

# (JC)2BIM 2018 Research School

## Variable Selection and Regularization

Julien Chiquet

Fréjus, 4–8 June 2018

<https://github.com/jchiquet/JC2BIM18>



# Outline

## ① Motivations

## ② Variable Selection

## ③ Regularisation

- The ridge estimator

- Model complexity and Tuning parameter

- Definition of the LASSO estimator

- Model complexity and Tuning parameter

# Outline

## ① Motivations

Assessing the quality of a regression model

Illustration: prostate cancer

## ② Variable Selection

## ③ Regularisation

The ridge estimator

Model complexity and Tuning parameter

Definition of the LASSO estimator

Model complexity and Tuning parameter

# Outline

## ① Motivations

Assessing the quality of a regression model

Illustration: prostate cancer

## ② Variable Selection

## ③ Regularisation

The ridge estimator

Model complexity and Tuning parameter

Definition of the LASSO estimator

Model complexity and Tuning parameter

# Statistical Learning

## Canonical scenario

- ① an **outcome** measurement (or *response*, *output*)
  - either quantitative (expression level, tumor size, survival time, etc.)
  - or categorical (presence/absence of a gene or of a disease, etc.)
- ② a set of **features** (or *predictors*, *inputs*)
  - clinical measurements (expression level, tumor size)
  - age, smoking or not, height, SNPs, etc.

## Learning problem

Given a training set of data (observed inputs and outputs), we aim to

- ① suggest a model,
- ② learn this model on the training set,
- ③ test this model on new outcomes/features.

⇒ A “good” model should accurately predict new outcomes.

# Notations

Let

- $Y$  be the output random variable,
- $X = (X_1, \dots, X_p)$  be the input random variables, where  $X_j$  is the  $j$  predictor.

## The data

Given a sample  $\{(y_i, x_i), i = 1, \dots, n\}$  of i.i.d. realizations of  $(Y, X)$ , denote

- $\mathcal{D} = \{i : (y_i, x_i) \in \text{training set}\},$
- $\mathcal{T} = \{i : (y_i, x_i) \in \text{test set}\},$
- $\mathbf{y} = (y_i)_{i \in \mathcal{D}},$  the *response* vector in  $\mathbb{R}^{|\mathcal{D}|},$
- $\mathbf{x}_j = (x_{ij})_{i \in \mathcal{D}}^\top$  the vector of data for the  $j$ th predictor in  $\mathbb{R}^{|\mathcal{D}|},$
- $\mathbf{X}$  the  $n \times p$  data (or design) matrix on the training set whose  $j$ th row is  $\mathbf{x}_j,$
- $(\mathbf{y}_{\mathcal{T}}, \mathbf{X}_{\mathcal{T}})$  are the test data.

# Regression models

We seek a function  $f$  that predicts  $Y$  through  $X$ .

## Proposition

*The model  $f(X) = \mathbb{E}[Y|X]$  minimizes the squared error loss, that is,*

$$f(X) = \arg \min_{\varphi} \text{err}(\varphi(X)), \quad \text{with } \text{err}(\varphi(X)) = \mathbb{E}[(Y - \varphi(X))^2].$$

*$\leadsto$  The best prediction of  $Y$  at any point  $X = x$  is the conditional mean, when best is measured by average squared error.*

This leads to the regression model

$$Y = f(X) + \varepsilon,$$

where

- $\varepsilon$  is an additive error with  $\mathbb{E}[\varepsilon] = \mathbf{0}$ ,  $\mathbb{V}[\varepsilon] = \sigma^2$ ,
- $f(x) = \mathbb{E}[Y|X = x]$  is the *regression* function.

# Learning strategy

## Problem

$\mathbb{P}(Y|X)$  and  $\mathbb{P}(X)$  are unknown thus  $\mathbb{E}(Y|X)$ ,  $\text{err}(f(X))$  unreachable: one should **estimate** this.

## Strategy

- 1 Fix a family  $\mathcal{F}$  of models

*For the linear model,  $\mathcal{F} = \{X^T \beta, \beta \in \mathbb{R}^p\}$ .*

- 2 Fit a model  $\hat{f} \in \mathcal{F}$  on the training set  $\mathcal{D}$

*With the least square, compute  $\hat{\beta}^{\text{ols}}$  and  $\hat{f} = \hat{Y} = \mathbf{X}\hat{\beta}^{\text{ols}}$*

- 3 Estimate the prediction error with the test set  $\mathcal{T}$ .

*For instance,  $\text{err}(\mathbf{X}_{\mathcal{T}}\hat{\beta}^{\text{ols}}) = \frac{1}{n} \left\| \mathbf{y}_{\mathcal{T}} - \mathbf{X}_{\mathcal{T}}\hat{\beta}^{\text{ols}} \right\|^2$ .*



# Learning strategy

## Problem

$\mathbb{P}(Y|X)$  and  $\mathbb{P}(X)$  are unknown thus  $\mathbb{E}(Y|X)$ ,  $\text{err}(f(X))$  unreachable: one should **estimate** this.

## Strategy

- 1 Fix a family  $\mathcal{F}$  of models

*For the linear model,  $\mathcal{F} = \{X^T \beta, \beta \in \mathbb{R}^p\}$ .*

- 2 Fit a model  $\hat{f} \in \mathcal{F}$  on the training set  $\mathcal{D}$

*With the least square, compute  $\hat{\beta}^{\text{ols}}$  and  $\hat{f} = \hat{Y} = \mathbf{X}\hat{\beta}^{\text{ols}}$*

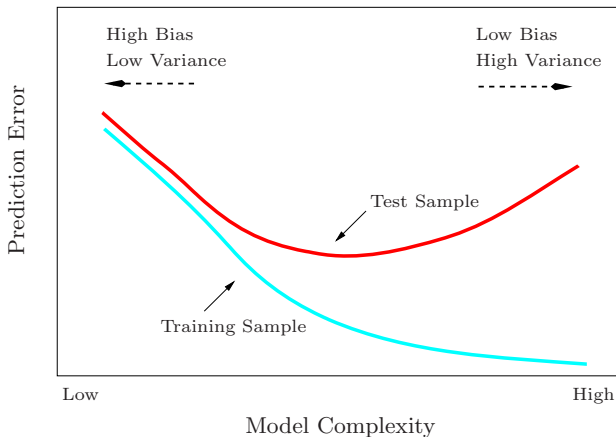
- 3 Estimate the prediction error with the test set  $\mathcal{T}$ .

$$\text{For instance, } \hat{\text{err}}(\mathbf{X}_{\mathcal{T}}\hat{\beta}^{\text{ols}}) = \frac{1}{n} \left\| \mathbf{y}_{\mathcal{T}} - \mathbf{X}_{\mathcal{T}}\hat{\beta}^{\text{ols}} \right\|^2.$$

# Bias/variance tradeoff

At an input point  $X = x$ ,

$$\text{err}(\hat{f}(x)) = \underbrace{\sigma^2}_{\text{incompressible error}} + \underbrace{\text{bias}^2(\hat{f}(x)) + \mathbb{V}(\hat{f}(x))}_{\text{MSE}(\hat{f}(x))}.$$



# Linear regression

## Prediction error

For a fixed  $\mathbf{X}$ , one has

$$\text{err}(\mathbf{X}\hat{\boldsymbol{\beta}}^{\text{ols}}) = \sigma^2 \frac{(p+1)}{n} + \sigma^2.$$

## Gauss-Markov Theorem

$\hat{Y} = \mathbf{X}^\top \hat{\boldsymbol{\beta}}^{\text{ols}}$  is the BLUE: the best model (i.e. with the smallest variance) among unbiased estimators of  $\boldsymbol{\beta}$ .

$\rightsquigarrow$  Are there some cases where we should **trade some bias for smaller variance**?

