Data Science - High dimensional regression

Summary

Linear models are popular methods for providing a regression of a response variable Y, that depends on covariates $(X_1, \ldots, X_p)$. We introduce the problem of high dimensional regression and provide some real examples where standard linear models methods are not well suited. Then, we propose some statistical resolution through the LASSO estimator and the Boosting algorithm. A practical session is proposed in the end of this Lecture, since the knowledge of these modern methods is needed in many fields.

1 Back to linear models

1.1 Sum of squares minimization

In a standard linear model, we have at our disposal (X_i, Y_i) supposed to be linked with

$$Y_i = X_i^t \beta^* + \epsilon_i, 1 \leq i \leq n.$$

In particular, each observation X_i is described by p variables $(X_i^1, \ldots, X_i^p)$, so that the former relation should be understood as

$$Y_i = \sum_{j=1}^{p} \beta_j^* X_i^j + \epsilon_i, 1 \leq i \leq n.$$

We aim to recover the unknown β^*.

- A classical "optimal" estimator is the MLE :

$$\hat{\beta}_{MLE} := \arg\max_{\beta \in \mathbb{R}^p} L(\beta, (X_i, Y_i)_{1 \leq i \leq n}),$$

  where L denotes the likelihood of the parameter β given the observations $(X_i, Y_i)_{1 \leq i \leq n}$.

- Generically, $(\epsilon_i)_{1 \leq i \leq n}$ is assumed to be i.i.d. replications of a centered and squared integrale noise

$$\mathbb{E}[\epsilon] = 0 \qquad \mathbb{E}[\epsilon^2] < \infty.$$

  A standard assumption even relies on the Gaussian structure of the errors $\epsilon_i \sim \mathcal{N}(0, 1)$ and in this case, the log-likelihood leads to the minimization of the sum of square and

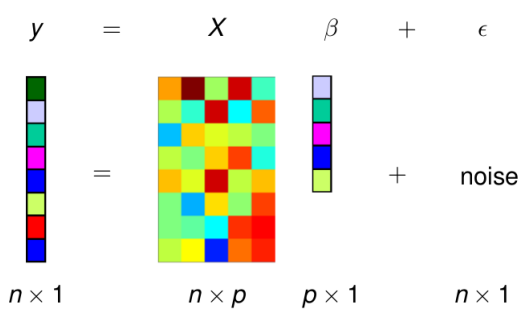
$$\hat{\beta}_{MLE} := \arg\min_{\beta \in \mathbb{R}^p} \underbrace{\sum_{i=1}^{n} \|Y_i - X_i^t \beta\|^2}_{:= J(\beta)}. \tag{1}$$

1.2 Matricial traduction & resolution

From a matricial point of view, the linear model can we written as follows :

$$Y = X\beta_0 + \epsilon, \qquad Y \in \mathbb{R}^n, X \in \mathcal{M}_{n,p}(\mathbb{R}), \beta_0 \in \mathbb{R}^p$$

In this lecture, we will consider situations where p varies (typically increases) with n.



It is an easy exercice to check that (1) leads to

$$\hat{\beta}_{MLE} := (X^t X)^{-1} X^t Y.$$

This can be obtained while remarking that J is a convex function, that possesses a unique minimizer if and only if X^tX has a full rank, meaning that J is indeed strongly convex :

$$D^2J = X^tX,$$

which is a squared $p \times p$ symmetric and positive matrix. It is non degenerate if X^tX has full rank, meaning that necessarily $p \leq n$.

PROPOSITION 1. — *$\hat{\beta}_{MLE}$ is an unbiased estimator of β_0 such that*

- *If $\epsilon \sim \mathcal{N}(0, \sigma^2)$: $\frac{\|X(\hat{beta}_{MLE} - \beta^*)\|_2^2}{\sigma^2} \sim \chi_p^2$*
- $$\mathbb{E}\left[\frac{\|X(\hat{\beta}_{MLE} - \beta^*)\|_2^2}{n}\right] = \frac{\sigma^2 p}{n}$$

Main requirement : X^tX must be full rank (invertible) !

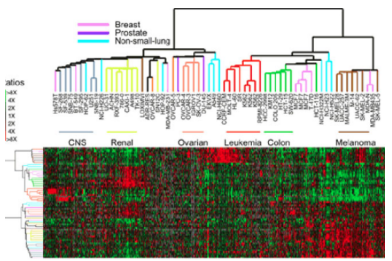
1.3 Difficulties in large dimensional case

Example One measures micro-array datasets built from a huge amount of profile genes expression. From a statistical point of view, we expect to find among the p variables that describe X important ones.

Number of genes p (of order thousands). Number of samples n (of order hundred).

-Y_i expression level of one gene on sample i

-$X_i = (X_{i,1}, \ldots, X_{i,p})$ biological signal (DNA micro-arrays)



Diagnostic help : healthy or ill ?

- Select among the genes meaningful elements : discover a cognitive link between DNA and the gene expression level.
- Find an algorithm with good prediction of the response ?

Linear model ? Difficult to imagine : $p > n$!

- X^tX is an $p \times p$ matrix, but its rank is lower than n. If $n << p$, then

$$rk(X^tX) \leq n << p.$$

- Consequence : the Gram matrix X^tX is not invertible and even very ill-conditionned (most of the eigenvalues are 0 !)
- The linear model $\hat{\beta}_{MLE}$ completely fails.
- One standard "improvement" : use the ridge regression with an additional penalty :

$$\hat{\beta}_n^{Ridge} = \arg\min_{\beta \in \mathbb{R}^p} \|Y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

  The ridge regression is a particular case of *penalized* regression. The penalization is still convex w.r.t. β and can be easily solved.
- We will attempt to describe a better suited penalized regression for high dimensional regression.
- Our goal : find a method that permits to find $\hat{\beta}_n$ such that :
  — Select features among the p variables.
  — Can be easily computed with numerical softs.
  — Possess some statistical guarantees.

