Chapitre 3 : sélection de modèles et pénalisations

Les données `swiss` du package `faraway`

Standardized fertility measure and socio-economic indicators for each of 47 French-speaking provinces of Switzerland at about 1888.
A data frame with 47 observations on 6 variables, each of which is in percent, i.e., in [0, 100].

- `Fertility` common standardized fertility measure
- `Agriculture` % of males involved in agriculture as occupation
- `Examination` % draftees receiving highest mark on army examination
- `Education` % education beyond primary school for draftees.
- `Catholic` % 'catholic' (as opposed to 'protestant').
- `Infant.Mortality` live births who live less than 1 year.

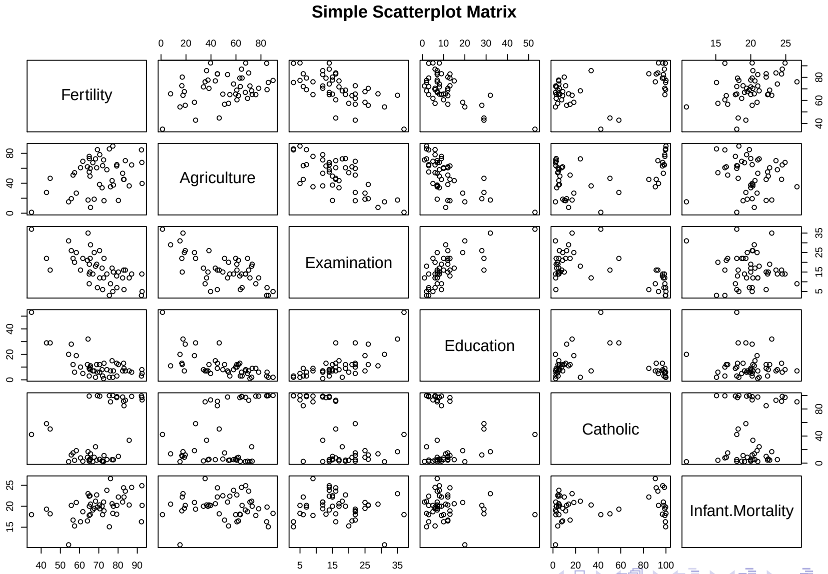
All variables but `Fertility` give proportions of the population.
Switzerland, in 1888, was entering a period known as the demographic transition ; i.e., its fertility was beginning to fall from the high level typical of underdeveloped countries.
The data collected are for 47 French-speaking "provinces" at about 1888.
Here, all variables are scaled to [0, 100], where in the original, all but "Catholic" were scaled to [0, 1].

```r
summary(swiss)
```

```
##    Fertility      Agriculture     Examination      Education    
##  Min.   :35.00   Min.   : 1.20   Min.   : 3.00   Min.   : 1.00  
##  1st Qu.:64.70   1st Qu.:35.90   1st Qu.:12.00   1st Qu.: 6.00  
##  Median :70.40   Median :54.10   Median :16.00   Median : 8.00  
##  Mean   :70.14   Mean   :50.66   Mean   :16.49   Mean   :10.98  
##  3rd Qu.:78.45   3rd Qu.:67.65   3rd Qu.:22.00   3rd Qu.:12.00  
##  Max.   :92.50   Max.   :89.70   Max.   :37.00   Max.   :53.00  
##     Catholic       Infant.Mortality
##  Min.   :  2.150   Min.   :10.80   
##  1st Qu.:  5.195   1st Qu.:18.15   
##  Median : 15.140   Median :20.00   
##  Mean   : 41.144   Mean   :19.94   
##  3rd Qu.: 93.125   3rd Qu.:21.70   
##  Max.   :100.000   Max.   :26.60   
```

Première vue des données

```r
pairs(swiss,  main="Simple Scatterplot Matrix")
```



Sélection de variables ℓ_0

Tests de Fisher dans le modèle linéaire

Test pour modèles emboîtés

Pour tester $H_0 : \beta_{k_1} = \ldots = \beta_{k_l} = 0$, on utilise le test de Fisher.

- On note $W = \text{vect}\{(\mathbf{1}\,, X_1\,, \ldots, X_p)/(X_{k_1}\,, \ldots, X_{k_l})\}$ de dimension $p+1-l$.
- On note $X\tilde{\beta} = \text{proj}_W^\top(Y)$ et on a toujours $X\hat{\beta} = \text{proj}_V^\top(Y)$.

