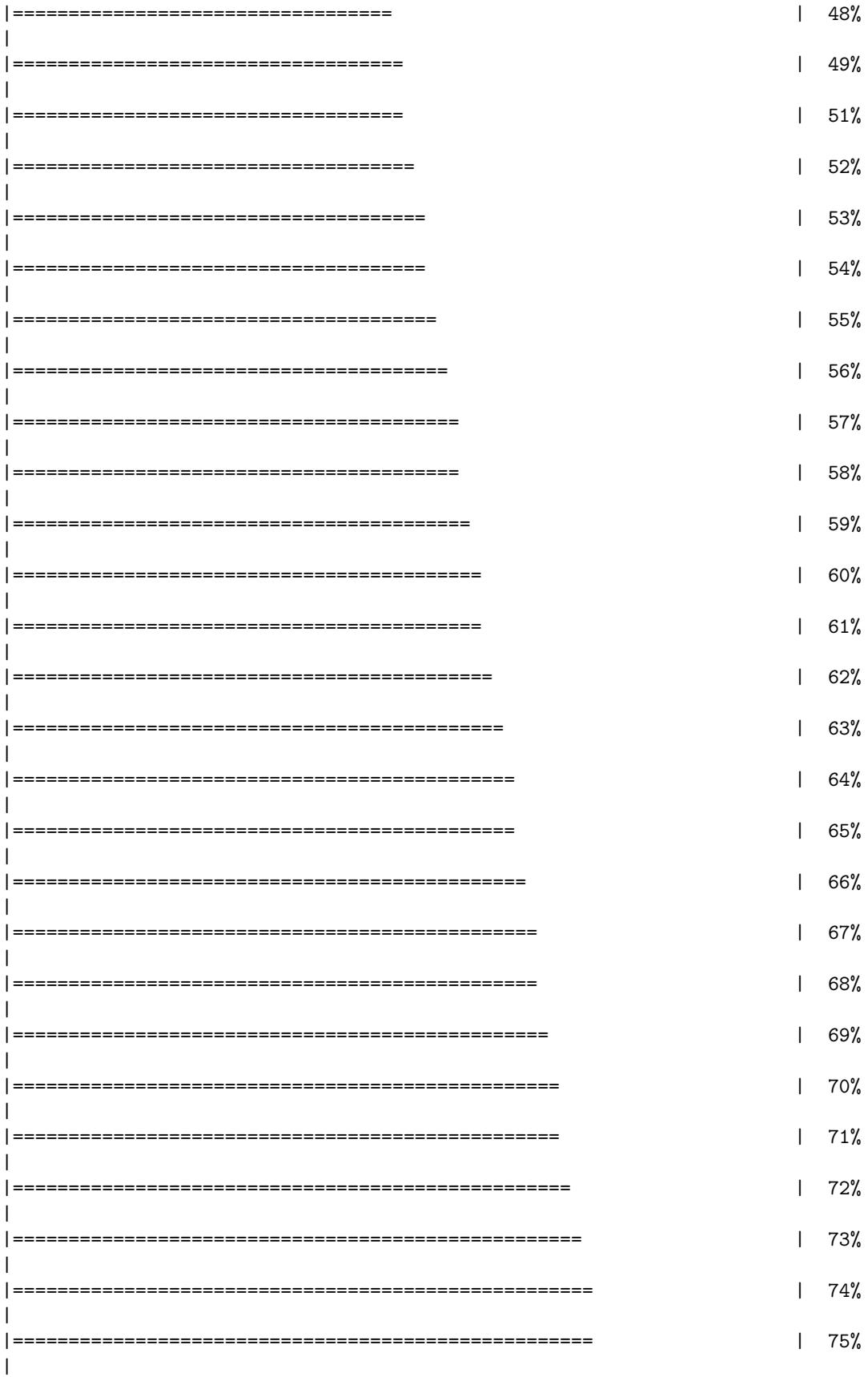
## 1.2 Problème de sélection de variables

1

Ici, on considère une matrice de données  $X$  de  $n = 25$  observations sur  $p = 100$  co-variables distribuées aléatoirement et une matrice  $XX$  de même dimension mais distribuée uniformément sur  $(0,1)$ . Puis une variable d'intérêt  $y$  constituée de  $n$  nombres aléatoires indépendants et issus d'une loi normale centrée-réduite. On sélectionne un modèle avec seulement 5 variables par BIC et on répète ceci 100 fois et on observe les estimations  $\hat{\beta}$  de  $\beta$ .

```
##  
|  
|  
|  
|= | 0%  
|  
|= | 1%  
|= | 2%  
|== | 3%  
|--- | 4%  
|---- | 5%  
|---- | 6%  
|----- | 7%  
|----- | 8%  
|----- | 9%  
|----- | 10%  
|----- | 11%  
|----- | 12%  
|----- | 13%  
|----- | 14%  
|----- | 15%  
|----- | 16%  
|----- | 17%  
|----- | 18%  
|----- | 19%  
|----- | 20%
```

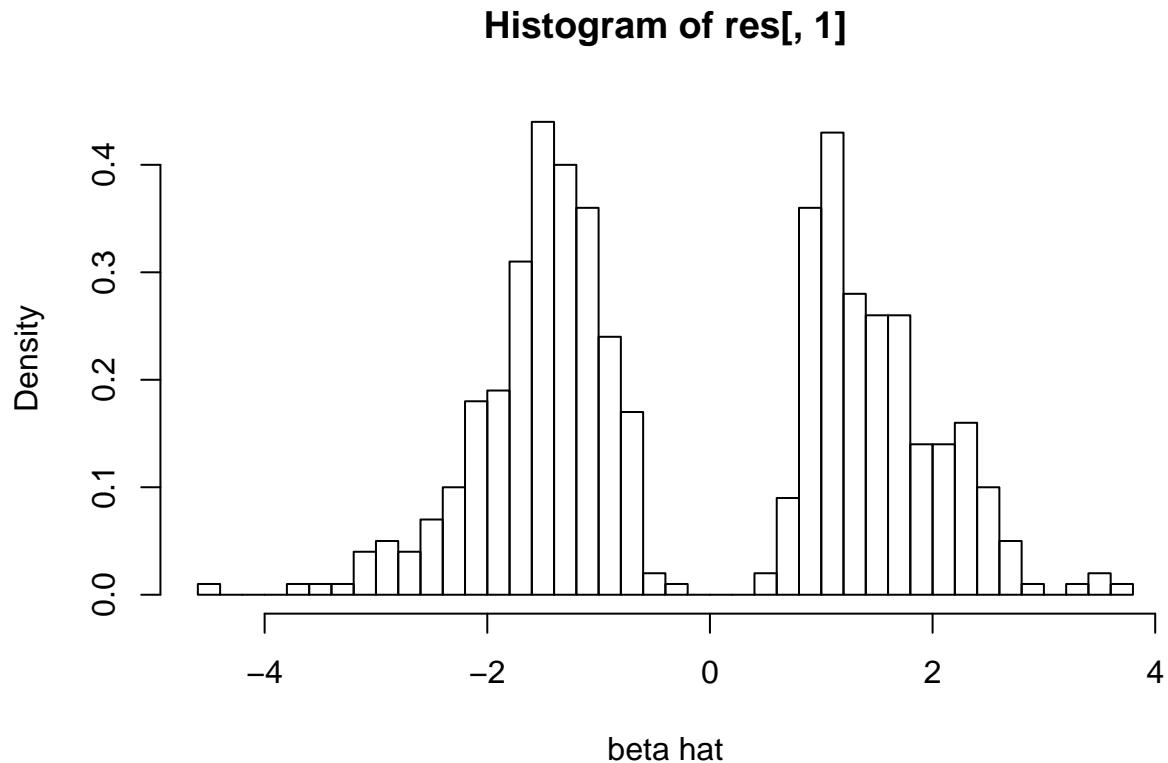




```
|=====| 76%
|=====| 77%
|=====| 78%
|=====| 79%
|=====| 80%
|=====| 81%
|=====| 82%
|=====| 83%
|=====| 84%
|=====| 85%
|=====| 86%
|=====| 87%
|=====| 88%
|=====| 89%
|=====| 90%
|=====| 91%
|=====| 92%
|=====| 93%
|=====| 94%
|=====| 95%
|=====| 96%
|=====| 97%
|=====| 98%
|=====| 99%
|=====| 100%
```

```
##      V54      V85      V63      V51      V72      V55
## -2.515185  1.529871 -1.645065  1.253193 -1.079182  2.575838
```

## 2 Tracé de l'histogramme du slide 17



Les valeurs estimées sont loin de réfléter la vraie valeur qui est égale à 0. La plupart de ces estimations tournent autour des valeurs +1.5 et -1.5. La distribution d'échantillonnage de  $\hat{\beta}$  est grossièrement déformée car nous avons utilisé l'ensemble des données pour la sélection du modèle. On est en présence d'un phénomène de surapprentissage puisque le vrai modèle dans ce cas est le modèle nul (intercept seulement). Donc nous pouvons dire que le critère de BIC ne peut pas être utilisé pour une sélection précise des variables dans les problèmes de grande dimension ou de malédictions de données.

## 3 Calculer les indicateurs du Slide 19 et d'autres slides

```
##      Min. 1st Qu. Median     Mean 3rd Qu.    Max.
## -4.4683 -1.5096 -0.7818 -0.1208  1.3370  3.7745
## [1] 5
## [1] 0.001316657
## [1] 3.347888e-10 3.824676e-01
## [1] 0.046
```

Le quartile est de -1.51 et le troisième quartile est de 1.337. En moyenne, les modèles sélectionnés ont obtenu une erreur quadratique moyenne (MSE) de 2.15.

## 1.3 DEVOIR

L'Analyse en composantes principales est une méthode statistique exploratoire permettant une description essentiellement graphique de l'information contenue dans des grands tableaux de données. Dans la plupart des applications, il s'agit d'étudier p variables mesurées sur un ensemble de n individus. Lorsque n et p sont grands on cherche à synthétiser la masse d'informations sous une forme exploitable et compréhensible.

1. Le problème majeur pouvant surgir dans une ACP en grande dimension concerne le choix du nombre d'axes à retenir pour l'interprétation. Les conséquences de ce choix sont importantes car si le nombre d'axes n'est pas correctement choisi, on peut introduire un bruit ou une perte d'informations importante dans l'analyse.
2. Une des étapes les plus importantes en statistique en grande dimension est la vérification des hypothèses et conclusions. Avec l'augmentation du nombre d'hypothèses, dont chacune est testée à un seuil  $\alpha$ , le risque d'erreur de rejet de l'hypothèse nulle augmente. Une analyse en grande dimension consiste souvent à considérer des centaines (voire des milliers) d'inférences.
3. Le *familiy-wise error rates* (FWER) est la probabilité de faire une ou plusieurs fausses découvertes, ou des erreurs de type I lors de l'exécution de tests d'hypothèses multiples. Les procédures de contrôle de type FWER ne sont efficaces que pour des petites familles d'inférences. Dans le cas de nombreuses familles d'inférences en grande dimension, il est plus efficace de contrôler par une méthode *false discovery rates* (FDR), c'est-à-dire la valeur attendue du taux de rejets erronés sur l'ensemble des rejets.

## TP2

### 2 - Sélection de modèle

#### 2-1 Les données

Ces données proviennent d'une étude qui a examiné la corrélation entre le niveau d'antigène spécifique de la prostate et un certain nombre de mesures cliniques chez les hommes sur le point de subir une prostatectomie radicale.

Téléchargement des données

```
## R Package to solve regression problems while imposing
##   an L1 constraint on the parameters. Based on S-plus Release 2.1
## Copyright (C) 1998, 1999
## Justin Lokhorst <jlokhors@stats.adelaide.edu.au>
## Berwin A. Turlach <bturlach@stats.adelaide.edu.au>
## Bill Venables <wvenable@stats.adelaide.edu.au>
##
## Copyright (C) 2002
## Martin Maechler <maechler@stat.math.ethz.ch>
```

Nous vérifions que les données correspondent bien au tableau de description et faisons quelques statistiques descriptives : dim, names, summary, cor, hist

```
##      lcavol          lweight           age            lbph
## Min. :-1.3471    Min. :2.375    Min. :41.00   Min. :-1.3863
## 1st Qu.: 0.5128   1st Qu.:3.376   1st Qu.:60.00   1st Qu.:-1.3863
## Median : 1.4469   Median :3.623   Median :65.00   Median : 0.3001
```

```

##  Mean    : 1.3500   Mean    :3.653   Mean    :63.87   Mean    : 0.1004
##  3rd Qu.: 2.1270   3rd Qu.:3.878   3rd Qu.:68.00   3rd Qu.: 1.5581
##  Max.    : 3.8210   Max.    :6.108   Max.    :79.00   Max.    : 2.3263
##      svi          lcp          gleason        pgg45
##  Min.    :0.0000   Min.    :-1.3863   Min.    :6.000   Min.    : 0.00
##  1st Qu.:0.0000   1st Qu.:-1.3863   1st Qu.:6.000   1st Qu.: 0.00
##  Median :0.0000   Median :-0.7985   Median :7.000   Median : 15.00
##  Mean    :0.2165   Mean    :-0.1794   Mean    :6.753   Mean    : 24.38
##  3rd Qu.:0.0000   3rd Qu.: 1.1787   3rd Qu.:7.000   3rd Qu.: 40.00
##  Max.    :1.0000   Max.    :2.9042   Max.    :9.000   Max.    :100.00
##      lpsa
##  Min.    :-0.4308
##  1st Qu.: 1.7317
##  Median : 2.5915
##  Mean    : 2.4784
##  3rd Qu.: 3.0564
##  Max.    : 5.5829

```

Les variables svi et gleason sont qualitatives.

```

## [1] 97 9

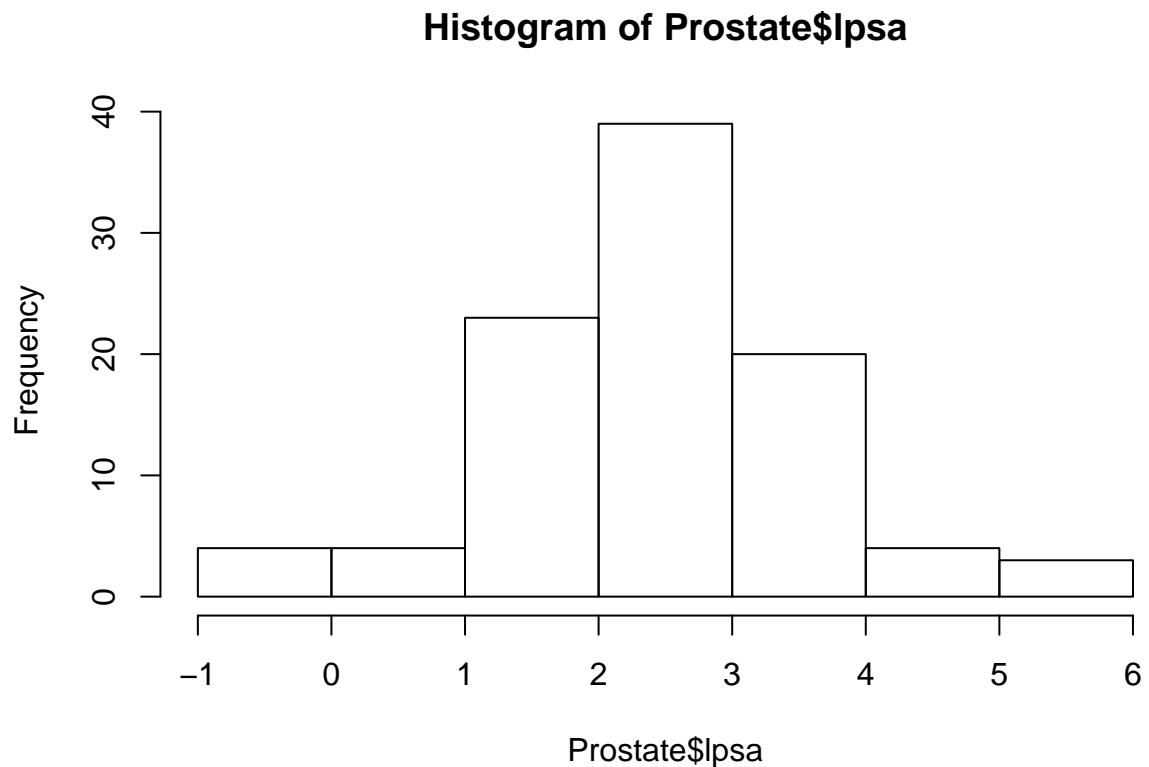
## [1] "lcavol"  "lweight" "age"       "lbph"     "svi"      "lcp"      "gleason"
## [8] "pgg45"   "lpsa"

##      lcavol        lweight        age         lbph        svi
##  Min.    :-1.3471   Min.    :2.375   Min.    :41.00   Min.    :-1.3863  0:76
##  1st Qu.: 0.5128   1st Qu.:3.376   1st Qu.:60.00   1st Qu.:-1.3863 1:21
##  Median : 1.4469   Median :3.623   Median :65.00   Median : 0.3001
##  Mean   : 1.3500   Mean   :3.653   Mean   :63.87   Mean   : 0.1004
##  3rd Qu.: 2.1270   3rd Qu.:3.878   3rd Qu.:68.00   3rd Qu.: 1.5581
##  Max.   : 3.8210   Max.   :6.108   Max.   :79.00   Max.   : 2.3263
##      lcp          gleason        pgg45        lpsa
##  Min.    :-1.3863   6:35     Min.    : 0.00   Min.    :-0.4308
##  1st Qu.:-1.3863   7:56     1st Qu.: 0.00   1st Qu.: 1.7317
##  Median :-0.7985   8: 1     Median : 15.00   Median : 2.5915
##  Mean   :-0.1794   9: 5     Mean   : 24.38   Mean   : 2.4784
##  3rd Qu.: 1.1787           3rd Qu.: 40.00   3rd Qu.: 3.0564
##  Max.   : 2.9042           Max.   :100.00   Max.   : 5.5829

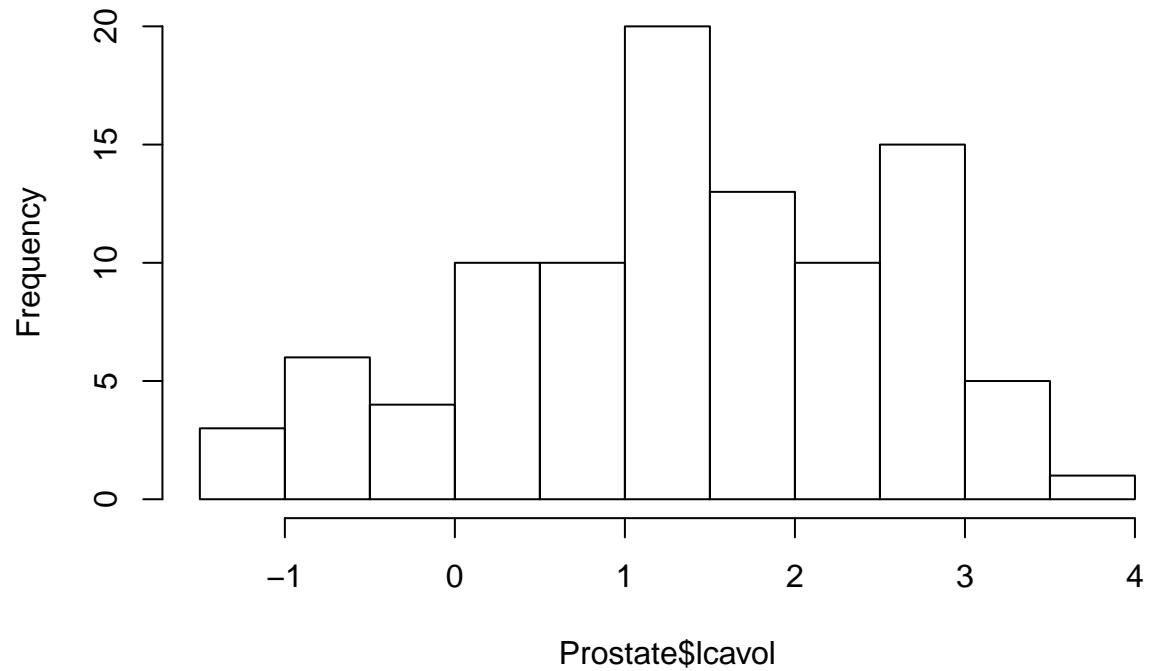
##      lcavol      lweight      age       lbph       lcp      pgg45
## lcavol  1.0000000 0.19412829 0.2249999  0.027349703 0.675310484 0.43365225
## lweight  0.1941283 1.00000000 0.3075286  0.434934636 0.100237795 0.05084682
## age     0.2249999 0.30752861 1.0000000  0.350185896 0.127667752 0.27611245
## lbph    0.0273497 0.43493464 0.3501859  1.000000000 -0.006999431 0.07846002
## lcp     0.6753105 0.10023780 0.1276678 -0.006999431 1.000000000 0.63152825
## pgg45  0.4336522 0.05084682 0.2761124  0.078460018 0.631528245 1.00000000
## lpsa   0.7344603 0.35412039 0.1695928  0.179809410 0.548813169 0.42231586
##      lpsa
## lcavol  0.7344603
## lweight 0.3541204
## age     0.1695928
## lbph    0.1798094

