Machine learning - Master ICFP 2019-2020

High-dimensional data and variable selection

Francis Bach

February 28, 2020

These notes are heavily based on notes from Alessandro Rudi and Pierre Gaillard.

1 Introduction

In statistics or machine learning, we often want to explain some output $Y \in \mathcal{Y}$ from input $X \in \mathcal{X} \subset \mathbb{R}^d$ by observing a data set $D_n = \{(X_i, Y_i)\}_{1 \leq i \leq n}$ of i.i.d. observations. In previous lessons, we saw methods such as ordinary least squares regression, K-nearest neighbors, probabilist models or kernel regression. Today, we would like to deal with high-dimensional input spaces, i.e., large d (possibly $d \gg n$). We will have two motivations in mind:

- *prediction, accuracy*: when $d \gg n$, classical models fail. Is it possible to have strong theoretical guarantees on the risk (i.e., generalization error)?
- *model interpretability*: by removing irrelevant features X_i (i.e., by setting the corresponding coefficients estimates to zero), the model will easier to understand.

Good references on this topic are [1] and [2].

Why high-dimensional data? The volume of available data is growing exponentially fast nowadays. According to IBM three years ago, 10^{18} bytes of data were created every day in the world and 90% of data is less than two years old. Many modern data record simultaneously thousands up to millions of features on each objects or individuals. In many applications, data is high-dimensional such as with DNA, images, video, text, cookies (data about consumer preferences) or in astrophysics.

The curse of dimensionality

- High-dimensional spaces are vast and data points are isolated in their immensity.
- The accumulation of small errors in many different directions can produce a large global error.

- An event that is an accumulation of rare events may be not rare in high-dimensional spaces.

Example 1.1 *In high-dimensional spaces, no point in your data set will be close from a new input you want to predict. Assume that your input space is $\mathcal{X} = [0,1]^d$. The number of points needed to cover the space at a radius ε in L_2 norm is of order $1/\varepsilon^d$ which increases exponentially with the dimension. Therefore, in high dimension, it is unlikely to have a point in you data set that will be close to any new input.*

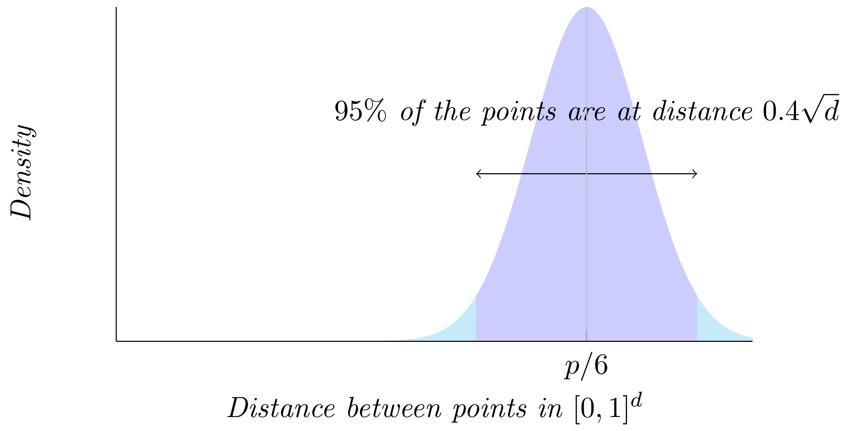
Example 1.2 *In high-dimensional spaces classical distances are often meaningless: all the points tends to be at similar distance from one another. Consider the following example to convince ourselves. Assume that X, X' follow uniform distribution on $[0,1]^d$. Then, the expected distance in square L_2-norm between X and X' is*

$$\mathbb{E}\big[\|X - X'\|^2\big] = \sum_{i=1}^d \mathbb{E}\big[(X_i - X_i')^2\big] = d\mathbb{E}\big[(X_1 - X_1')^2\big] = d\int_0^1\int_0^1 (x-x')^2 dx dx' = \frac{d}{6}.$$