Statistique de Fisher

On a

$$\frac{(n-p-1)\big(\|Y - X\tilde{\beta}\|^2 - \|Y - X\hat{\beta}\|^2\big)}{l\|Y - X\hat{\beta}\|^2} \underset{H_0}{\sim} \mathcal{F}(l, n-p-1).$$

Test du rapport de vraisemblance dans les glm

Test du rapport de vraisemblance (LRT)

Pour tester $H_0 : \beta_{k_1} = \ldots = \beta_{k_l} = 0$, dans un modèle linéaire généralisé,on utilise le test du rapport de vraisemblance (LRT).

- On note $\hat{\beta}$ l'estimateur du maximum de vraisemblance dans le modèle complet
- et $\tilde{\beta}$ l'estimateur du maximum de vraisemblance dans le modèle sans les variables $X^{k_1}, \ldots, X^{k_l}$

Statistique du LRT

- On sait que

$$D^0(\hat{\beta}) - D^0(\tilde{\beta}) = -2\log(\mathcal{L}(\tilde{\beta})) + 2\log(\mathcal{L}(\hat{\beta}))$$

  suit, sous $\mathcal{H}_0$, si n est grand, une loi du $\chi^2(l)$
- On rejette $\mathcal{H}_0$ au niveau α si $D^0(\hat{\beta}) - D^0(\tilde{\beta}) > z'$ où z' est un fractile de la loi du $\chi^2(l)$.

Sélection pas à pas

Si le modèle de départ a p variables, on peut définir 2^p sous-modèles, tous ceux de

$$\mathcal{M} = \mathcal{P}\{1, \ldots, p\} = \left\{1, (1, X^1), (1, X^2), \ldots, (1, X^1, X^2, \ldots, X^p)\right\}$$

.

Dans l'exemple `swiss`, on peut former les modèles

- `Fertility ~Agriculture+Examination+Education+ Catholic+Infant.Mortality` (modèle complet)
- `Fertility ~Examination+Education+ Catholic+Infant.Mortality`
- `Fertility ~Agriculture+Education+ Catholic+Infant.Mortality`
- ...
- `Fertility ~Agriculture`
- `Fertility ~Education`
- ....
- `Fertility ~1` (modèle `null`).

Il y a 32 modèles possibles.

Algorithmes de recherche de sous-modèles

2^p peut être très grand, il faut des algorithmes efficaces :

- **forward** : on part du modèle avec seulement l'intercept et on ajoute les variables une à une. A chaque pas, on ajoute celle qui a la plus grande statistique de Fisher. On s'arrête quand la p-value associée devient > 0.1 (seuil arbitraire)
- **backward** : on part du modèle avec toutes les variables et on retire les variables une à une. A chaque pas, on retire celle qui a la plus petite statistique de Fisher. On s'arrête quand la p-value associée devient < 0.1 (seuil arbitraire)
- **stepwise** mixte des deux premières (on ajoute, puis on permet une élimination, etc)

Modèles complet et nul

- Modèle complet

```r
fit_full = lm(Fertility~Agriculture+Examination+Education+
                Catholic+Infant.Mortality ,data = swiss)

summary(fit_full)$coefficients
```

```
##                    Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept)      66.9151817 10.70603759  6.250229 1.906051e-07
## Agriculture      -0.1721140  0.07030392 -2.448142 1.872715e-02
## Examination      -0.2580082  0.25387820 -1.016268 3.154617e-01
## Education        -0.8709401  0.18302860 -4.758492 2.430605e-05
## Catholic          0.1041153  0.03525785  2.952969 5.190079e-03
## Infant.Mortality  1.0770481  0.38171965  2.821568 7.335715e-03
```

- Modèle nul

```r
fit_null = lm(Fertility ~ 1, data = swiss)
summary(fit_null)$coefficients
```

```
##             Estimate Std. Error  t value     Pr(>|t|)
## (Intercept) 70.14255   1.822101 38.49542 1.212895e-36
```