1.4 Goals

Important and nowadays questions :

- What is a good framework for high dimensional regression ? A good model is required.
- How can we estimate ? An efficient algorithm is necessary.
- How can we measure the performances : prediction of Y ? Feature selection in β ? What are we looking for ?
- Statistical guarantees ? Some mathematical theorems ?

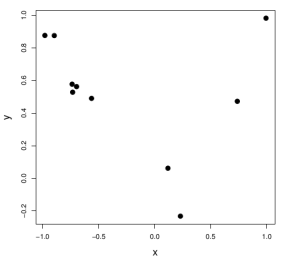
2 Penalized regression

2.1 Important balance : bias-variance tradeoff

A classical result in statistics states that a good estimator should achieve a balance between bias and variance.

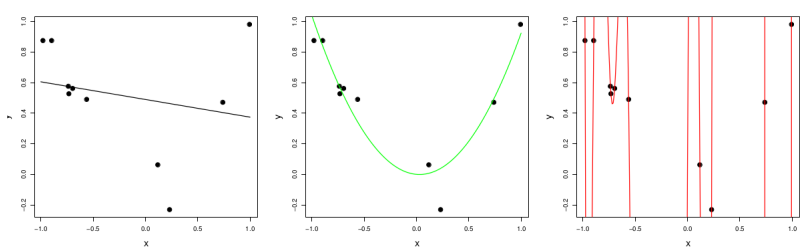
Example : In high dimension :

- Optimize the fit to the observed data ?
- Reduce the variability ?

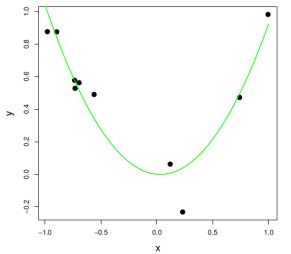


Standard question : find the best curve... In what sense ? Several regressions :

- Left : fit the best line (1-D regression)
- Middle : fit the best quadratic polynomial
- Right : fit the best 10-degree polynomial



Now I am interested in the prediction at point $x = 0.5$. What is the best ?

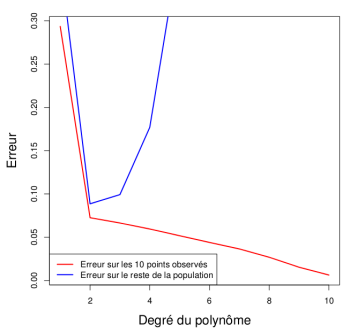


If we are looking for the best possible fit, a high dimensional regressor will be convenient.

Nevertheless, our goal is to generally to predict y for new points x and a standard matching criterion is

$$C(\hat{f}) := \mathbb{E}_{(X,Y)}[Y - \hat{f}(X)]^2.$$

It is a quadratic loss here, and should be replaced by other criteria (in classification for example).



- When the degree increases, the fit to the observed data (red curve) is always decreasing.
- Over the rest of the population, the generalization error starts decreasing, and after increases.
- Too simple sets of functions cannot contain the good function, and optimization over simple sets introduces a bias.

- Too complex sets of functions may contain the good function but are too rich and generates high variance.

The former balance is illustrated by a very simple theorem.

$$Y = f(X) + \epsilon \qquad \text{with} \qquad \mathbb{E}[\epsilon] = 0.$$

THÉORÈME 2. — *For any estimator $\hat{f}$, one has*

$$\begin{aligned} C(\hat{f}) = \mathbb{E}[Y - \hat{f}(X)]^2 \quad = \quad & \mathbb{E}\left[Y - \mathbb{E}[\hat{f}(X)]\right]^2 \\ & + \mathbb{E}\left[\mathbb{E}[\hat{f}(X)] - \hat{f}(X)\right]^2 \\ & + \mathbb{E}\left[Y - f(X)\right]^2 \end{aligned}$$

- The blue term is a bias term.
- The red term is a variance term.
- The green term is the Bayes risk and is independent on the estimator $\hat{f}$.

Statistical principle :

The empirical squared loss $\|Y - \hat{f}(X)\|_{2,n}^2$ mimics the bias. It is the sum of squares in (1). Important needs to introduce something to quantify the variance of estimation : this is provided by a penalty term.

2.2 Ridge regression as a preliminary (insufficient) response

Ridge Ridge regression is like ordinary linear regression, but it shrinks the estimated coefficients towards zero. The ridge coefficients are defined by solving

$$\hat{\beta}_{Ridge} := \arg\min_{\beta \in \mathbb{R}^p} \|Y - X^t\beta\|_2^2 + \lambda\|\beta\|_2^2$$

Here $\lambda \geq 0$ is a tuning parameter, which controls the strength of the penalty term. Write $\hat{\beta}_{Ridge}$ as the ridge solution. Note that :

- When $\lambda = 0$, we get the linear regression estimate
- When $\lambda = +\infty$, we get $\hat{\beta}_{Ridge} = 0$
- For λ in between, we are balancing two ideas : fitting a linear model of Y on X, and shrinking the coefficients.

Ridge with intercept When including an intercept term in the regression, we usually leave this coefficient unpenalized. Otherwise we could add some constant amount c to the vector Y, and this would not result in the same solution. Hence ridge regression with intercept solves

$$\hat{\beta}_{Ridge} := \arg\min_{c \in \mathbb{R}, \beta \in \mathbb{R}^p} \|Y - c - X^t\beta\|_2^2 + \lambda\|\beta\|_2^2$$

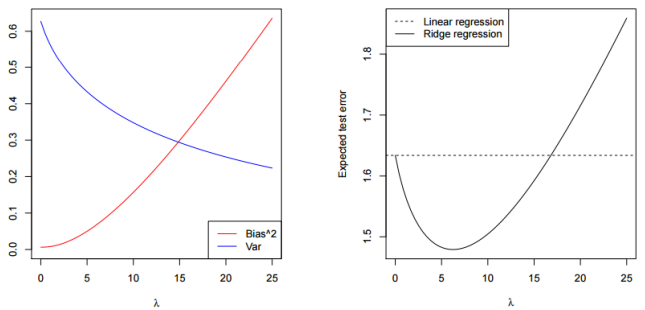
If we center the columns of X, then the intercept estimate ends up just being $\hat{c} = \bar{Y}$, so we usually just assume that Y and X have been centered and don't include an intercept.