Therefore, the average distance between the points increases with the dimension. Furthermore, the standard deviation of this square distance is

$$\sqrt{\mathrm{Var}\big(\|X - X'\|^2\big)} = \sqrt{\sum_{i=1}^d \mathrm{Var}\big((Xi - X_i')^2\big)} = \sqrt{p\mathrm{Var}\big((X_1 - X_1')^2\big)} = \frac{\sqrt{7d}}{6\sqrt{5}} \simeq 0.2\sqrt{d}\,.$$

Thus, if we plot the distribution of the square distance, we get something like:



Therefore, relatively to their distance, all points seem to be at similar distance from one another. The notion of nearest point distance vanishes. As a consequence, K-Nearest Neighbors gets poor performance in large dimension.

Example 1.3 *Let us consider another example in high-dimensional linear regression. We consider the ordinary least square estimator (OLS) for the linear model*

$$\widehat{\beta} \in \operatorname*{argmin}_{\beta \in \mathbb{R}^d} \big\|Y - X\beta\big\|^2 \qquad \textit{where} \quad Y_i = x_i^\top \beta^* + \varepsilon_i, \quad X = (x_1, \dots, x_n)^\top \in \mathbb{R}^{n\times p} \quad \textit{and} \quad \varepsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)\,.$$

If $rg(X) = d$ (which imposes $d \leq n$) then $\widehat{\beta} = (X \top X)^{-1} X^\top Y$ and as we saw in previous lecture the estimator satisfies

$$\mathbb{E}\big[\|\widehat{\beta} - \beta^*\|^2\big] = \mathrm{Tr}\big((X^\top X)^{-1}\big)\sigma^2\,.$$

In particular, in the very gentle case of an orthogonal design, we get $\mathbb{E}\big[\|\widehat{\beta} - \beta^*\|^2\big] = d\sigma^2$. Therefore, the variance of the estimator increases linearly with the dimension and the later gets unstable for high-dimensional data. Furthermore, OLS only works for $d \gg n$ because otherwise the matrix $X^\top X$ is not *invertible and using pseudo-inverse would lead to highly unstable estimator and over-fitting. One needs to regularize.*

The previous examples seem to show that the curse of dimensionality is unavoidable and we are doomed to poor estimators in large dimension. Hopefully, in many cases, data has an intrinsic low complexity (sparsity, low dimensional structure,...). This is the case of the data (for instance with images) or of the machine learning methods which is used (for instance Kernel regression).

What can we do with high-dimensional data? There are three classes of methods to deal with large dimensional input spaces:

- *Model selection*: we identify a subset of $s \ll d$ predictors that we believe to be related to the response. We then fit a model (for instance OLS) on the s variables only.
- *Regularization*: Ridge, Lasso,...
- *Dimension reduction*: the objective is to find a low-dimensional representation of the data. If we consider linear transformation, we may project the d predictors into a s-dimensional space with $s \ll d$. This is achieved by computing s different linear combination or projections of the variables. Then these projections are used as new features to fit a simple model for instance by least squares. Examples of such methods are PCA, PLS, etc. (see last lecture).

2 Model selection

The high level idea is to compare different statistical models corresponding to different possible hidden structure and select the best. This is theoretically very powerful, however the computational complexity is often prohibitive. Here, we will consider the example of the sparse linear model

$$Y = X\beta^* + \varepsilon\,, \quad Y = (y_1, \dots, y_n) \in \mathbb{R}^n, \quad X \in \mathbb{R}^{n \times d}, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 I_n)\,. \tag{1}$$

We consider $d \gg n$ but we assume that β^* has only $s < d$ non-zero coordinates.