Première sélection de modèle “à la main” descendante (1)

```r
drop1(fit_full, test = "F")
```

```
## Single term deletions
##
## Model:
## Fertility ~ Agriculture + Examination + Education + Catholic +
##     Infant.Mortality
##                  Df Sum of Sq    RSS    AIC F value    Pr(>F)
## <none>                        2105.0 190.69
## Agriculture       1    307.72 2412.8 195.10  5.9934  0.018727 *
## Examination       1     53.03 2158.1 189.86  1.0328  0.315462
## Education         1   1162.56 3267.6 209.36 22.6432 2.431e-05 ***
## Catholic          1    447.71 2552.8 197.75  8.7200  0.005190 **
## Infant.Mortality  1    408.75 2513.8 197.03  7.9612  0.007336 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Première sélection de modèle “à la main” descendante (2)

```r
drop1(update(fit_full, ~ . -Examination), test = "F")
```

```
## Single term deletions
##
## Model:
## Fertility ~ Agriculture + Education + Catholic + Infant.Mortality
##                  Df Sum of Sq    RSS    AIC F value    Pr(>F)
## <none>                        2158.1 189.86
## Agriculture       1    264.18 2422.2 193.29  5.1413   0.02857 *
## Education         1   2249.97 4408.0 221.43 43.7886 5.140e-08 ***
## Catholic          1    956.57 3114.6 205.10 18.6165 9.503e-05 ***
## Infant.Mortality  1    409.81 2567.9 196.03  7.9757   0.00722 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Première sélection de modèle “à la main” ascendante (1)

```
add1(fit_null, scope =  ~ Agriculture+Examination+Education
     +Catholic+Infant.Mortality,
     test = "F")
```

```
## Single term additions
##
## Model:
## Fertility ~ 1
##                  Df Sum of Sq    RSS    AIC F value    Pr(>F)
## <none>                        7178.0 238.34
## Agriculture       1     894.8 6283.1 234.09  6.4089  0.014917 *
## Examination       1    2994.4 4183.6 214.97 32.2087 9.450e-07 ***
## Education         1    3162.7 4015.2 213.04 35.4456 3.659e-07 ***
## Catholic          1    1543.3 5634.7 228.97 12.3251  0.001029 **
## Infant.Mortality  1    1245.5 5932.4 231.39  9.4477  0.003585 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Première sélection de modèle “à la main” ascendante (2)

```r
add1(update(fit_null,~. + Examination),
     scope =  ~ Agriculture+Examination+Education
     +Catholic+Infant.Mortality,
     test = "F")
```

```
## Single term additions
##
## Model:
## Fertility ~ Examination
##                  Df Sum of Sq    RSS    AIC F value   Pr(>F)
## <none>                        4183.6 214.97
## Agriculture       1    110.83 4072.7 215.71  1.1973 0.279810
## Education         1    633.96 3549.6 209.25  7.8584 0.007497 **
## Catholic          1     93.91 4089.7 215.91  1.0103 0.320322
## Infant.Mortality  1    855.16 3328.4 206.22 11.3048 0.001608 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Première sélection de modèle “à la main” ascendante (3)

```r
add1(update(fit_null,~. + Examination + Infant.Mortality
            +Catholic+Education),
     scope =  ~ Agriculture+Examination+Education
     +Catholic+Infant.Mortality,
     test = "F")
```

```
## Single term additions
##
## Model:
## Fertility ~ Examination + Infant.Mortality + Catholic + Education
##             Df Sum of Sq    RSS    AIC F value   Pr(>F)
## <none>                   2412.8 195.10
## Agriculture  1    307.72 2105.0 190.69  5.9934 0.01873 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Critères pénalisés

Modèles, vrai modèle

On se donne une famille de modèles $\mathcal{M}$, par exemple

$$\mathcal{M} = \mathcal{P}\{1, \ldots, p\} = 1, (1, X^1), (1, X^2), \ldots, (1, X^1, X^2 n \ldots, X^p).$$

On suppose qu'il existe un vrai modèle $m^* \in \mathcal{M}$ tel que :

$$\mathbb{E}(Y) = g^{-1}(X^{(m^*)}\beta^{(m^*)}).$$

On veut retrouver m^*.

- **Attention :** le R^2 ou la log-vraisemablence ne sont pas des bons critères pour ce problème car ils choisiront toujours le modèle complet (avec toutes les covariables)
- On note $m^{\text{full}} = (1\,, X^1\,, X^2\,, \ldots, X^p)$ le modèle complet.

Sélection via le R2 : pas une bonne idée

```r
library(leaps)

choix <- regsubsets(Fertility~Agriculture+Examination+
                      Education+Catholic+Infant.Mortality,
                    data=swiss,
                    method = "exhaustive")
summary(choix)$rsq
```