Also, the penalty term $\|\beta\|_2^2$ is unfair is the predictor variables are not on the same scale. (Why ?) Therefore, if we know that the variables are not measured in the same units, we typically scale the columns of X (to have sample variance 1), and then we perform ridge regression.

Bias and variance of the ridge regression The bias and variance are not quite as simple to write down for ridge regression as they were for linear regression (see Proposition 1) but closed-form expressions are still possible. The general trend is :

- The bias increases as λ (amount of shrinkage) increases
- The variance decreases as λ (amount of shrinkage) increases

Think : what is the bias at $\lambda = 0$? The variance at $\lambda = +\infty$?



- What you may (should) be thinking now : this only work for some values of λ so how would we choose λ in practice ? As you can imagine, one way to do this involves cross-validation.
- What happens when we none of the true coefficients are small ? In other words, if all the true coefficients are moderate or large, is it still helpful to shrink the coeffi- cient estimates ? The answer is (perhaps surprisingly) still "yes". But the advantage of ridge regression here is less dramatic, and the corresponding range for good values of λ is smaller.

Variable selection To the other extreme (of a subset of small coefficients), suppose that there is a group of true coefficients that are identically zero. That is, that the mean outcome doesn't depend on these predictors at all, so they are completely extraneous.

The problem of picking out the relevant variables from a larger set is called variable selection. In the linear model setting, it means estimating some coefficients to be exactly zero. Aside from predictive accuracy, this can be very important for the purposes of model interpretation.

So how does ridge regression perform if a group of the true coefficients was exactly zero ? The answer depends whether on we are interested in prediction or interpretation. In terms of prediction, the answer is effectively exactly the same as what happens with a group of small true coefficients–there is no real difference in the case of a large number of covariates with a null effect.

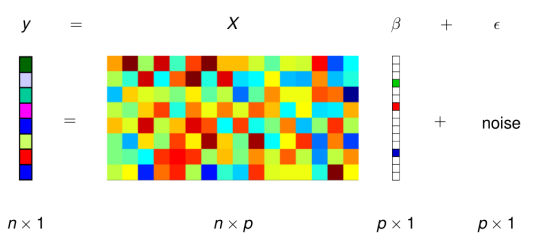
But for interpretation purposes, ridge regression does not provide as much help as we would like. This is because it shrinks components of its estimate toward zero, but never sets these components to be zero exactly (unless $\lambda = +\infty$, in which case all components are zero). So strictly speaking, ridge regression does not perform variable selection.

3 Sparsity : the Lasso re(s)volution

3.1 Sparsity assumption

An introductory example :

- In many applications, $p >> n$ but ...
- Important prior : many extracted feature in X are irrelevant
- In an equivalent way : many coefficients in β_0 are "exactly zero".
- For example, if Y is the size of a tumor, it might be reasonable to suppose that it can be expressed as a linear combination of genetic information in the genome described in X. BUT most components of X will be zero and most genes will be unimportant to predict Y :
  — We are looking for meaningful few genes
  — We are looking for the prediction of Y as well.



Dogmatic approach :

- Sparsity : assumption that the unknown β_0 we are looking for possesses its major coordinates null. Only s of them are important :

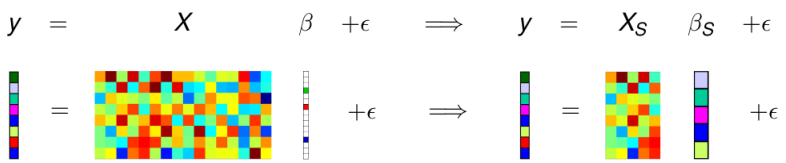
$$s := \text{Card}\left\{1 \leq i \leq p | \beta_0(i) \neq 0\right\}.$$

- Sparsity assumption :

$$s << n$$

- It permits to reduce the effective dimension of the problem.

- Assume that the effective support of β_0 is known, then



- If $\mathcal{S}$ is the support of β_0, maybe $X_{\mathcal{S}}^t X_{\mathcal{S}}$ is full rank, and linear model can be applied.

Major issue : How could we find $\mathcal{S}$?

3.2 Lasso relaxation

Ideally, we would like to find β such that

$$\hat{\beta}_n = \arg \min_{\beta : \|\beta\|_0 \leq s} \|Y - X\beta\|_2^2,$$

meaning that the minimization is embbeded in a ℓ_0 ball.

In the previous lecture, we have seen that it is a constrained minimization problem of a convex function ... A dual formulation is

$$\arg \min_{\beta : \|Y - X\beta\|_2 \leq \epsilon} \{\|\beta\|_0\}$$

But : The ℓ_0 balls are not convex and not smooth !

- First (illusive) idea : explore all ℓ_0 subsets and minimize ! Bullshit since :

$$C_p^s \quad \text{subsets and} \quad p \quad \text{is large !}$$

- Second idea (existing methods) : run some heuristic and greedy methods to explore ℓ_0 balls and compute an approximation of $\hat{\beta}_n$. (See below)
- Good idea : use a convexification of the $\|\|_0$ norm (also referred to as a convex relaxation method). How ?

Idea of the convex relaxation : instead of considering a variable $z \in \{0, 1\}$, imagine that $z \in [0, 1]$.