```

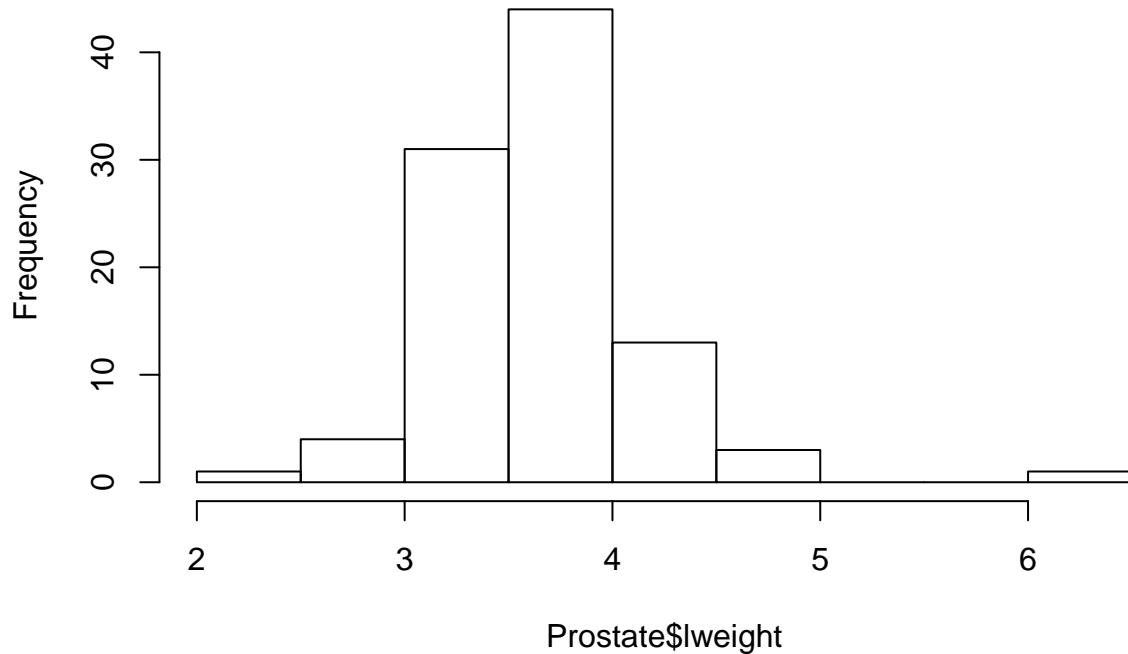
```
## lcp      0.5488132
## pgg45    0.4223159
## lpsa     1.0000000
```



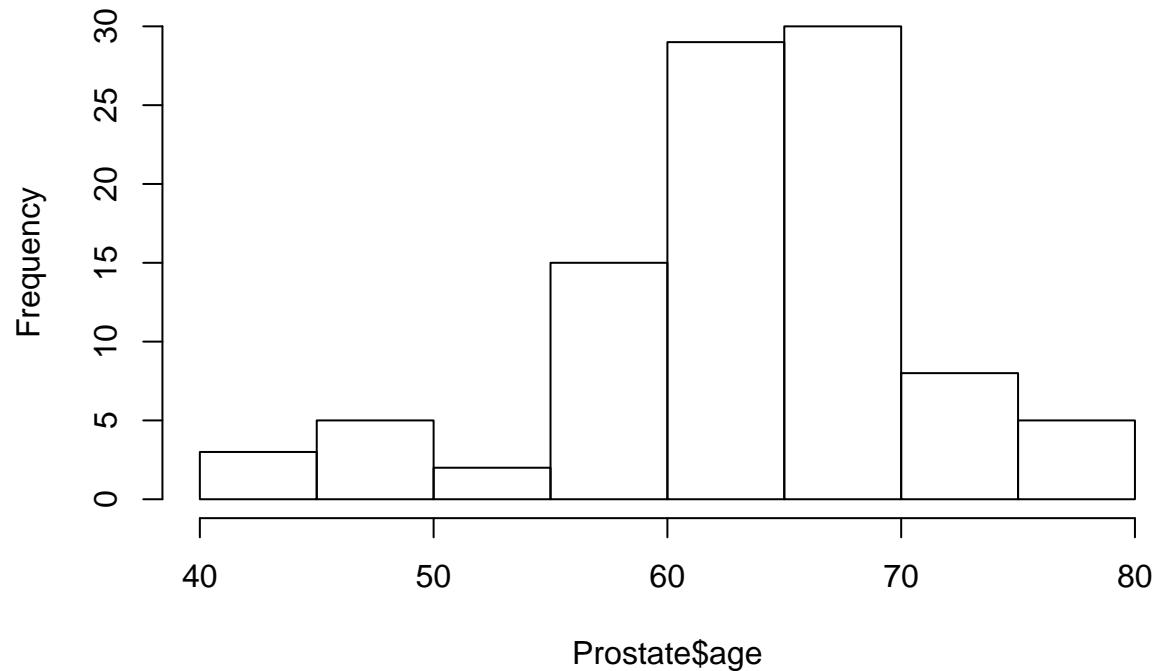
**Histogram of Prostate\$lcavol**



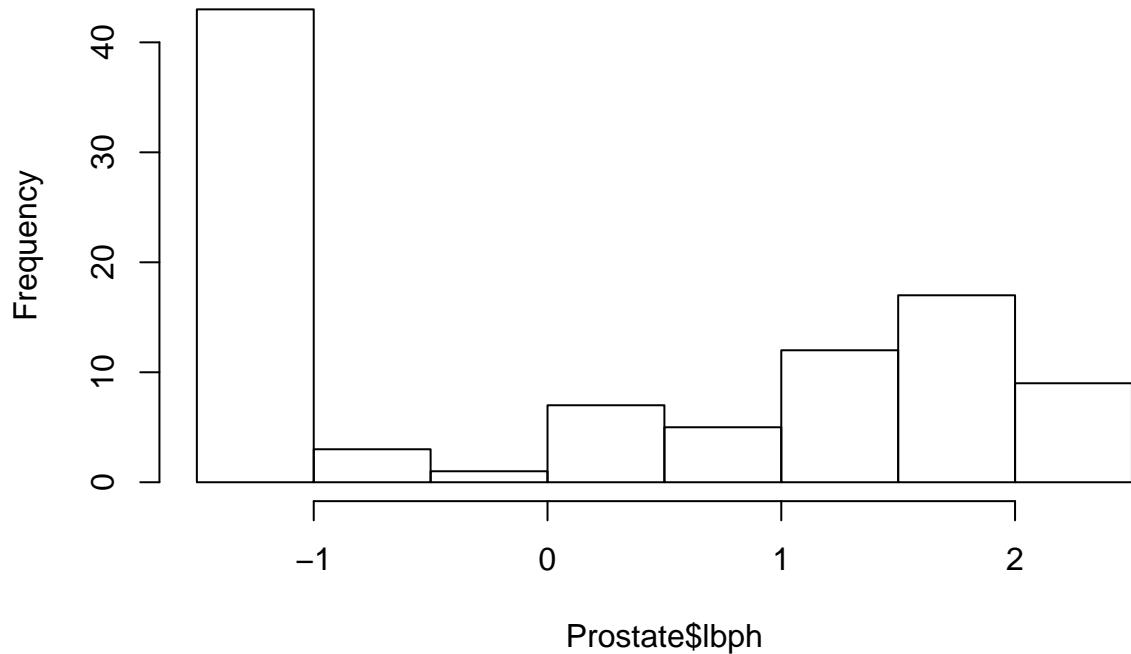
**Histogram of Prostate\$lweight**



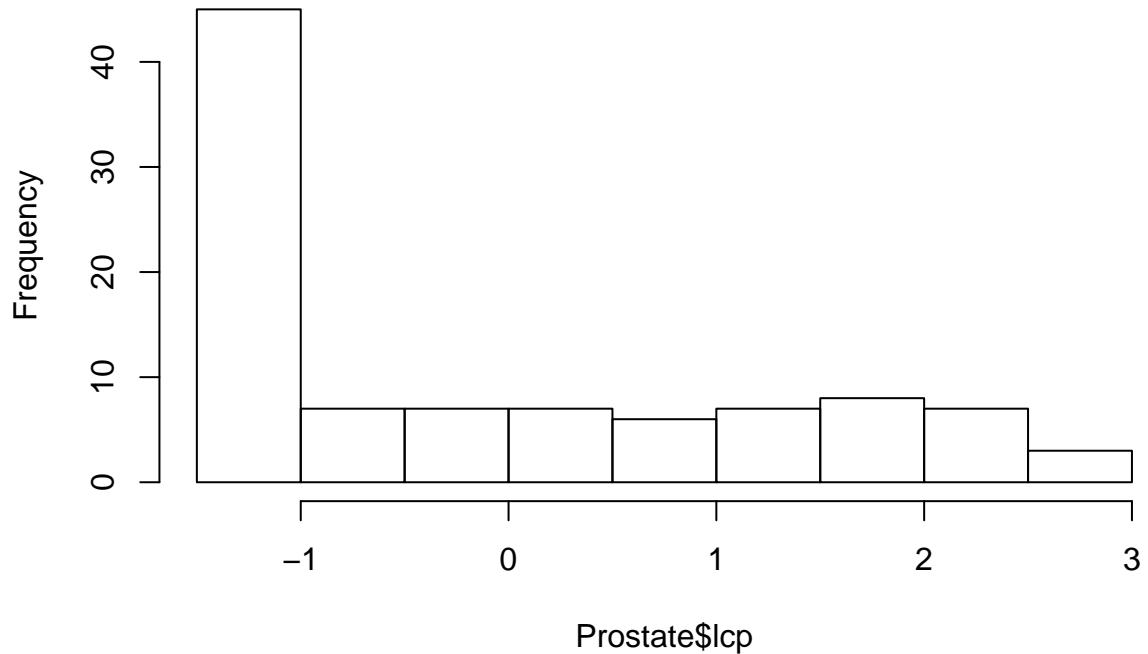
### Histogram of Prostate\$age



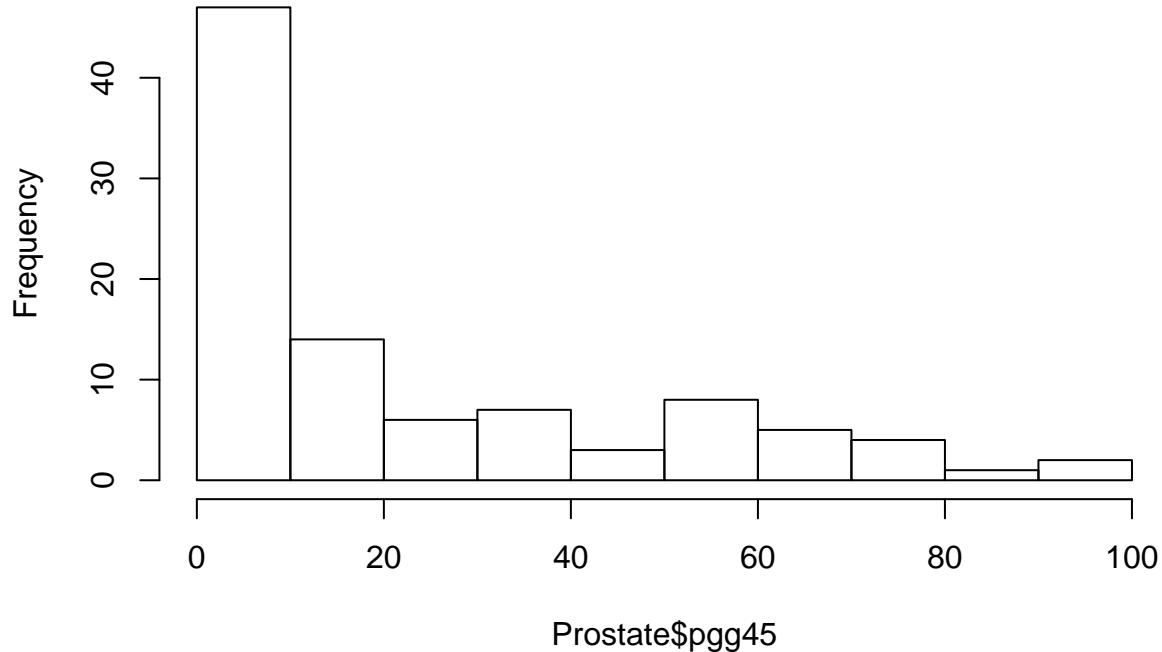
**Histogram of Prostate\$lbph**



**Histogram of Prostate\$lcp**



## Histogram of Prostate\$pgg45



Il s'agit d'une matrice de données comportant 97 lignes et 9 colonnes. On remarque aussi que la distribution des variables lpsa, lcavol, weight et âge semblent suivre une loi normale.

### Données train et test

Les valeurs de lpsa sont rangées par ordre croissant. Nous divisons ici le jeu de données en un échantillon d'apprentissage pour estimer les modèles et en un échantillon de test pour comparer les erreurs de prédiction. On conserve 1/4 des données pour l'échantillon de test.

On donne quelques statistiques descriptives des données : Prostate.app et Prostate.test

```
## [1] 22  9

## [1] 75  9

##      lcavol      lweight       age       lbph       svi
##  Min. :-1.0498  Min. 2.375   Min. 41.00  Min. -1.3863 0:57
##  1st Qu.: 0.5128 1st Qu.:3.376   1st Qu.:60.00 1st Qu.:-1.3863 1:18
##  Median : 1.4586  Median :3.593   Median :65.00  Median :-1.3863
##  Mean   : 1.3969  Mean   :3.607   Mean   :63.63  Mean   :-0.1226
##  3rd Qu.: 2.3491 3rd Qu.:3.858   3rd Qu.:68.50  3rd Qu.: 1.3737
##  Max.   : 3.8210  Max.   :4.780   Max.   :79.00  Max.   : 2.3263
##      lcp      gleason     pgg45      lpsa
##  Min. :-1.3863  6:28    Min. : 0.00  Min. :-0.4308
##  1st Qu.:-1.3863 7:42    1st Qu.: 0.00  1st Qu.: 1.7490
```

```

## Median : -0.7985   8: 1   Median : 15.00   Median : 2.5915
## Mean   : -0.1157   9: 4   Mean   : 25.05   Mean   : 2.5244
## 3rd Qu.: 1.3218          3rd Qu.: 40.00   3rd Qu.: 3.1751
## Max.   : 2.9042          Max.   : 100.00  Max.   : 5.5829

##      lcavol       lweight        age       lbph       svi
## Min. : -1.3471   Min. : 3.013   Min. : 56.00   Min. : -1.38629 0:19
## 1st Qu.: 0.7192   1st Qu.: 3.390   1st Qu.: 62.25   1st Qu.: 0.07109 1: 3
## Median : 1.2876   Median : 3.832   Median : 64.50   Median : 1.41580
## Mean   : 1.1900   Mean   : 3.808   Mean   : 64.68   Mean   : 0.86047
## 3rd Qu.: 1.7792   3rd Qu.: 4.006   3rd Qu.: 66.75   3rd Qu.: 1.79964
## Max.   : 3.1411   Max.   : 6.108   Max.   : 77.00   Max.   : 2.30757
##      lcp        gleason     pgg45       lpsa
## Min. : -1.3863   6: 7   Min. : 0.00   Min. : -0.1625
## 1st Qu.: -1.3863  7:14   1st Qu.: 0.00   1st Qu.: 1.7395
## Median : -1.3863   8: 0   Median : 12.50   Median : 2.4718
## Mean   : -0.3963   9: 1   Mean   : 22.09   Mean   : 2.3216
## 3rd Qu.: 0.3537          3rd Qu.: 40.00   3rd Qu.: 2.9425
## Max.   : 2.4204          Max.   : 70.00   Max.   : 3.7124

```

On obtient donc un jeu d'apprentissage comportant 75 lignes et 9 colonnes et un jeu de test de 22 lignes et 9 colonnes.

## 2.2 Modèle linéaire complet

Une fonction utile de graphe des résidus

### 2.2.1 Estimation du modèle et graphes des résidus

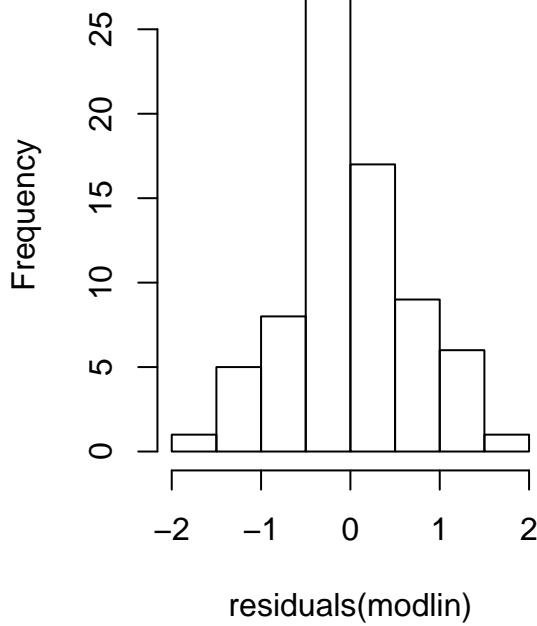
```

##
## Call:
## lm(formula = lpsa ~ ., data = Prostate.app)
##
## Residuals:
##      Min      1Q      Median      3Q      Max
## -1.86180 -0.34983 -0.07532  0.42418  1.52730
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.710840  0.991216  0.717  0.47590
## lcavol      0.549812  0.107883  5.096  3.3e-06 ***
## lweight     0.668793  0.235243  2.843  0.00599 **
## age         -0.028194  0.012643 -2.230  0.02927 *
## lbph        0.124275  0.070797  1.755  0.08398 .
## svi1        0.752437  0.291861  2.578  0.01225 *
## lcp        -0.128543  0.110444 -1.164  0.24880
## gleason7    0.173159  0.264526  0.655  0.51507
## gleason8    0.464916  0.815868  0.570  0.57078
## gleason9   -0.342973  0.604472 -0.567  0.57243
## pgg45      0.006458  0.005407  1.194  0.23681
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

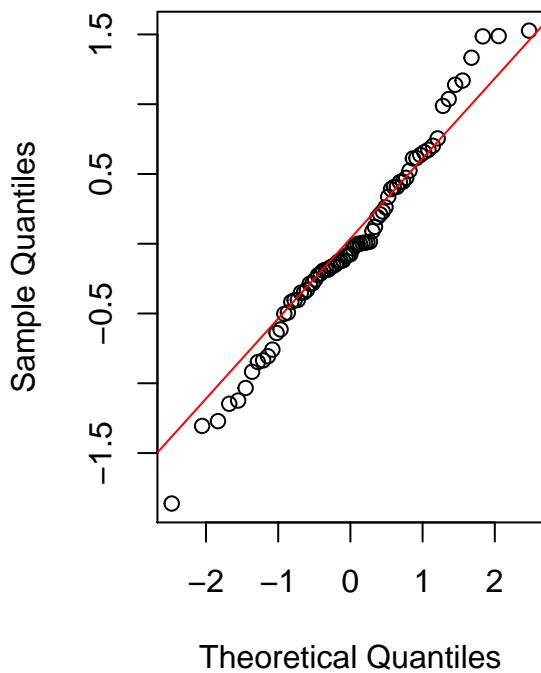
```

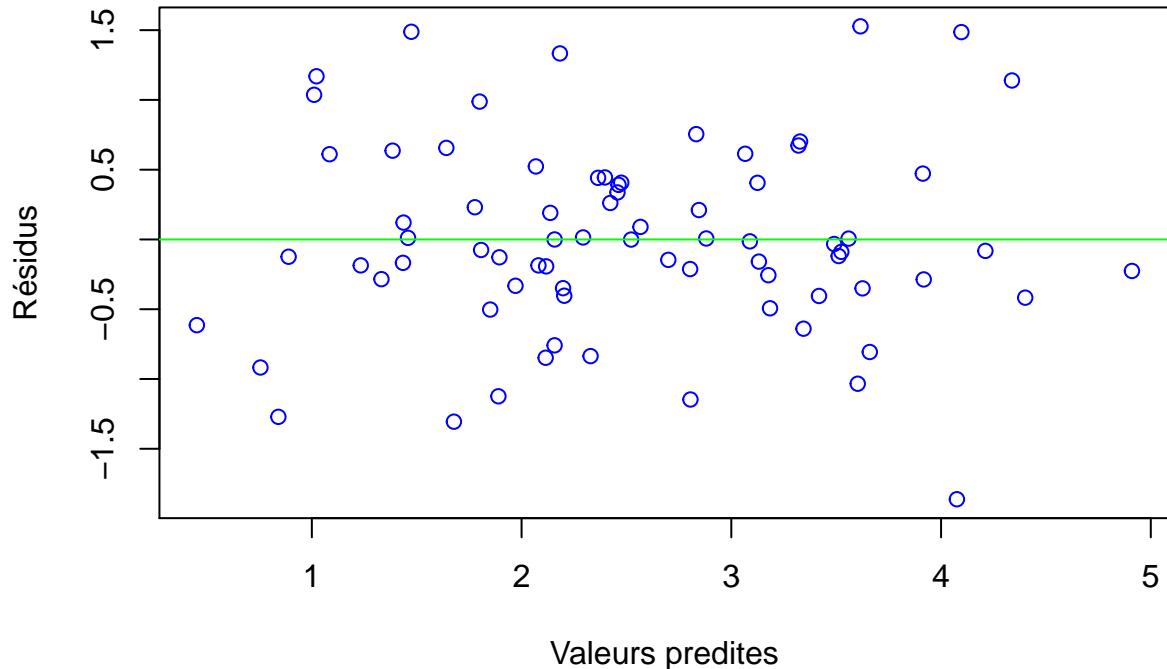
```
##  
## Residual standard error: 0.7363 on 64 degrees of freedom  
## Multiple R-squared:  0.6786, Adjusted R-squared:  0.6283  
## F-statistic: 13.51 on 10 and 64 DF,  p-value: 2.231e-12
```

**Histogram of residuals(modlin)**



**Normal Q-Q Plot**





Les p-valeurs de certaines variables comme lcavol, lweight, age et svil sont très significatives. Selon la répartition des points (uniformément répartis), on peut dire que les résidus sont sans biais. En regardant l'histogramme des résidus, nous pouvons dire que leur distribution suit une loi normale avec des quantiles compris entre -1.86 et +1.5.

### 2.2.2 Erreur d'apprentissage

Calculons l'erreur d'apprentissage

```
## [1] 0.4626117
```

L'erreur d'apprentissage obtenue est de 0.46.

### 2.2.3 Erreur sur l'échantillon test

```
## [1] 0.4500765
```

On obtient 0.45 comme valeur de l'erreur sur l'échantillon de test.

Les erreurs d'apprentissage et de test sont donc presqu'égales.