If we knew in advance the non-zero coordinates of β^* say $m^* \subset \{1, \dots, d\}$, we could consider the simpler linear regression problem $y_i = \sum_{j \in m^*} \beta_j^* X_{i,j} + \varepsilon_i$ and use the estimator

$$\widehat{\beta}_m \in \underset{\beta \in \mathbb{R}^d,\ \beta_j = 0 \forall j \notin m}{\operatorname{argmin}} \big\|Y - X\beta\big\|^2. \tag{2}$$

More generally, this would work if we know that β belongs to some vector space of dimension $s < d$. We then get a risk which is scaling with s instead of d and the estimator has good statistical properties.

If we do not know m^* in advance, assuming the algorithmic complexity is not a problem, we can

1. consider a collection $\mathcal{M}$ of possible models $m \subset \{1, \dots, d\}$;
2. compute $\widehat{\beta}_m$ for each $m \in \mathcal{M}$ as defined in (2);

3. estimate β^* by the best estimator among the collection $\widehat{\beta}_m$.

A natural candidate for the best model is the minimizer of the empirical risk:

$$\widehat{\beta}_{\widehat{m}} \qquad \text{with} \quad \widehat{m} \in \operatorname*{argmin}_{m \in \mathcal{M}} \left\{ \big\|Y - X\widehat{\beta}_m\big\|^2 \right\}$$

The issue is that larger models $m \supset m'$ will always get smaller empirical risk because minimizing over a bigger space always leads to a smaller objective function. One needs to penalize models according to their complexity and choose the penalized estimator

$$\widehat{\beta}_{\widehat{m}} \qquad \text{with} \quad \widehat{m} \in \operatorname*{argmin}_{m \in \mathcal{M}} \left\{ \big\|Y - X\widehat{\beta}_m\big\|^2 + \operatorname{pen}(m) \right\} \tag{3}$$

There are several well known penalization criteria.

The Akaike Information Criterion (AIC). It defines the penalization

$$\operatorname{pen}(m) = 2|m|\sigma^2 \,.$$

The AIC criterion is motivated by the following lemma.

Lemma 1 *In least square linear regression with Gaussian model (see* (1)*),* $\|Y - \widehat{X}\beta_m\|^2 + (2|m| - n)\sigma^2$ *is an unbiased estimator of the risk* $R(\widehat{\beta}_m) := \mathbb{E}\big[\|X\beta^* - X\widehat{\beta}_m\|^2\big]$.

Proof We show that in least square regression the risk equals

$$R(\widehat{\beta}_m) := \mathbb{E}\big[\|X\beta^* - X\widehat{\beta}_m\|^2\big] = \mathbb{E}\big[\|Y - X\widehat{\beta}_m\|^2\big] + (2|m| - n)\sigma^2 \,.$$

Let us first give some useful notation an equalities. For each $m \subset \{1, \dots, d\}$, we define the sub-vector space $S_m := \{X\beta \in \mathbb{R}^n : \beta \in \mathbb{R}^d, \beta_j = 0 \ \forall j \notin m\}$ and $\Pi_{S_m} \in \mathbb{R}^{n \times n}$ the orthogonal projection matrix on S_m. Then, by definition of $\widehat{\beta}_m$, we have $X\widehat{\beta}_m = \Pi_{S_m} Y$ and we recall that $Y = X\beta^* + \varepsilon$. Furthermore, we will also use that:

$$\begin{aligned}\mathbb{E}\big[\|\Pi_{S_m}\varepsilon\|^2\big] = \mathbb{E}\big[\varepsilon^\top \Pi_{S_m}^\top \Pi_{S_m}\varepsilon\big] = \mathbb{E}\big[\varepsilon^\top \Pi_{S_m}\varepsilon\big] &= \mathbb{E}\big[\operatorname{Tr}(\varepsilon^\top \Pi_{S_m}\varepsilon)\big] \\ &= \mathbb{E}\big[\operatorname{Tr}(\Pi_{S_m}\varepsilon\varepsilon^\top)\big] = \sigma^2 \operatorname{Tr}(\Pi_{S_m}) = |m|\sigma^2 \,. \end{aligned} \tag{4}$$