DÉFINITION 3. — *[Convex Envelope] The convex envelope f^* of a function f is the largest convex function below f.*

THÉORÈME 4. — *[Envelope of $\beta \longmapsto \|\beta\|_0$]*

- *On $[-1, 1]^d$, the convex envelope of $\beta \longmapsto \|\beta\|_0$ is $\beta \longmapsto \|\beta\|_1$.*
- *On $[-R, R]^d$, the convex envelope of $\beta \longmapsto \|\beta\|_0$ is $\beta \longmapsto \frac{\|\beta\|_1}{R}$.*

Idea : Instead of solving the minimization problem :

$$\forall s \in \mathbb{N} \qquad \min_{\|\beta\|_0 \leq s} \|Y - X\beta\|_2^2, \tag{2}$$

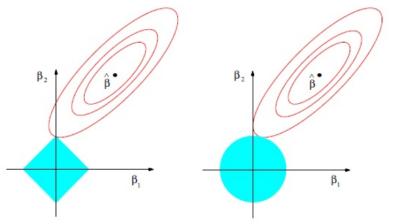
we are looking for

$$\forall C > 0 \qquad \min_{\|.\|_0^*(\beta) \leq C} \|Y - X\beta\|_2^2, \tag{3}$$

What's new ?

- The function $\|.\|_0^*$ is convex and thus the above problem is a convex minimization problem with convex constraints.
- Since $\|.\|_0^*(\beta) \leq \|\beta\|_0$, it is rather reasonnable to obtain sparse solutions. In fact, solutions of (3) with a given C provide a lower bound of solutions of (2) with $s \leq C$.
- If we are looking for good solutions of (2), then there must exists even better solution to (3).

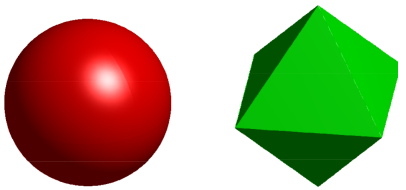
3.3 Geometrical interpretation (in 2 D)



Left : Level sets of $\|Y - X\beta\|_2^2$ and intersection with ℓ^1 ball. Right : Same with ℓ^2 ball.

The left constraint problem is likely to obtain a sparse solution. Oppositely, the right constraint no !

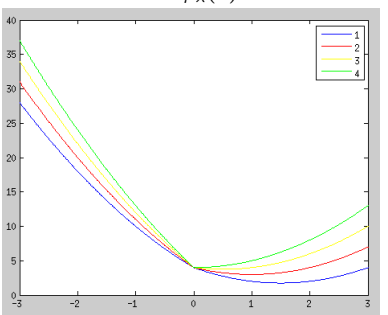
In larger dimensions the balls are even more different :



- Analytic point of view : why does the ℓ^1 norm induce sparsity ?
- From the KKT conditions (see Lecture 1), it leads to a **penalized criterion**

$$\min_{\beta\in\mathbb{R}^p:\|\beta\|_1\leq C} \|Y - X\beta\|_2^2 \Longleftrightarrow \min_{\beta\in\mathbb{R}^p} \underbrace{\|Y - X\beta\|_2^2}_{\text{Mimics the bias}} + \overbrace{\lambda\|\beta\|_1}^{\text{Controls the variance}}$$

- In the 1d case : $\arg\min_{\alpha\in\mathbb{R}} \frac{1}{2}\underbrace{|x-\alpha|^2 + \lambda|x|}_{:=\varphi_\lambda(x)}$:



- The minimal value of φ_λ is reached at point x^* when $0 \in \partial\varphi_\lambda(x^*)$. We can check that x^* is minimal iff
  — $x^* \neq 0$ and $(x^* - \alpha) + \lambda sgn(x^*) = 0$.
  — $x^* = 0$ and $d\varphi_\lambda^+(0) > 0$ and $d\varphi_\lambda^-(0) < 0$.
  PROPOSITION 5. — *[Analytical minimization of φ_λ]*

$$x^* = sgn(\alpha)[|\alpha| - \lambda]_+ = \arg\min_{x\in\mathbb{R}} \left\{\frac{1}{2}|x-\alpha|^2 + \lambda|x|\right\}$$

- For large values of λ, the minimum of φ_λ is reached at point 0.

3.4 Lasso estimator

We introduce the *Least Absolute Shrinkage and Selection Operator* :

$$\forall\lambda > 0 \qquad \hat{\beta}_n^{Lasso} = \arg\min_{\beta\in\mathbb{R}^p} \|Y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

The above criterion is convex w.r.t. β.

- Efficient algorithms to solve the LASSO, even for very large p.
- The minimizer may not be unique since the above criterion is not strongly convex.
- Predictions $X\hat{\beta}_n^{Lasso}$ are always unique.
- λ is a penalty constant that must be carefully chosen.
- A large value of λ leads to a very sparse solution, with an important bias.
- A low value of λ yields overfitting with no penalization (too much important variance).
- We will see that a careful balance between s, n and p exists. These parameters as well as the variance of the noise σ^2 influence a "good " choice of λ.

Alternative formulation :

$$\hat{\beta}_n^{Lasso} = \arg\min_{\beta\in\mathbb{R}^p:\|\beta\|_1\leq C} \|Y - X\beta\|_2^2$$

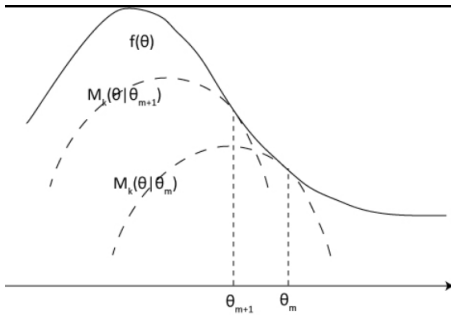
3.5 Principle of the MM algorithm

Algorithm is needed to solve the minimization problem

$$\arg\min_{\beta\in\mathbb{R}^p} \underbrace{\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1}_{:=\varphi_\lambda(\beta)} .$$

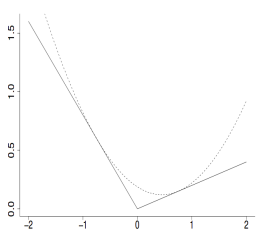
An efficient method follows the method of "Minimize Majorization" and is referred to as MM method.