# Outline

## ① Motivations

Assessing the quality of a regression model

**Illustration: prostate cancer**

## ② Variable Selection

## ③ Regularisation

The ridge estimator

Model complexity and Tuning parameter

Definition of the LASSO estimator

Model complexity and Tuning parameter

# Example: prostate cancer data set

The data set: 97 patient with prostate cancer

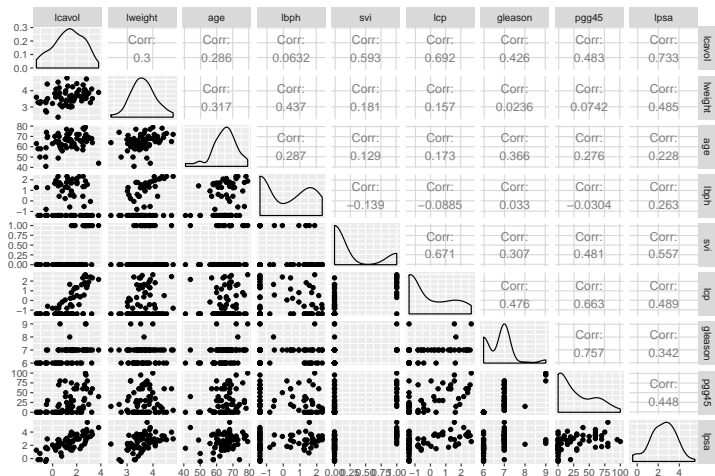
Examine the correlation between the level of cancer-specific antigen (y) and various clinical measures.

```
load("prostate.rda")
prostate %>% as_tibble() %>% print()

## # A tibble: 97 x 10
##   lcavol lweight age lbph svi lcp gleason pgg45 lpsa train
## *   <dbl>   <dbl> <int> <dbl> <int> <dbl>   <int> <int>   <dbl> <lgl>
## 1 -0.580    2.77   50 -1.39    0 -1.39     6     0 -0.431 TRUE
## 2 -0.994    3.32   58 -1.39    0 -1.39     6     0 -0.163 TRUE
## 3 -0.511    2.69   74 -1.39    0 -1.39     7    20 -0.163 TRUE
## 4 -1.20     3.28   58 -1.39    0 -1.39     6     0 -0.163 TRUE
## 5  0.751    3.43   62 -1.39    0 -1.39     6     0  0.372 TRUE
## 6 -1.05     3.23   50 -1.39    0 -1.39     6     0  0.765 TRUE
## 7  0.737    3.47   64  0.615    0 -1.39     6     0  0.765 FALSE
## 8  0.693    3.54   58  1.54     0 -1.39     6     0  0.854 TRUE
## 9 -0.777    3.54   47 -1.39    0 -1.39     6     0  1.05 FALSE
## 10 0.223    3.24   63 -1.39    0 -1.39     6     0  1.05 FALSE
## # ... with 87 more rows
```

# Correlations between predictors

```
prostate %>% filter(train == TRUE) %>% select(-train) %>%
  ggpairs(upper = list(continuous="cor", combo="box"))
```



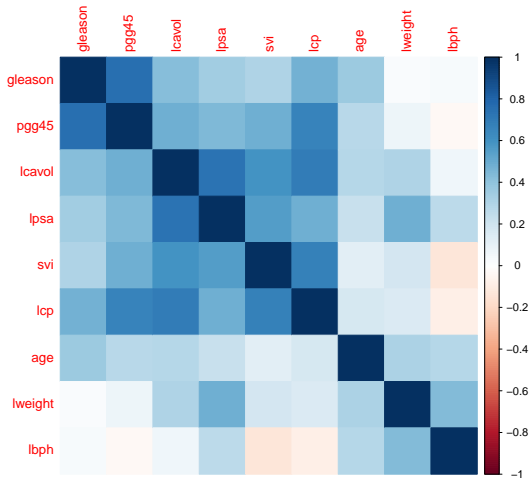
# Correlations between predictors II

```
prostate %>%  
  filter(train == TRUE) %>%  
  select(-train) %>%  
  cor %>% as.dist() %>% print()
```

```
##           lcavol      lweight      age      lbph      svi  
## lweight  0.30023199  
## age      0.28632427  0.31672347  
## lbph     0.06316772  0.43704154  0.28734645  
## svi      0.59294913  0.18105448  0.12890226 -0.13914680  
## lcp      0.69204308  0.15682859  0.17295140 -0.08853456  0.67124021  
## gleason  0.42641407  0.02355821  0.36591512  0.03299215  0.30687537  
## pgg45    0.48316136  0.07416632  0.27580573 -0.03040382  0.48135774  
## lpsa     0.73315515  0.48521519  0.22764238  0.26293763  0.55688643  
##           lcp      gleason      pgg45  
## lweight  
## age  
## lbph  
## svi  
## lcp  
## gleason  0.47643684  
## pgg45    0.66253335  0.75705650  
## lpsa     0.48920320  0.34242781  0.44804795
```

# Correlations between predictors III

```
prostate %>% filter(train == TRUE) %>%  
  select(-train) %>% cor() %>% corrplot(method = "color", order = "hclust")
```





# OLS and limitations I

For studying the correlation effect, we normalize and create test and train sets

```
prostate_train <-  
  prostate %>% filter(train == TRUE) %>% select(-train) %>%  
  scale(FALSE, TRUE) %>% as_data_frame()  
prostate_test <-  
  prostate %>% filter(train == FALSE) %>% select(-train) %>%  
  scale(FALSE, TRUE) %>% as_data_frame()  
model.full <- lm(lpsa~., prostate_train)
```

## Estimating prediction error

```
y_hat <- predict(model.full, newdata=prostate_test)  
y_test <- prostate_test$lpsa  
err_ols <- mean((y_test-y_hat)^2)  
print(err_ols)  
  
## [1] 0.0673631
```

# OLS and limitations II

```
summary(model.full)

##
## Call:
## lm(formula = lpsa ~ ., data = prostate_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59947 -0.12416 -0.01972  0.16341  0.54059
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.15605     0.56489   0.276  0.78334
## lcavol        0.38055     0.07092   5.366 1.47e-06 ***
## lweight       0.82258     0.29903   2.751  0.00792 **
## age          -0.45367     0.32500  -1.396  0.16806
## lbph          0.07718     0.03754   2.056  0.04431 *
## svi           0.12779     0.05175   2.469  0.01651 *
## lcp          -0.10632     0.05695  -1.867  0.06697 .
## gleason      -0.07315     0.49870  -0.147  0.88389
## pgg45         0.13589     0.07820   1.738  0.08755 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.259 on 58 degrees of freedom
## Multiple R-squared:  0.6944, Adjusted R-squared:  0.6522
## F-statistic: 16.47 on 8 and 58 DF,  p-value: 2.042e-12
```

# Comments

Why do some coefficients in  $\beta$  are not well estimated/ have large variance?  
(pgg45, gleason)

## Statistical issue

Correlated variables are not well estimated,  
 $\rightsquigarrow$  they carry the same information regarding the response.

## Numerical issue

Correlated variables leads to bad conditioning of  $\mathbf{X}^T \mathbf{X}$ ,  
 $\rightsquigarrow$  OLS cannot be computed when they are redundant variables in  $\mathbf{X}$  or when  $n < p$ .

$\rightsquigarrow$  interpretation becomes rather difficult

# Solutions

## Variable selection

If the underlying model is assumed to have only few predictors truly related to the outcome, we may want to **select** those with the highest effect. We are looking for both

- better interpretability.
- better predictive performances.

## Regularization

If all the predictors have similar or close effects on the response, selection (and thus interpretability) is out of reach.

We may **regularize** the problem by **constraining** the parameters  $\beta$  to live in an appropriate set that will make the  $\mathbf{X}^T \mathbf{X}$  invertible.

# Outline

## ① Motivations

## ② Variable Selection

- Criteria for model comparison

- Algorithms for variable subset selection

- Illustration: prostate cancer

## ③ Regularisation

- The ridge estimator

- Model complexity and Tuning parameter

- Definition of the LASSO estimator

- Model complexity and Tuning parameter

# Variable Selection

## Problematic

With many regressor,

- we integrate more and more information in the model ;
- we have more and more parameters to estimate and  $\mathbb{V}(\hat{Y}_i) \nearrow$ .

## Idea

Look for a (small) set  $\mathcal{S}$  with  $k$  variables among  $p$  such that

$$Y \approx X_{\mathcal{S}}^T \hat{\beta}_{\mathcal{S}}.$$

## Ingredients

To find this tradeoff, we need

- ① a **criterion** to evaluate the performance ;
- ② an **algorithm** to determine the subset of  $k$  variables optimising the criterion.

# Outline

## ① Motivations

## ② Variable Selection

- Criteria for model comparison

- Algorithms for variable subset selection

- Illustration: prostate cancer

## ③ Regularisation

- The ridge estimator

- Model complexity and Tuning parameter

- Definition of the LASSO estimator

- Model complexity and Tuning parameter

# Estimation of the prediction error by cross-validation

For the regression: PRESS (*predicted residual sum of squares*)

## Principe

- ① Split the data into  $K$  subsets,
- ② Successively use each subset as the test set,
- ③ Compute the test error for the  $K$  subsets,
- ④ Average the  $K$  error to get the final estimate.