Similarly, $\mathbb{E}\big[\|(I - \Pi_{S_m})\varepsilon\|^2\big] = (n - |m|)\sigma^2$. From the decomposition $Y - X\widehat{\beta}_m = (I - \Pi_{S_m})(X\beta^* + \varepsilon)$, we have

$$\begin{aligned}\mathbb{E}\big[\|Y - X\widehat{\beta}_m\|^2\big] &= \mathbb{E}\big[\|(I - \Pi_{S_m})X\beta^*\|^2 + \cancel{2\varepsilon^\top(I - \Pi_{S_m})X\beta^*} + \|(I - \Pi_{S_m})\varepsilon\|^2\big] \\ &= \|(I - \Pi_{S_m})X\beta^*\|^2 + (n - |m|)\sigma^2 \,. \\ &= \|(I - \Pi_{S_m})X\beta^*\|^2 + \mathbb{E}\big[\|\Pi_{S_m}\varepsilon\|^2\big] + (n - 2|m|)\sigma^2 \\ &= \mathbb{E}\big[\|(I - \Pi_{S_m})X\beta^* - \Pi_{S_m}\varepsilon\|^2\big] + (n - 2|m|)\sigma^2 \qquad \leftarrow \text{Pythagore's theorem} \\ &= \mathbb{E}\big[\|X\beta^* - \Pi_{S_m}(X\beta^* + \varepsilon)\|^2\big] + (n - 2|m|)\sigma^2 \\ &= \mathbb{E}\big[\|X\beta^* - X\widehat{\beta}_m\|^2\big] + (n - 2|m|)\sigma^2 \,. \end{aligned}$$

■

Prior-based penalization. Another popular penalization is to assign a prior weight π_m for each $m \in \mathcal{M}$, choose a regularization parameter $K > 1$ and select

$$\mathrm{pen}(m) = K\sigma^2\big(\sqrt{|m|} + \sqrt{2\log(1/\pi_m)}\big)^2 . \tag{5}$$

Theorem 1 (Thm. 2.2, [1]) *Under the model 1, there exists some constant $C_K > 1$ depending only on K such that the penalized estimator $\widehat{\beta}_{\widehat{m}}$ defined in* (3) *with penalty* (5) *satisfies*

$$R(\widehat{\beta}_{\widehat{m}}) := \mathbb{E}\big[\|X\beta^* - X\widehat{\beta}_{\widehat{m}}\|^2\big] \le C_K \min_{m\in\mathcal{M}} \Big\{\mathbb{E}\big[\|X\beta^* - X\widehat{\beta}_m\|^2\big] + \sigma^2 \log\frac{1}{\pi_m} + \sigma^2\Big\} .$$

A possible choice motivated by minimum description length (see lecture on PAC-Learning with infinite number of models) for the prior is $\log(1/\pi_m) \approx 2|m|\log d$, i.e., the number of bits needed to encode $m \subset \{1,\dots,d\}$. Remark that this choice of prior leads up to the $\log d$ to a similar criterion that for *AIC*. Yet, it is worth pointing out that the previous theorem is valid for general models $m \in \mathcal{M}$ (it is not restricted to the estimators (2)) and priors π_m. Other priors can promote different types of assumptions such as group sparsity.

Computational issues. The estimator (3) has very nice statistical properties even when $p \gg n$. However we need to compute $\widehat{\beta}_m$ for all models $m \in \mathcal{M}$. This is often prohibitive. We can understand it by rewriting it as an optimization problem of the form

$$\widehat{\beta}_{\widehat{m}} \in \operatorname*{argmin}_{\beta\in\mathbb{R}^d} \Big\{ \|Y - X\beta\|^2 + \lambda\|\beta\|_0 \Big\}, \tag{6}$$

which is non-convex because of the $\|\cdot\|_0$. The estimator of AIC corresponds to the choice $\lambda = 2\sigma^2$. In some cases, such as orthogonal design, we can approximate efficiently the solution or find an efficient implementation. However, this is not true in general. A approximate implementation which is sometimes used to solve (3) is the *forward-backward algorithm.* It consists in alternatively trying to add or remove variables in the model one by one. It quickly converges in practice, but there is no theoretical guarantees.

