Machine Learning for High-Dimensional Data

Sparse Regression

Charlotte Laclau

General Setting: Linear Regression

We consider the following regression model

$$\boxed{Y = X\theta^* + \epsilon}$$

- $Y \in \mathbb{R}^n$ is the target vector
- $X \in \mathbb{R}^{n \times d}$ is a matrix of predictors
- $\theta^* \in \mathbb{R}^d$ are the ground truth parameters of the model
- $\epsilon \in \mathbb{R}^n$ is some noise
- We assume that X and Y are normalized

General Setting: Linear Regression

$$X = [\mathbf{x}_1, \ldots, \mathbf{x}_d] = \begin{pmatrix} x_{1,1} & \cdots & x_{1,d} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,d} \end{pmatrix} \in \mathbb{R}^{n \times d}, \boldsymbol{\theta}^\star \in \mathbb{R}^d$$

$$\mathbf{y} = [y_1, \ldots, y_n]$$

$$\boldsymbol{\theta}^\star = [\theta_1, \ldots, \theta_d]$$

Objective

For a new observation $x^{(n+1)}$ predict the associated $y^{(n+1)}$

$$\Leftrightarrow$$

Learn $\hat{\theta}$ such that $\hat{y}^{(n+1)} = x^{(n+1)\,T}\hat{\theta} \approx y^{(n+1)}$

Linear Regression: an example

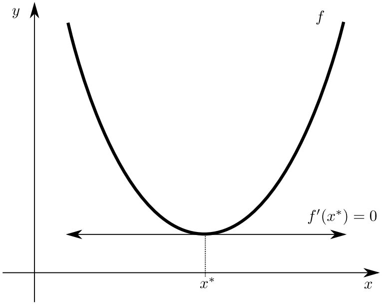
Let's consider the following problem

- We have a population of $n = 120$ patients
- For each patient i we have a list of $d = 6$ physiological index
  - Age, in years
  - Weight, in kg
  - Body surface area (BSA), in m²
  - Duration of hypertension (Dur), in years
  - Basal Pulse (Pulse), in beats per minute
  - Stress index (Stress)
- For each patient we want to **predict** the Blood pressure (BP) expressed in mm Hg

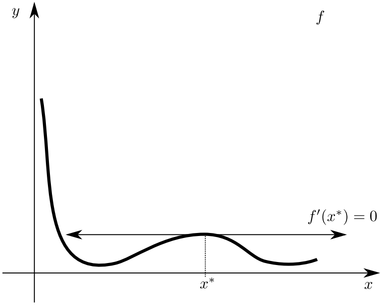
We assume that BP (Y) is the result of a linear combination of each of the index (X) and our objective is to learn their weight θ.

Solving Linear Regression

Theorem: Fermat's rule
If f is differentiable, then at a local minimum x^* the gradient of f vanishes at x^*, i.e. $\nabla f(x^*) = 0$.



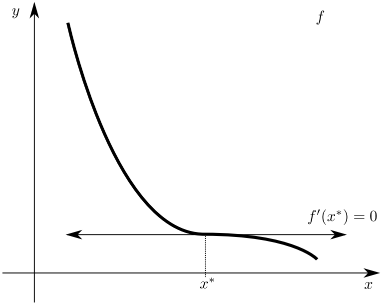
Theorem: Fermat's rule
If f is differentiable, then at a local minimum x^* the gradient of f vanishes at x^*, i.e. $\nabla f(x^*) = 0$.



Rem: sufficient condition when f is convex!

Solving Linear Regression

Theorem: Fermat's rule
If f is differentiable, then at a local minimum x^* the gradient of f vanishes at x^*, i.e. $\nabla f(x^*) = 0$.



Rem: sufficient condition when f is convex!

Back to least squares: 1D case

$$\boxed{\widehat{\boldsymbol{\theta}} = (\hat{\theta}_0, \hat{\theta}_1) \in \operatorname{argmin}_{(\theta_0, \theta_1) \in \mathbb{R}^2} \frac{1}{2} \sum_{i=1}^{n} (y_i - \theta_0 - \theta_1 x_i)^2}$$

For least squares, minimize the function of two variables:

$$f(\theta_0, \theta_1) = f(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^{n} (y_i - \theta_0 - \theta_1 x_i)^2$$

First order condition / Fermat's rule:

$$\begin{cases} \frac{\partial f}{\partial \theta_0}(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n} (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i) = 0 \\ \frac{\partial f}{\partial \theta_1}(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n} (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i) x_i = 0 \end{cases}$$