- MM are useful for the minimization of a convex function/maximization of a concave one.
- Geometric illustration



- Idea : Build a sequence $(\beta_k)_{k\geq 0}$ that converges to the minimum of φ_λ.
- A particular case of such a method is encountered with the E.M. algorithm useful for clustering and mixture models.
- MM algorithms are powerful, especially they can convert non-differentiable problems to smooth ones.

1. A function $g(\beta, \beta_k)$ is said to *majorize* f at point β_k if
$$g(\beta_k|\beta_k) = f(\beta_k) \qquad \text{and} \qquad g(\beta|\beta_k) \geq f(\beta), \forall \beta \in \mathbb{R}^p.$$
2. Then, we define
$$\beta_{k+1} = \arg\min_{\beta\in\mathbb{R}^p} g(\beta|\beta_k)$$
3. We wish to find each time a function $g(., \beta_k)$ whose minimization is easy.
4. An example with a quadratic majorizer of a non-smooth function :



5. Important remark : The MM is a descent algorithm :
$$\begin{aligned} f(\beta_{k+1}) &= g(\beta_{k+1}|\beta_k) + f(\beta_{k+1}) - g(\beta_{k+1}|\beta_k) \\ &\leq g(\beta_k|\beta_k) = f(\beta_k) \end{aligned} \tag{4}$$

3.6 MM algorithm for the LASSO

We can deduce for the LASSO the coordinate descent algorithm

1. Define a sequence $(\beta_k)_{k\geq 0} \Longleftrightarrow$ find a suitable majorization.
2. $g : \beta \longmapsto \|Y - X\beta\|^2$ is convex, whose Hessian matrix is X^tX. A Taylor's expansion leads to
$$\forall y \in \mathbb{R}^p \qquad g(y) \leq g(x) + \langle \nabla g(x), y - x\rangle + \rho(X)\|y - x\|^2,$$
where $\rho(X)$ is the spectral radius of X.
3. We are naturally driven to upper bound φ_λ as
$$\begin{aligned} \varphi_\lambda(\beta) &\leq \varphi_\lambda(\beta_k) + \langle \nabla g(\beta_k), \beta - \beta_k\rangle + \rho(X)\|\beta - \beta_k\|_2^2 + \lambda\|\beta\|_1 \\ &= \psi(\beta_k) + \rho(X)\left\|\beta - \left(\beta_k - \frac{\nabla g(\beta_k)}{\rho(X)}\right)\right\|_2^2 + \lambda\|\beta\|_1 := \varphi_k(\beta) \end{aligned}$$
The important point with this majorization is that it is "tensorized" : each coordinates acts separately on $\varphi_k(\beta)$.
4. To minimize the majorization of φ_λ, we then use the above proposition of soft-thresholding :
   - Define
   $$\tilde{\beta}_k^j := \beta_k^j - \nabla g(\beta_k)^j/\rho(X).$$
   - Compute
   $$\beta_{k+1}^j = sgn(\tilde{\beta}_k^j)\max\left[|\beta_k^j| - \frac{2\lambda}{\rho(X)}\right]_+$$

4 Running the Lasso

4.1 Choice of the regularization parameter

It is an important issue to obtain a good performance of the method, and could be almost qualified as a "tarte à la crême" issue.

We won't provide a sharp presentation of the best known results to keep the level understandable.

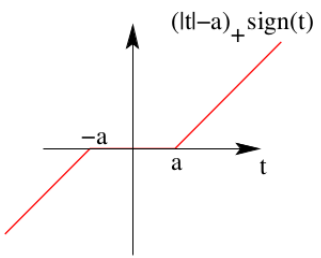
It is important to have in mind the extremely favorable situation of an almost orthogonal design :

$$\frac{X^t X}{n} \simeq I_p.$$

In this case solving the lasso is equivalent to

$$\min_w \frac{1}{2n}\|X^t y - w\|_2^2 + \lambda\|w\|_1$$

Solutions are given by ST (Soft-Thresholding) :



$$w_j = ST_\lambda\left(\frac{1}{n}X_j^t y\right) = ST_\lambda\left(\theta_j^0 + \frac{1}{n}X_j^t\epsilon\right)$$

We would like to keep the useless coefficients to 0, which requires that

$$\lambda \geq \frac{1}{n}X_j^t\epsilon, \forall j \in J_0^c.$$

The random variables $\frac{1}{n}X_j^t\epsilon$ are i.i.d. with a variance σ^2/n.

PROPOSITION 6. — *The expectation of the maximum of $p-s$ Gaussian standard variables is*

$$\mathbb{E}[\max_{1\leq i\leq p-s} X_i] \sim \sqrt{2\log(p-s)}.$$

We are naturally driven to the choice

$$\lambda = A\sigma\sqrt{\frac{\log p}{n}}, \qquad \text{with} \qquad A > \sqrt{2}.$$

Precisely :

$$\mathbb{P}\left(\forall j \in J_0^c : |X_j^t\epsilon| \leq n\lambda\right) \geq 1 - p^{1-A^2/2}.$$

4.2 Theoretical consistency

An additionnal remark is that we expect $ST_\lambda \longmapsto Id$ to obtain a consistency result. It means that $\lambda \longmapsto 0$, so that

$$\frac{\log p}{n} \longmapsto 0$$

Hence, a good behaviour of the lasso can be expected only if we have the next dimensional settings :

$$p_n = \mathcal{O}(\exp(n^{1-\xi})).$$

THÉORÈME 7. — *Assume that $\log p << n$, X has norm 1 and $\epsilon_i \sim \mathcal{N}(0,\sigma^2)$, then under a coherence assumption on the design matrix $X^t X$, one has*

i) With high probability, $J(\hat{\theta}_n) \subset J_0$.

ii) There exists C such that, with high probability,

$$\frac{\|X(\theta_n - \theta_0)\|_2^2}{n} \leq \frac{C}{\kappa^2}\frac{\sigma^2 s_0 \log p}{n},$$

where κ^2 is a positive constant that depends on the correlations in $X^t X$.