```
## [1] 0.4406156 0.5745071 0.6625438 0.6993476 0.7067350
```

Estimation dans le modèle m

Dans le modèle m, on note $|m|$ le nombre de covariables qu'il contient

$$\hat{\beta}^{(m)}$$

l'estimateur au maximum de vraisemblance dans ce modèle.

Pour le modèle linéaire

AIC/BIC

On choisit $\hat{m}_{AIC} \in \mathcal{M}$ et $\hat{m}_{BIC} \in \mathcal{M}$tel que :

$$\hat{m}_{AIC} = \underset{m \in \mathcal{M}}{\text{argmin}}\, AIC(m),$$
$$\hat{m}_{BIC} = \underset{m \in \mathcal{M}}{\text{argmin}}\, BIC(m),$$

avec

$$AIC(m) = \log\left(\|Y - X\hat{\beta}^{(m)}\|^2\right) + 2\frac{|m|}{n} = \log\left(\text{RSS}^{(m)}\right) + 2\frac{|m|}{n}$$
$$BIC(m) = \log\left(\|Y - X\hat{\beta}^{(m)}\|^2\right) + \frac{\log(n)|m|}{n} = \log\left(\text{RSS}^{(m)}\right) + \frac{\log(n)|m|}{n}.$$

Pour les glm

AIC/BIC

On choisit $\hat{m}_{AIC} \in \mathcal{M}$ et $\hat{m}_{BIC} \in \mathcal{M}$tel que :

$$\hat{m}_{AIC} = \underset{m \in \mathcal{M}}{\text{argmin}}\, AIC(m),$$
$$\hat{m}_{BIC} = \underset{m \in \mathcal{M}}{\text{argmin}}\, BIC(m),$$

avec

$$AIC(m) = -2 \log \mathcal{L}(\hat{\beta}^{(m)}) + 2\frac{|m|}{n}$$
$$BIC(m) = -2 \log \mathcal{L}(\hat{\beta}^{(m)}) + \frac{\log(n)|m|}{n}$$

où $\log \mathcal{L}(\hat{\beta}^{(m)})$ est la log-vraisemblance dans le modèle m .

Sélection automatique stepwise par AIC(1)

```
model.aic.both <- step(fit_null, direction = "both" ,
                          scope =  ~ Agriculture+Examination+Education
                            +Catholic+Infant.Mortality, trace = TRUE)
```

```
## Start:  AIC=238.35
## Fertility ~ 1
##
##                    Df Sum of Sq    RSS    AIC
## + Education         1    3162.7 4015.2 213.04
## + Examination       1    2994.4 4183.6 214.97
## + Catholic          1    1543.3 5634.7 228.97
## + Infant.Mortality  1    1245.5 5932.4 231.39
## + Agriculture       1     894.8 6283.1 234.09
## <none>                          7178.0 238.34
##
## Step:  AIC=213.04
## Fertility ~ Education
##
##                    Df Sum of Sq    RSS    AIC
## + Catholic          1     961.1 3054.2 202.18
## + Infant.Mortality  1     891.2 3124.0 203.25
## + Examination       1     465.6 3549.6 209.25
## <none>                          4015.2 213.04
```

Sélection automatique stepwise par AIC (2)