### 2.2.4 Nouvelle paramétrisation

Afin de faciliter l'interprétation des résultats concernant les variables qualitatives, on introduit une nouvelle paramétrisation à l'aide de contrastes. Par défaut, la référence est prise pour la valeur 0 de svil et 6 de

gleason, qui sont les plus petites valeurs. Les paramètres indiqués pour les variables svi1 gleason 7, 8 et 9 indiquent l'écart estimé par rapport à cette référence. Il est plus intéressant en pratique de se référer à la moyenne des observations sur toutes les modalités des variables qualitatives, et d'interpréter les coefficients comme des écarts à cette moyenne.

```
##
## Call:
## lm(formula = lpsa ~ ., data = Prostate.app)
##
## Residuals:
##    Min      1Q  Median      3Q     Max 
## -1.86180 -0.34983 -0.07532  0.42418  1.52730
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 1.160834  1.057759  1.097  0.27656  
## lcavol       0.549812  0.107883  5.096 3.3e-06 *** 
## lweight      0.668793  0.235243  2.843  0.00599 **  
## age          -0.028194  0.012643 -2.230  0.02927 *   
## lbph         0.124275  0.070797  1.755  0.08398 .    
## svi1        -0.376219  0.145931 -2.578  0.01225 *  
## lcp          -0.128543  0.110444 -1.164  0.24880  
## gleason1     -0.073775  0.302549 -0.244  0.80813  
## gleason2     0.099384  0.237560  0.418  0.67709  
## gleason3     0.391140  0.620394  0.630  0.53063  
## pgg45        0.006458  0.005407  1.194  0.23681  
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7363 on 64 degrees of freedom
## Multiple R-squared:  0.6786, Adjusted R-squared:  0.6283 
## F-statistic: 13.51 on 10 and 64 DF,  p-value: 2.231e-12
```

Nom des variables : gleason1 =6, gleason2 =7, gleason3 =8, gleason4 =9 , la somme des coefficients associés à ces variables est nulle. svi1=0, svi2=1. La somme des deux coefficients est nulle.

### 3 Sélection de modèle par sélection de variables

#### 3.1 Sélection par AIC et backward

```
## Start:  AIC=-35.82
## lpsa ~ lcavol + lweight + age + lbph + svi + lcp + gleason +
##       pgg45
##
##              Df Sum of Sq   RSS   AIC
## - gleason  3    1.2305 35.926 -39.201
## - lcp      1    0.7344 35.430 -36.244
## - pgg45   1    0.7731 35.469 -36.162
## <none>          34.696 -35.815
## - lbph    1    1.6705 36.366 -34.288
## - age     1    2.6957 37.392 -32.203
## - svi     1    3.6032 38.299 -30.405
## - lweight 1    4.3817 39.078 -28.895
```

```

## - lcavol 1 14.0807 48.777 -12.268
##
## Step: AIC=-39.2
## lpsa ~ lcavol + lweight + age + lbph + svi + lcp + pgg45
##
##          Df Sum of Sq    RSS     AIC
## - lcp      1  0.4553 36.382 -40.257
## - pgg45    1  0.8821 36.809 -39.382
## <none>           35.926 -39.201
## - lbph     1  1.7700 37.696 -37.594
## - age      1  2.4344 38.361 -36.284
## - svi      1  3.7232 39.650 -33.806
## - lweight   1  4.6200 40.546 -32.128
## - lcavol   1 15.8909 51.817 -13.732
##
## Step: AIC=-40.26
## lpsa ~ lcavol + lweight + age + lbph + svi + pgg45
##
##          Df Sum of Sq    RSS     AIC
## - pgg45    1  0.5203 36.902 -41.192
## <none>           36.382 -40.257
## - lbph     1  1.6557 38.037 -38.919
## - age      1  2.1996 38.581 -37.854
## - svi      1  3.2777 39.659 -35.787
## - lweight   1  4.5751 40.957 -33.373
## - lcavol   1 16.8805 53.262 -13.670
##
## Step: AIC=-41.19
## lpsa ~ lcavol + lweight + age + lbph + svi
##
##          Df Sum of Sq    RSS     AIC
## <none>           36.902 -41.192
## - age      1  1.8283 38.730 -39.565
## - lbph     1  1.8859 38.788 -39.453
## - lweight   1  4.2368 41.139 -35.040
## - svi      1  4.8021 41.704 -34.017
## - lcavol   1 18.1823 55.084 -13.147

##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + age + lbph + svi, data = Prostate.app)
##
## Residuals:
##       Min     1Q Median     3Q    Max 
## -1.93986 -0.37190  0.02246  0.47021  1.43568
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 1.06729   0.99041   1.078  0.28495    
## lcavol      0.52960   0.09083   5.831 1.61e-07 ***  
## lweight     0.63898   0.22702   2.815  0.00636 **   
## age        -0.02172   0.01175  -1.849  0.06875 .    
## lbph       0.12957   0.06900   1.878  0.06463 .    
## svi1       -0.36548   0.12197  -2.997  0.00379 **  

```

```

## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7313 on 69 degrees of freedom
## Multiple R-squared: 0.6581, Adjusted R-squared: 0.6334
## F-statistic: 26.57 on 5 and 69 DF, p-value: 7.267e-15

```

La sélection par critère AIC et backward nous fournit comme meilleur modèle celui en fonction des variables lcavol, lweight, age, lbph, svi. Ceci avec un AIC = 41.19. On note que certaines variables comme âge et lbph restent non significatifs.

### 3.2 Sélection par AIC et forward

```

## Start: AIC=29.31
## lpsa ~ 1
##
##          Df Sum of Sq    RSS      AIC
## + lcavol  1   57.008  50.934 -25.0218
## + svi     1   37.507  70.435 -0.7097
## + lcp     1   33.011  74.931  3.9314
## + lweight  1   24.626  83.317 11.8870
## + gleason  3   22.657  85.285 17.6383
## + pgg45    1   16.316  91.626 19.0171
## + lbph     1   6.358  101.584 26.7552
## <none>           107.942 29.3081
## + age      1   2.448  105.494 29.5878
##
## Step: AIC=-25.02
## lpsa ~ lcavol
##
##          Df Sum of Sq    RSS      AIC
## + lweight  1   6.8773  44.057 -33.901
## + svi     1   4.8638  46.070 -30.549
## + lbph     1   3.1103  47.824 -27.747
## + pgg45    1   1.3620  49.572 -25.055
## <none>           50.934 -25.022
## + lcp     1   0.8523  50.082 -24.287
## + age     1   0.0509  50.883 -23.097
## + gleason  3   1.9581  48.976 -21.962
##
## Step: AIC=-33.9
## lpsa ~ lcavol + lweight
##
##          Df Sum of Sq    RSS      AIC
## + svi     1   4.1621  39.895 -39.343
## + pgg45   1   1.3831  42.674 -34.293
## + age     1   1.1984  42.858 -33.969
## <none>           44.057 -33.901
## + lcp     1   0.7377  43.319 -33.167
## + lbph     1   0.6054  43.451 -32.938
## + gleason  3   2.0371  42.020 -31.451
##
## Step: AIC=-39.34

```

```

## lpsa ~ lcavol + lweight + svi
##
##          Df Sum of Sq    RSS     AIC
## + lbph     1   1.16439 38.730 -39.565
## + age      1   1.10678 38.788 -39.453
## <none>            39.895 -39.343
## + pgg45    1   0.31242 39.582 -37.933
## + lcp      1   0.02440 39.870 -37.389
## + gleason  3   1.11456 38.780 -35.469
##
## Step:  AIC=-39.56
## lpsa ~ lcavol + lweight + svi + lbph
##
##          Df Sum of Sq    RSS     AIC
## + age      1   1.82830 36.902 -41.192
## <none>            38.730 -39.565
## + pgg45    1   0.14896 38.581 -37.854
## + lcp      1   0.07421 38.656 -37.709
## + gleason  3   0.85165 37.879 -35.233
##
## Step:  AIC=-41.19
## lpsa ~ lcavol + lweight + svi + lbph + age
##
##          Df Sum of Sq    RSS     AIC
## <none>            36.902 -41.192
## + pgg45    1   0.52027 36.382 -40.257
## + lcp      1   0.09347 36.809 -39.382
## + gleason  3   1.13158 35.770 -37.528

##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + svi + lbph + age, data = Prostate.app)
##
## Residuals:
##       Min     1Q     Median      3Q     Max 
## -1.93986 -0.37190  0.02246  0.47021  1.43568
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 1.06729   0.99041   1.078  0.28495  
## lcavol       0.52960   0.09083   5.831 1.61e-07 ***
## lweight      0.63898   0.22702   2.815  0.00636 ** 
## svi1        -0.36548   0.12197  -2.997  0.00379 ** 
## lbph         0.12957   0.06900   1.878  0.06463 .  
## age         -0.02172   0.01175  -1.849  0.06875 .  
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7313 on 69 degrees of freedom
## Multiple R-squared:  0.6581, Adjusted R-squared:  0.6334 
## F-statistic: 26.57 on 5 and 69 DF,  p-value: 7.267e-15

```

Avec la sélection par AIC et forward, on obtient presque les mêmes résultats qu'avec la sélection par AIC backward.

### 3.3 Sélection par AIC et stepwise

```

## Start:  AIC=-35.82
## lpsa ~ lcavol + lweight + age + lbph + svi + lcp + gleason +
##      pgg45
##
##          Df Sum of Sq    RSS     AIC
## - gleason  3   1.2305 35.926 -39.201
## - lcp      1   0.7344 35.430 -36.244
## - pgg45   1   0.7731 35.469 -36.162
## <none>        34.696 -35.815
## - lbph     1   1.6705 36.366 -34.288
## - age      1   2.6957 37.392 -32.203
## - svi      1   3.6032 38.299 -30.405
## - lweight   1   4.3817 39.078 -28.895
## - lcavol   1  14.0807 48.777 -12.268
##
## Step:  AIC=-39.2
## lpsa ~ lcavol + lweight + age + lbph + svi + lcp + pgg45
##
##          Df Sum of Sq    RSS     AIC
## - lcp      1   0.4553 36.382 -40.257
## - pgg45   1   0.8821 36.809 -39.382
## <none>        35.926 -39.201
## - lbph     1   1.7700 37.696 -37.594
## - age      1   2.4344 38.361 -36.284
## + gleason  3   1.2305 34.696 -35.815
## - svi      1   3.7232 39.650 -33.806
## - lweight   1   4.6200 40.546 -32.128
## - lcavol   1  15.8909 51.817 -13.732
##
## Step:  AIC=-40.26
## lpsa ~ lcavol + lweight + age + lbph + svi + pgg45
##
##          Df Sum of Sq    RSS     AIC
## - pgg45   1   0.5203 36.902 -41.192
## <none>        36.382 -40.257
## + lcp      1   0.4553 35.926 -39.201
## - lbph     1   1.6557 38.037 -38.919
## - age      1   2.1996 38.581 -37.854
## + gleason  3   0.9515 35.430 -36.244
## - svi      1   3.2777 39.659 -35.787
## - lweight   1   4.5751 40.957 -33.373
## - lcavol   1  16.8805 53.262 -13.670
##
## Step:  AIC=-41.19
## lpsa ~ lcavol + lweight + age + lbph + svi
##
##          Df Sum of Sq    RSS     AIC
## <none>        36.902 -41.192
## + pgg45   1   0.5203 36.382 -40.257
## - age      1   1.8283 38.730 -39.565
## - lbph     1   1.8859 38.788 -39.453
## + lcp      1   0.0935 36.809 -39.382

```

```

## + gleason  3    1.1316 35.770 -37.528
## - lweight   1    4.2368 41.139 -35.040
## - svi       1    4.8021 41.704 -34.017
## - lcavol    1   18.1823 55.084 -13.147

##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + age + lbph + svi, data = Prostate.app)
##
## Residuals:
##      Min      1Q Median      3Q      Max
## -1.93986 -0.37190  0.02246  0.47021  1.43568
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  1.06729   0.99041   1.078  0.28495  
## lcavol       0.52960   0.09083   5.831 1.61e-07 ***
## lweight      0.63898   0.22702   2.815  0.00636 ** 
## age          -0.02172   0.01175  -1.849  0.06875 .  
## lbph         0.12957   0.06900   1.878  0.06463 .  
## svi1        -0.36548   0.12197  -2.997  0.00379 ** 
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7313 on 69 degrees of freedom
## Multiple R-squared:  0.6581, Adjusted R-squared:  0.6334 
## F-statistic: 26.57 on 5 and 69 DF,  p-value: 7.267e-15

```

On retrouve ici le même modèle qu'avec l'algorithme backward.

### 3.4 Sélection par BIC et stepwise

k=log(napp) pour BIC au lieu de AIC.

```

## Start:  AIC=-10.32
## lpsa ~ lcavol + lweight + age + lbph + svi + lcp + gleason +
##       pgg45
##
##             Df Sum of Sq     RSS     AIC
## - gleason  3    1.2305 35.926 -20.6613
## - lcp      1    0.7344 35.430 -13.0693
## - pgg45   1    0.7731 35.469 -12.9873
## - lbph     1    1.6705 36.366 -11.1135
## <none>          34.696 -10.3227
## - age      1    2.6957 37.392  -9.0283
## - svi      1    3.6032 38.299  -7.2298
## - lweight   1    4.3817 39.078  -5.7205
## - lcavol   1   14.0807 48.777  10.9070
##
## Step:  AIC=-20.66
## lpsa ~ lcavol + lweight + age + lbph + svi + lcp + pgg45
##
##             Df Sum of Sq     RSS     AIC

```

```

## - lcp      1    0.4553 36.382 -24.0342
## - pgg45    1    0.8821 36.809 -23.1595
## - lbph     1    1.7700 37.696 -21.3718
## <none>          35.926 -20.6613
## - age      1    2.4344 38.361 -20.0614
## - svi      1    3.7232 39.650 -17.5831
## - lweight   1    4.6200 40.546 -15.9056
## + gleason   3    1.2305 34.696 -10.3227
## - lcavol    1    15.8909 51.817    2.4901
##
## Step: AIC=-24.03
## lpsa ~ lcavol + lweight + age + lbph + svi + pgg45
##
##              Df Sum of Sq    RSS      AIC
## - pgg45     1    0.5203 36.902 -27.2868
## - lbph      1    1.6557 38.037 -25.0139
## <none>          36.382 -24.0342
## - age      1    2.1996 38.581 -23.9490
## - svi      1    3.2777 39.659 -21.8820
## + lcp      1    0.4553 35.926 -20.6613
## - lweight   1    4.5751 40.957 -19.4678
## + gleason   3    0.9515 35.430 -13.0693
## - lcavol    1    16.8805 53.262    0.2354
##
## Step: AIC=-27.29
## lpsa ~ lcavol + lweight + age + lbph + svi
##
##              Df Sum of Sq    RSS      AIC
## - age       1    1.8283 38.730 -27.9775
## - lbph      1    1.8859 38.788 -27.8660
## <none>          36.902 -27.2868
## + pgg45     1    0.5203 36.382 -24.0342
## - lweight   1    4.2368 41.139 -23.4528
## + lcp       1    0.0935 36.809 -23.1595
## - svi       1    4.8021 41.704 -22.4292
## + gleason   3    1.1316 35.770 -16.6701
## - lcavol    1    18.1823 55.084 -1.5593
##
## Step: AIC=-27.98
## lpsa ~ lcavol + lweight + lbph + svi
##
##              Df Sum of Sq    RSS      AIC
## - lbph      1    1.1644 39.895 -30.0734
## <none>          38.730 -27.9775
## + age       1    1.8283 36.902 -27.2868
## - lweight   1    3.3237 42.054 -26.1201
## + pgg45     1    0.1490 38.581 -23.9490
## + lcp       1    0.0742 38.656 -23.8039
## - svi       1    4.7211 43.451 -23.6684
## + gleason   3    0.8516 37.879 -16.6926
## - lcavol    1    17.0987 55.829 -4.8696
##
## Step: AIC=-30.07
## lpsa ~ lcavol + lweight + svi

```

```

##          Df Sum of Sq    RSS      AIC
## <none>            39.895 -30.0734
## + lbph     1   1.1644 38.730 -27.9775
## + age      1   1.1068 38.788 -27.8660
## - svi      1   4.1621 44.057 -26.9482
## + pgg45    1   0.3124 39.582 -26.3456
## + lcp      1   0.0244 39.870 -25.8018
## - lweight   1   6.1755 46.070 -23.5966
## + gleason   3   1.1146 38.780 -19.2461
## - lcavol    1  17.5291 57.424  -7.0747

##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + svi, data = Prostate.app)
##
## Residuals:
##       Min     1Q Median     3Q    Max
## -1.80400 -0.46271 -0.00071  0.44944  1.55985
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.50668   0.74740 -0.678  0.50002
## lcavol       0.51598   0.09238  5.585 4.02e-07 ***
## lweight      0.68900   0.20783  3.315  0.00144 **
## svi1        -0.33677   0.12374 -2.722  0.00817 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7496 on 71 degrees of freedom
## Multiple R-squared:  0.6304, Adjusted R-squared:  0.6148
## F-statistic: 40.37 on 3 and 71 DF,  p-value: 2.452e-15

```

Avec la méthode BIC et stepwise, on obtient moins de variables dans le meilleur sélectionné comparé aux méthodes précédentes. Le modèle sélectionné est plus parcimonieux dans le cas de la méthode BIC et stepwise.

### 3.5 Erreur sur l'échantillon d'apprentissage

Modèle stepwise AIC

```
## [1] 0.4920266
```

Avec la méthode stepwise AIC, on trouve pour valeur 0.49 comme erreur sur l'échantillon d'apprentissage.

Modèle stepwise BIC

```
## [1] 0.531929
```

Avec la méthode stepwise BIC, on trouve pour valeur 0.53 comme erreur sur l'échantillon d'apprentissage.

### 3.6 Calcul de l'erreur sur l'échantillon test

Modèle stepwise AIC

```
## [1] 0.4655829
```

Avec la méthode stepwise AIC, on trouve pour valeur 0.46 comme erreur sur l'échantillon test.

Modèle stepwise BIC

```
## [1] 0.4008493
```

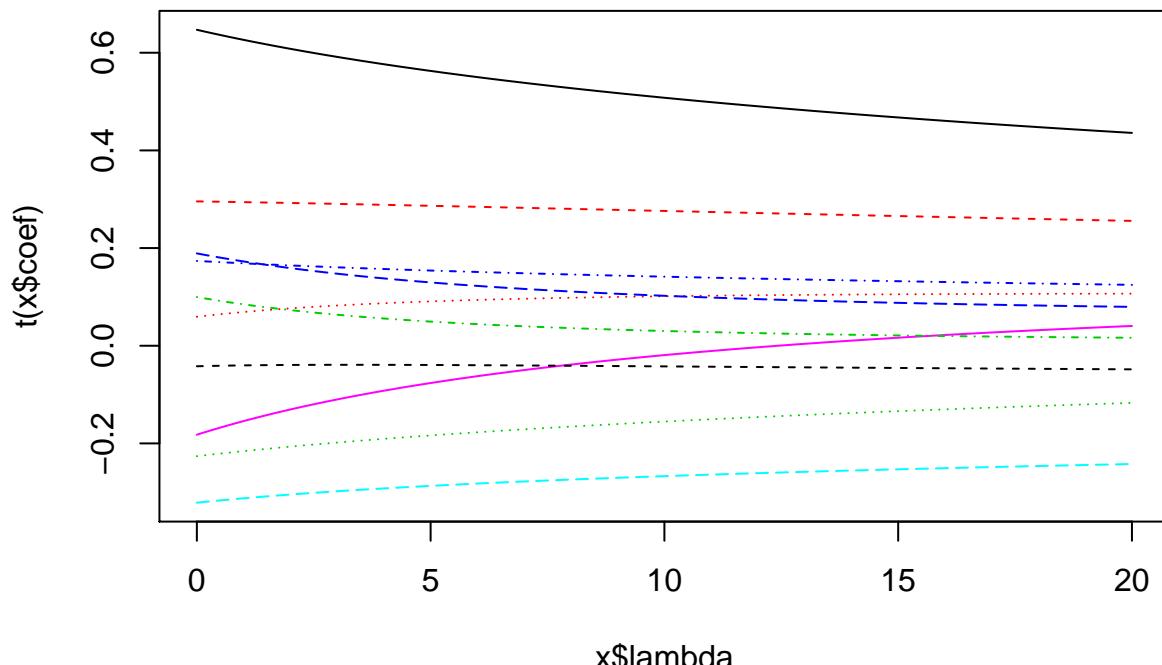
Avec la méthode stepwise BIC, on trouve pour valeur 0.40 comme erreur sur l'échantillon test.

Les modèles sélectionnés ont une erreur plus grande que le modèle linéaire comprenant toutes les variables sur l'échantillon d'apprentissage. Sur l'échantillon test, le modèle qui minimise le critère BIC a de meilleures performances que le modèle initial. Les deux modèles sélectionnés sont beaucoup plus parcimonieux que le modèle initial.

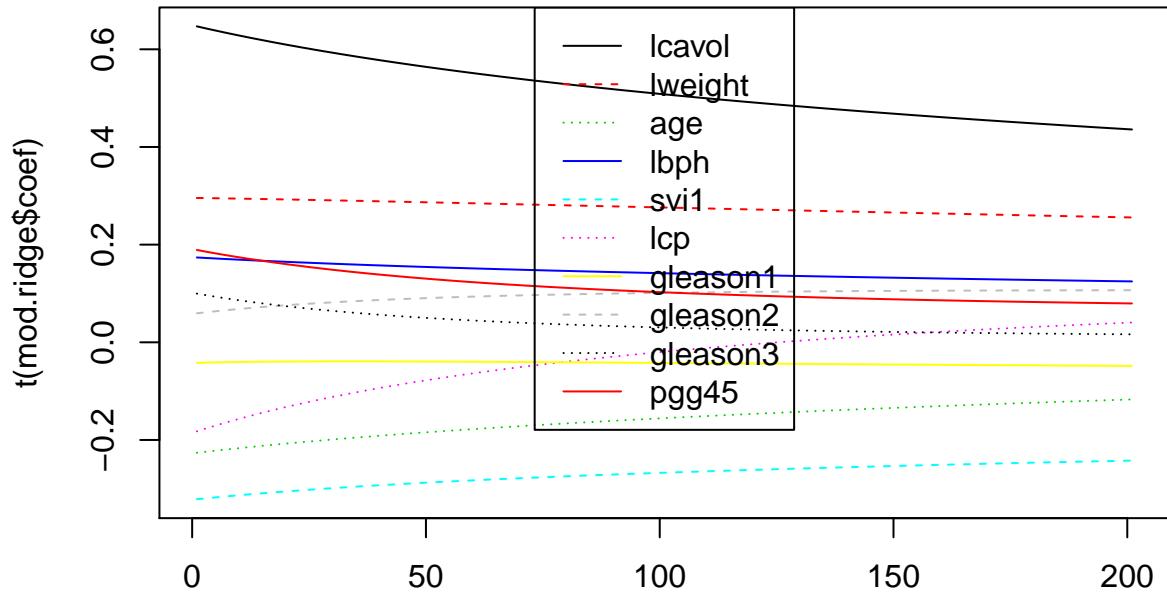
## 4-Sélection de modèle par pénalisation Ridge

### 4.1 Comportement des coefficients

Calcul des coefficients pour différentes valeurs du paramètre lambda



Evolution des coefficients



#### 4.2 Pénalisation optimale par validation croisée

```
## modified HKB estimator is 5.597491
## modified L-W estimator is 4.440824
## smallest value of GCV at 10.4
```

#### 4.3 Prévision et erreur d'apprentissage

Ici, on calcule les valeurs prédites à partir des coefficients.