One can also find results on the exact support recovery, as well as some weaker results without any coherence assumption.

N.B. : Such a coherence is measured through the almost orthogonality of the colums of X. It can be traduced in terms of

$$|\sup_{i\neq j}\langle X_i, X_j\rangle| \leq \epsilon.$$

4.3 Practical calibration of λ

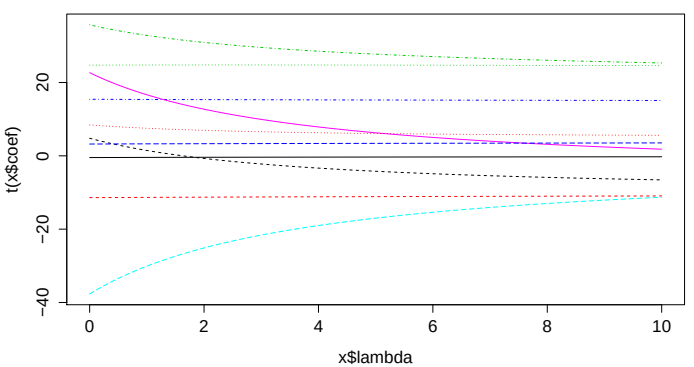
In practice, λ is generally chosen according to a criterion that is *data dependent*, *e.g.* a criterion that is calibrated on the observations through a cross-validation approach. In general, the packages implement this automatic choice of the regularization parameter with a CV option.

5 Numerical example

5.1 Very brief R code

5.1.1 About the use of the Ridge regression

```
library(lars)
data(diabetes)
library(MASS)
diabetes.ridge <- lm.ridge(diabetes$y ~ diabetes$x,
        lambda=seq(0,10,0.05))
plot(diabetes.ridge, lwd=3)
```

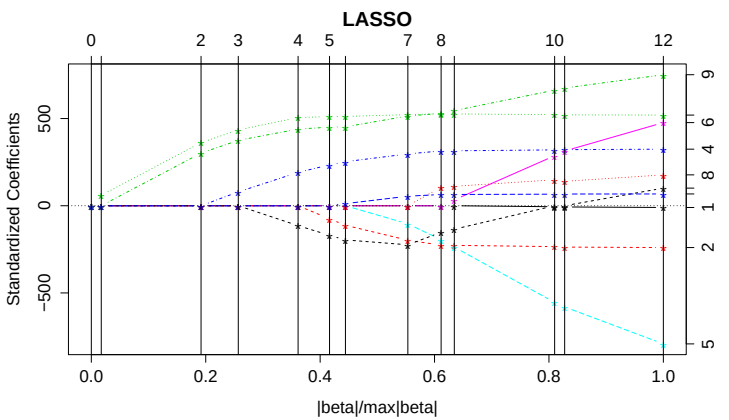


We can see that the influence of the regularization parameter λ of the ridge regression is important ! But a good choice of λ is difficult and should be data-driven. That is why a cross-validation procedure is needed. Does the ridge regression performs variable selection ?

5.1.2 About the use of the Lasso regression

```
library(lars)
data(diabetes)
diabetes.lasso = lars(diabetes$x, diabetes$y,
                      type='lasso')
plot(diabetes.lasso)
```

Lars algorithm : solves the Lasso less efficiently than the coordinate descent algorithm.



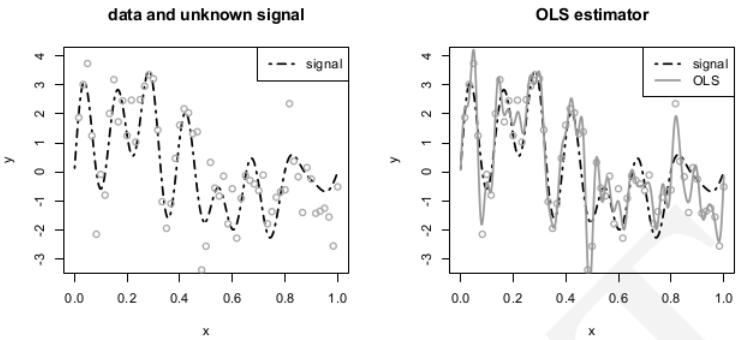
Typical output of the Lars software :

- The greater ℓ^1 norm, the lower λ
- Sparse solution with small values of the $\|.\|_1$ norm.

We can see that each variable of the diabetes dataset enter the model successively as long as λ decreases to 0. Again, the choice of λ should be done carefully with a data-driven criterion.

5.2 Removing the bias of the Lasso

Signal processing example :



We have $n = 60$ noisy observations $Y(i) = f(i/n) + \epsilon_i$. f is an unknown periodic function defined on $[0,1]$, sampled at points (i/n). ϵ_i are independent realizations of Gaussian r.v. We use the 50 first Fourier coefficients :

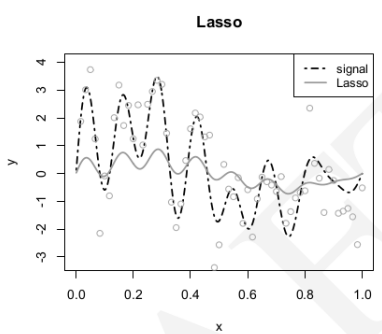
$$\varphi_0(x) = 1, \qquad \varphi_{2j}(x) = \sin(2j\pi x) \qquad \varphi_{2j+1}(x) = \cos(2j\pi x),$$

to approximate f. The OLS estimator is

$$\hat{f}^{OLS}(x) = \sum_{j=1}^{p} \hat{\beta}_j^{OLS} \varphi_j(x) \qquad \text{with} \qquad \hat{\beta}^{OLS} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - \sum_{j=0}^{p} \beta_j \varphi_j(i/n))^2.$$

The OLS does not perform well on this example.