```r
summary(model.aic.both)
```

```
##
## Call:
## lm(formula = Fertility ~ Education + Catholic + Infant.Mortality +
##     Agriculture, data = swiss)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.6765  -6.0522   0.7514   3.1664  16.1422
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      62.10131    9.60489   6.466 8.49e-08 ***
## Education        -0.98026    0.14814  -6.617 5.14e-08 ***
## Catholic          0.12467    0.02889   4.315 9.50e-05 ***
## Infant.Mortality  1.07844    0.38187   2.824  0.00722 **
## Agriculture      -0.15462    0.06819  -2.267  0.02857 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.168 on 42 degrees of freedom
## Multiple R-squared:  0.6993,  Adjusted R-squared:  0.6707
```

Régressions pénalisées

Introduction

On observe Y et X de dimension $n \times p$. On fait l'hypothèse qu'il existe un vrai modèle m^* tel que

$$Y = X^{m^*} \beta^{m^*} + \epsilon^{m^*} = X^* \beta^* + \epsilon^*,$$

et que $|m^*|$ est petit devant p et n.
Quand p devient grand par rapport à n, il y a trois problèmes potentiels :

- $\hat{\beta} = (X^\top X)^{-1} X^\top Y$ peut ne pas être défini car $X^\top X$ n'est alors plus inversible (c'est aussi le cas si des colonnes de X sont colinéaires)
- La variance d'estimation à partir de X devient trop grande.
- Les algorithmes de sélection de variables ℓ_0 ne peuvent plus être appliqués à cause des 2^p modèles à comparer.

Ridge

Régression Ridge

Hoerl et Kennard (1970) ont l'idée d'ajouter à $X^\top X$ une matrice diagonale $\lambda \mathrm{Id}_n$ pour retrouver l'inversibilité pour

$$X^\top X + \lambda \mathrm{Id}_p = PDP^{-1} + \lambda \mathrm{Id}_p = P(D + \lambda \mathrm{Id}_p)P^{-1}.$$

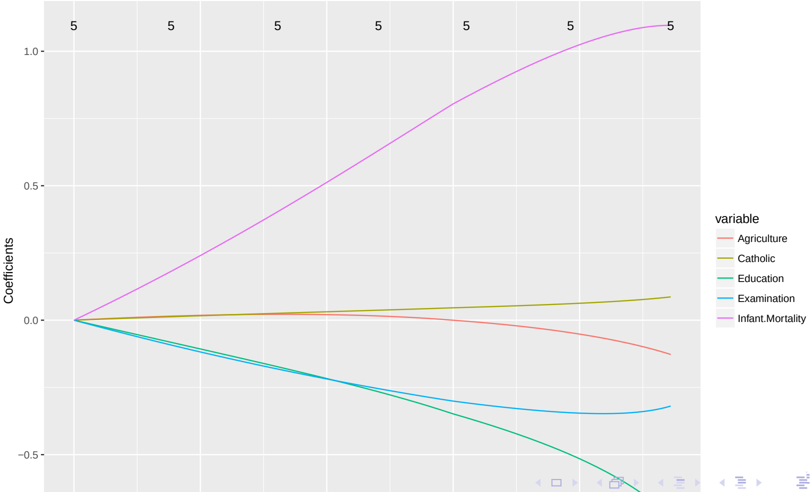
Pénalité ridge

$$\hat{\beta}_\lambda^{\text{ridge}} = \underset{\beta \in \mathbb{R}^p}{\text{argmin}} \|Y - X\beta\|^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

$$\hat{\beta}_\lambda^{\text{ridge}} = \underset{\beta \in \mathbb{R}^p}{\text{argmin}} - \log \mathcal{L}(\beta) + \lambda \sum_{j=1}^{p} \beta_j^2$$

on formate les données pour les passer à glmnet

```
#?glmnet
Y =  Fertility
X = as.matrix(swiss[,2:6],ncol = 4)
fit_ridge =  glmnet(X,Y, alpha = 0)
autoplot(fit_ridge)
```



Cross-validation pour λ

On sait qu'il existe une valeur λ^* du paramètre de régularisation qui minimise l'erreur de l'estimateur ridge. Cette valeur idéale dépend de quantités inconnues, il faut donc l'estimer.
Si on avait à disposition d'autres données, on aurait

- des données d'apprentissage (training, learning set) $\mathcal{S}_L = \{(Y_1\,, X_i)\,, \ldots, (Y_n\,, X_n)\}$
- des données de validation, de test (testing, validation set) $\mathcal{S}_T = \{(Y_{+,1}\,, X_{+,1})\,, \ldots, (Y_{+,n'}\,, X_{+,n'})\}$ avec $Y_+ = X_+\beta^* + \epsilon_+$ **ET** ϵ et ϵ_+ indépendants.

On pourrait calculer $\hat{\beta}_\lambda^{\text{ridge}}$ pour chaque valeur de λ sur l'échantillon d'apprentissage, puis regarder pour quelle valeur de λ il y a le moins d'erreurs sur l'échantillon de test. Pour le modèle linéaire, on calcule

$$\frac{1}{n'}\sum_{i=1}^{n'}(Y_{+,i} - X_{+,i}\hat{\beta}^{(\lambda)})^2,$$

chaque valeur λ du paramètre de régularisation.

création d'un jeu de données d'apprentissage et de test