## Formalism

Let  $\kappa : \{1, \dots, n\} \rightarrow \{1, \dots, K\}$  be an indexing function that indicates the partition to which observation  $i$  is allocated by randomization. Denote by  $\hat{f}^{-\kappa(i)}$  the fitted model, computed with the  $k$ th part of the data removed. Then

$$\text{CV}(\hat{\beta}) = \frac{1}{n} \sum_{i=1}^n (y_i - x_i^T \hat{\beta}^{-\kappa(i)})^2$$

provides an estimate of the prediction error.



# Penalized Criterion

## Principle

### Idea

Rather than estimating the prediction error with the test error, we estimate how much the training error underestimates the true prediction error.

### General form

Based on the available model fit, compute

$$\hat{err} = err_{\mathcal{D}} + \text{"optimism"}.$$

### Remarks

- “penalize” too much complex models

# Penalized Criteria

The most Popular in linear regression

Let  $k$  be the size of the current model (i.e. the current number of predictors).

Criterion for the Linear regression model  $\sigma$  known

We choose the model with size  $k$  minimizing one of the following

- **Akaïke Information Criteria** equivalent to  $C_p$  when  $\sigma$  is known

$$\text{AIC} = -2\log\text{lik} + 2k = \frac{n}{\sigma^2}\text{err}_{\mathcal{D}} + 2k.$$

- **Bayesian Information Criterion**

$$\text{BIC} = -2\log\text{lik} + k \log(n) = \frac{n}{\sigma^2}\text{err}_{\mathcal{D}} + k \log(n).$$

# Penalized Criteria

The most Popular in linear regression

Let  $k$  be the size of the current model (i.e. the current number of predictors).

Criterion for the Linear regression model  $\sigma$  unknown

We choose the model with size  $k$  minimizing one of the following

- **Akaike Information Criteria**  $\sigma^2$  estimated by  $\text{err}_{\mathcal{D}}/n$

$$\text{AIC} = -2\log\text{lik} + 2k = n \log(\text{err}_{\mathcal{D}}) + 2k.$$

- **Bayesian Information Criterion**  $\sigma^2$  estimated by  $\text{err}_{\mathcal{D}}/n$

$$\text{BIC} = -2\log\text{lik} + k \log(n) = n \log(\text{err}_{\mathcal{D}}) + k \log(n).$$

# Outline

## ① Motivations

## ② Variable Selection

Criteria for model comparison

**Algorithms for variable subset selection**

Illustration: prostate cancer

## ③ Regularisation

The ridge estimator

Model complexity and Tuning parameter

Definition of the LASSO estimator

Model complexity and Tuning parameter

# Exhaustive search (best-subset)

## Algorithm

For  $k = 0, \dots, p$ , find the subset with  $k$  variables with the smallest  $SCR$  among  $2^k$  models.

## Properties

- Generalize to any criterion ( $R^2$ , AIC, BIC...)
- Efficient algorithm with pruning (“Leaps and Bound”)
- impossible as soon as  $p > 30$ .

# (Forward regression)

## Algorithm

1. Begin with  $\mathcal{S} = \emptyset$
2. at step  $k$  find the variable which, added to  $\mathcal{S}$ , gives the best model
- 2'. At step  $k$  find the best model by either adding or removing one variable.
- 3 etc. until  $p$  variables enter the model

## Properties

- Best model is understood as adjusted  $R^2$ , AIC, BIC. . .
- useful when  $p$  is large
- large bias, but variance/complexity controlled.
- “greedy” algorithm

# Forward-stepwise

## Algorithm

1. Begin with  $\mathcal{S} = \emptyset$
2. at step  $k$  find the variable which, added to  $\mathcal{S}$ , gives the best model
- 2'. At step  $k$  find the best model by either adding or removing one variable.
- 3 etc. until  $p$  variables enter the model

## Properties

- Best model is understood as adjusted  $R^2$ , AIC, BIC. . .
- useful when  $p$  is large
- large bias, but variance/complexity controlled.
- “greedy” algorithm

# Outline

## ① Motivations

## ② Variable Selection

Criteria for model comparison

Algorithms for variable subset selection

**Illustration: prostate cancer**

## ③ Regularisation

The ridge estimator

Model complexity and Tuning parameter

Definition of the LASSO estimator

Model complexity and Tuning parameter



# Exhaustive search I

```
library(leaps)
```

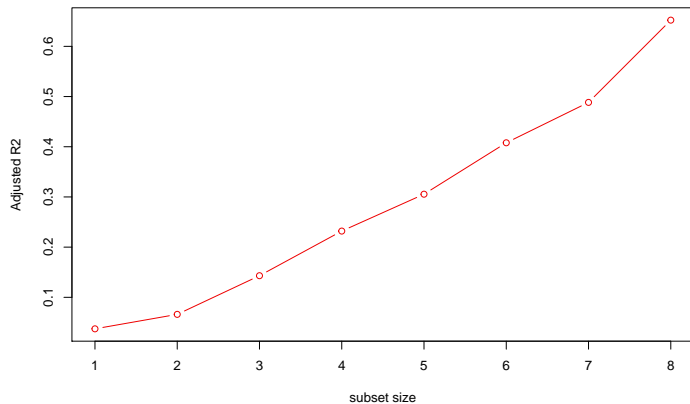
Get all possible models

```
out <- regsubsets(  
  lpsa ~ . ,  
  data = prostate_train,  
  nbest = 100,  
  really.big = TRUE  
)  
bss <- summary(out)
```

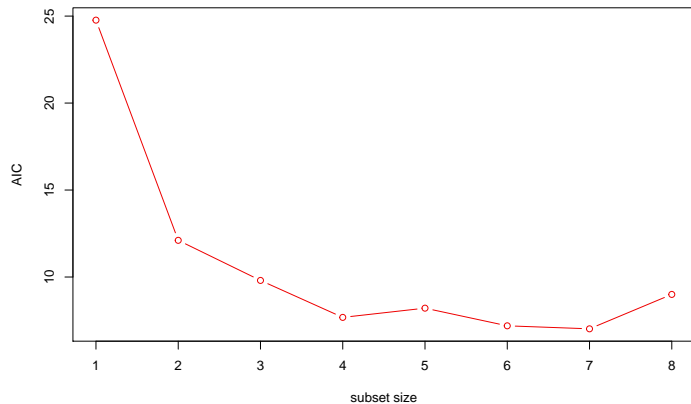
Extract size and RSS. Add the null model (with just the intercept)

```
intercept <- lm(lpsa ~ 1, data = prostate_train)
```

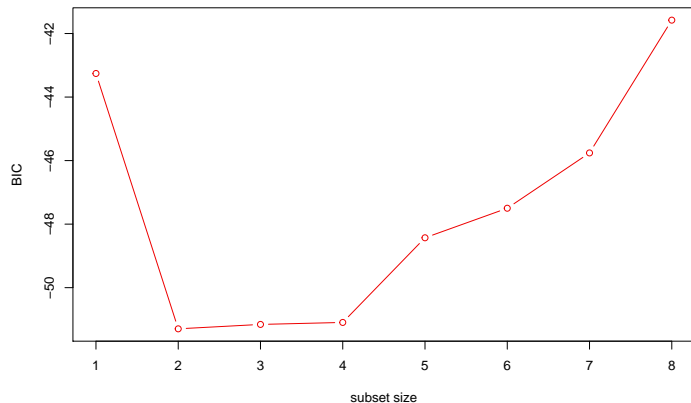
## Exhaustive search II



# Exhaustive search III



# Exhaustive search IV



# Forward-Stepwise (I)

Create the nul model and the full model

```
null <- lm(lpsa ~ 1, data = prostate_train)
full  <- lm(lpsa ~ ., data = prostate_train)
```

Create the scope of models

```
lower <- ~1
upper <- ~lcavol+lwght+age+lbph+svi+lcp+gleason+pgg45
scope <- list(lower = lower, upper = upper)
```

Stepwise with AIC: forward, backward, both

```
fwd  <- step(null, scope, direction = "forward", trace=FALSE)
bwd  <- step(full, scope, direction = "backward", trace=FALSE)
both <- step(null, scope, direction = "both",    trace=FALSE)
```