3 The Lasso

The high-level idea of the Lasso is to transform the non-convex optimization problem (6) into a convex problem. This is done by replacing the ℓ_0-norm $\|\beta\|_0 = \sum_{j=1}^d 1_{\beta_j\neq 0}$ with the ℓ_1-norm $\|\beta\|_1 = \sum_{j=1}^d |\beta|_j$ which is convex. We define the LASSO estimator

$$\widehat{\beta}_\lambda \in \operatorname*{argmin}_{\beta\in\mathbb{R}^d} \Big\{ \frac{1}{2}\|Y - X\beta\|^2 + \lambda\|\beta\|_1 \Big\} . \tag{LASSO}$$

The solution $\widehat{\beta}_\lambda$ may not be unique but the prediction $X\widehat{\beta}_\lambda$ is.

3.1 Geometric insight

By convex duality, the Lasso is also the solution of

$$\widehat{\beta}_\lambda \in \underset{\beta \in \mathbb{R}^d : \|\beta\|_1 \leq R_\lambda}{\operatorname{argmin}} \Big\{ \|Y - X\beta\|^2 \Big\} ,$$

for some radius $R_\lambda > 0$. The non-smoothness of the ℓ_1-norm puts some coefficients to zero. In Figure 1, we can see that because of the corners of the ℓ_1-ball, the solution $\widehat{\beta}_\lambda$ gets zero coefficients which is not the case when regularizing with the ℓ_2-norm (on the right).

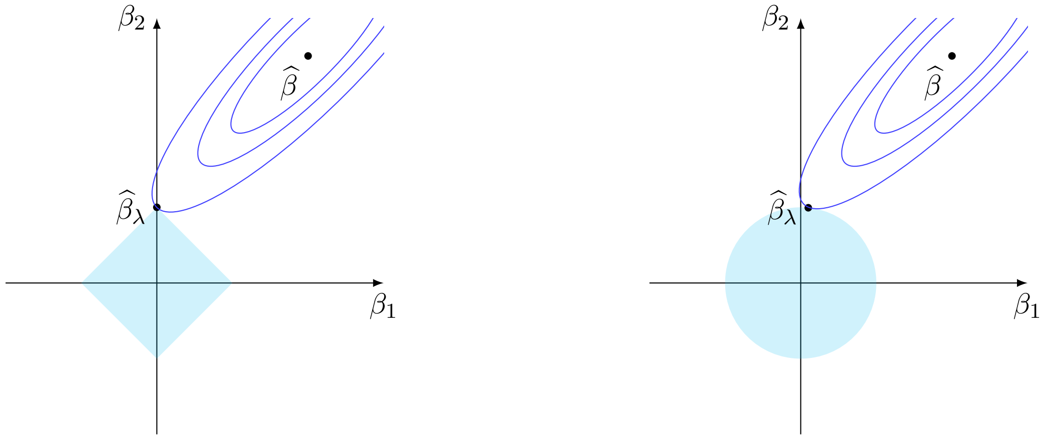


Figure 1: $\widehat{\beta}$ denotes the minimizer of the empirical risk and the blue lines denote level lines of the empirical risk. [left] Regularization with a ℓ_1-ball, [right] Regularization with a ℓ_2-ball.

3.2 What does the solution of the Lasso looks like?

To solve the problem of Lasso, if the objective function $\mathcal{L} : \beta \mapsto \|Y - X\beta\|^2 + \lambda\|\beta\|_1$ was differentiable, one would cancel the gradient. However, because of the ℓ_1-norm the later is not differentiable and one needs to generalize the notion of gradient to convex functions which are not necessarily differentiable. This is done with the following definition.