```
shuffle = sample(1:nrow(swiss))
apprentissage = shuffle[1:33]
print(apprentissage)
```

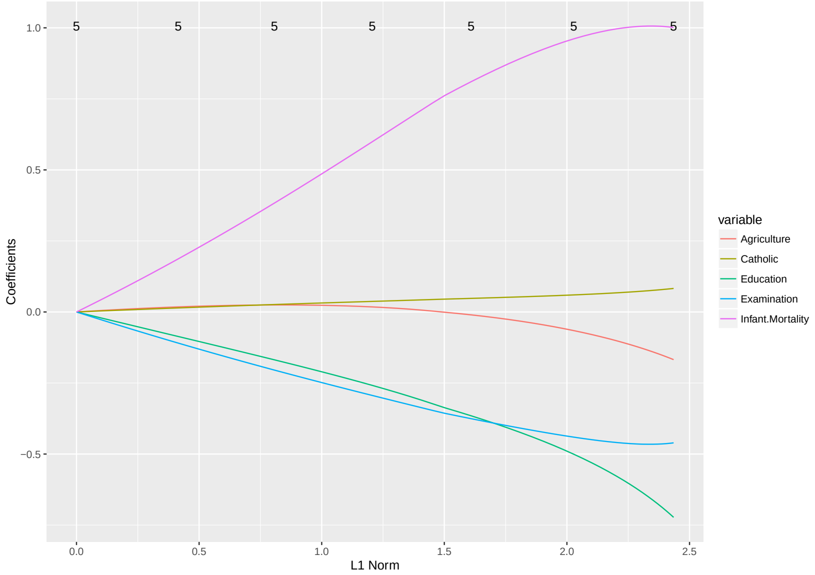
```
##  [1] 24  1 35  9  6 43 26  3 46 36 28 42 16 23 10 37 15 34 31  2 47
## [24] 29 18 25  4 22 41 30  8 19 45
```

```
test = shuffle[34:nrow(swiss)]
print(test)
```

```
##  [1] 11 33 27  7 17 38 13 32 44 20 21 14 12 39
```

```
Y_apprentissage =  Fertility[apprentissage]
X_apprentissage = as.matrix(swiss[apprentissage,2:6],ncol = 4)
Y_test =  Fertility[test]
X_test = as.matrix(swiss[test,2:6],ncol = 4)
```

```
fit_ridge_apprentissage =  glmnet(X_apprentissage,Y_apprentissage, alph
autoplot(fit_ridge_apprentissage)
```



```r
predictions = predict(fit_ridge_apprentissage,newx=X_test)
erreurs_test = rep(0,100)
for (l in 1:100){
  erreurs_test[l] = mean((Y_test - predictions[,l])^2)
}

indice_ideal = which(erreurs_test == min(erreurs_test))
lambda_ideal = fit_ridge_apprentissage$lambda[indice_ideal]
lambda_ideal
```

```
## [1] 4.304976
```

En pratique

Même en l'absence de données de validation (situation fréquente en pratique), on peut vouloir créer des données qui "ressemblent" à des données de test pour appliquer ce qui précède.

Leave-one-out (jackknife)

Chaque observation joue à tour de rôle le rôle d'échantillon de validation.

Estimation de l'erreur de généralisation par leave-one-out

$$\mathbb{E}(\| \widehat{Y_+ - \hat{Y}_+^{(\lambda)}} \|^2)_{loo} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - X_i\hat{\beta}_{(-i)}^{(\lambda)})^2,$$

où $\hat{\beta}_{(-i)}^{(\lambda)}$ a été calculé sur l'échantillon $\mathcal{S}_L \backslash (Y_i, X_i)$ et pour la valeur λ du paramètre.

K-fold cross-validation

On découpe l'échantillon initial en K sous-ensembles pour obtenir la partition $\mathcal{S}_L = \mathcal{S}_{L,1} \cup \ldots \cup \mathcal{S}_{L,K}$. Dans le cas, où $n = Kn_K$, on tire aléatoirement et sans remise dans $\mathcal{S}_L$ pour former les $\mathcal{S}_{L,k}$.

Estimation de l'erreur de généralisation par K-fold cross-validation

$$\widehat{eg(\lambda)}_{Kfold-cv} = \mathbb{E}(\widehat{\| Y_+ - \hat{Y}_+^{(\lambda)} \|^2})_{Kfold-cv} = \frac{1}{n_K K} \sum_{k=1}^{K} \sum_{i=1}^{n_K} (Y_{k,i} - X_{k,i} \hat{\beta}_{(-k)}^{(\lambda)})^2,$$

où $\hat{\beta}_{(-k)}^{(\lambda)}$ a été calculé sur l'échantillon $\mathcal{S}_L \backslash \mathcal{S}_{L,k}$ et pour la valeur λ du paramètre.

On choisit alors

$$\hat{\lambda} = \arg\min_{\lambda \in \Lambda} \widehat{eg(\lambda)}_{Kfold-cv}$$

Cross-validation et comparaison avec l'AIC