⇒ 3 equivalent models

# Forward regression

```
fwd

##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + svi + lbph, data = prostate_train)
##
## Coefficients:
## (Intercept)      lcavol      lweight          svi          lbph
##    -0.1185      0.3337      0.7219      0.1165      0.0746

fwd$anova

##           Step Df  Deviance Resid.  Df Resid. Dev      AIC
## 1              NA      NA        66  12.729028 -109.2741
## 2 + lcavol    -1  6.8420623    65   5.886966 -158.9408
## 3 + lweight   -1  0.9831846    64   4.903781 -169.1840
## 4 + svi       -1  0.2887517    63   4.615030 -171.2501
## 5 + lbph     -1  0.2766756    62   4.338354 -173.3922
```

# Backward regression

```
bwd

##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + age + lbph + svi + lcp +
##      pgg45, data = prostate_train)
##
## Coefficients:
## (Intercept)      lcavol      lweight          age          lbph
##    0.09420      0.37883      0.82953     -0.46510      0.07695
##          svi          lcp          pgg45
##    0.12858     -0.10586      0.12842

bwd$anova

##      Step Df    Deviance Resid. Df Resid. Dev      AIC
## 1      NA      NA         58    3.890358 -172.6948
## 2 - gleason  1 0.001443147      59    3.891801 -174.6700
```

# Stepwise regression

```
both

##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + svi + lbph, data = prostate_train)
##
## Coefficients:
## (Intercept)          lcavol          lweight             svi             lbph
##      -0.1185           0.3337           0.7219           0.1165           0.0746
```

```
both$anova
```

##	Step	Df	Deviance	Resid. Df	Resid. Dev	AIC
## 1		NA	NA	66	12.729028	-109.2741
## 2	+ lcavol	-1	6.8420623	65	5.886966	-158.9408
## 3	+ lweight	-1	0.9831846	64	4.903781	-169.1840
## 4	+ svi	-1	0.2887517	63	4.615030	-171.2501
## 5	+ lbph	-1	0.2766756	62	4.338354	-173.3922



# Performance on test data

```
print(err_ols)

## [1] 0.0673631

print(err_AIC.fwd <- mean((y_test - predict(fwd , prostate_test))^2))

## [1] 0.05838145

print(err_AIC.bwd <- mean((y_test - predict(bwd , prostate_test))^2))

## [1] 0.06680086

print(err_AIC <- mean((y_test - predict(both, prostate_test))^2))

## [1] 0.05838145
```

# Stepwise: BIC modification

More sparse model

```
BIC <- step(null, scope, k = log(n <- nrow(prostate)), trace=FALSE)
BIC

##
## Call:
## lm(formula = lpsa ~ lcavol + lweight, data = prostate_train)
##
## Coefficients:
## (Intercept)      lcavol      lweight
##      -0.3816       0.4143       0.9892

print(err_BIC <- mean((y_test - predict(BIC, prostate_test))^2))

## [1] 0.06342134
```

# Comments

## Interpretability

- ① If the true  $\mathcal{S}$  only contains a **few variables linked to the response**,  
     $\rightsquigarrow$  variable selection algorithms can retrieve relevant predictors.
- ② If the true  $\mathcal{S}$  contains **many correlated predictors**  
     $\rightsquigarrow$  the selected variables will be hardly interpretable.

## Stability issue

With strong correlation or when  $n < p$ , **small changes** in the data can induce **large discrepancies** between the sets of selected variables.

# Outline

## 1 Motivations

## 2 Variable Selection

## 3 Regularisation

- Motivations et principe

- Ridge regression

  - The ridge estimator

  - Model complexity and Tuning parameter

- Lasso Regression

  - Definition of the LASSO estimator

  - Model complexity and Tuning parameter

# Outline

## ① Motivations

## ② Variable Selection

## ③ Regularisation

### Motivations et principe

#### Ridge regression

The ridge estimator

Model complexity and Tuning parameter

#### Lasso Regression

Definition of the LASSO estimator

Model complexity and Tuning parameter

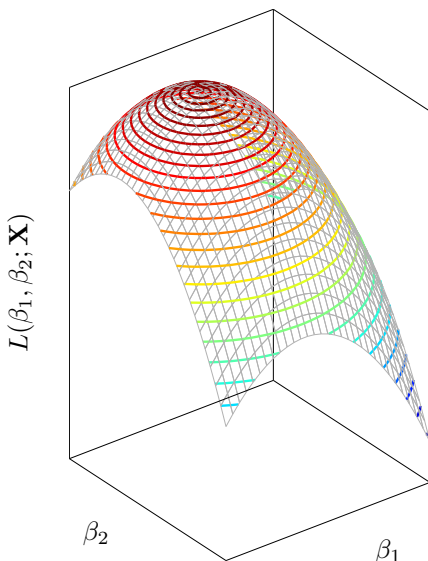
# Several goals

Control the parameter  $\hat{\beta}$  to

- ① **Regularize** the problem
  - For numerical purpose, (conditioning of  $\mathbf{X}^T \mathbf{X}$ ),
  - For stability purpose, (correlation between  $(X_1, \dots, X_p)$ ).
- ② **Enhance** the prediction
  - By trading a little bias vs variance
  - By controlling irrelevant variables
- ③ **Looking towards** interpretability
  - By controlling model complexity,
  - By embedding the variable selection (Lasso).

# A Geometric View of Shrinkage

## Constrained Optimization



We basically want to solve a problem of the form

$$\underset{\beta_1, \beta_2}{\text{maximize}} \quad L(\beta_1, \beta_2; \mathbf{X})$$

where  $L$  is typically a concave likelihood function.

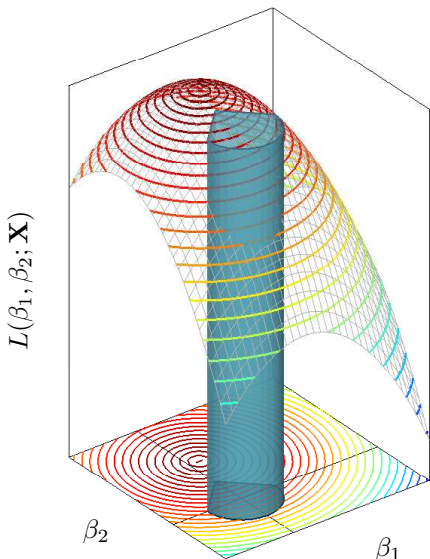
This is strictly equivalent to solve

$$\underset{\beta_1, \beta_2}{\text{minimize}} \quad L'(\beta_1, \beta_2; \mathbf{X})$$

where  $L' = -L$  is convex ! For instance the squared error loss in the OLS.

# A Geometric View of Shrinkage

## Constrained Optimization



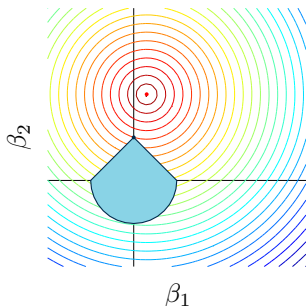
$$\begin{cases} \underset{\beta_1, \beta_2}{\text{maximize}} & L(\beta_1, \beta_2; \mathbf{X}) \\ \text{s.t.} & \Omega(\beta_1, \beta_2) \leq c \end{cases},$$

where  $\Omega$  defines a domain that *constrains*  $\beta$ .



# A Geometric View of Shrinkage

## Constrained Optimization



$$\begin{cases} \underset{\beta_1, \beta_2}{\text{maximize}} & L(\beta_1, \beta_2; \mathbf{X}) \\ \text{s.t.} & \Omega(\beta_1, \beta_2) \leq c \end{cases},$$

where  $\Omega$  defines a domain that *constrains*  $\beta$ .



$$\underset{\beta_1, \beta_2}{\text{minimize}} J(\beta),$$

with  $J$  the convex objective defined by

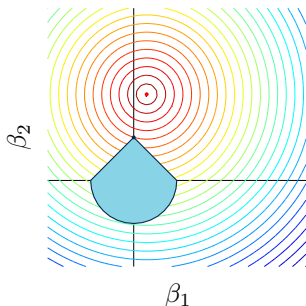
$$J(\beta) = -L(\beta_1, \beta_2; \mathbf{X}) + \lambda\Omega(\beta_1, \beta_2)$$

# A Geometric View of Shrinkage

## Constrained Optimization

$$\begin{cases} \underset{\beta_1, \beta_2}{\text{maximize}} & L(\beta_1, \beta_2; \mathbf{X}) \\ \text{s.t.} & \Omega(\beta_1, \beta_2) \leq c \end{cases},$$

where  $\Omega$  defines a domain that *constrains*  $\beta$ .



$$\underset{\beta_1, \beta_2}{\text{minimize}} J(\beta),$$

with  $J$  the convex objective defined by

$$J(\beta) = -L(\beta_1, \beta_2; \mathbf{X}) + \lambda\Omega(\beta_1, \beta_2)$$

How shall we define  $\Omega$  ?