Definition 1 (Subdifferential) *A subgradient of a convex function $f : \mathbb{R}^d \to \mathbb{R}$ at a point $\beta_0 \in \mathbb{R}^d$ is a vector $z \in \mathbb{R}^d$ such that for any $\beta \in \mathbb{R}^d$ the convex inequality holds*

$$f(\beta) - f(\beta_0) \geq z^\top (\beta - \beta_0) .$$

The set of all subgradients of f at β_0 is denoted $\partial f(\beta_0)$ and is called the subdifferential of f at β_0.

The subdifferential of the ℓ_1-norm is

$$\partial \|\beta\|_1 = \Big\{ z \in [-1,1]^d \text{ s.t. for all } 1 \leq j \leq d \quad z_j = \operatorname{sign}(\beta_j) \text{ if } \beta_j \neq 0 \Big\}$$

and the subdifferential of the objective function of the Lasso is

$$\partial \mathcal{L}(\beta) = \Big\{ -X^\top (Y - X\beta) + \lambda z : \; z \in \partial \|\beta\|_1 \Big\} .$$

Any solution of the Lasso should cancel the subdifferential. Therefore, if $\widehat{\beta}_\lambda$ is a solution of the Lasso, it exists $\widehat{z} \in \partial\|\widehat{\beta}_\lambda\|_1$ (i.e., $\widehat{z}_j = \operatorname{sign}(\widehat{\beta}_\lambda(j))$ if $\widehat{\beta}_\lambda(j) \neq 0$ and $\widehat{z}_j \in [-1,1]$ otherwise) such that

$$-2X^\top(Y - X\beta) + \lambda\widehat{z} = 0 \quad \Rightarrow \quad X^\top X\widehat{\beta}_\lambda = X^\top Y - \lambda\widehat{z}\,. \tag{7}$$

Orthonormal design. If the gram matrix $X^\top X$ is general, it is not possible to solve the later in close form. To get some insights about the solution of the Lasso, let us assume the orthonormal setting $X^\top X = I_p$. Then, from (7), we get for all $j \in \{1, \dots, d\}$ such that $\widehat{\beta}_\lambda(j) \neq 0$

$$\widehat{\beta}_\lambda(j) = X_j^\top Y - \lambda \operatorname{sign}(\widehat{\beta}_\lambda(j))\,.$$

Therefore, $X_j^\top Y = \widehat{\beta}_\lambda(j) + \operatorname{sign}(\widehat{\beta}_\lambda(j))$ and $\widehat{\beta}_\lambda(j)$ have same sign and we obtain for all $1 \le j \le p$

$$\widehat{\beta}_\lambda(j) = \begin{cases} X_j^\top Y - \frac{\lambda}{2}\operatorname{sign}(X_j^\top Y) & \text{if } |X_j^\top Y| \ge \lambda \\ 0 & \text{if } |X_j^\top Y| \le \lambda \end{cases}$$

In the orthonormal setting, the Lasso performs thus a soft threshold of the coordinates of the OLS.

Statistical property of the Lasso estimator. For λ large enough $\lambda \simeq \sigma\sqrt{\log d}$, under some additional condition on the design (relaxed version of orthonormal design), it is possible to show that the Lasso does not assign any weight to coefficients that are not in m^*. If λ is properly chosen, it recovers exactly the coefficients of β^* and its risk is controlled with high probability as

$$R(\widehat{\beta}_\lambda) = \left\|X\beta^* - X\widehat{\beta}_\lambda\right\|^2 \le \inf_{\beta\in\mathbb{R}^d\setminus\{0\}} \left\{ \|X\beta - X\beta^*\|^2 + \square_X \lambda^2 \|\beta\|_0 \right\},$$

where $\lambda^2 \simeq \sigma^2 \log d$ and $\square_X$ is the compatibility constant depending on the design X. It can be bad for non-orthogonal design. We recover a similar result than the one obtained for model selection in Theorem 1 but with $\square_X$ and with an efficient procedure. It can be shown that it is not possible to avoid $\square_X$ for efficient (polynomial time) procedures.