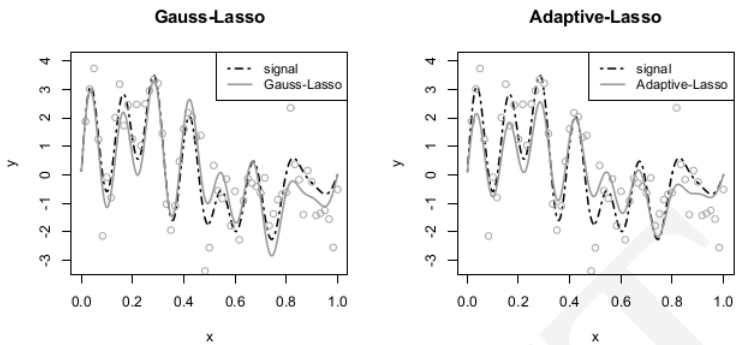
We experiment here the Lasso estimator with $\lambda = 3\sigma\sqrt{\frac{2\log p}{n}}$ and obtain



We define

$$\hat{f}^{\text{Gauss}} = \pi_{\hat{J}_0}(Y) \qquad \text{with} \qquad \hat{J}_0 = \text{Supp}(\hat{\theta}^{\text{Lasso}}),$$

where $\pi_{\hat{J}_0}$ is the $\mathbb{L}^2$ projection of the observations on the features selected by the Lasso.



The Adaptive Lasso is almost equivalent :

$$\beta^{\text{Adaptive Lasso}} = \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_2^2 + \mu \sum_{j=1}^{p} \frac{|\beta_j|}{|\hat{\beta}_j^{\text{Gauss}}|} \right\}$$

This minimization remains convex and the penalty term aims to mimic the ℓ^0 penalty.

The Adaptive Lasso is very popular and tends to select more accurately the variables than the Gauss-Lasso estimator.

- Lasso estimator reproduces the oscillations of f but these oscillations are shrunked to 0.
- When considering the initial minimization problem, the ℓ^1 penalty select nicely the good features, but introduces also a bias (introduces a shrinkage of the parameters).
- Strategy : select features with the Lasso and run an OLS estimator using the good variables.

6 Aggregation Boosting

6.1 Aggregation of estimators

Boosting refers to a widespread family of method for aggregating weak learners and producing reliable estimators. The underlying idea of aggregation is as follows. You assume you have in your hands a set of predictors $(f_i)_{1\leq i\leq k}$. Each predictor is a function from $\mathcal{X} \longrightarrow \mathbb{R}$ and we observe some realizations of

$$Y = g(X) + \epsilon.$$

We assume that each predictor f_i is estimated according to a training set $\mathcal{D}_n := (X_1, Y_1), \ldots, (X_n, Y_n)$. The idea of aggregation is to produce a robust estimation according to the preliminary estimators $(f_i)_{i\leq k}$ while using a set of weights w_i that are "optimal" from a statistical point of view. The final estimator obtained at the end of the algorithm is then of the form

$$\hat{f}_{Aggregation} := \sum_{i=1}^{k} w_i f_i.$$

and we expect that the loss of $\hat{f}_{Aggregation}$ is much better than the loss of each individual predictor f_i. Therefore, the main questions are :

- How to build the preliminary estimators (f_i) ?
- How to compute the weights of aggregation ?

We need to design good procedures for these two steps, and to do this, we have to use the learning set $\mathcal{D}_n$.

We should end this introductory paragraph saying that we can boost some weak predictors for optimizing regression task as well as classification task. Below, we will focus on the particular case of regression, as it is the main subject of this lecture.

6.2 $\mathbb{L}^2$-Boosting (Buhlmann & Yu)

6.2.1 Approximation procedure

The $\mathbb{L}^2$-Boosting estimator works as follows. We approach a (centered) linear function

$$m = \sum_{j=1}^{p} a_j f_j$$

with a recursive strategy.

We need a set of weak learners (dictionary) **that are centered and normalized** :

$$\forall 1 \leq j \leq p \qquad < f_j >= \int_{\mathcal{X}} f_j = 0 \qquad \text{and} \qquad \|f_j\| = \int_{\mathcal{X}} f_j^2 = 1$$

Algorithm 1 Weak greedy approximation - WGA

Input Shrinkage parameter ν. Function m. Weak learners $(f_j)_{1\leq j\leq p}$
Initialization : Define your prediction as $G_0 = 0$ and your rest $R_0 = m$.
Iterate Choose $\hat{j}_k$ such that

$$|\langle f_{\hat{j}_k}, R_{k-1}\rangle| = \max_{1\leq j\leq p} |\langle f_j, R_{k-1}\rangle| \tag{5}$$

Update the prediction as

$$G_k = G_{k-1} + \nu\langle R_{k-1}, f_{\hat{j}_k}\rangle \qquad \text{and} \qquad R_k = R_{k-1} - \nu\langle R_{k-1}, f_{\hat{j}_k}\rangle$$

Output : $\lim_{k\longrightarrow+\infty} G_k$

6.2.2 Estimation procedure

Of course, to adapt this approximation method to the empirical setting, we need to handle the empirical inner product instead of the theoretical one. This adaptation produces the $\mathbb{L}^2$-boosting algorithm.

6.2.3 Theoretical result

The $\mathbb{L}^2$ boosting algorithm is shown to be efficient from a statistical point of view, again under statistical sparsity assumption on the linear model or on the decomposition with any dictionary.