Coefficients du modèle sélectionné:

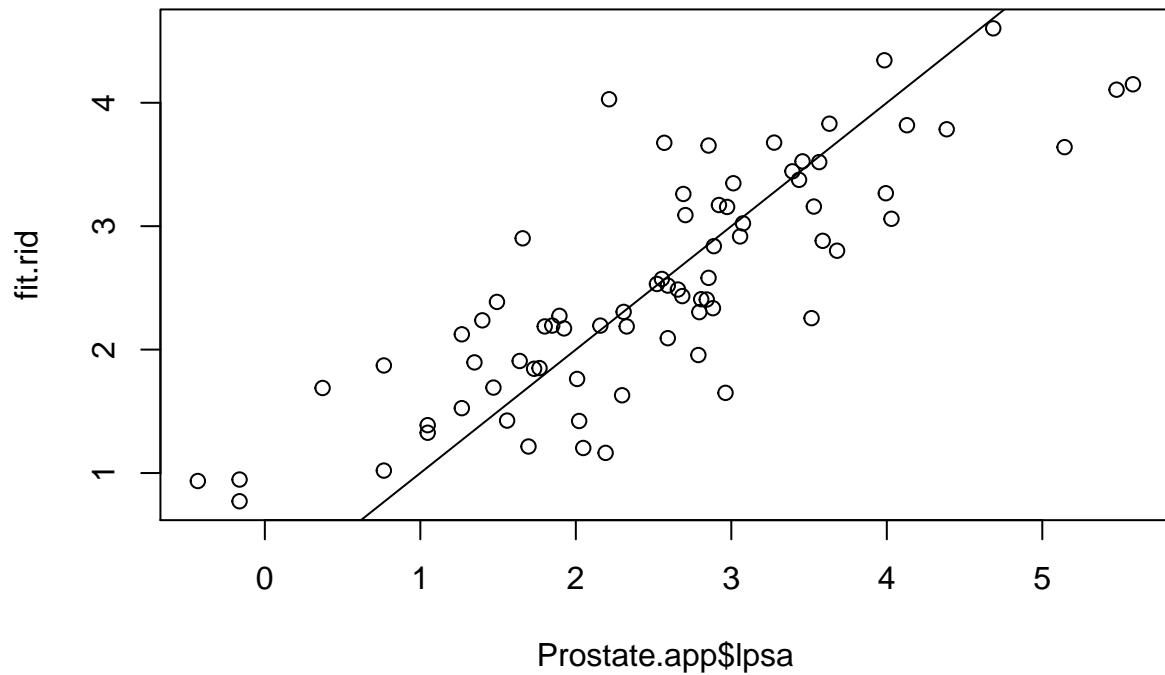
On crée des vecteurs pour :

- les variables qualitatives

- les variables quantitatives

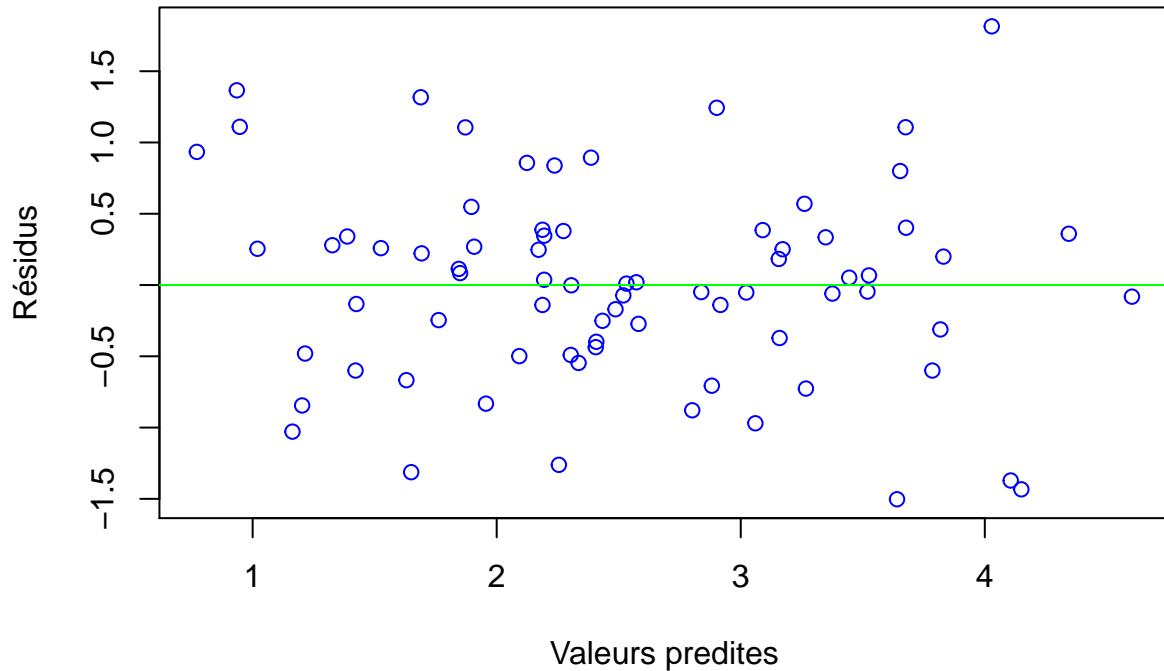
On calcule ici des valeurs prédites

Nous traçons ensuite des valeurs prédites en fonction des valeurs observées



On calcule et on fait le tracé des résidus

## Tracé des résidus



Selon la répartition des points (uniformément répartis), on peut dire que les résidus sont sans biais.  
Erreurd'apprentissage

```
## [1] 0.4789025
```

On trouve pour valeur 0.478 comme erreur d'apprentissage.

### 4.4 Prévision sur l'échantillon test

Les variables qualitatives

Les variables quantitatives

Erreur sur l'échantillon test

```
## [1] 0.424398
```

On trouve pour valeur 0.424 comme erreur sur l'échantillon test.

L'erreur d'apprentissage est légèrement plus élevée que pour le modèle linéaire sans pénalisation. L'erreur de test est plus faible. Les performances sont comparables sur l'échantillon test au modèle sélectionné par le critère BIC. En terme d'interprétation, les modèles sélectionnés par AIC et BIC sont préférables.

## 5 Sélection de modèle par pénalisation Lasso

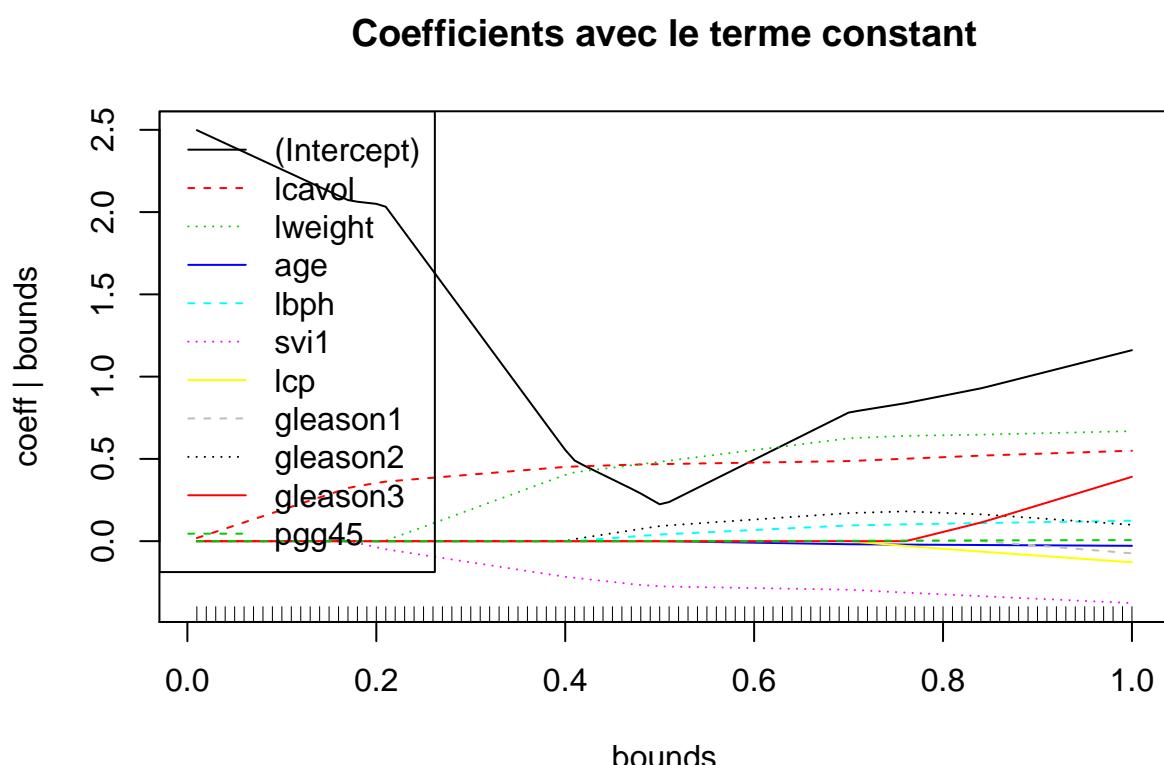
### 5.1 Librairie Lasso2

### 5.2 Construction du modèle

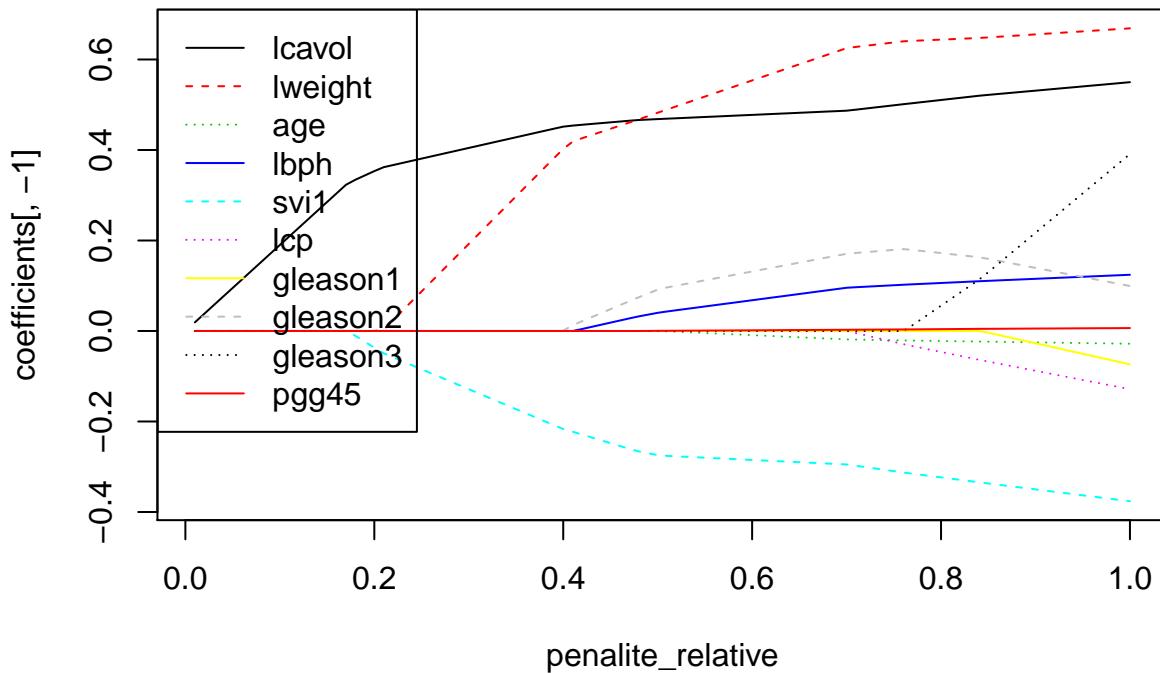
La borne est ici relative, elle correspond à une certaine proportion de la norme L1 du vecteur des coefficients des moindres carrés. Une borne égale à 1 correspond donc à l'absence de pénalité, on retrouve l'estimateur des moindres carrés.

### 5.3 Visualisation des coefficients

On visualise ici les coefficients du modèle



## Coefficients après suppression du terme constant



En fonction de différentes valeurs de la pénalité, on obtient différentes valeurs pour les coefficients. Afin de faire un bon choix de la pénalité, nous procédons par la méthode de validation croisée.

### 5.4 Sélection de la pénalité par validation croisée

On procède à la validation croisée pour sélectionner la pénalité.

Par la méthode de validation croisée, on obtient 0.95 comme valeur de la meilleure pénalité qui nous a servi à optimiser notre modèle précédent en utilisant que cette meilleure pénalité dans le modèle.

### 5.5 Erreur d'apprentissage

```
## [1] 0.4629727
```

On trouve pour valeur 0.46 comme erreur d'apprentissage.

### 5.6 Erreur sur l'échantillon test

```
## [1] 0.4438783
```

On trouve pour valeur 0.44 comme erreur sur l'échantillon test.

## **5.7 Librairie glmnet**

L'utilisation de la librairie glmnet fournit des résultats plus rapides, ce qui peut s'avérer important pour des données de grande dimension. Par contre, on ne peut pas traiter à priori des variables qualitatives. Nous allons donc devoir créer des vecteurs avec des variables indicatrices des diverses modalités pour les variables qualitatives. Nous ne prendrons pas en compte les contrastes.

## **5.8 Mise en forme des variables**

on construit une matrice xx.app d'apprentissage et xx.test de test

On construit ici des vecteurs indicatrices pour les variables qualitatives

On crée une matrice avec les vecteurs indicatrices

On nomme les colonnes avec les noms des variables

On fait de même pour l'échantillon test

On construit une matrice avec les vecteurs indicatrices

## **5.9 Construction du modèle**

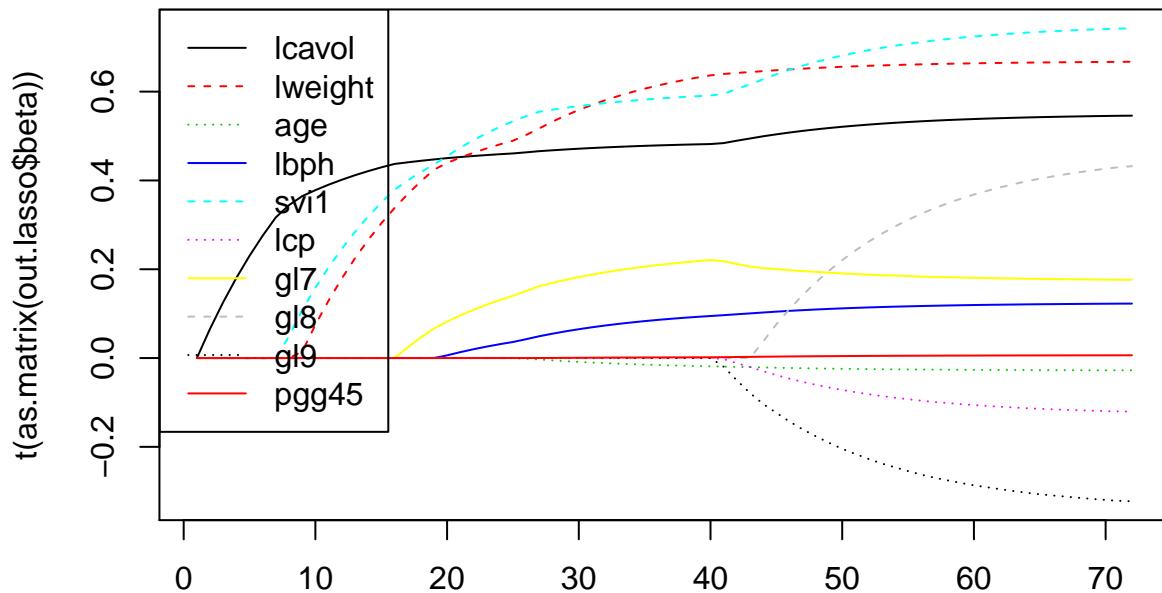
```
## Loading required package: Matrix
```

```
## Loaded glmnet 3.0-2
```

## **5.10 Visualisation des coefficients**

Chemin de régularisation du lasso

## Lasso



Ici, on a tracé les coefficients des différentes variables du modèle Lasso.

### 5.11 Sélection de la pénalité par validation croisée

Nous appliquons la méthode de validation croisée afin de sélectionner la meilleure pénalité. Ensuite, on optimise le modèle avec la valeur de la meilleure pénalité trouvée par la méthode de validation croisée.

Nous trouvons la valeur de 0.0444 comme meilleure pénalité.

### 5.12 Erreur d'apprentissage

```
## [1] 0.4869519
```

Avec le modèle optimisé, nous trouvons 0.48 comme valeur de l'erreur d'apprentissage.

### 5.13 Erreur sur l'échantillon test

```
## [1] 0.3878855
```

Avec le modèle optimisé, nous trouvons environ 0.39 comme erreur de prédiction commise sur le jeu de test.

Il est à noter que l'erreur de test est un peu plus petite que l'erreur d'apprentissage.

## 5.14 Elastic Net

La méthode Elastic Net est une méthode qui permet de résoudre les problèmes liés à la méthode Lasso (comme par exemple le problème de colinéarité entre les variables et le problème du nombre de variables à sélectionner lié au fléau de la dimension).

On peut jouer avec le paramètre alpha de glmnet

Ici, nous avons créé un modèle avec 0.5 comme paramètre alpha. On a procédé ensuite par la méthode de validation croisée pour choisir le meilleur paramètre lambda. Ceci nous a permis d'optimiser notre modèle. On utilise donc le modèle optimisé pour faire des prédictions et calculer les erreurs commises sur le jeu d'apprentissage et sur le jeu de test.

Erreur d'apprentissage

```
## [1] 0.6232078
```

Nous trouvons la valeur 0.6 comme erreur d'apprentissage.

Erreur de prédiction

```
## [1] 0.3840482
```

Nous trouvons la valeur 0.37 comme erreur de test.

L'erreur commise sur le jeu de test est plus petite que celle commise sur le jeu d'apprentissage.

# TP3 Chap3 Pratique de la régression Ridge et Elasticnet

Dans ce TP, nous nous situons dans le cadre de la régression logistique avec une variable cible qualitative binaire. Les propriétés de régularisation de ridge et elasticnet devraient se révéler décisives. Encore faut-il savoir/pouvoir déterminer les valeurs adéquates des paramètres de ces algorithmes. Ils pèsent fortement sur la qualité des résultats.

## 3.1 Données et régression logistique glm()

### 3.1.1 Base de données “adult”

Nos données sont dérivées de la base “Adult Data Set”. Elle a été retravaillée: les variables quantitatives ont été discrétisées, puis transformées en variables indicatrices via un codage disjonctif complet; les variables catégorielles ont également été binarisées via le même procédé. En définitive, nous disposons de  $p = 123$  variables explicatives, toutes binaires. La variable cible est également binaire, définie dans positive, negative.

### 3.1.2 Importation des bases d'apprentissage et de test - Quelques vérifications

```
##   age      workclass educatoin education_num marital_status      occupation race
## 1  39      Government Bachelors          13           Single White-Collar White
## 2  50    Self-Employed Bachelors          13           Married White-Collar White
## 3  38        Private   HS-grad            9           Divorced Blue-Collar White
## 4  53        Private     11th             7           Married Blue-Collar Black
## 5  28        Private Bachelors          13           Married Professional Black
## 6  37        Private    Masters          14           Married White-Collar White
```

```

##      sex hours_per_week income
## 1    Male           40   <=50K
## 2    Male           13   <=50K
## 3    Male           40   <=50K
## 4    Male           40   <=50K
## 5 Female          40   <=50K
## 6 Female          40   <=50K

## [1] 32561     10

## [1] FALSE

```

On sépare les données en un jeu d'apprentissage et de test

```

## [1] 24423     10

## [1] 8138     10

## [1] 8138

```

Nous vérifions la répartition des classes. Le modèle par défaut consisterait à prédire systématiquement la classe majoritaire négative.

Répartition de la variable income

```

##
##      <=50K      >50K
## 0.756541 0.243459

##
##      <=50K      >50K
## 0.7671418 0.2328582

```

La répartition des classes de l'échantillon de test est très similaire à celle de l'échantillon d'apprentissage. Avec le modèle par défaut, prédiction systématique de la classe majoritaire négative, le taux d'erreur serait de 23.28%. Il s'agit de faire mieux avec les différentes variantes de la régression logistique que nous mettrons en oeuvre dans ce qui suit.

Nous plaçons les variables explicatives et la variable cible dans deux structures distinctes, une matrice pour les premières, un vecteur pour la seconde.