# Outline

## ① Motivations

## ② Variable Selection

## ③ Regularisation

- Motivations et principe

### **Ridge regression**

- The ridge estimator

- Model complexity and Tuning parameter

### **Lasso Regression**

- Definition of the LASSO estimator

- Model complexity and Tuning parameter

# Outline

## ① Motivations

## ② Variable Selection

## ③ Regularisation

- Motivations et principe

- Ridge regression**

  - The ridge estimator

  - Model complexity and Tuning parameter

- Lasso Regression**

  - Definition of the LASSO estimator

  - Model complexity and Tuning parameter

# Definition

## Fact

If the  $\beta_j$  are unconstrained, they can have very high magnitude and thus large variances.

## Idea

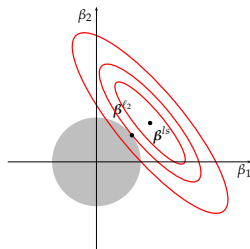
To control the variance, we should control the size of the coefficients in  $\beta$ . This could induce a large decrease of the prediction error.

## Ridge as a regularization problem

The ridge estimate of  $\beta$  is the solution to

$$\hat{\beta}^{\text{ridge}} = \arg \min_{\beta \in \mathbb{R}^{p+1}} \text{RSS}(\beta), \quad \text{s.t.} \quad \sum_{j=1}^p \beta_j^2 \leq s,$$

where  $s$  is a shrinkage factor.



## A 2-dimensional toy example

Consider that the true relationship is  $Y = X_1\beta_1 + X_2\beta_2 + \varepsilon$ . If  $X_1$  and  $X_2$  are strongly correlated, then  $X_1 \approx X_2$  and for any  $\gamma \geq 0$

$$\begin{aligned} Y &= X_1(\beta_1 + \gamma) + X_2(\beta_2 - \gamma) + \gamma(X_1 - X_2) + \varepsilon \\ &\approx X_1(\beta_1 + \gamma) + X_2(\beta_2 - \gamma) + \varepsilon. \end{aligned}$$

A large panel of fit with estimated  $\beta$  varying according to  $\gamma$  will produce the same prediction error.

For small  $s$  (or large  $\lambda$  in the Lagrangian form), the ridge controls

$$(\beta_1 + \gamma)^2 + (\beta_2 - \gamma)^2$$

which is minimal for  $\gamma = (\beta_2 - \beta_1)/2$ , and in this case  $\beta_j = (\beta_1 + \beta_2)/2$ .

## A 2-dimensional toy example

Consider that the true relationship is  $Y = X_1\beta_1 + X_2\beta_2 + \varepsilon$ . If  $X_1$  and  $X_2$  are strongly correlated, then  $X_1 \approx X_2$  and for any  $\gamma \geq 0$

$$\begin{aligned} Y &= X_1(\beta_1 + \gamma) + X_2(\beta_2 - \gamma) + \gamma(X_1 - X_2) + \varepsilon \\ &\approx X_1(\beta_1 + \gamma) + X_2(\beta_2 - \gamma) + \varepsilon. \end{aligned}$$

A large panel of fit with estimated  $\beta$  varying according to  $\gamma$  will produce the same prediction error.

For small  $s$  (or large  $\lambda$  in the Lagrangian form), the ridge controls

$$(\beta_1 + \gamma)^2 + (\beta_2 - \gamma)^2$$

which is minimal for  $\gamma = (\beta_2 - \beta_1)/2$ , and in this case  $\beta_j = (\beta_1 + \beta_2)/2$ .

# A 2-dimensional toy example (in R) I

Generate two correlated predictors

```
suppressMessages(library(quadrupen)) # use github version
x1 <- rnorm(5)
x2 <- x1 + rnorm(5,0, 0.5)
cor(x1,x2)

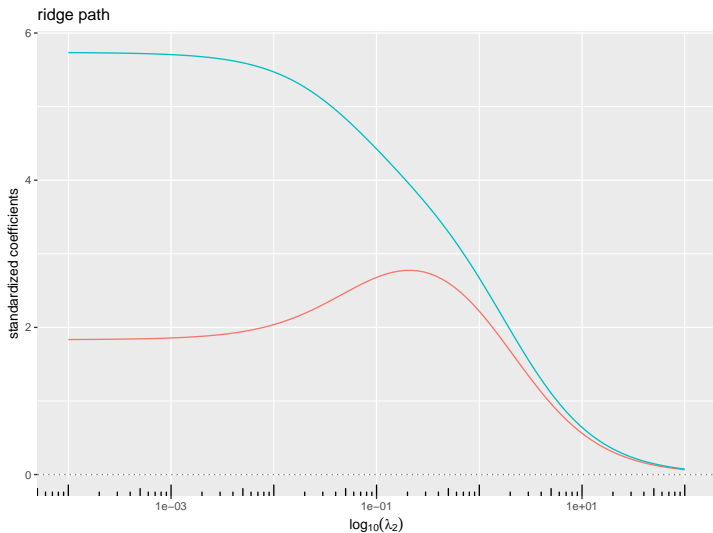
## [1] 0.934117
```

Draw  $Y$  and plot the **ridge regularisation path**

```
library(glmnet)
y <- x1 + x2 + rnorm(5)
plot(quadrupen::ridge(cbind(x1,x2),y))
```



## A 2-dimensional toy example (in $\mathbb{R}$ ) II



# Ridge as penalized regression

Dont penalize the intercept thus consider  $\beta = (\beta_1, \dots, \beta_p)$  and set

- $\hat{\beta}_0 = \bar{y} - \bar{x}\hat{\beta}$
- center  $y$  and  $x_j$ ,  $j = 1, \dots, p$ .

Standardize the  $x_j$  for the fit and send back  $\hat{\beta}^{\text{ridge}}$  to the original scale.

Convex Lagrangian form

$$\begin{aligned}\hat{\beta}^{\text{ridge}} &= \arg \min_{\beta \in \mathbb{R}^p} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda \|\beta\|^2 \\ &= (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}_p)^{-1} \mathbf{X}^\top \mathbf{y} = \mathbf{H}_\lambda \mathbf{y}.\end{aligned}$$

Strong convexity

Oppositely to the least squares, a non-singular solution always exists when  $\lambda > 0$  whatever the conditioning of  $\mathbf{X}^\top \mathbf{X}$  (original proposal).

# Ridge fit for the prostate cancer data

Compute the ridge path

```
ridge_path <- quadrupen::ridge(x_train, y_train)
```

Compute the prediction error on the test set for all  $\lambda$

```
err <- colMeans((y_test - predict(ridge_path, as.matrix(x_test)))^2)
```

Then,  $\lambda^*$  that minimizes this error

```
ridge_path@lambda2[which.min(err)]
```

```
## [1] 0.5722368
```

The prediction error is smaller than with the OLS

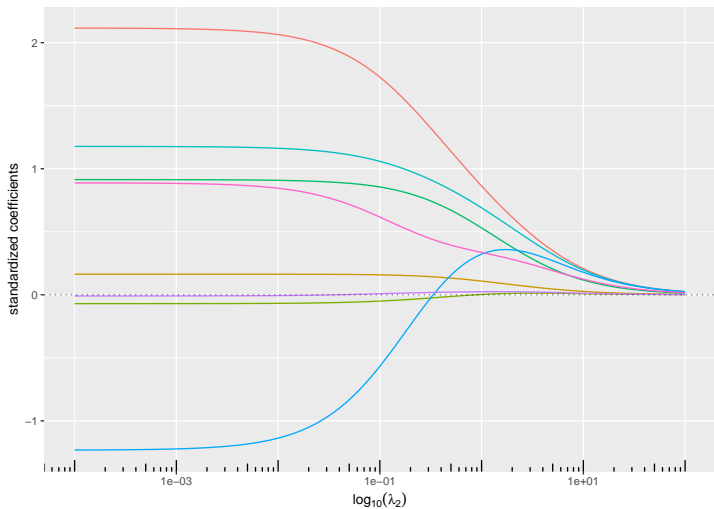
```
err_ridge <- err[which.min(err)]  
print(err_ridge)
```

```
##      0.572  
## 0.06618371
```

```
print(err_ols)
```

```
## [1] 0.0673631
```

ridge path



# Outline

## ① Motivations

## ② Variable Selection

## ③ Regularisation

- Motivations et principe

- Ridge regression**

  - The ridge estimator

  - Model complexity and Tuning parameter**

- Lasso Regression

  - Definition of the LASSO estimator

  - Model complexity and Tuning parameter

# Classical options

## Cross-validation

We compute  $CV(\lambda)$ , the CV error along the  $\lambda$  path

- ① if  $K = n$ , this is the LOOCV,
- ② if  $K = 2$ , this is the hold out estimation,
- ③ in a high dimensional setup, we must choose  $K$  “carefully”,