THÉORÈME 8. — *Assume that $\|f_j\|_2^2 = 1$ and that a $\xi > 0$ exists such that*

$$\log(p_n) = O(n^{1-\xi})$$

. At last, assume that $\|a\|_0 \leq s$, then if $|a_j| > n^{-\kappa}\mathbf{1}_{a_j\neq 0}$ with $\kappa > 0$ and some coherence assumption holds on X^tX, then a early stopping procedure at iteration $k_n \propto C\log(n)$ satisfies with high probability :

Algorithm 2 $\mathbb{L}^2$-boosting algorithm.

Input Shrinkage parameter ν. Dataset (X_i, Y_i). Weak learners $(f_j)_{1\leq j\leq p}$
Initialization : Define your prediction as $G_0 = 0$.
Iterate Choose $\hat{j}_k$ such that

$$|\langle f_{\hat{j}_k}, Y - G_{k-1}\rangle| = \max_{1\leq j\leq p} |\langle f_j, Y - G_{k-1}\rangle| \tag{6}$$

Update the prediction as

$$G_k = G_{k-1} + \nu\langle Y - G_{k-1}, f_{\hat{j}_k}\rangle$$

Output : G_{k_n}

- *The estimated support is included in the one of a.*
- $\|a_{k_n} - a\| \longrightarrow 0$.

What should be kept in mind is that Boosting procedures are shown to be consistent when the number of iterations is reasonable, meaning that it can overfit when the number of iteration becomes too large.

6.2.4 Implementation

The boosting procedure is very easy to implement. Try it ! ν should be chosen of the order $0.1, 0.2$...

6.3 Exponential weighting Aggregation

6.3.1 Bayesian point of view

We briefly discuss on another family of aggregation procedures that relies on a Bayesian point of view which is defined through a prior distribution on $\theta \in \mathbb{R}^p$ and a posterior distribution π_n.

A prior distribution π_0 is a probability distribution on $\mathbb{R}^p$ and the posterior distribution is described through the Bayes rule. This posterior distribution is proportional to

$$\mathbb{P}[\mathcal{D}_n|\theta]\pi_0(\theta)d\theta$$

where $\mathcal{D}_n = (X_1, Y_1), \ldots, (X_n, Y_n)$. The Gaussian assumption on the relationship between Y and $\langle X, \theta\rangle$ then shows that

$$\mathbb{P}[\mathcal{D}_n|\theta] \propto \exp\left(-\sum_{i=1}^n [Y_i - \langle X_i, \theta\rangle]^2\right).$$

Hence, the posterior distribution is then proportional to

$$\pi_n(\theta) \propto \exp\left(-\sum_{i=1}^n [Y_i - \langle X_i, \theta\rangle]^2 + \log(\pi_0(\theta))\right).$$

6.3.2 What to do with a Bayesian approach ?

Bayesian approaches are powerful because they make it possible to produce estimators, as well as confidence intervals. A natural Bayesian estimator is generally obtained with the help of the posterior mean :

$$\hat{\theta}_{posterior} := \int_\Theta \theta\pi_n(\theta).$$

Indeed, it is expected that the posterior distribution π_n converges when $n \longrightarrow +\infty$ towards $\delta_{\theta^\star}$ where $\theta^\star$ is the true parameter that links X and Y. Hence, the posterior mean should also behave as a consistant estimator of $\theta\star$.

It is thus intuitive trying computing $\hat{\theta}_{posterior}$ with the help of simulations of π_n. In some cases, this simulation is possible exactly (as if we were trying to sample a Gaussian distribution, or a Laplace distribution, ...

In a more general situation, π_n may not be sampled exactly, but thanks to Markov chains approximation, it is however possible to produce rapidly an approximation of π_n with the help of Langevin Markov stochastic processes or Metropolis-Hastings Markov chains. We will not explain why such approaches produce these invariant distributions and instead explain an efficient algorithm to sample π_n and then compute $\hat{\theta}_{posterior}$ in our large dimensional settings.

6.3.3 Prior distribution

The prior distribution is defined by

$$\log \pi_0(\theta) = \sum_{j=1}^p w(\alpha\theta_j) + 2\log(\tau^2 + \theta_j^2).$$

where $w(u)$ equals u^2 if $|u| \leq 1$ and equals $2|u|-1$ otherwise. Hence, the prior distribution involves an heavy tail distribution (the Cauchy distribution with $\exp(-2\log(\tau^2+\theta_j^2))$ and a Laplace distribution close to the Lasso penalty ℓ_1.

It can be shown the following result.

THÉORÈME 9. — *Assume that $\theta^\star$ is sparse and $\tau^2 = 16/(np)$ and $\alpha = \sqrt{\frac{n}{p}}/16$, then*

$$\mathbb{E}[|\hat{\theta}_{posterior} - \theta^\star|^2] \lesssim \frac{\sigma^2 s \log(p)}{n}.$$

6.3.4 Algorithm

We introduce

$$U_n(\theta) = \sum_{i=1}^n [Y_i - \langle X_i, \theta\rangle]^2 + \log(\pi_0(\theta)) + \sum_{j=1}^p w(\alpha\theta_j) + 2\log(\tau^2+\theta_j^2).$$

Now, the baseline stochastic process (X_t) solves the differential equation

$$dX_t = -\nabla U_n(X_t)dt + dB_t$$

that can be efficiently sampled with the following iterative scheme :

$$X_{(k+1)\delta} = X_{k\delta} - \delta\nabla U_n(X_{k\delta}) + \sqrt{\delta}\xi_k \tag{7}$$

where ξ_k is a sequence of i.i.d. Gaussian distributions $\mathcal{N}(0,1)$. Then, $\hat{\theta}_{posterior}$ is approximated by

$$\hat{\theta}_{\delta,T} = \frac{1}{T}\sum_{t=1}^T X_{t\delta},$$

i.e. the Cesaro average of the simulated iterative scheme (7)

6.3.5 Challenging simulation part

Program this estimation ! Code with Python !