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

Nous plaçons également la matrice des descripteurs dans une structure dédiée. Matrice en test

La fonction `glm()` du package stats est l'outil privilégié pour la régression logistique sous R. Nous l'appliquons à l'échantillon d'apprentissage.

```

##
## Call:
## glm(formula = as.factor(DTrain$income) ~ ., family = "binomial",
##      data = DTrain)

```

```

##
## Deviance Residuals:
##      Min       1Q   Median      3Q      Max
## -2.6785  -0.5843  -0.2565  -0.0624  3.3796
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.634022  0.341467 -22.357 < 2e-16 ***
## age          0.028427  0.001713  16.595 < 2e-16 ***
## workclassOther/Unknown -1.458578  0.805022 -1.812  0.07001 .
## workclassPrivate     0.020255  0.055402  0.366  0.71467
## workclassSelf-Employed -0.205897  0.071770 -2.869  0.00412 **
## educatoin11th        0.090970  0.239824  0.379  0.70445
## educatoin12th        0.631116  0.289482  2.180  0.02925 *
## educatoin1st-4th     -0.633326  0.464097 -1.365  0.17237
## educatoin5th-6th     -0.531761  0.349604 -1.521  0.12825
## educatoin7th-8th     -0.655933  0.272987 -2.403  0.01627 *
## educatoin9th         -0.172457  0.283881 -0.607  0.54352
## educatoinAssoc-acdm   1.732330  0.198827  8.713 < 2e-16 ***
## educatoinAssoc-voc    1.662357  0.192500  8.636 < 2e-16 ***
## educatoinBachelors    2.249563  0.181216 12.414 < 2e-16 ***
## educatoinDoctorate    3.128623  0.236541 13.227 < 2e-16 ***
## educatoinHS-grad      0.968660  0.177816  5.448 5.11e-08 ***
## educatoinMasters      2.624642  0.191862 13.680 < 2e-16 ***
## educatoinPreschool    -11.691910 124.758727 -0.094  0.92533
## educatoinProf-school   3.252062  0.224460 14.488 < 2e-16 ***
## educatoinSome-college   1.401063  0.179759  7.794 6.49e-15 ***
## education_num          NA        NA        NA        NA
## marital_statusMarried   1.952260  0.068901 28.334 < 2e-16 ***
## marital_statusSeparated -0.328895  0.178682 -1.841  0.06567 .
## marital_statusSingle    -0.583666  0.086104 -6.779 1.21e-11 ***
## marital_statusWidowed   0.042718  0.152877  0.279  0.77992
## occupationOther/Unknown  1.100864  0.805011  1.368  0.17146
## occupationProfessional   0.652886  0.071152  9.176 < 2e-16 ***
## occupationSales          0.442028  0.065950  6.702 2.05e-11 ***
## occupationService         0.143457  0.068898  2.082  0.03733 *
## occupationWhite-Collar   0.778961  0.054628 14.259 < 2e-16 ***
## raceAsian-Pac-Islander   0.576864  0.278030  2.075  0.03800 *
## raceBlack                 0.557420  0.267838  2.081  0.03742 *
## raceOther                 -0.007340  0.380715 -0.019  0.98462
## raceWhite                 0.794517  0.258167  3.078  0.00209 **
## sexMale                   0.408306  0.053586  7.620 2.54e-14 ***
## hours_per_week            0.030346  0.001675 18.112 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 27111 on 24422 degrees of freedom
## Residual deviance: 17972 on 24388 degrees of freedom
## AIC: 18042
##
## Number of Fisher Scoring iterations: 13

```

Le modèle est inutilisable. Certains coefficients n'ont pas été estimés et correspondent à la valeur NA (NotAvailable). Nous ne pouvons ni les interpréter, ni les déployer sur des individus supplémentaires. De fait, nous n'avons même pas essayé d'appliquer le modèle sur l'échantillon test pour en mesurer les performances.

### 3.1.3 La librairie glmnet

Elle est efficace (rapidité des calculs, qualité des résultats), et surtout, elle propose toute une panoplie d'outils qui se révèlent précieux dans l'analyse exploratoire, lors de la recherche des combinaisons adéquates des paramètres  $\lambda$  et  $\alpha$ .

Régression logistique non-régularisée

Nous désactivons la standardisation (standardize = FALSE) des variables explicatives parce que nous savons qu'elles sont toutes binaires, définies de facto sur la même échelle. Sur une base quelconque, en l'absence d'informations précises sur les variables, il est plus prudent d'activer l'option.

Nous lancons la régression logistique sans paramètre de régularisation ( $\lambda = 0$ )

```
## Loading required package: lattice
## Loading required package: ggplot2
##
##           1           2
## 0.7730061 0.2269939
##
##           1           2
## 0.7590507 0.2409493
##
## Call: glmnet(x = XTrain, y = yTrain, family = "binomial", nlambda = 1,      lambda = 0, standardize
##               Df %Dev Lambda
## 1 156     1     0
```

Nous appliquons le modèle sur l'échantillon test avec predict().

```
## yp2
##   0    1
## 30804 1431
```

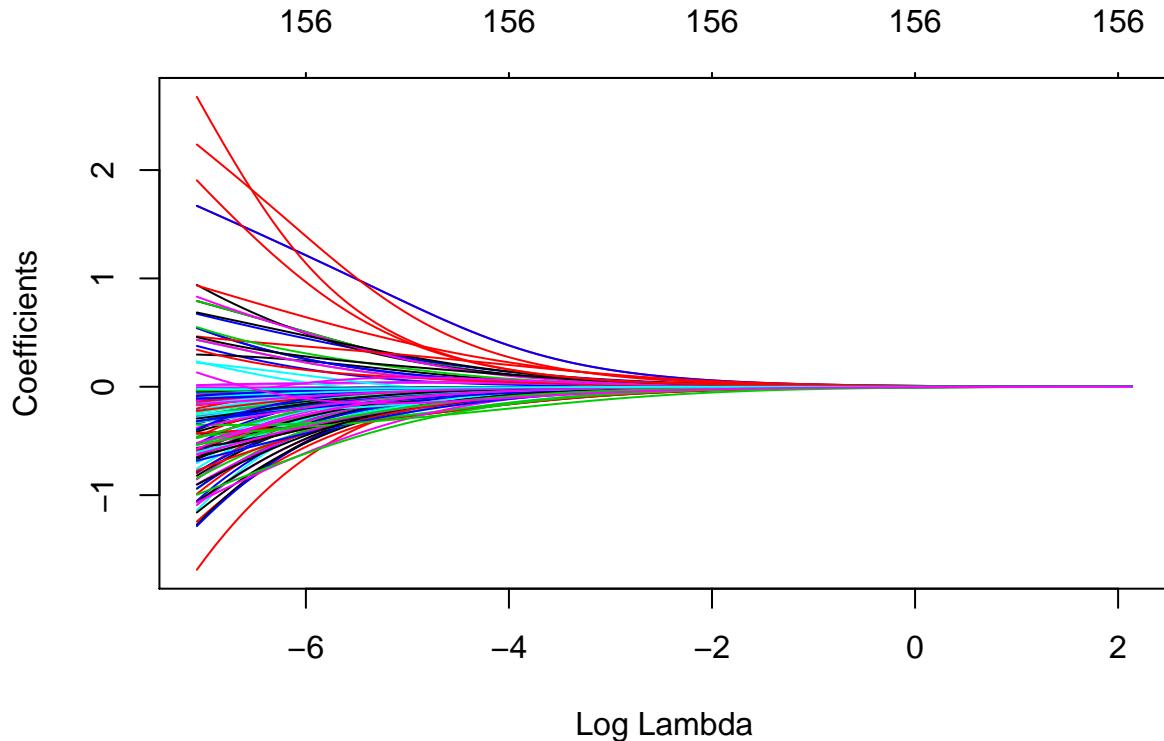
Les prédictions sont reparties entre les deux classes, il n'y a pas de prédiction systématique.

Taux d'erreur

```
## [1] 0.9621219
```

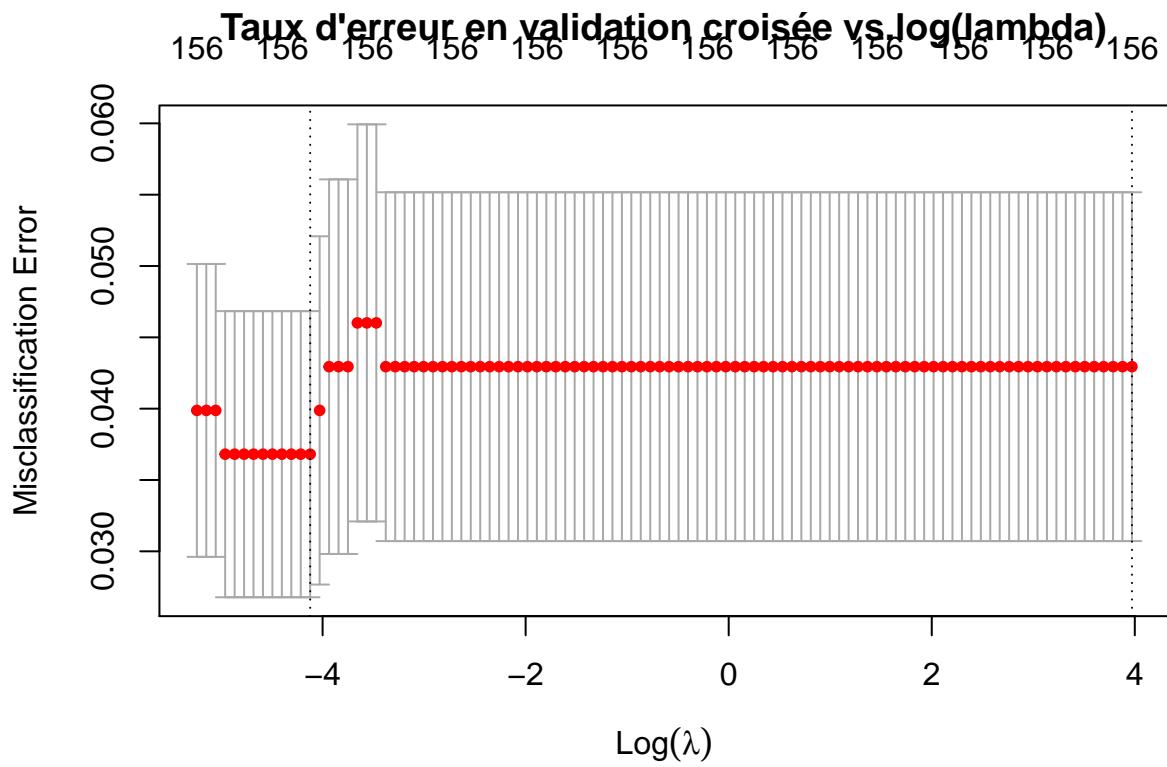
Les difficultés (le ratio p/ntrain élevé et, surtout, les variables constituées de constantes) ont eu raison de l'algorithme d'apprentissage en l'absence de contraintes sur les coefficients. L'inspection des variables préalablement à l'apprentissage est donc très importante. Il aurait fallu exclure délibérément ces variables à problème. Nous continuons en l'état néanmoins en espérant que les mécanismes de régularisation des régressions ridge et elasticnet nous aident à résoudre le problème.

## Régression Ridge



A l'issue de l'apprentissage, nous disposons d'un vecteur de coefficients  $\beta^i$  pour chaque  $\lambda_i$ . Tous les coefficients sont nuls lorsque  $\lambda = \lambda_{max}$ . Nous pouvons afficher une "Ridge path coefficients" permettant de juger de la dispersion des coefficients au regard de  $\lambda$ . Contrairement à la régression Lasso, Ridge ne sait pas fixer sélectivement les coefficients à la valeur 0 et ne permet donc pas de réaliser une sélection de variables.

Afin d'optimiser le paramètre lambda, nous procédons par une validation croisée.



Nous affichons ci-dessous la suite de  $\lambda_i$ ; le taux d'erreur associé; le nombre de coefficients non-nuls (qui n'évolue pas puisque nous utilisons une régression ridge). Affichage

```

##          [,1]      [,2] [,3]
## s0  53.087871541 0.04294479 156
## s1  48.371691353 0.04294479 156
## s2  44.074483615 0.04294479 156
## s3  40.159027969 0.04294479 156
## s4  36.591410610 0.04294479 156
## s5  33.340730545 0.04294479 156
## s6  30.378831937 0.04294479 156
## s7  27.680060237 0.04294479 156
## s8  25.221039977 0.04294479 156
## s9  22.980472300 0.04294479 156
## s10 20.938950479 0.04294479 156
## s11 19.078791831 0.04294479 156
## s12 17.383884549 0.04294479 156
## s13 15.839548159 0.04294479 156
## s14 14.432406358 0.04294479 156
## s15 13.150271155 0.04294479 156
## s16 11.982037311 0.04294479 156
## s17 10.917586141 0.04294479 156
## s18 9.947697879 0.04294479 156
## s19 9.063971817 0.04294479 156
## s20 8.258753543 0.04294479 156
## s21 7.525068641 0.04294479 156

```

```

## s22 6.856562284 0.04294479 156
## s23 6.247444188 0.04294479 156
## s24 5.692438466 0.04294479 156
## s25 5.186737922 0.04294479 156
## s26 4.725962421 0.04294479 156
## s27 4.306120945 0.04294479 156
## s28 3.923577029 0.04294479 156
## s29 3.575017260 0.04294479 156
## s30 3.257422580 0.04294479 156
## s31 2.968042136 0.04294479 156
## s32 2.704369453 0.04294479 156
## s33 2.464120724 0.04294479 156
## s34 2.245215031 0.04294479 156
## s35 2.045756316 0.04294479 156
## s36 1.864016963 0.04294479 156
## s37 1.698422834 0.04294479 156
## s38 1.547539630 0.04294479 156
## s39 1.410060475 0.04294479 156
## s40 1.284794589 0.04294479 156
## s41 1.170656979 0.04294479 156
## s42 1.066659039 0.04294479 156
## s43 0.971899990 0.04294479 156
## s44 0.885559074 0.04294479 156
## s45 0.806888446 0.04294479 156
## s46 0.735206701 0.04294479 156
## s47 0.669892964 0.04294479 156
## s48 0.610381519 0.04294479 156
## s49 0.556156907 0.04294479 156
## s50 0.506749461 0.04294479 156
## s51 0.461731235 0.04294479 156
## s52 0.420712305 0.04294479 156
## s53 0.383337383 0.04294479 156
## s54 0.349282747 0.04294479 156
## s55 0.318253430 0.04294479 156
## s56 0.289980673 0.04294479 156
## s57 0.264219589 0.04294479 156
## s58 0.240747049 0.04294479 156
## s59 0.219359745 0.04294479 156
## s60 0.199872430 0.04294479 156
## s61 0.182116314 0.04294479 156
## s62 0.165937602 0.04294479 156
## s63 0.151196163 0.04294479 156
## s64 0.137764313 0.04294479 156
## s65 0.125525711 0.04294479 156
## s66 0.114374353 0.04294479 156
## s67 0.104213650 0.04294479 156
## s68 0.094955597 0.04294479 156
## s69 0.086520003 0.04294479 156
## s70 0.078833805 0.04294479 156
## s71 0.071830427 0.04294479 156
## s72 0.065449210 0.04294479 156
## s73 0.059634882 0.04294479 156
## s74 0.054337084 0.04294479 156
## s75 0.049509927 0.04294479 156

```

```

## s76 0.045111602 0.04294479 156
## s77 0.041104011 0.04294479 156
## s78 0.037452444 0.04294479 156
## s79 0.034125272 0.04294479 156
## s80 0.031093677 0.04601227 156
## s81 0.028331401 0.04601227 156
## s82 0.025814517 0.04601227 156
## s83 0.023521226 0.04294479 156
## s84 0.021431666 0.04294479 156
## s85 0.019527735 0.04294479 156
## s86 0.017792945 0.03987730 156
## s87 0.016212269 0.03680982 156
## s88 0.014772016 0.03680982 156
## s89 0.013459711 0.03680982 156
## s90 0.012263987 0.03680982 156
## s91 0.011174488 0.03680982 156
## s92 0.010181777 0.03680982 156
## s93 0.009277256 0.03680982 156
## s94 0.008453091 0.03680982 156
## s95 0.007702141 0.03680982 156
## s96 0.007017904 0.03680982 156
## s97 0.006394453 0.03987730 156
## s98 0.005826387 0.03987730 156
## s99 0.005308787 0.03987730 156

```

Identifier visuellement les éléments importants dans cette liste n'est pas aisé. Nous le faisons par le calcul. Tout dabord, nous affichons l'erreur minimale

```

## [1] 0.03680982
## [1] 0.01621227
## [1] -4.121987
## [1] 53.08787
## [1] 3.971948

```

Prédiction sur l'échantillon test

```

##    1    2
## 1 "0" "0"
## 2 "0" "0"
## 3 "0" "0"
## 5 "0" "0"
## 6 "0" "0"
## 7 "0" "0"

```

Nous avons une matrice avec autant de colonnes que de valeurs de  $\lambda$  essayé. Le nombre de lignes correspond à ntest, l'effectif de l'échantillon test.

Nous calculons les taux d'erreur pour les deux prédictions.

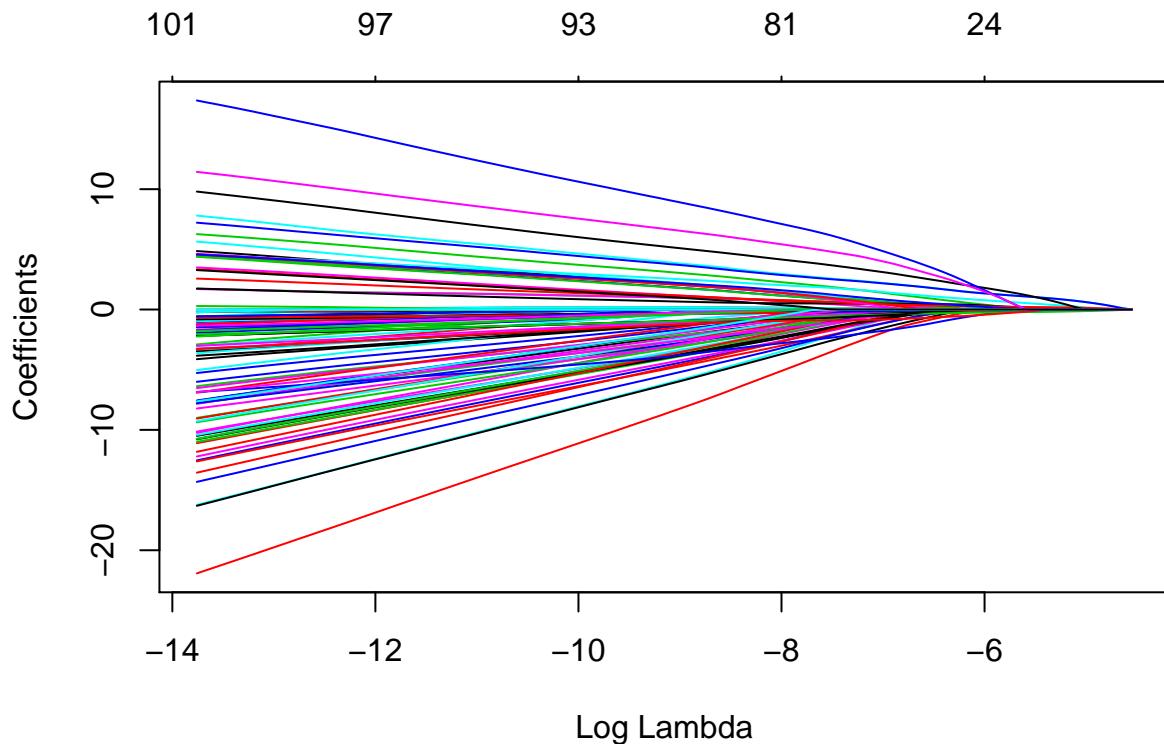
```

## [1] 0.9928959
## [1] 1

```

Les deux font mieux que la prédiction systématique. Malgré les embuches, la régression ridge a su tirer parti des données.

### 3.1.4 Régression elasticnet



```

## [1] 1.058564e-02 9.645245e-03 8.788388e-03 8.007652e-03 7.296274e-03
## [6] 6.648094e-03 6.057495e-03 5.519364e-03 5.029039e-03 4.582273e-03
## [11] 4.175196e-03 3.804284e-03 3.466321e-03 3.158383e-03 2.877801e-03
## [16] 2.622145e-03 2.389201e-03 2.176951e-03 1.983557e-03 1.807343e-03
## [21] 1.646784e-03 1.500488e-03 1.367189e-03 1.245731e-03 1.135064e-03
## [26] 1.034228e-03 9.423501e-04 8.586343e-04 7.823556e-04 7.128533e-04
## [31] 6.495254e-04 5.918233e-04 5.392474e-04 4.913421e-04 4.476926e-04
## [36] 4.079209e-04 3.716823e-04 3.386631e-04 3.085772e-04 2.811640e-04
## [41] 2.561862e-04 2.334273e-04 2.126903e-04 1.937955e-04 1.765792e-04
## [46] 1.608924e-04 1.465992e-04 1.335757e-04 1.217092e-04 1.108969e-04
## [51] 1.010451e-04 9.206854e-05 8.388943e-05 7.643692e-05 6.964648e-05
## [56] 6.345927e-05 5.782173e-05 5.268500e-05 4.800461e-05 4.374002e-05
## [61] 3.985427e-05 3.631373e-05 3.308772e-05 3.014830e-05 2.747001e-05
## [66] 2.502965e-05 2.280608e-05 2.078005e-05 1.893401e-05 1.725196e-05
## [71] 1.571935e-05 1.432288e-05 1.305048e-05 1.189111e-05 1.083474e-05