```
fit_ridge_cv  = cv.glmnet(X,Y, lambda = seq(0,3,0.01),alpha = 0)
coef(fit_ridge_cv)
```

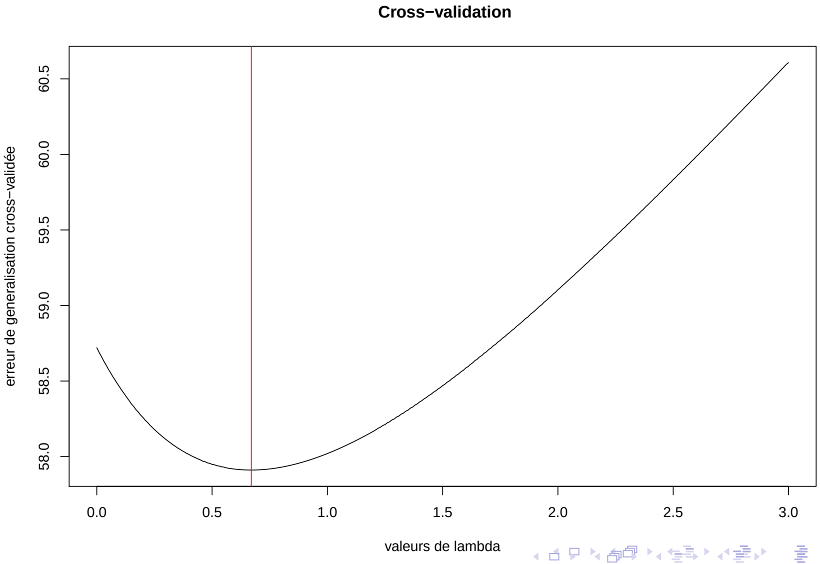
```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                              1
## (Intercept)      61.46350702
## Agriculture      -0.06316741
## Examination      -0.34720834
## Education        -0.54782492
## Catholic          0.06627933
## Infant.Mortality  1.04759689
```

```
summary(model.aic.both)$coef[,1]
```

```
##      (Intercept)        Education         Catholic Infant.Mortality
##       62.1013116       -0.9802638        0.1246664        1.0784422
##      Agriculture
##       -0.1546175
```

Grille en lambda, learning curve