3.3 Computing the Lasso estimator

Since this is the solution of a convex optimization problem, the solution of the Lasso can be obtained efficiently. There are three main algorithms used by the community.

Coordinate descent. (cf. practical session) the idea is to repeatedly minimize the objective function $\mathcal{L}(\beta)$ with respect to each coordinate. It converges thanks to the convexity of $\mathcal{L}$. As we saw in Equation (7), the solution of the Lasso satisfies

$$X^\top X\widehat{\beta}_\lambda = X^\top Y - \lambda\widehat{z}$$

where $\widehat{z} \in \partial\|\beta\|_1$. We saw that the solution equals $\widehat{\beta}_\lambda(j) = S_\lambda(X_j^\top Y)$ when $X^\top X = I_p$. This equation has however no closed-form solution in general. The idea of coordinate descent is to solve this equation only for one coordinate, fixing all the other coordinates.

Let $1 \leq i \leq n$ and fix coordinates $\beta_j \in \mathbb{R}$ for $j \neq i$. Solving the i-th coordinate optimisation problem given by

$$\min_{\beta_i} \mathcal{L}(\beta) = \min_{\beta_i \in \mathbb{R}} \left\{ \frac{1}{2} \big\|Y - X\beta\big\|^2 + \lambda \|\beta\|_1 \right\} ,$$

we get that the i-th partial sub-derivative of $\mathcal{L}$ should cancel, which gives similarly to previously

$$X_i^\top X\beta = X_i^\top Y - \lambda z_i \,,$$

where $z_i \in \partial|\beta_i|$. This can be rewritten as

$$X_i^\top X_i \beta_i + X_i^\top X_{-i}\beta_{-i} = X_i^\top Y - \lambda z_i \,,$$

where $X_{-i} \in \mathbb{R}^n \times (p-1)$ is the input matrix without column i and $\beta_{-i} \in \mathbb{R}^{p-1}$ is the fixed parameter vector without coordinate i.

Assume $\beta_i \neq 0$, then $z_i = \text{sign}(\beta_i)$ and

$$X_i^\top X_i \beta_i + \lambda \,\text{sign}(\beta_i) = X_i^\top (Y - X_{-i}\beta_{-i}) \,,$$

Since $X_i^\top X_i > 0$, we have $z_i = \text{sign}\big(X_i^\top (Y - X_i^\top X_{-i}\beta_{-i})\big)$ which implies

$$\beta_i = \frac{S_\lambda\big(X_i^\top (Y - X_i^\top X_{-i}\beta_{-i})\big)}{X_i^\top X_i} . \tag{8}$$

where S_λ is the soft-threshold function:

$$S_\lambda(x) = \begin{cases} 0 & \text{if} \quad |x| \leq \lambda \\ x - \lambda \,\text{sign}(x) & \text{otherwise} \end{cases} .$$

The algorithm of coordinate descent consists in sequentially repeating the update (8) for $i = 1, \dots, p, 1 \dots, p, \dots$ minmizing the objective function with respect to each coordinate at a time.