```

```

## [76] 9.872208e-06 8.995188e-06 8.196080e-06 7.467963e-06 6.804530e-06
## [81] 6.200034e-06 5.649240e-06 5.147377e-06 4.690099e-06 4.273443e-06
## [86] 3.893802e-06 3.547888e-06 3.232703e-06 2.945519e-06 2.683847e-06
## [91] 2.445421e-06 2.228177e-06 2.030232e-06 1.849872e-06 1.685534e-06
## [96] 1.535796e-06 1.399360e-06 1.275045e-06 1.161773e-06 1.058565e-06

##          [,1] [,2]
## [1,] 1.058564e-02    0
## [2,] 9.645245e-03    4
## [3,] 8.788388e-03    4
## [4,] 8.007652e-03    5
## [5,] 7.296274e-03    5
## [6,] 6.648094e-03    5
## [7,] 6.057495e-03    6
## [8,] 5.519364e-03    6
## [9,] 5.029039e-03    8
## [10,] 4.582273e-03   10
## [11,] 4.175196e-03   11
## [12,] 3.804284e-03   13
## [13,] 3.466321e-03   19
## [14,] 3.158383e-03   19
## [15,] 2.877801e-03   21
## [16,] 2.622145e-03   21
## [17,] 2.389201e-03   24
## [18,] 2.176951e-03   24
## [19,] 1.983557e-03   28
## [20,] 1.807343e-03   35
## [21,] 1.646784e-03   37
## [22,] 1.500488e-03   39
## [23,] 1.367189e-03   42
## [24,] 1.245731e-03   50
## [25,] 1.135064e-03   54
## [26,] 1.034228e-03   59
## [27,] 9.423501e-04   62
## [28,] 8.586343e-04   62
## [29,] 7.823556e-04   64
## [30,] 7.128533e-04   67
## [31,] 6.495254e-04   69
## [32,] 5.918233e-04   72
## [33,] 5.392474e-04   74
## [34,] 4.913421e-04   75
## [35,] 4.476926e-04   78
## [36,] 4.079209e-04   81
## [37,] 3.716823e-04   81
## [38,] 3.386631e-04   81
## [39,] 3.085772e-04   81
## [40,] 2.811640e-04   81
## [41,] 2.561862e-04   81
## [42,] 2.334273e-04   81
## [43,] 2.126903e-04   82
## [44,] 1.937955e-04   82
## [45,] 1.765792e-04   85
## [46,] 1.608924e-04   84
## [47,] 1.465992e-04   85

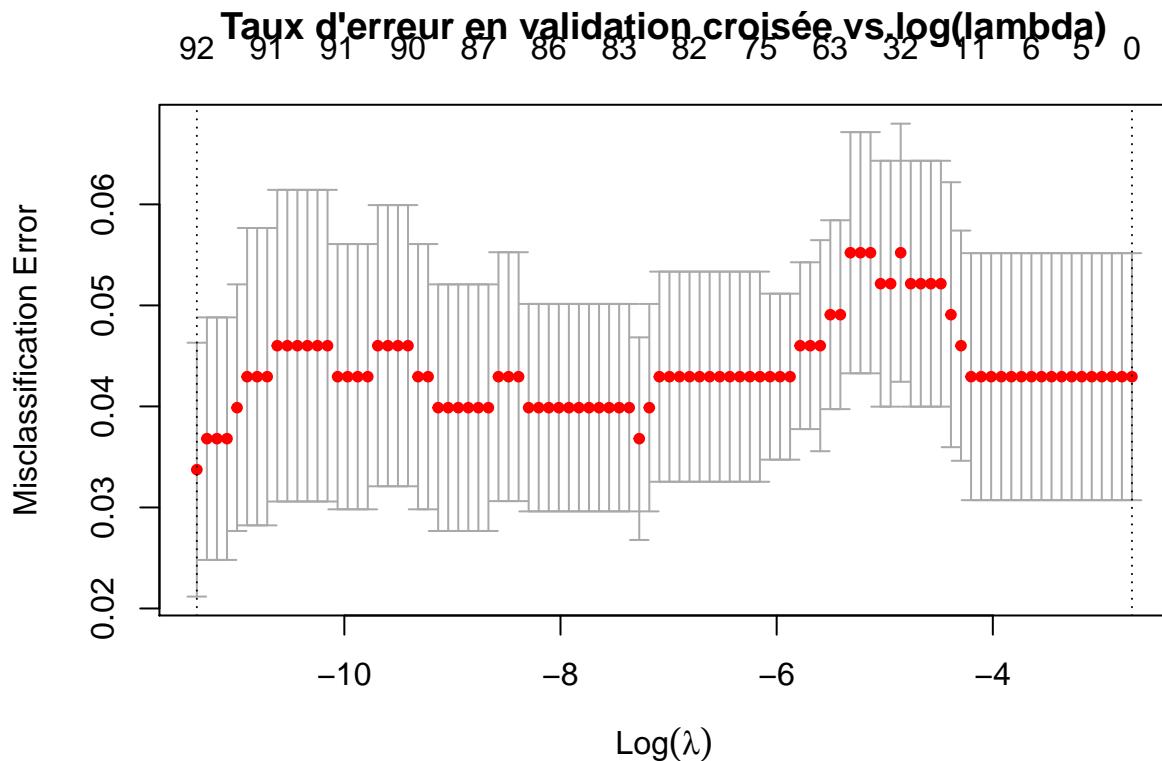
```

```

## [48,] 1.335757e-04 85
## [49,] 1.217092e-04 85
## [50,] 1.108969e-04 86
## [51,] 1.010451e-04 87
## [52,] 9.206854e-05 87
## [53,] 8.388943e-05 87
## [54,] 7.643692e-05 87
## [55,] 6.964648e-05 87
## [56,] 6.345927e-05 89
## [57,] 5.782173e-05 93
## [58,] 5.268500e-05 93
## [59,] 4.800461e-05 93
## [60,] 4.374002e-05 93
## [61,] 3.985427e-05 93
## [62,] 3.631373e-05 93
## [63,] 3.308772e-05 94
## [64,] 3.014830e-05 94
## [65,] 2.747001e-05 94
## [66,] 2.502965e-05 95
## [67,] 2.280608e-05 96
## [68,] 2.078005e-05 97
## [69,] 1.893401e-05 97
## [70,] 1.725196e-05 97
## [71,] 1.571935e-05 97
## [72,] 1.432288e-05 97
## [73,] 1.305048e-05 97
## [74,] 1.189111e-05 97
## [75,] 1.083474e-05 97
## [76,] 9.872208e-06 97
## [77,] 8.995188e-06 97
## [78,] 8.196080e-06 97
## [79,] 7.467963e-06 97
## [80,] 6.804530e-06 97
## [81,] 6.200034e-06 97
## [82,] 5.649240e-06 97
## [83,] 5.147377e-06 96
## [84,] 4.690099e-06 96
## [85,] 4.273443e-06 97
## [86,] 3.893802e-06 96
## [87,] 3.547888e-06 98
## [88,] 3.232703e-06 98
## [89,] 2.945519e-06 98
## [90,] 2.683847e-06 98
## [91,] 2.445421e-06 98
## [92,] 2.228177e-06 98
## [93,] 2.030232e-06 98
## [94,] 1.849872e-06 99
## [95,] 1.685534e-06 99
## [96,] 1.535796e-06 99
## [97,] 1.399360e-06 101
## [98,] 1.275045e-06 101
## [99,] 1.161773e-06 101
## [100,] 1.058565e-06 101

```

On procède par validation croisée pour l'optimisation



```
## [1] 1.159657e-05
```

```
## ye3
##      0      1
## 30980 1255
```

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

Après la validation croisée, on a fait la prédiction en utilisant le jeu de test et en prenant la valeur minimale de  $\lambda$  obtenue par validation croisée. L'erreur de prédiction est ensuite calculée.

## TP4 Exemples sur le lien du mail.

### Exemple 1

#### Generate data

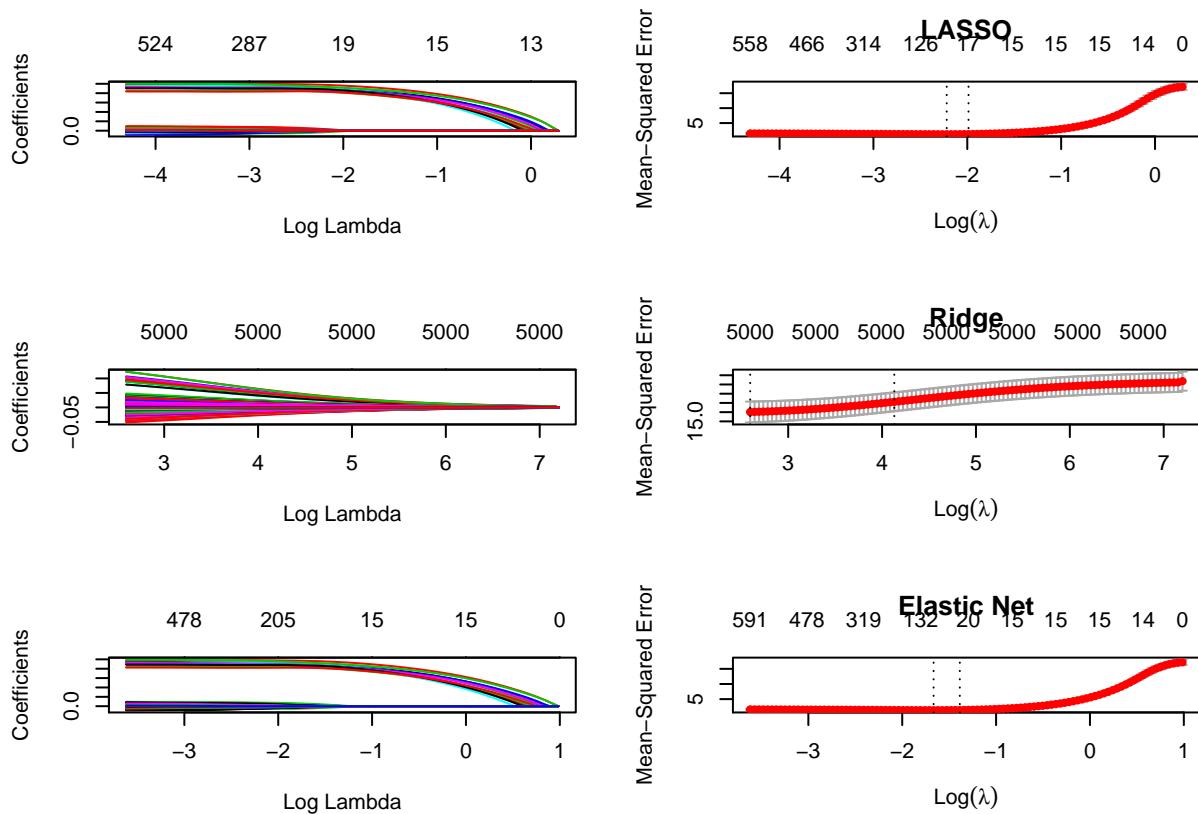
A partir de la loi normale, nous générerons une matrice de données de taille  $n = 1000$  (individus) et  $p = 5000$  (variables). Nous la partageons ensuite en un jeu d'apprentissage (2/3) et un jeu de test(1/3). Nous prenons dans l'exemple 1, 15 comme nombre de vrais prédicteurs.

## Fit models

Ici, dans un premier temps, nous implémentons les modèles Lasso, Ridge et Elastic Net avec notre jeu d'apprentissage. Dans un second temps, nous procérons par la méthode de validation croisée en utilisant 10-folds pour différentes valeurs du paramètre alpha pris dans la séquence de 0 à 1 par pas de 0.1.

Plot solution path and cross-validated MSE as function of  $\lambda$

On trace ici les erreurs quadratiques moyennes des différents modèles.



## MSE on test set

Nous faisons des prédictions sur le jeu de test en utilisant les différents modèles et nous calculons par la suite les erreurs quadratiques moyennes commises afin de savoir lequel de ces modèles est le meilleur.

Parmi les modèles implantés, compte tenu des erreurs obtenues, le modèle Lasso est le meilleur dans cet exemple. C'est le Lasso qui marche parce que le nombre de vrais prédicteurs choisi est très petit (15 vrais prédicteurs). Lasso est un modèle parcimonieux qui ne choisit qu'une variable dès qu'il y a colinéarité entre certaines variables. De plus il ne sélectionne pas plus que  $n$  variables quand  $p$  est très supérieur à  $n$ . Le lasso est bon pour capter un petit nombre de prédicteurs à travers plusieurs.

## Exemple 2

Nous refaisons la même chose qu'à l'exemple 1 en prenant 1500 comme nombre de vrais prédicteurs.

## Generate Data

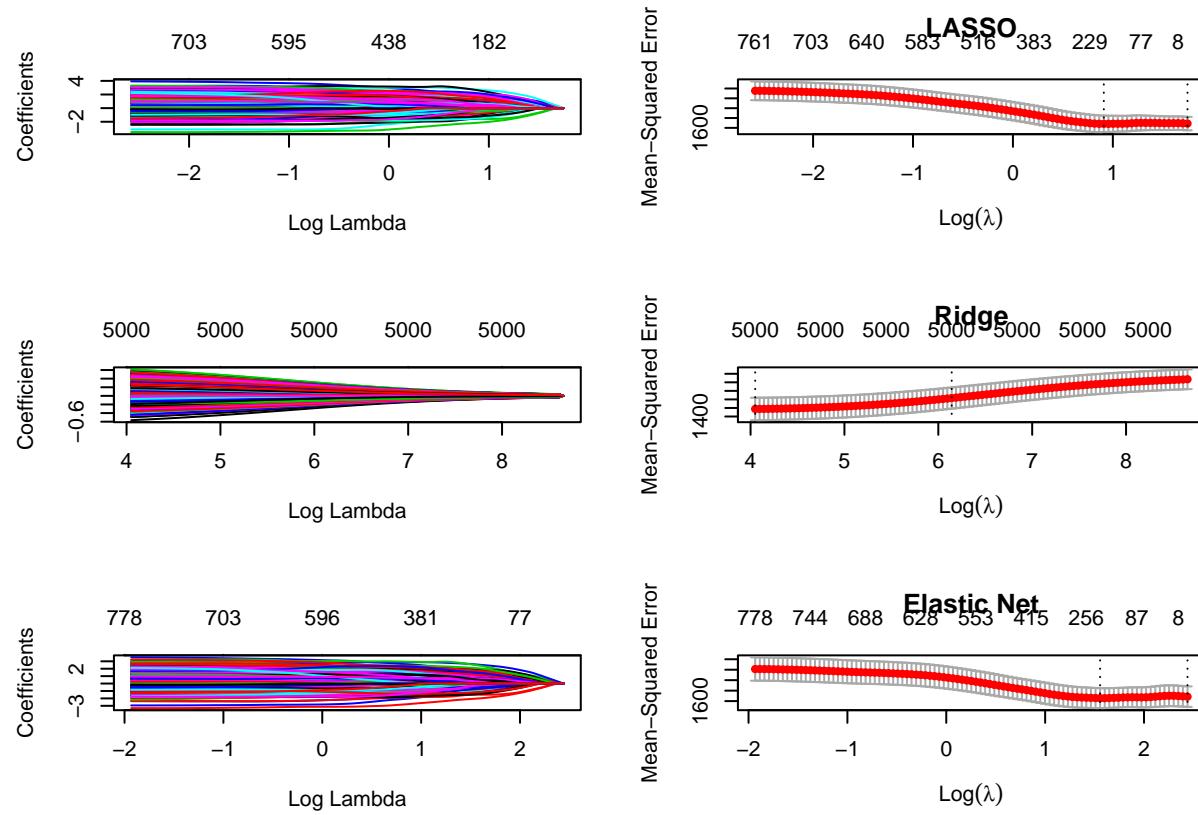
A partir de la loi normale, nous générerons une matrice de données de taille  $n = 1000$  (individus) et  $p = 5000$  (variables). Nous la partageons ensuite en un jeu d'apprentissage (2/3) et un jeu de test(1/3). Nous prenons dans l'exemple 2, 1500 comme nombre de vrais prédicteurs.

## Fit models

Nous implémentons les modèles Lasso, Ridge et Elastic Net avec notre jeu d'apprentissage. Ensuite, nous procérons par la méthode de validation croisée en utilisant 10-folds pour différentes valeurs du paramètre alpha pris dans la séquence de 0 à 1 par pas de 0.1

Plot solution path and cross-validated MSE as function of  $\lambda$

On trace ici les erreurs quadratiques moyennes des différents modèles.



## MSE on test set

Nous faisons des prédictions sur le jeu de test en utilisant les différents modèles et nous calculons par la suite les erreurs quadratiques moyennes commises afin de savoir lequel de ces modèles est le meilleur.

Le modèle Lasso ne marche pas dans ce cas car le nombre de vrais prédicteurs (1500) est supérieur au nombre d'individus (1000) et Laaso ne peut pas choisir plus de 1000 prédicteurs. Compte-tenu des valeurs des obtenues, dans cet exemple c'est le modèle Ridge qui est le meilleur modèle. Ce modèle est généralement très bon pour la prédiction (même en grande dimension) mais n'est pas très interprétable.

### Exemple 3

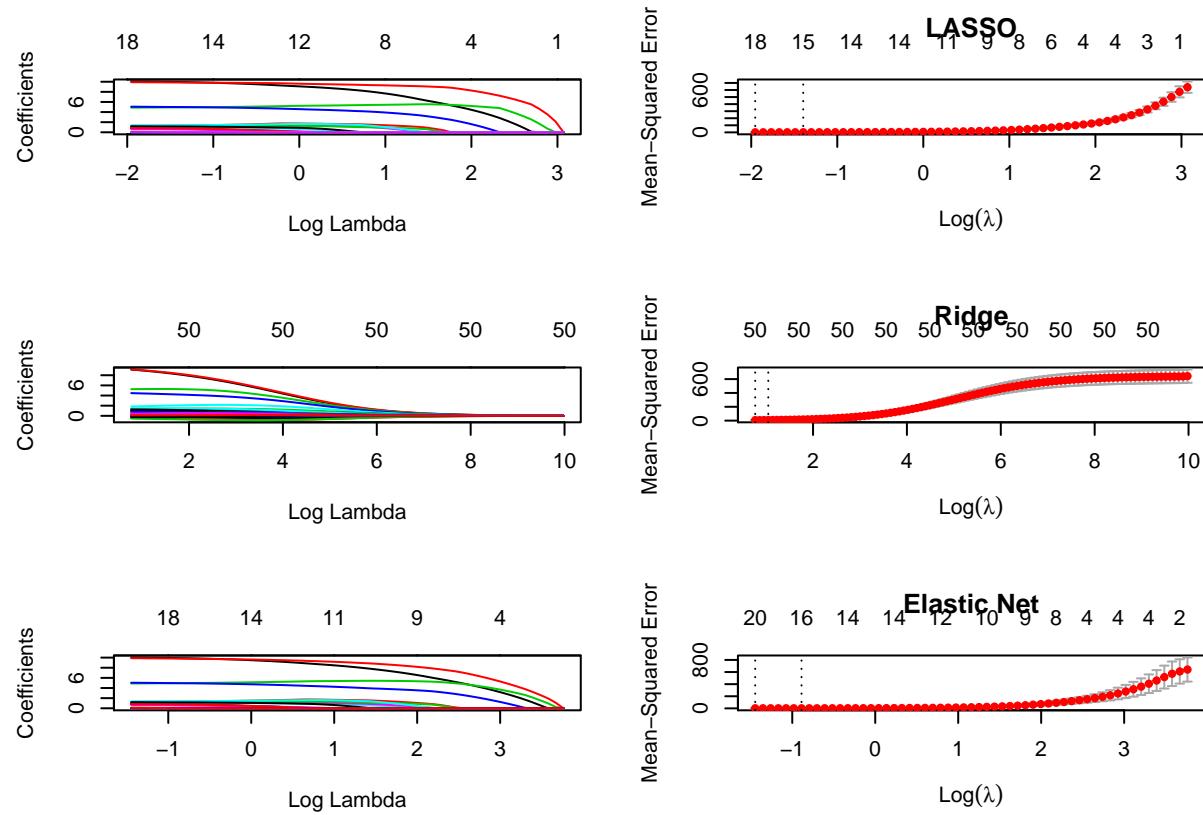
Dans cet exemple, à partir de la loi normale, nous générerons une matrice de données de taille  $n = 100$  (individus) et  $p = 50$  (variables). Nous la partagerons ensuite en un jeu d'apprentissage et un jeu de test.

#### Fit models

Nous implémentons les modèles Lasso, Ridge et Elastic Net avec notre jeu d'apprentissage. Ensuite, nous procéderons par la méthode de validation croisée en utilisant 10-folds pour différentes valeurs du paramètre alpha pris dans la séquence de 0 à 1 par pas de 0.1

Plot solution path and cross-validated MSE as function of  $\lambda$

On trace ici les erreurs quadratiques moyennes des différents modèles.



#### MSE on test set

Nous faisons des prédictions sur le jeu de test en utilisant les différents modèles et nous calculons par la suite les erreurs quadratiques moyennes commises afin de savoir lequel de ces modèles est le meilleur.

Dans cet exemple, compte-tenu des erreurs obtenues, nous pouvons dire que c'est le modèle Elastic Net qui est le meilleur modèle. On peut aussi voir que la meilleure solution est proche de Ridge mais pour  $\alpha = 0$  on obtient une erreur quadratique un peu plus grande comparée aux erreurs pour les autres valeurs de  $\alpha$ .

On peut conclure, au regard de nos divers résultats, qu'en fonction de la complexité du modèle, chacune des trois méthodes (Lasso, Ridge et Elastic Net) peut marcher.

## **DEVOIR: Différence entre la descente de gradient et la descente de coordonnées**

La Descente de Gradient est un algorithme d'optimisation qui permet de trouver le minimum de n'importe quelle fonction convexe, c'est-à-dire une fonction qui a l'allure d'une belle vallée qui descend progressivement vers un unique minimum. A l'inverse, une fonction non-convexe est une fonction qui présente plusieurs minimums locaux et l'algorithme de Gradient Descent ne doit pas être utilisé sur ces fonctions, au risque de se bloquer au premier minima rencontré. En Machine Learning, on va utiliser l'algorithme de la Descente de Gradient dans les problèmes d'apprentissage supervisé pour minimiser la fonction de coût, qui justement est une fonction convexe (par exemple l'erreur quadratique moyenne). C'est cet algorithme qui fait que la machine apprend, c'est-à-dire trouve le meilleur modèle. L'algorithme est itératif et procède donc par améliorations successives. L'algorithme du gradient est également connu sous le nom d'algorithme de la plus forte pente ou de la plus profonde descente (steepest descent, en anglais) parce que le gradient est la pente de la fonction linéarisée au point courant et est donc, localement, sa plus forte pente (notion qui dépend du produit scalaire).

Alors que l'algorithme de descente de coordonnées typique utilise le gradient le long de chaque coordonnée, ou le long d'un hyperplan obtenu à partir d'un groupe de coordonnées, pour trouver le minimiseur le long de cette direction. La descente de coordonnées est un algorithme d'optimisation qui minimise successivement le long des directions de coordonnées pour trouver le minimum d'une fonction. À chaque itération, l'algorithme détermine une coordonnée ou un bloc de coordonnées via une règle de sélection de coordonnées, puis minimise exactement ou inexactement sur l'hyperplan de coordonnées correspondant tout en fixant toutes les autres coordonnées ou blocs de coordonnées. Une recherche de ligne le long de la direction des coordonnées peut être effectuée à l'itération actuelle pour déterminer la taille de pas appropriée. La descente de coordonnées est applicable à la fois dans des contextes différentiables et sans dérivé. La descente de coordonnées est basée sur l'idée que la minimisation d'une fonction multivariable peut être atteint en le minimisant dans une direction à la fois, c'est-à-dire en résolvant des problèmes d'optimisation univariés (ou du moins beaucoup plus simples) dans une boucle. Dans le cas le plus simple de descente de coordonnées cycliques, on itère cycliquement dans les directions, une à la fois, en minimisant la fonction objectif par rapport à chaque direction de coordonnées à la fois.