Calculus continued

$$\text{Usual mean notation: } \overline{x}_n = \frac{1}{n}\sum_{i=1}^{n} x_i \text{ and } \overline{y}_n = \frac{1}{n}\sum_{i=1}^{n} y_i$$

With that, Fermat's rule states (dividing by n) :

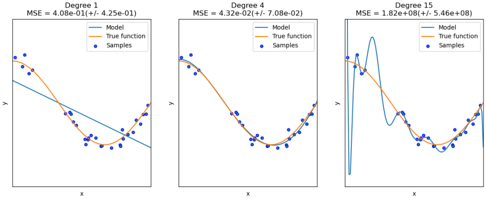
$$\begin{cases} \frac{\partial f}{\partial \theta_0}(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}(y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i) = 0 \\ \frac{\partial f}{\partial \theta_1}(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}(y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i) x_i = 0 \end{cases}$$

$$\Leftrightarrow$$

$$\begin{cases} \hat{\theta}_0 = \overline{y}_n - \hat{\theta}_1 \overline{x}_n & \text{(CNO1)} \\ \hat{\theta}_1 = \dfrac{\sum_{i=1}^{n}(x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2} & \text{(CNO2)} \end{cases}$$

Penalized Linear Regression: Initial Motivation

- Prevent overfitting: sacrifies bias to reduce the variance



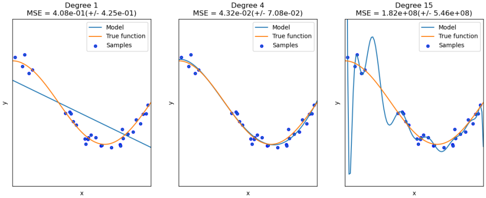
1

- Numerical (in)stability - colinearity

Intermediate objective - shrink the values of $\boldsymbol{\theta}$

Penalized Linear Regression: Initial Motivation

- Prevent overfitting: sacrifies bias to reduce the variance



[1]

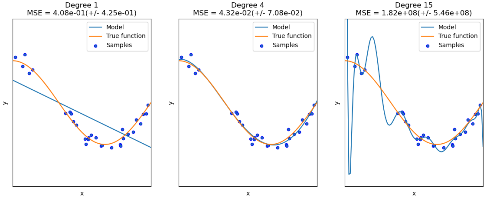
- Numerical (in)stability - colinearity

Intermediate objective - shrink the values of $\boldsymbol{\theta}$

$$\boxed{\hat{\boldsymbol{\theta}}_{\lambda}^{\mathrm{rdg}} = \mathrm{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^p} \left(\underbrace{\|Y - X\boldsymbol{\theta}\|_2^2}_{\text{data fitting}} + \underbrace{\lambda \phi(\boldsymbol{\theta})}_{\text{regularisation}} \right)}$$

Penalized Linear Regression: Initial Motivation

- Prevent overfitting: sacrifies bias to reduce the variance



[1]

- Numerical (in)stability - colinearity

Intermediate objective - shrink the values of $\boldsymbol{\theta}$

$$\boxed{\hat{\boldsymbol{\theta}}_{\lambda}^{\mathrm{rdg}} = \operatorname{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^p} \left(\underbrace{\|Y - X\boldsymbol{\theta}\|_2^2}_{\textbf{data fitting}} + \underbrace{\lambda \phi(\boldsymbol{\theta})}_{\textbf{regularisation}} \right)}$$

How to choose ϕ?

Penalizing the norm of θ

- $\phi(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2^2$

Constraint interpretation

A "Lagrangian" formulation is as follows:

$$\boxed{\operatorname{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^p} \left(\underbrace{\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2}_{\textbf{data fitting}} + \underbrace{n\lambda\|\boldsymbol{\theta}\|_2^2}_{\textbf{regularization}} \right)}$$

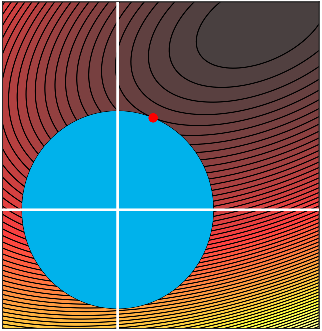
has for a certain $T > 0$ the same solution as:

$$\begin{cases} \operatorname{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^p} \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 \\ \text{s.t. } \|\boldsymbol{\theta}\|_2^2 \leq T \end{cases}$$

<u>Rem</u>: the link $T \leftrightarrow \lambda$ is not explicit!