We choose  $\lambda$  minimising the CV

## Penalized criteria

We choose  $\lambda$  minimizing a criterion with the form

$$\text{crit}(\lambda) = \text{err}_{\mathcal{D}}(\lambda) + \text{pen}(\text{df}_{\lambda})$$

↪ What sens give to the degrees of freedom for ridge regression?

# Effective degrees of freedom

- Degrees of freedom of a model describes its complexity level.
- For the least squares,  $df = p$  (plus 1 for the intercept).
- Need a definition adapted to shrinkage methods.

## Definition (Efron and others)

Consider a fitted vector  $\hat{\mathbf{y}}$  from an observation  $\mathbf{y}$ . We define its degrees of freedom as

$$df(\hat{\mathbf{y}}) = \frac{1}{\sigma^2} \sum_{i=1}^n \text{cov}(\hat{y}_i, y_i).$$

$\leadsto$  The harder the fit to the data, the higher the covariance.

# Effective degrees of freedom: the ridge case

## Proposition

*Consider a linear fitting method that predicts  $\hat{\mathbf{y}}$  for entry  $\mathbf{y}$  through the smoother matrix  $\mathbf{H}$ :*

$$\hat{\mathbf{y}} = \mathbf{H}\mathbf{y}.$$

*The effective degrees of freedom of the model  $\hat{\mathbf{y}}$  verifies*

$$\text{df}(\hat{\mathbf{y}}) = \text{Tr}(\mathbf{H}).$$

## Ridge: effective degrees of freedom

For ridge regression,  $\text{df}$  is a decreassig function of  $\lambda$  which tends to 0 (or 1 when considering the intercept):

$$\text{df}(\hat{\mathbf{y}}_\lambda) = \sum_{i=1}^p \frac{d_i^2}{d_i^2 + \lambda}.$$



# Cross-Validation

Cross-validation is easily parallelized and is fast on small data sets

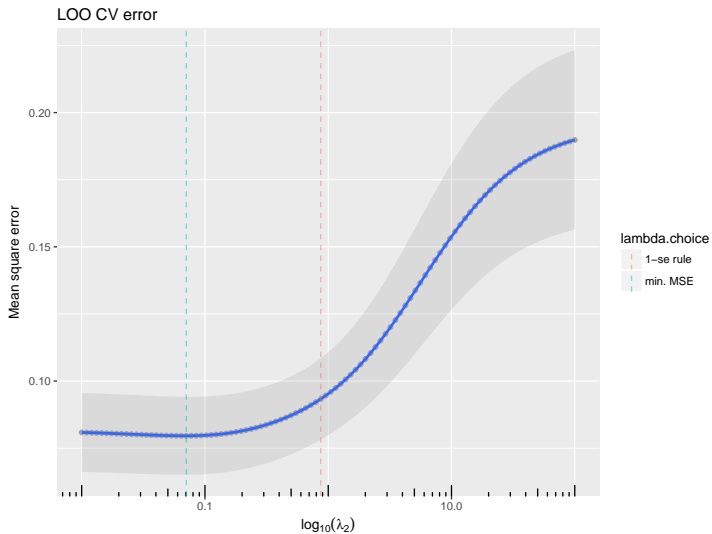
```
system.time(looc <- quadrupen::crossval(x_train,y_train, penalty = "ridge", K = n))
```

```
##  
## CROSS-VALIDATION FOR ridge REGULARIZER  
##  
## 67-fold CV on the lambda2 grid.  
##      user  system elapsed  
##    0.364   0.092   0.274
```

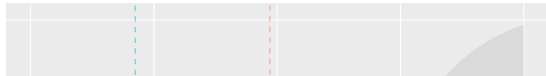
```
system.time(CV10 <- quadrupen::crossval(x_train,y_train, penalty = "ridge", K = 10))
```

```
##  
## CROSS-VALIDATION FOR ridge REGULARIZER  
##  
## 10-fold CV on the lambda2 grid.  
##      user  system elapsed  
##    0.488   0.228   0.173
```

# Leave one out

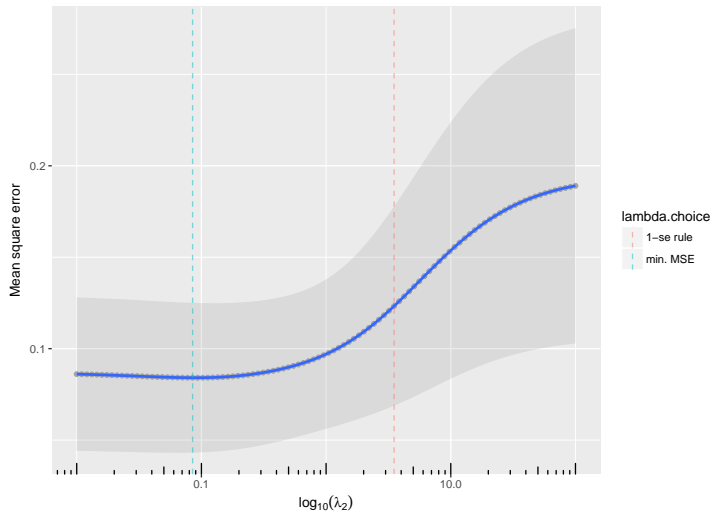


LOO CV error



# Ten fold

10-fold CV error



10-fold CV error



# Outline

## ① Motivations

## ② Variable Selection

## ③ Regularisation

- Motivations et principe

- Ridge regression

  - The ridge estimator

  - Model complexity and Tuning parameter

- Lasso Regression**

  - Definition of the LASSO estimator

  - Model complexity and Tuning parameter

# Outline

## ① Motivations

## ② Variable Selection

## ③ Regularisation

- Motivations et principe

- Ridge regression

  - The ridge estimator

  - Model complexity and Tuning parameter

- Lasso Regression

  - Definition of the LASSO estimator

  - Model complexity and Tuning parameter

# The Lasso

Least Absolute Shrinkage and Selection Operator

## Fact

Ridge performs regularization... but we also would like to select the most significant variables.

## Idea

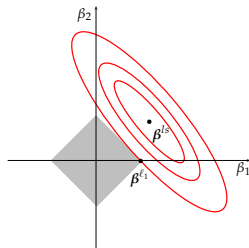
Suggest an admissible set that induces **sparsity** (force several entries to exactly zero in  $\hat{\beta}$ ).

Lasso as a convex optimization problem

The Lasso estimate  $\hat{\beta}^{\text{lasso}}$  solves

$$\underset{\beta \in \mathbb{R}^{p+1}}{\text{minimize}} \text{RSS}(\beta), \quad \text{s.t.} \quad \sum_{j=1}^p |\beta_j| \leq s,$$

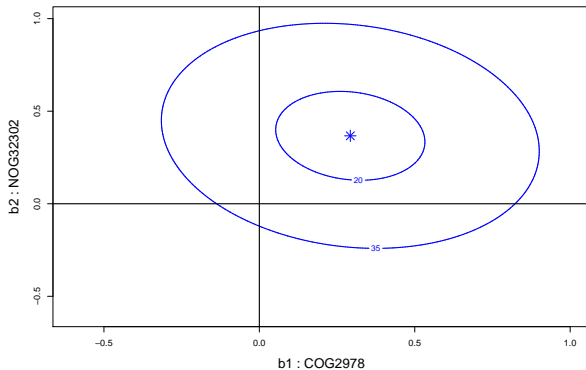
where  $s$  is a shrinkage factor.