Malgré le taux d'apprentissage de la procédure de descente de gradient (qui pourrait en effet accélérer la convergence), la comparaison entre les deux est juste au moins en termes de complexité.

## **TP5 Analyse de données fonctionnelles**

### **1.2 Partie I**

```
## Loading required package: splines

##
## Attaching package: 'fda'

## The following object is masked from 'package:graphics':
## 
##      matplot
```

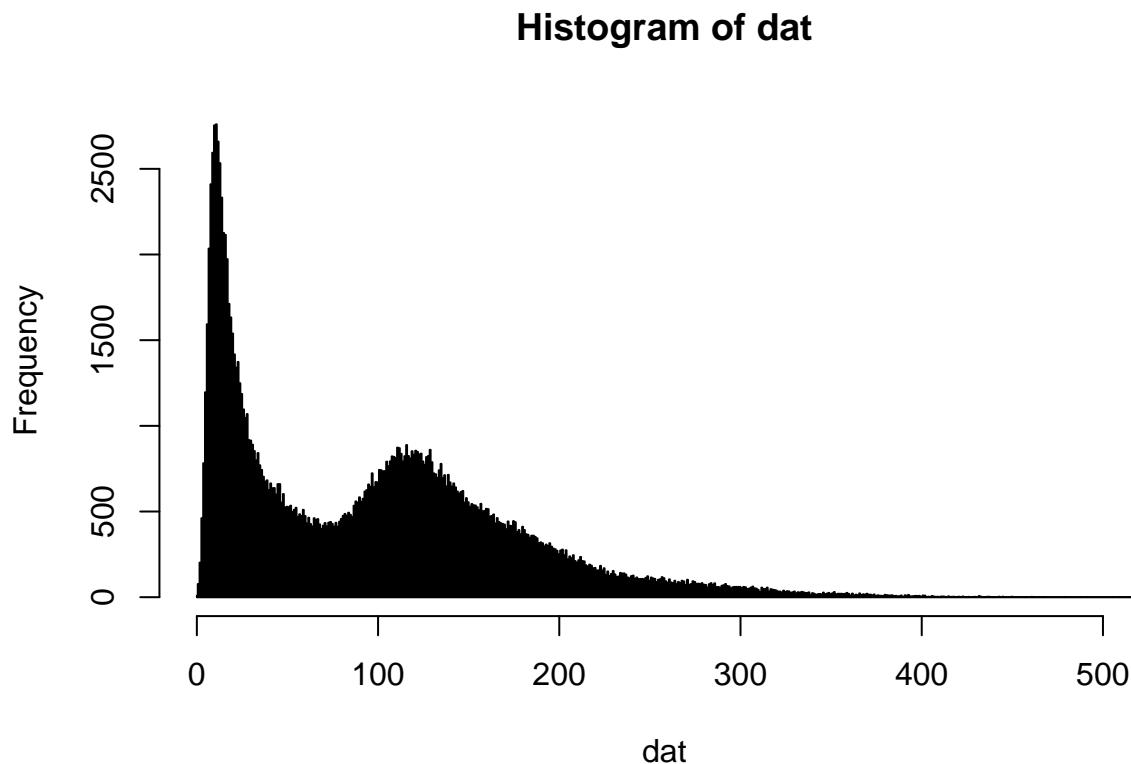
#### **1.2.1 Examen des données**

On charge les données et on retire les individus ayant des données manquantes de la manière suivante

La variable heures contient le temps dans la journée en heures : toutes les 6 minutes (240 points).

1.

(a) On vérifie la répartition des valeurs pour les débits.



La distribution présente une queue qui nous fait penser à un processus de Poisson.

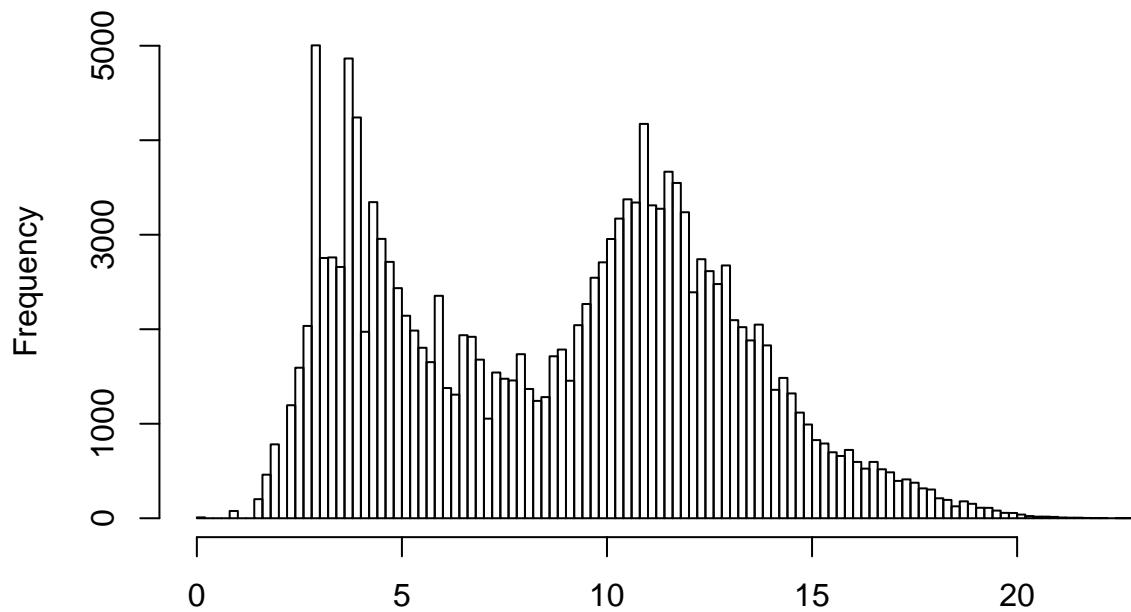
(b) Expliquer pourquoi on peut appliquer la méthode de stabilisation de la variance pour un processus de Poisson.

On peut appliquer la méthode de stabilisation de la variance pour un processus de Poisson car ce faisant, on obtient une variance égale à 1 dans le cas du processus de Poisson. Ce qui permet de réduire les grandes valeurs.

(c) Est-ce que cela améliore la queue de distribution?

Pour en savoir plus, traçons l'histogramme de la racine carrée de nos données

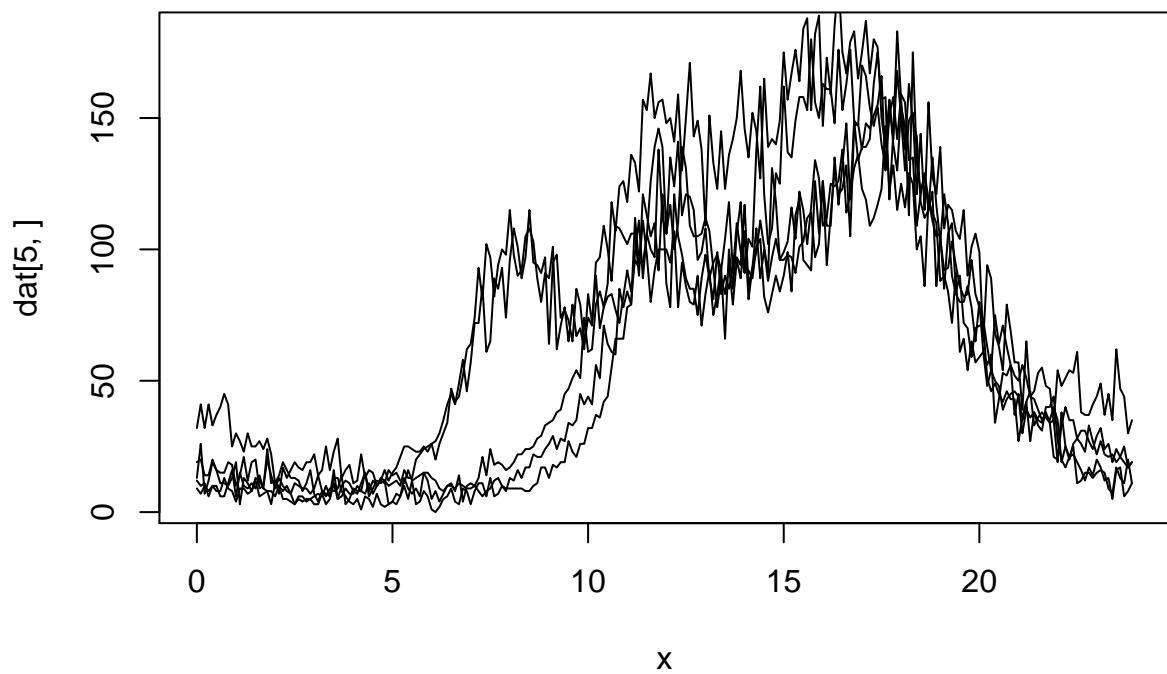
## Histogramme de la racine carrée des données

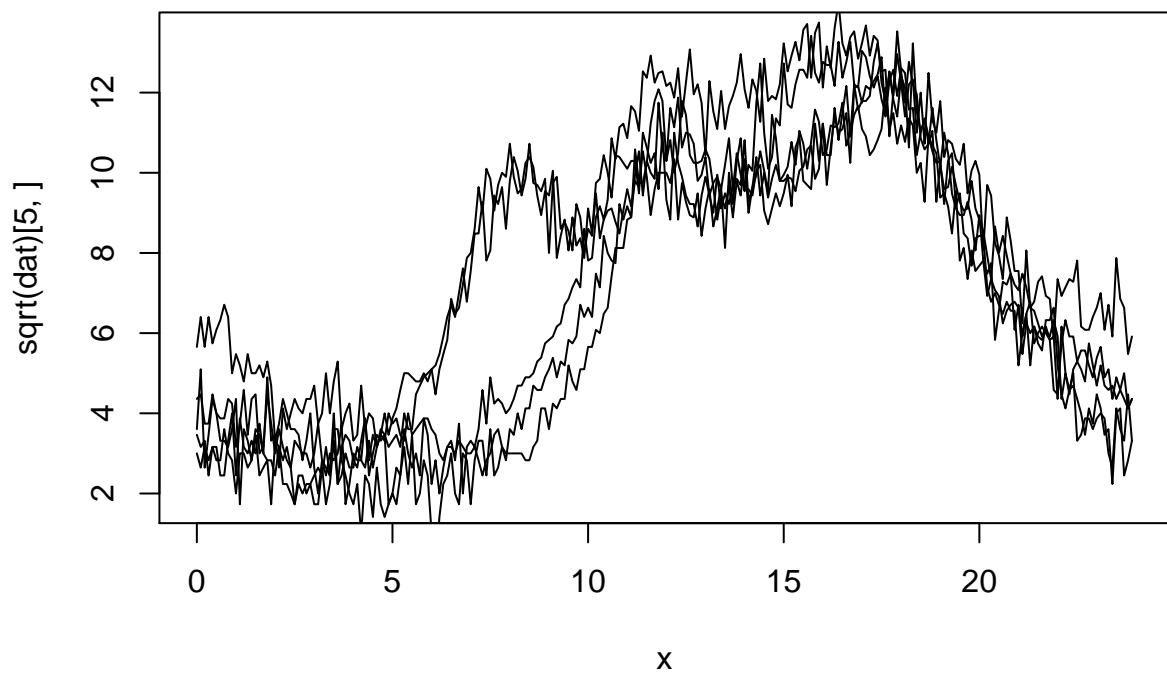


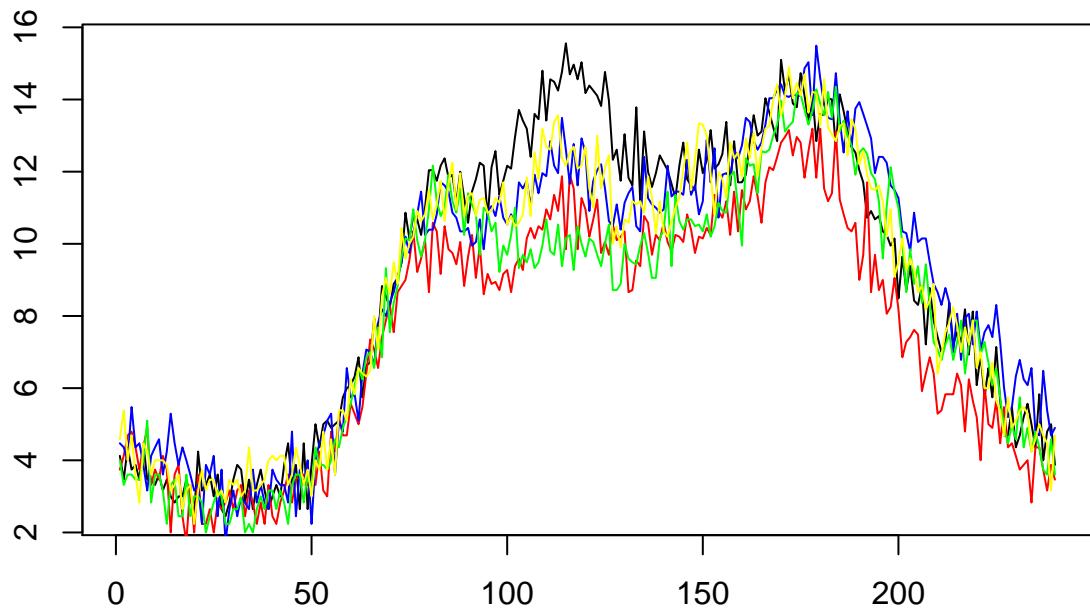
Cette répartition est moins asymétrique comparée au graphe précédent. On remarque aussi que la stabilisation améliore bien la queue de la distribution.

### 2. Traçons des courbes pour quelques journées (au moins 5)

Nous traçons ici quelques courbes des données brutes et des données stabilisées.

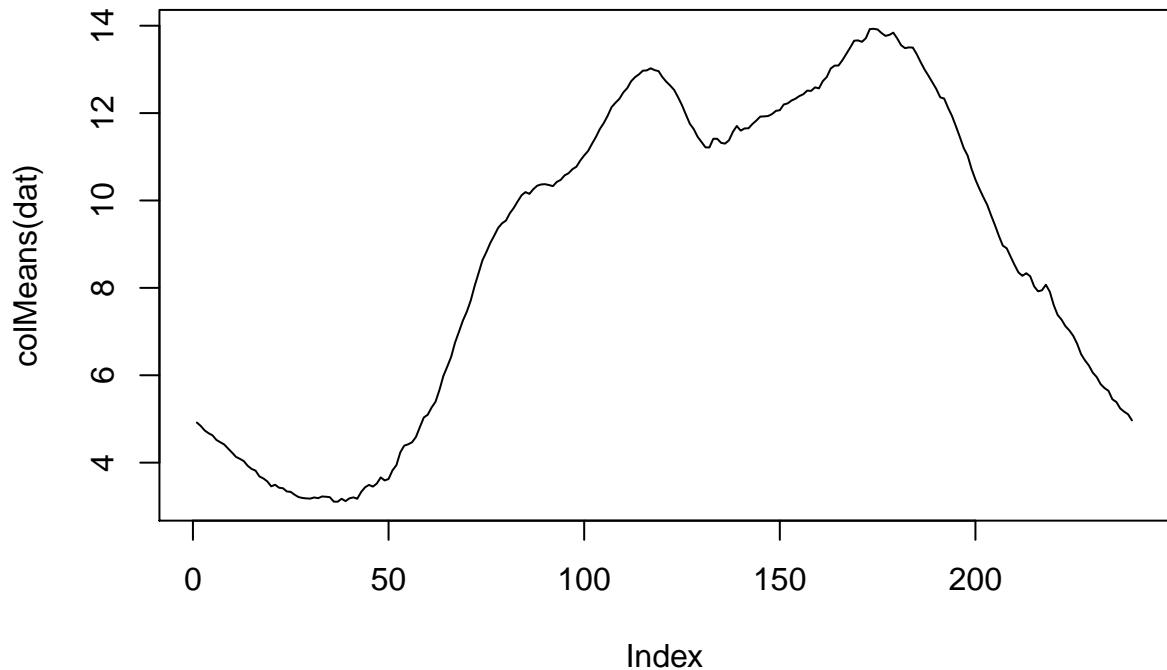






Les courbes des données stabilisées présentent moins de bruit comparées à celles des données brutes (bien sûr sans les données manquantes).

3. Traçons la courbe du débit moyen au cours de la journée.

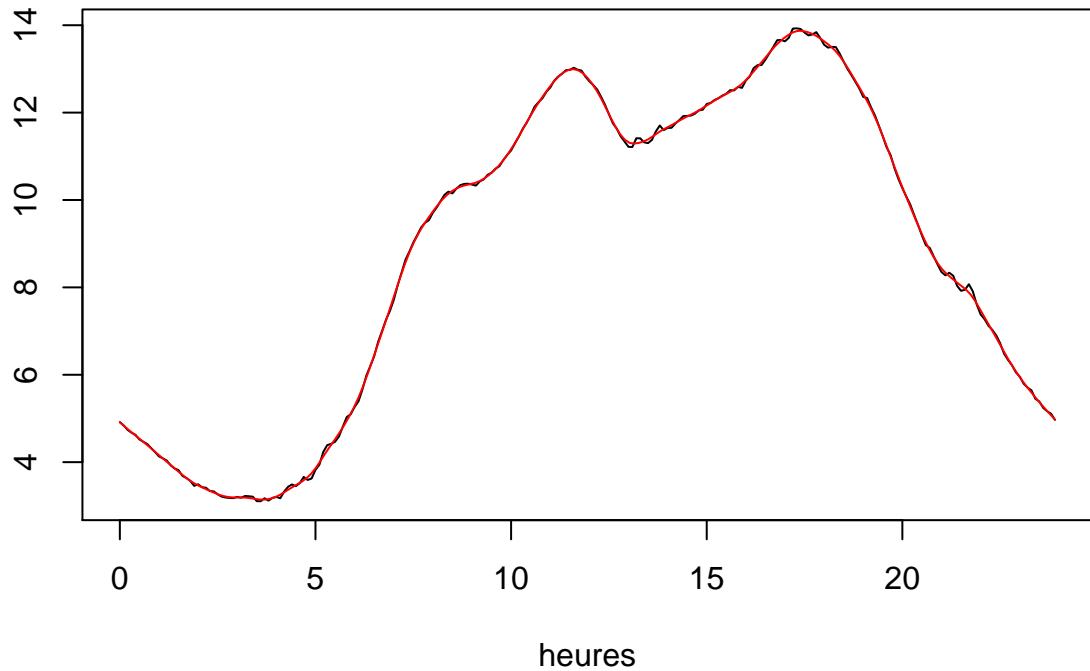


Cette courbe présente une forme bimodale. Les deux pics représentent une augmentation du trafic aux horaires correspondants. Elle est un peu proche de la répartition des débits pour les données stabilisées avec la racine carrée.

### 1.2.2 Lissage des données

#### 1. Lissage de la moyenne

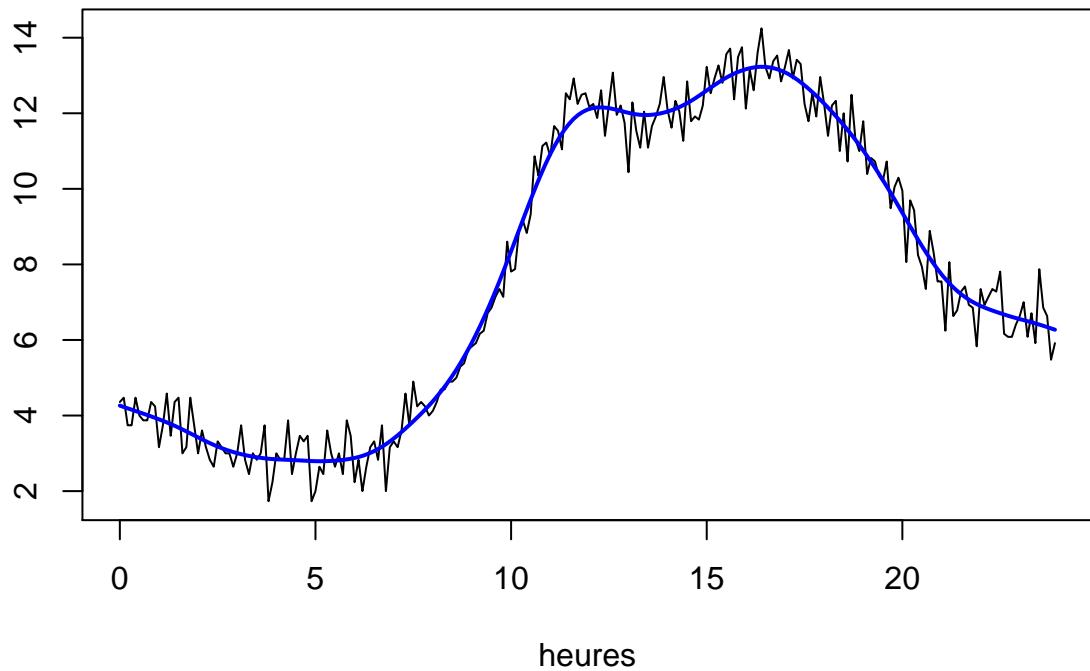
Ici, on essaie avec plusieurs valeurs de  $\lambda$  pour en trouver une qui lisse correctement la moyenne tout en gardant un *SSE* correct. Ensuite, nous traçons la moyenne correspondante.



En utilisant le code ci-dessus, pour les valeurs de  $\lambda$  choisies parmi (0.01,0.1,0.5,1,2,), on constate que pour  $\lambda = 0.01$ , on obtient une courbe plus lisse de la moyenne.