Fista. (fast iterative shrinkage thresholding algorithmn) It uses the explicit formula in the orthogonal design setting for computing recursively an approximation of the solution

Homotopy methods. The insight of the algorithm comes from equation (7): $X^\top X \widehat{\beta}_\lambda = X^\top Y - \frac{\lambda}{2}\widehat{z}$. We then consider the function $\lambda \mapsto \widehat{\beta}_\lambda$. For non-zero coefficients, $\widehat{z}_j = \text{sign}(\widehat{\beta}_\lambda(j))$ and is constant while $\lambda \mapsto \widehat{\beta}_\lambda(j)$ does not change sign. Therefore, the function $\lambda \mapsto \widehat{\beta}_\lambda$ is piecewise linear in λ with a change when for some coordinate $\widehat{\beta}_\lambda(j)$ changes sign. Homotopy methods compute the sequence $\{\widehat{\beta}_{\lambda_1}, \widehat{\beta}_{\lambda_2}, \dots\}$ of the Lasso estimator corresponding to the break points of the path $\lambda \mapsto \widehat{\beta}_\lambda$. At each break point, the model $m_\lambda = \{i \in \{1, \dots, d\} : \widehat{\beta}_\lambda(i) \neq 0\}$ is updated and we solve the linear equation

$$X_{m_\lambda}^\top X_{m_\lambda} \widehat{\beta}_\lambda(m_\lambda) = X_{m_\lambda}^\top Y - \lambda \,\text{sign}(\widehat{\beta}_\lambda(m)) \,,$$

until the next break point. This algorithm is slower than the other two algorithms but it provides the full regularization path $\lambda \mapsto \widehat{\beta}_\lambda$ (see Figure 2).

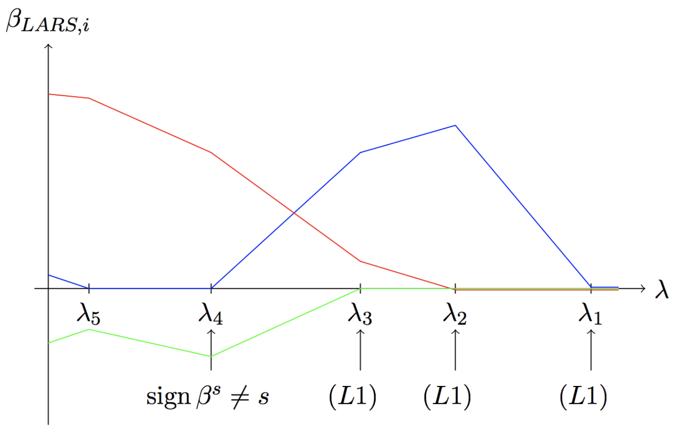


Figure 2: Lasso regularization path computed with LARS

3.4 Final remarks and variants

Removing the bias of the Lasso. The Lasso estimator $\widehat{\beta}_\lambda$ is biased. Often one might want to remove the bias for instance by first computing $\widehat{\beta}_\lambda$ to select to good model $\widehat{m}_\lambda$ and then solve the OLS or Ridge on the model $\widehat{m}_\lambda$ only.

No penalization of the intercept. In practice, the intercept is often no penalized and the Lasso solves

$$\widehat{\beta}_\lambda \in \operatorname*{argmin}_{\beta \in \mathbb{R}^d} \left\{ \sum_{i=1}^n (Y_i - \beta_0 - \beta^\top X_i)^2 + \lambda \|\beta\|_1 \right\}.$$

Group Lasso. It is an extension when coordinates are sparse by groups. In other words, we have some groups $G_k \subset \{1, \dots, d\}$ and we assume that all coordinates β_i for $i \in G_k$ are either all zero or all non-zero.

Elastic net. It is a mix of ℓ_1 and ℓ_2 regularization

$$\widehat{\beta} \in \operatorname*{argmin}_{\beta \in \mathbb{R}^d} \left\{ \|Y - X\beta\|^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2 \right\}.$$

It also selects variables thanks to sharp corners and it is heavily used in practice.

Calibration of λ. It is a crucial point in practice. A common solution is to perform K-fold cross validation. There are a few other techniques such as the slopes heuristic.

References

[1] Christophe Giraud. *Introduction to high-dimensional statistics*. Chapman and Hall/CRC, 2014.

[2] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA:, 2001.