# Some more insights: 2-dimensional example

Thanks to Sylvie Huet

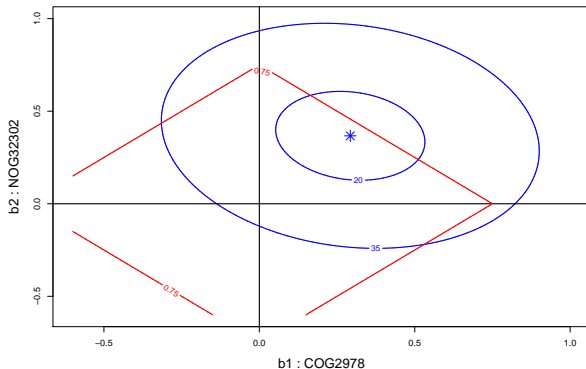
$$\sum_{i=1}^n (y_i - x_i^1 \beta_1 - x_i^2 \beta_2)^2, \quad \text{no constraints}$$



# Some more insights: 2-dimensional example

Thanks to Sylvie Huet

$$\sum_{i=1}^n (y_i - x_i^1 \beta_1 - x_i^2 \beta_2)^2, \quad \text{s.c. } |\beta_1| + |\beta_2| < 0.75$$

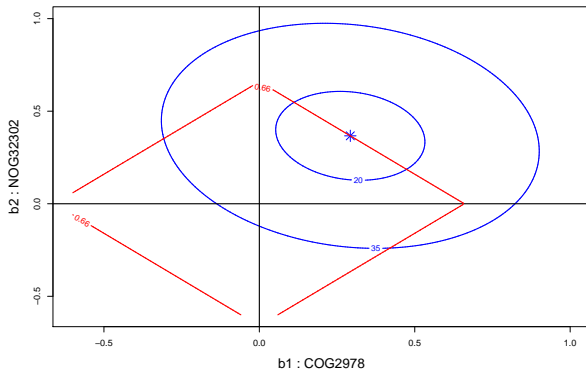




# Some more insights: 2-dimensional example

Thanks to Sylvie Huet

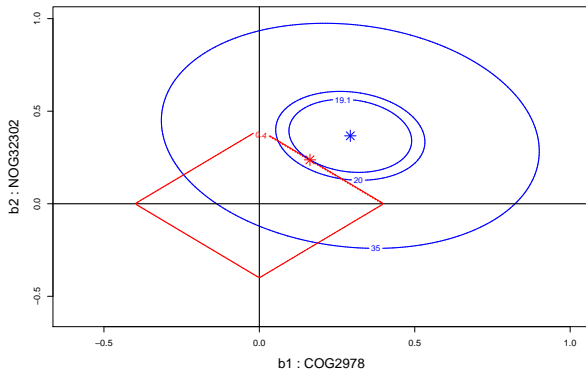
$$\sum_{i=1}^n (y_i - x_i^1 \beta_1 - x_i^2 \beta_2)^2, \quad \text{s.c. } |\beta_1| + |\beta_2| < 0.66$$



# Some more insights: 2-dimensional example

Thanks to Sylvie Huet

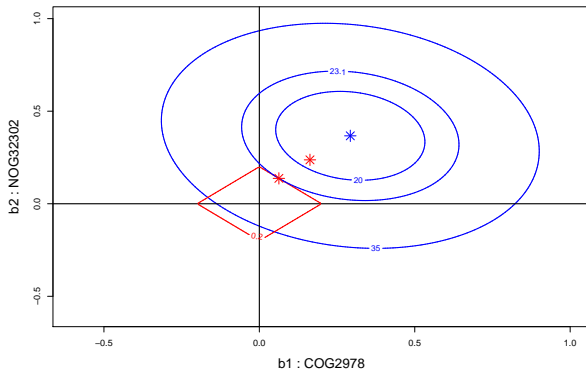
$$\sum_{i=1}^n (y_i - x_i^1 \beta_1 - x_i^2 \beta_2)^2, \quad \text{s.c. } |\beta_1| + |\beta_2| < 0.4$$



# Some more insights: 2-dimensional example

Thanks to Sylvie Huet

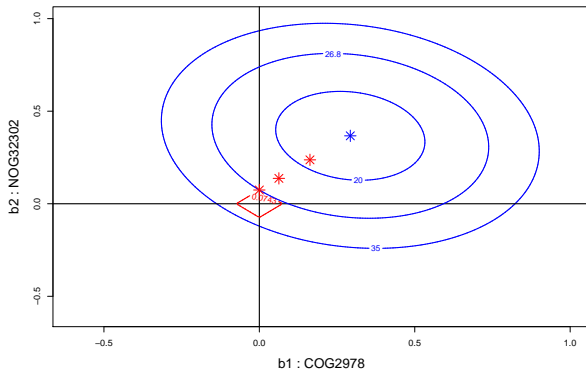
$$\sum_{i=1}^n (y_i - x_i^1 \beta_1 - x_i^2 \beta_2)^2, \quad \text{s.c. } |\beta_1| + |\beta_2| < 0.2$$



# Some more insights: 2-dimensional example

Thanks to Sylvie Huet

$$\sum_{i=1}^n (y_i - x_i^1 \beta_1 - x_i^2 \beta_2)^2, \quad \text{s.t. } |\beta_1| + |\beta_2| < 0.0743$$



# Lasso as penalized regression

Get rid of the intercept

We should not penalize the intercept term, thus

- $\hat{\beta}_0 = \bar{\mathbf{y}}$ ,
- center  $\mathbf{y}$  and  $\mathbf{x}_j$ ,  $j = 1, \dots, p$ ,
- scale the predictor before the fit,
- send  $\hat{\beta}$  back to the original scale.

Solve the convex,  $\ell_1$ -penalized problem

$$\hat{\beta}^{\text{lasso}} = \arg \min_{\beta \in \mathbb{R}^p} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda \|\beta\|_1,$$

whose solution has no close form, but always exists and is unique as soon as  $\mathbf{X}^\top \mathbf{X}$  has full rank.

$\rightsquigarrow$  Lasso performs regularization and variable selection but has no analytical solution.

# Lasso fit on the prostate cancer data I

## Compute the LASSO path

```
library(glmnet)
lasso_path <- quadrupen::lasso(x_train,y_train)
```

## Compute the prediction error on the test set for all $\lambda$

```
err <- colMeans((y_test - predict(lasso_path, x_test))^2)
```

Then,  $\lambda^*$  that minimizes this error

```
lasso_path@lambda1[which.min(err)]
```

```
## [1] 0.337835
```

# Lasso fit on the prostate cancer data II

The prediction error is smaller than with the OLS with only 5 coefficients

```
err[which.min(err)]
```

```
##      0.338
```

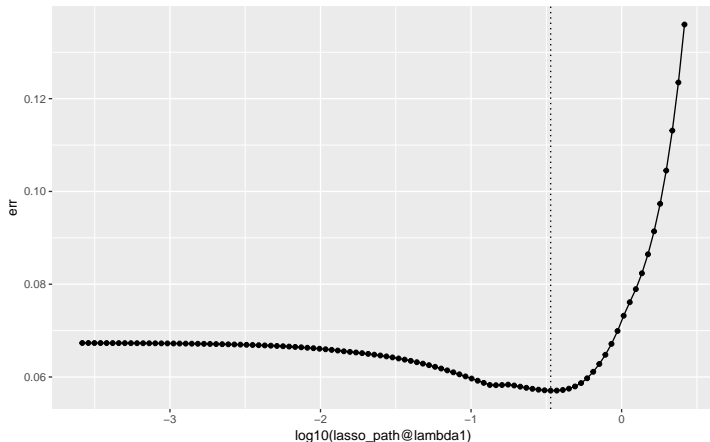
```
## 0.05705363
```

```
lasso_path@coefficients[which.min(err), ]
```

```
##      lcavol      lweight      age      lbph      svi      lcp  
## 0.30457102 0.63292397 0.00000000 0.03384766 0.06716404 0.00000000  
##      gleason      pgg45  
## 0.00000000 0.02819954
```