- If $T \to 0$ we recover the null vector: $0 \in \mathbb{R}^p$
- If $T \to \infty$ we recover $\hat{\boldsymbol{\theta}}^{\mathrm{OLS}}$ (un-constrained)

Level lines and and constraints set



Optimization under ℓ_2 constraints

Solving Ridge Regression

$$\boxed{\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = \operatorname{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^p} \left(\underbrace{\|Y - X\boldsymbol{\theta}\|_2^2}_{\textbf{data fitting}} + \underbrace{\lambda\|\boldsymbol{\theta}\|_2^2}_{\textbf{regularisation}} \right)}$$

- Computation of the solution using the necessary condition of optimality (Fermat rule), we have

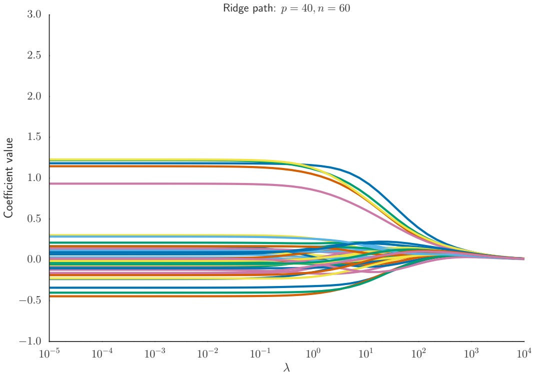
$$f(\boldsymbol{\theta}) = \frac{\|Y - X\boldsymbol{\theta}\|_2^2}{2} + \frac{\lambda\|\boldsymbol{\theta}\|_2^2}{2}.$$

$$\begin{aligned} \text{CNO} \ : \nabla f(\boldsymbol{\theta}) &= X^\top(X\boldsymbol{\theta} - Y) + \lambda\boldsymbol{\theta} = 0 \\ &\Leftrightarrow (X^\top X + \lambda \operatorname{Id}_p)\boldsymbol{\theta} = X^\top Y \end{aligned}$$

- We recover the regularized normal equation.

Choosing λ

```
n_features = 50; n_samples = 50
X = np.random.randn(n_samples, n_features)
theta_true = np.zeros([n_features, ])
theta_true[0:5] = 2.
y_true = np.dot(X, theta_true)
y = y_true + 1. * np.random.rand(n_samples,)
```



Additional Hypothesis: Assuming Sparsity

Estimators $\hat{\boldsymbol{\theta}}$ with many zero coefficients are useful:

- for interpretation
- for computational efficiency if d is huge

Underlying idea: **variable selection**

Rem: also useful if $\boldsymbol{\theta}^\star$ has few non-zero coefficients

Rem: quadratic penalisation shrink the values of θ

Additional Hypothesis: Assuming Sparsity

Estimators $\hat{\boldsymbol{\theta}}$ with many zero coefficients are useful:

- for interpretation
- for computational efficiency if d is huge

Underlying idea: **variable selection**

Rem: also useful if $\boldsymbol{\theta}^\star$ has few non-zero coefficients

Rem: quadratic penalisation shrink the values of θ
$\rightarrow$ Can we go further?

Variable selection overview

- **Screening**: remove the $\mathbf{x}_j$'s whose correlation with $\mathbf{y}$ is weak
  - pros: fast (+++), i.e. one pass over data, intuitive (+++)
  - cons: neglect variables interactions $\mathbf{x}_j$, weak theory (- - -)
- **Greedy** methods aka stagewise / stepwise
  - pros: fast (++), intuitive (++)
  - cons: propagates wrong selection forward; weak theory (-)
- Sparsity enforcing **penalized** methods (*e.g.* Lasso)
  - pros: better theory for convex cases (++)
  - cons: can be still slow (-)

The ℓ_0 pseudo-norm

Definition

The **support** of $\boldsymbol{\theta} \in \mathbb{R}^d$ is the set of indexes of non-zero coordinates:

$$\mathrm{Supp}(\boldsymbol{\theta}) = \{j \in [\![1, d]\!], \theta_j \neq 0\}$$

The ℓ_0 **pseudo-norm** of a $\boldsymbol{\theta} \in \mathbb{R}^d$ is the number of non-zero coordinates:

$$\|\boldsymbol{\theta}\|_0 = \mathrm{card}\{j \in [\![1, d]\!], \theta_j \neq 0\}$$

Rem: $\|\cdot\|_0$ is not a norm, $\forall t \in \mathbb{R}^*, \|t\boldsymbol{\theta}\|_0 = \|\boldsymbol{\theta}\|_0$