```
plot(fit_ridge_cv$lambda,fit_ridge_cv$cvm, type = 'l', xlab = "valeurs
abline(v =fit_ridge_cv$lambda.min,col="red")
```



Lasso et elastic-net

Le lasso

Introduit en 1996 par Tibshirani, la lasso peut être vu comme un intermédiaire entre la régression ridge et la sélection ℓ_0.

Pénalité lasso

$$\begin{aligned}\hat{\beta}_\lambda^{\text{lasso}} &= \operatorname*{argmin}_{\beta \in \mathbb{R}^p} \|Y - X\beta\|^2 + \lambda \sum_{j=1}^{p} |\beta_j| \\ &= \operatorname*{argmin}_{\beta \in \mathbb{R}^p} \|Y - X\beta\|^2 + \lambda \|\beta\|_1 .\end{aligned}$$

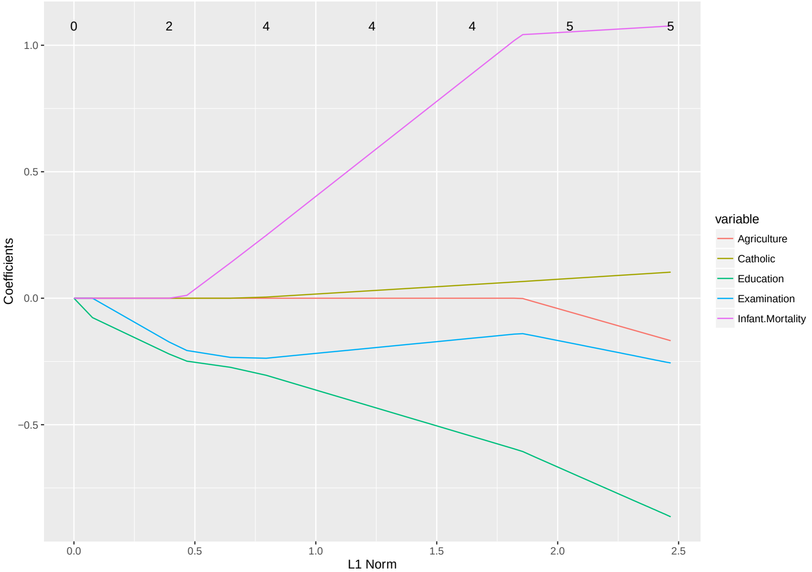
Solution lasso sur design orthogonal

Quand $X^\top X = Id_p$, on sait que

$$\hat{\beta}_{\text{lasso}}^{(\lambda)} = \mathcal{S}_{\lambda/2}(Y^\top X),$$

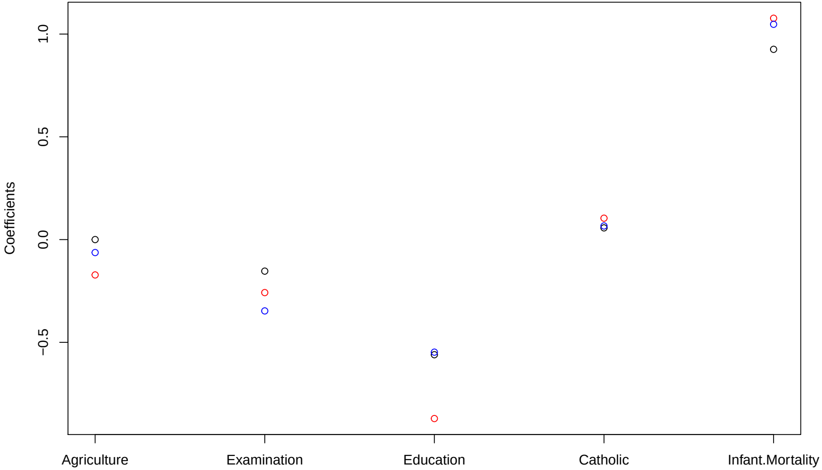
avec $\mathcal{S}_\tau(x) = sign(x)(|x| - \tau)_+$.

```
fit_lasso =  glmnet(X,Y, alpha = 1)
autoplot(fit_lasso)
```



Cross-validation

```
fit_lasso_cv  = cv.glmnet(X,Y,alpha = 1)
plot(coef(fit_full)[2:6],col = "red",ylab = "Coefficients", xaxt="n")
axis(1, 1:5, labels=names(coef(fit_full))[2:6])
points(coef(fit_lasso_cv)[2:6])
points(coef(fit_ridge_cv)[2:6],col = "blue")
```



L'elastic net

Elastic net

$$\hat{\beta}_\lambda^{\mathsf{en}} = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \|Y - X\beta\|^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2$$

Attention dans `glmnet`, l'elastic-net est défini par

$$\hat{\beta}_\lambda^{\mathsf{en}} = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \|Y - X\beta\|^2 + \lambda\left(\alpha \|\beta\|_1 + \frac{1-\alpha}{2} \|\beta\|_2^2\right)$$

Cross validation en lambda et alpha

```
trControlCV <- trainControl(method = "CV",number = 5)
swissGlmnet <- train(Fertility ~ ., data = swiss, method = "glmnet",
                     trControl = trControlCV,
                     tuneGrid  =  expand.grid(alpha = exp(seq(-8,0, lengt
                                              lambda = exp(seq(-8,0, length
swissGlmnet
```

```
## glmnet
##
## 47 samples
##  5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 38, 38, 38, 38, 36
## Resampling results across tuning parameters:
##
##   alpha         lambda        RMSE      Rsquared
##   0.0003354626  0.0003354626  7.434016  0.6474605
##   0.0003354626  0.0008159878  7.434016  0.6474605
##   0.0003354626  0.0019848296  7.434016  0.6474605
##   0.0003354626  0.0048279500  7.434016  0.6474605
##   0.0003354626  0.0117436285  7.434016  0.6474605
```