# Prediction error on the test set

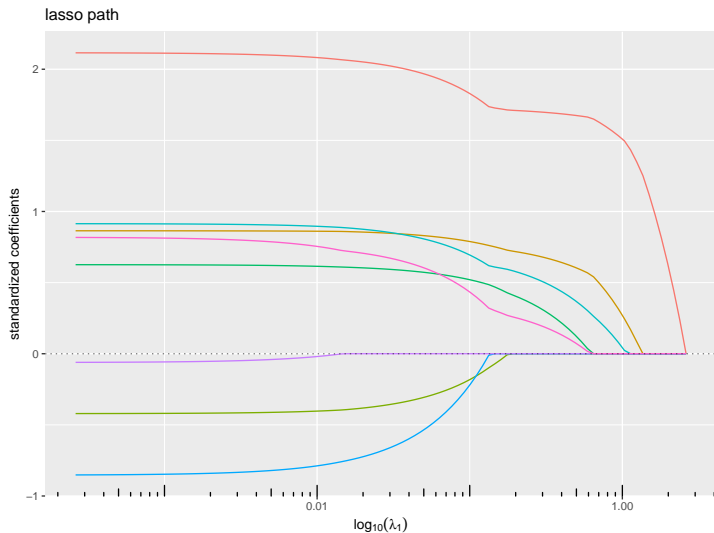
```
qplot(log10(lasso_path@lambda1), err) + geom_line() +  
geom_vline(xintercept = log10(lasso_path@lambda1[which.min(err)]), lty=3)
```





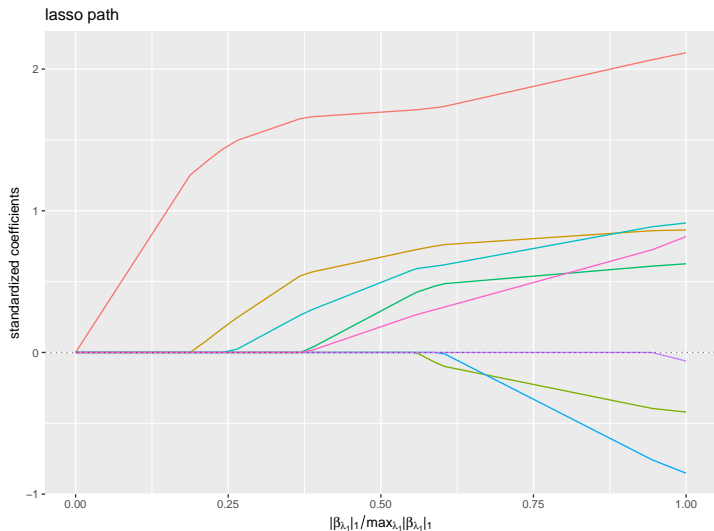
# Path of solution ( $\lambda$ )

```
plot(lasso_path)
```



# Path of solution (amount of shrinkage $s$ )

```
plot(lasso_path, xvar="fraction")
```



# Outline

## ① Motivations

## ② Variable Selection

## ③ Regularisation

- Motivations et principe

- Ridge regression

  - The ridge estimator

  - Model complexity and Tuning parameter

- Lasso Regression**

  - Definition of the LASSO estimator

  - Model complexity and Tuning parameter

# Critères pénalisés

## LASSO degrees of freedom

It simply equals the number of active (non-null) coefficients)

$$\text{df}(\hat{\mathbf{y}}_{\lambda}^{\text{lasso}}) = \text{card}(\{j : \beta_j(\lambda) \neq 0\}) = |\mathcal{A}|.$$

- Akaike Information Criterion

$$\text{AIC} = -2\log\text{lik} + 2\frac{|\mathcal{A}|}{n},$$

- Bayesian Information Criterion

$$\text{BIC} = -2\log\text{lik} + |\mathcal{A}| \log(n),$$

- modified BIC (when  $n < p$ )

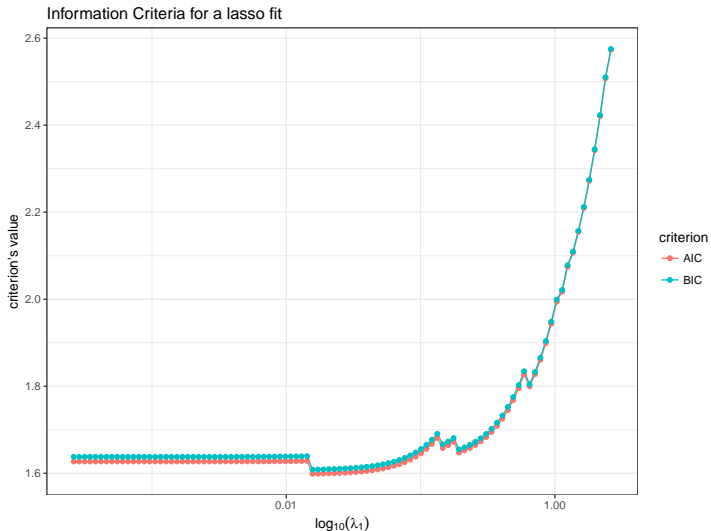
$$\text{mBIC} = -2\log\text{lik} + |\mathcal{A}| \log(p),$$

- Extended BIC add a prior on the number of model with size  $|\mathcal{A}|$

$$\text{eBIC} = -2\log\text{lik} + |\mathcal{A}|(\log(n) + 2\log(p)).$$

# Cancer de la prostate

Calcul de l' AIC/BIC en estimant  $\sigma$  (plot)



# Cross-validation

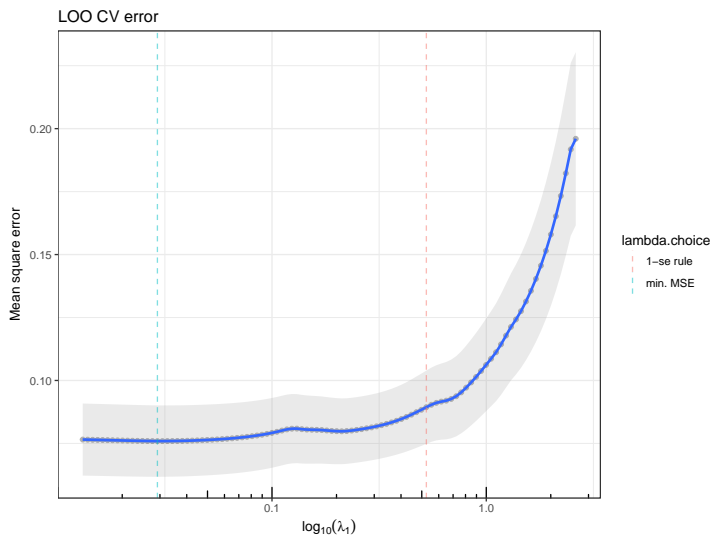
```
system.time(looc <- crossval(x_train, y_train, penalty = "lasso", K = n))
```

```
##  
## CROSS-VALIDATION FOR  lasso  REGULARIZER  
##  
## 67-fold CV on the lambda1 grid, lambda2 is fixed.  
##      user  system elapsed  
##    0.532    0.136    0.299
```

```
system.time(CV10 <- crossval(x_train, y_train, penalty = "lasso", K = 10))
```

```
##  
## CROSS-VALIDATION FOR  lasso  REGULARIZER  
##  
## 10-fold CV on the lambda1 grid, lambda2 is fixed.  
##      user  system elapsed  
##    0.476    0.168    0.243
```

# Leave one out

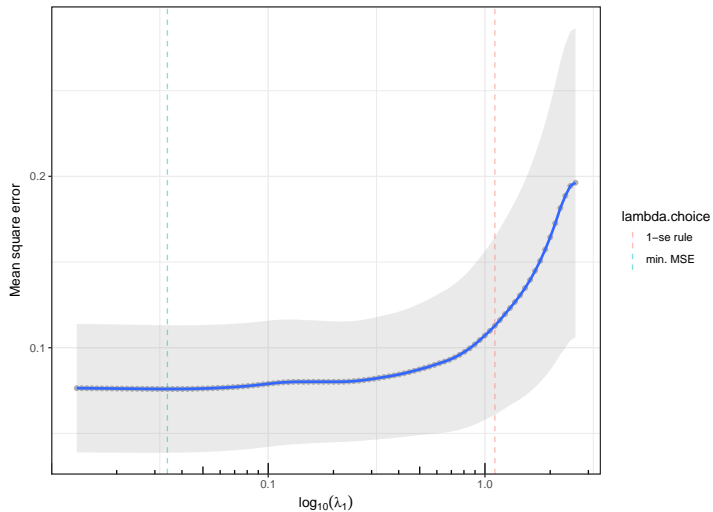


LOO CV error



# Ten fold

10-fold CV error



10-fold CV error