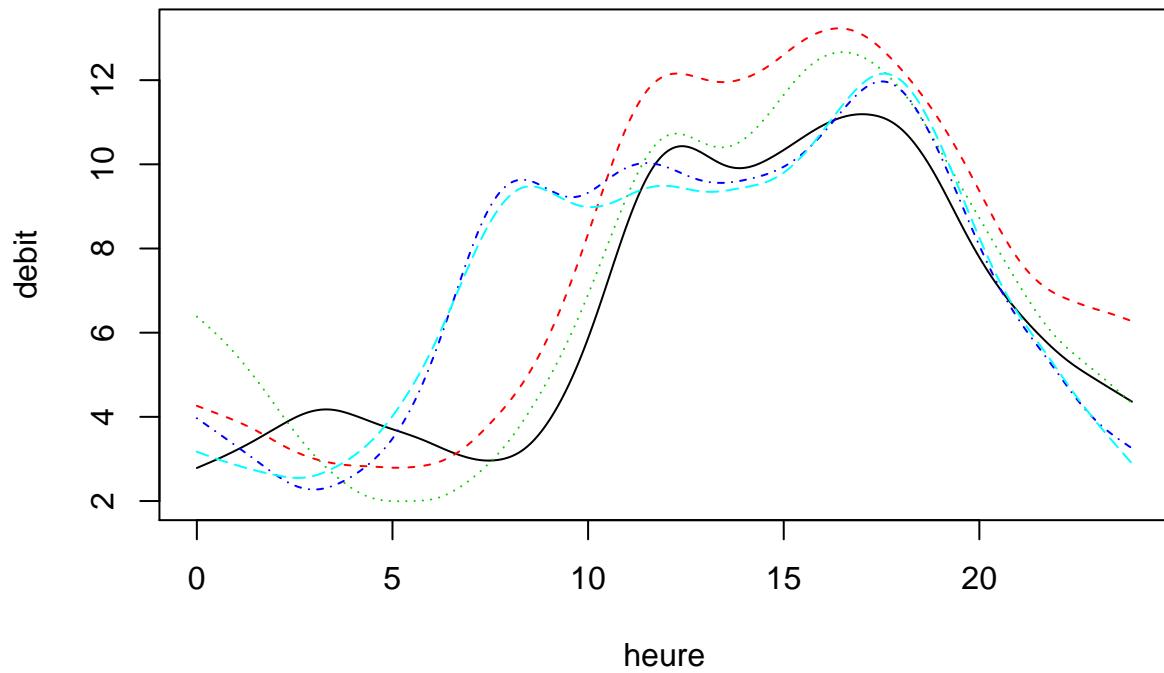
## 2. Lissage d'une courbe

On choisit de lisser la deuxième courbe du jeu de données. Nous faisons évoluer arbitrairement la valeur de  $\lambda$  jusqu'à obtenir une courbe assez lisse. On s'arrête à  $\lambda = 1.5$ . On représente en bleu sur le graphe ci-dessous, la courbe lissée.



### 3. Lissage de plusieurs courbes

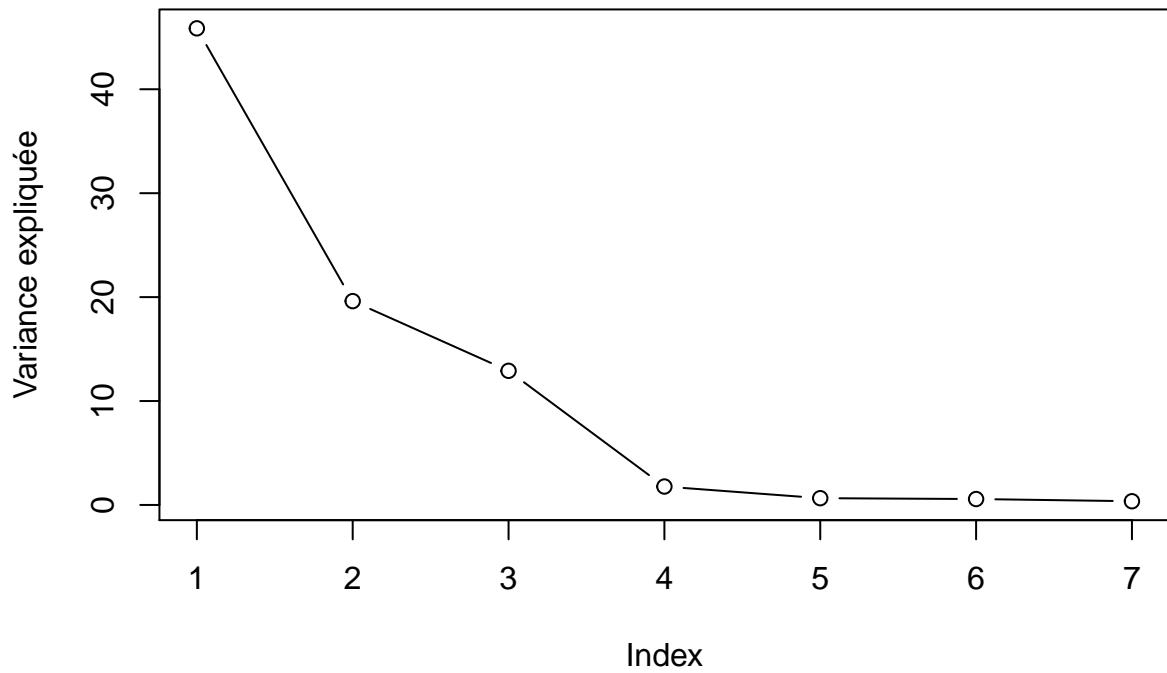
On lisse les mêmes courbes qui on été tracées dans la section précédente et on fait un tracé joint des courbes lissées (pour une valeur raisonnable de  $\lambda$ )



```
## [1] "done"
```

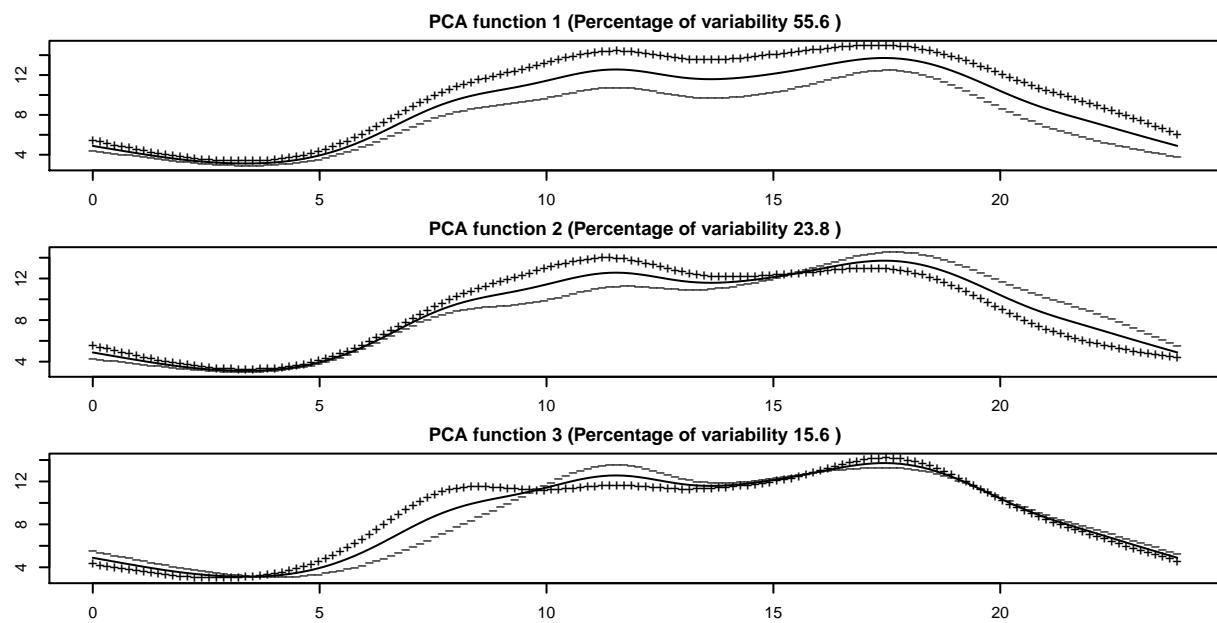
### 1.3 Partie II

On fait une ACP fonctionnelle sur les données disponibles (avec  $\lambda = 0$ ), et on représente la courbe de variance expliquée par les 7 premières valeurs propres.



On remarque que que  $\%lambda = 0$  marche très bien: la courbe de départ est bien lisse. Compte tenu de la décroissance de la courbe, on peut considérer les 3 ou les 4 premières composantes.

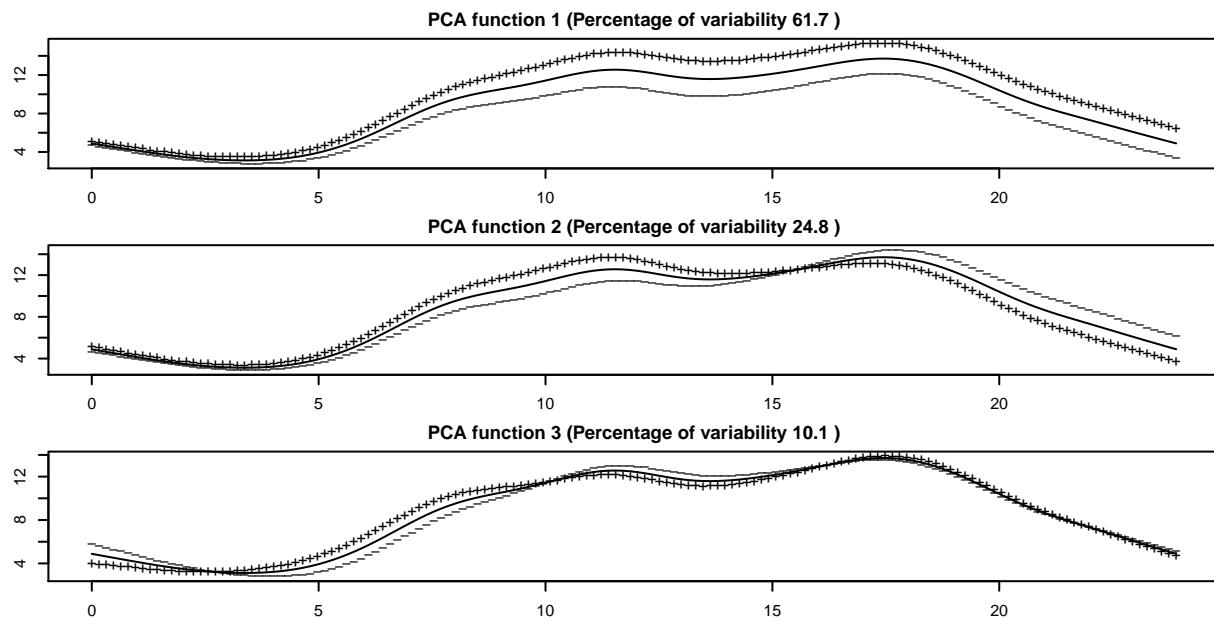
Représentons les 3 premiers axes principaux



Pour observer l'influence du lissage, on applique le même procédé avec un paramètre de lissage significatif. Prenons  $\lambda = 10$ .

```
## [1] 62 87 97 99 99 100 100 100
```

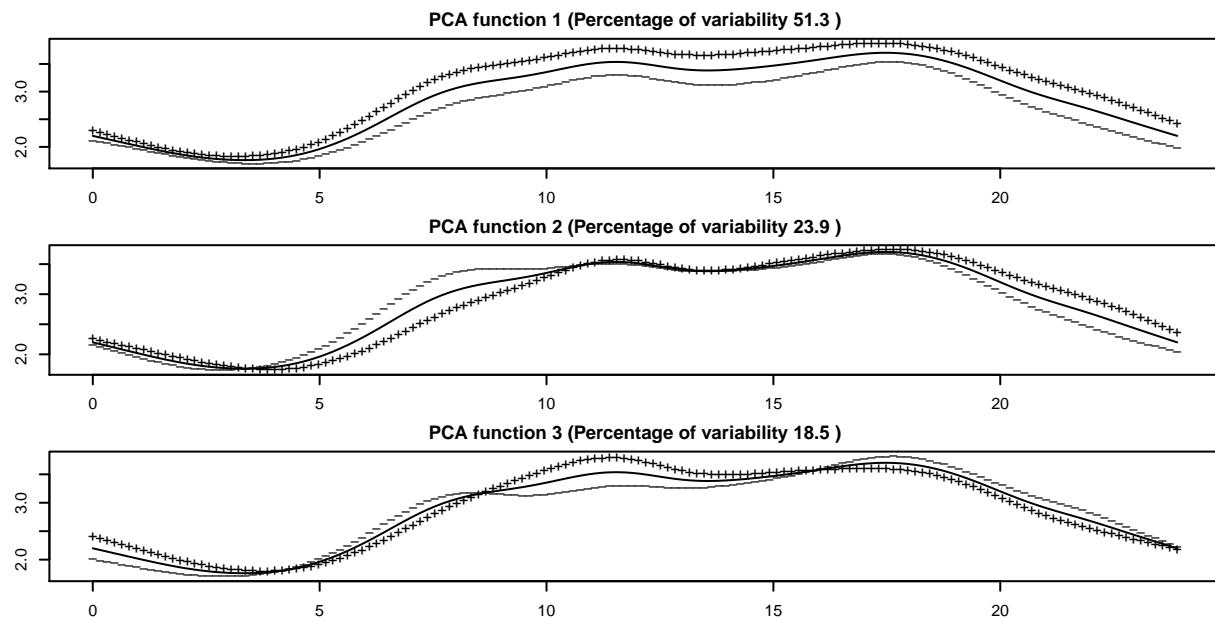
Les 3 premiers axes principaux sont représentés ci-dessous :



Dans cette partie, on essaie de regarder si l'ACP fonctionnelle sur la racine carrée des données est plus intéressante. On conserve la valeur  $\lambda = 0$ .

```
## [1] 51 75 94 96 97 98 99 99
```

On voit que la répartition des valeurs propres est plus régulière, même si l'inertie expliquée par les 3 premiers axes est presque la même. Les composantes principales ci-dessous sont toujours interprétables de la même manière, mais sont plus marquées. Globalement, cette analyse n'apporte pas grand-chose de neuf.



## Gestion des valeurs manquantes

On voudrait analyser aussi les journées dont le nombre de données manquantes est inférieur à 10 (mais non nul).

Nous faisons une sélection de ces données puis on affiche le nombre d'individus des différents jeux de données.

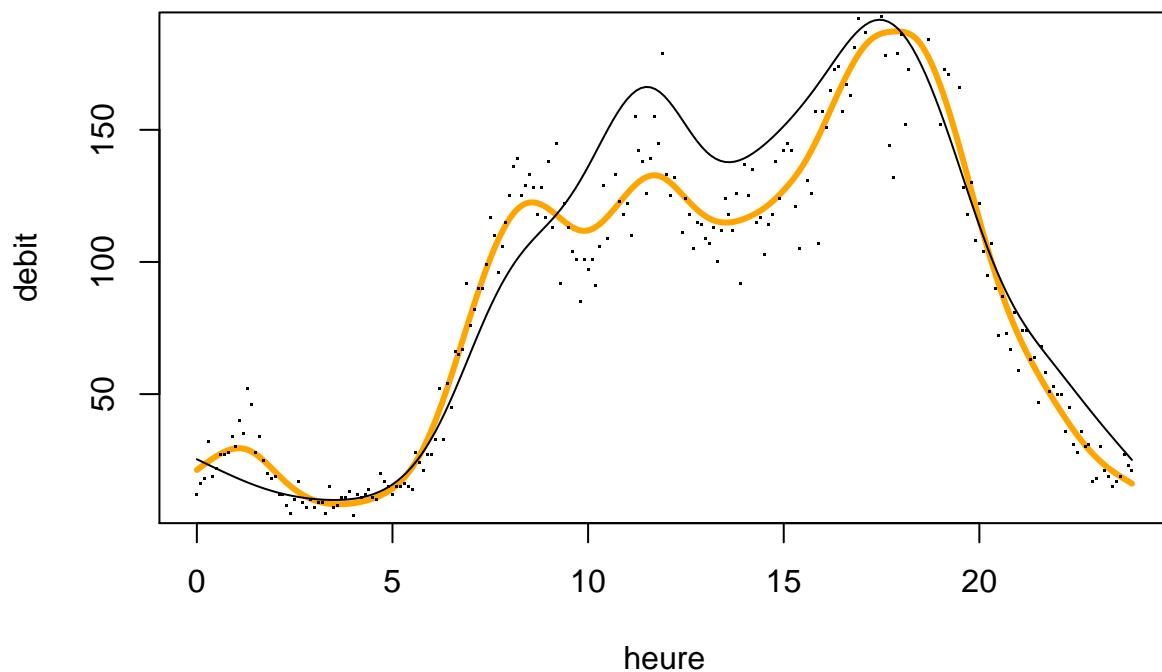
```
## [1] 761
## [1] 661
## [1] 77
```

Les données sélectionnées représentent à peu près 10% des données d'origine. Il y a un intérêt donc de les conserver.

Le code ci-dessous permet de lisser une journée avec des données manquantes. On choisit arbitrairement la courbe.

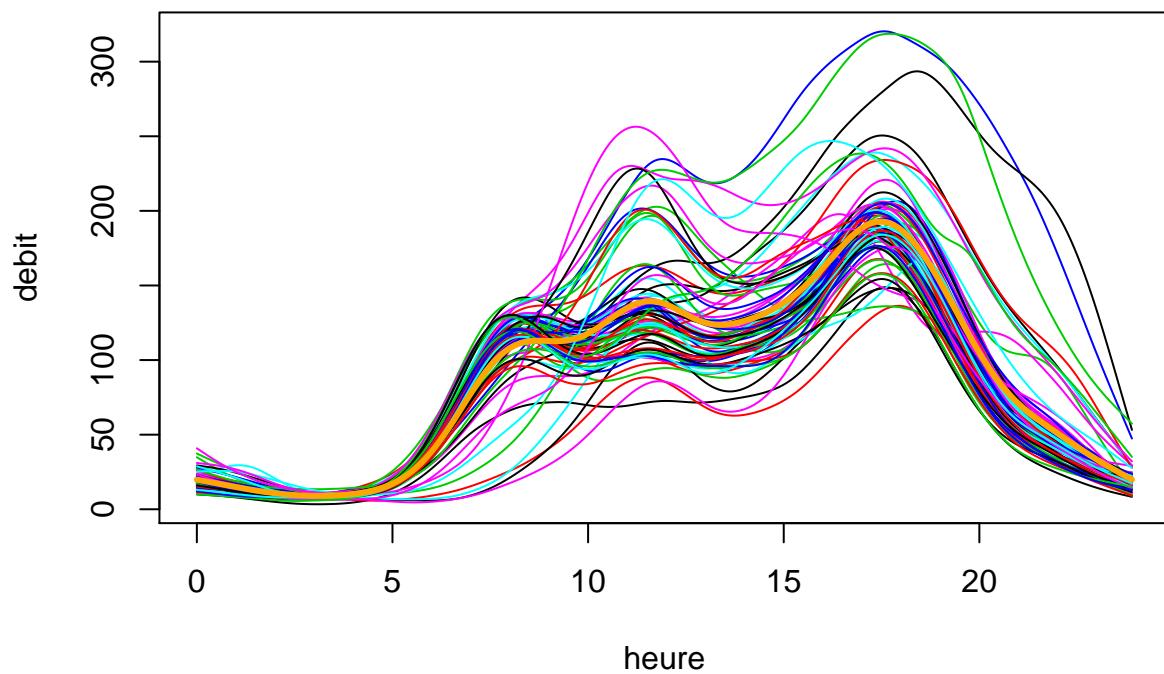
Afin de faire une comparaison, on représente les données avec la moyenne

## **donnees brutes et version lissee pour une journee reconstituee**

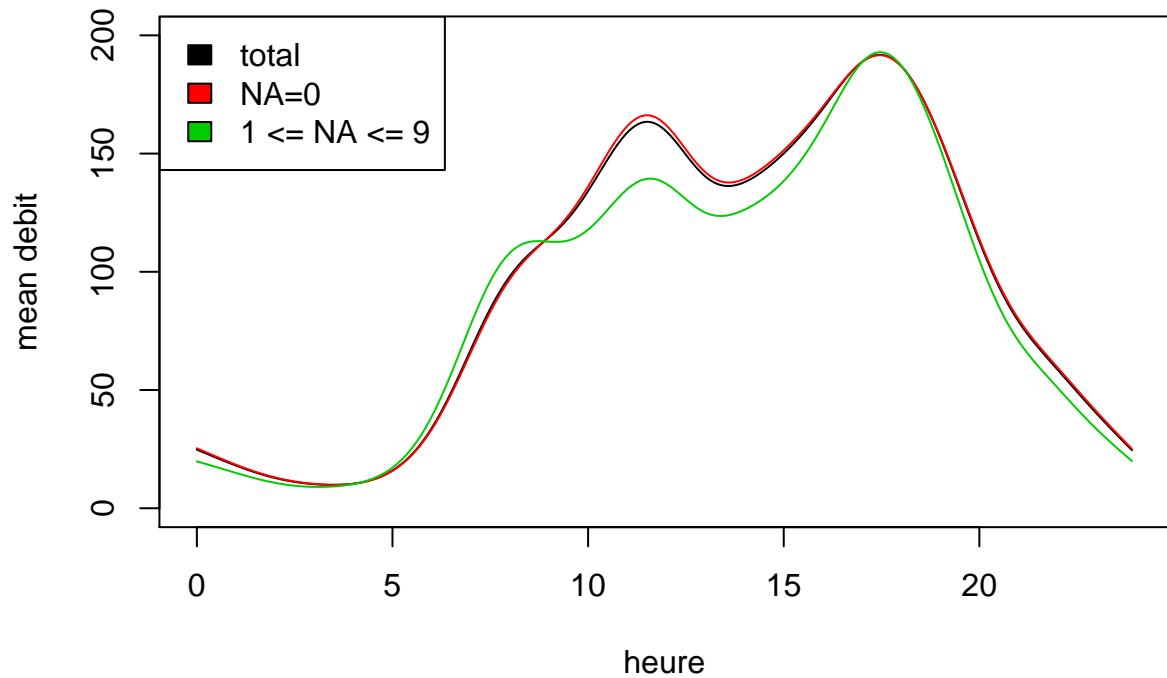


On fait la même chose dans une boucle. On va créer une matrice de coefficients dont les colonnes seront formées par les vecteurs de coefficients des courbes lisses individuelles. On utilise la fonction cbind() pour placer les coefficients de chaque courbe dans une matrice commune.

```
## [1] 242 77
```



On trace ensuite les courbes des moyennes des différentes données



On constate que les données pour lesquelles des données sont manquantes ont une moyenne de débit plus basse entre 8 et 16h.