Rem: $\|\cdot\|_0$ it is not even convex, $\boldsymbol{\theta}_1 = (1, 0, 1, \ldots, 0)$
$\boldsymbol{\theta}_2 = (0, 1, 1, \ldots, 0)$ and $3 = \|\frac{\boldsymbol{\theta}_1 + \boldsymbol{\theta}_2}{2}\|_0 \geq \frac{\|\boldsymbol{\theta}_1\|_0 + \|\boldsymbol{\theta}_2\|_0}{2} = 2$

The ℓ_0 penalty

Sparsity enforcing penalty: use ℓ_0 as a penalty (or regularization)

$$\boxed{\hat{\boldsymbol{\theta}}_\lambda = \operatorname{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^d} \left(\underbrace{\frac{1}{2}\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2}_{\text{data fitting}} + \underbrace{\lambda\|\boldsymbol{\theta}\|_0}_{\text{regularization}} \right)}$$

Combinatorial problem!!!

Exact solution: require considering all sub-models, i.e. computing OLS for all possible support; meaning one might need 2^d least squares computation!

$d = 10$ possible: $\approx 10^3$ least squares

$d = 30$ impossible: $\approx 10^{10}$ least squares

Rem: problem "NP-hard", can be solved for small problems by mixed integer programming.

The ℓ_1 penalty as an alternative

What about $\phi = \|\boldsymbol{\theta}\|_1$?

Lasso: penalty point of view

Lasso: *Least Absolute Shrinkage and Selection Operator*

$$\boxed{\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{Lasso}} = \mathrm{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^p} \left(\underbrace{\frac{1}{2}\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2}_{\textbf{data fitting}} + \underbrace{\lambda\|\boldsymbol{\theta}\|_1}_{\textbf{regularization}} \right)}$$

où $\|\boldsymbol{\theta}\|_1 = \sum_{j=1}^{p} |\theta_j|$ sum of absolute values of the coefficients)

- We recover the limiting cases:

$$\lim_{\lambda\to 0} \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{Lasso}} = \hat{\boldsymbol{\theta}}^{\mathrm{OLS}}$$
$$\lim_{\lambda\to +\infty} \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{Lasso}} = 0 \in \mathbb{R}^p$$

Beware: the Lasso estimator is not always **unique** for a fixed λ (consider cases with two equals columns in X)

Constraint point of view

The following problem:

$$\boxed{\hat{\boldsymbol{\theta}}_{\lambda}^{\mathrm{Lasso}} = \mathrm{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^p} \left(\underbrace{\frac{1}{2}\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2}_{\textbf{data fitting}} + \underbrace{\lambda\|\boldsymbol{\theta}\|_1}_{\textbf{regularization}} \right)}$$

shares the same solutions as the constrained formulation:

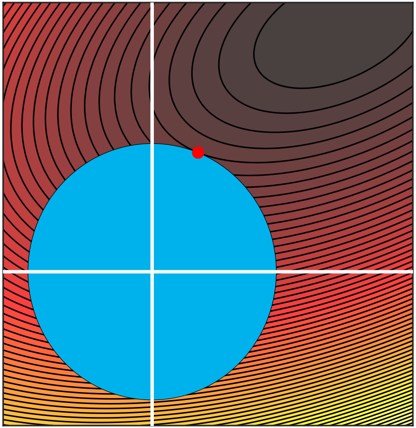
$$\begin{cases} \mathrm{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^p} \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 \\ \text{ s.t. } \|\boldsymbol{\theta}\|_1 \leq T \end{cases}$$

for some $T > 0$.

Rem: the link $T \leftrightarrow \lambda$ is not explicit

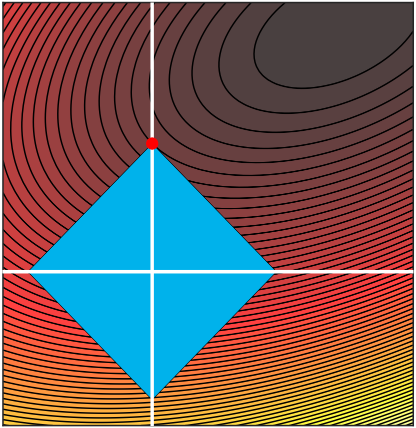
- If $T \to 0$ one recovers the null vector: $0 \in \mathbb{R}^d$
- If $T \to \infty$ one recovers $\hat{\boldsymbol{\theta}}^{\mathrm{OLS}}$ (unconstrained)

Zeroing coefficients



Optimization under ℓ_2 constraint : non sparse solution

Zeroing coefficients



Optimization under ℓ_1 constraint : sparse solution

Solving Lasso - Intuition

The function $\mathcal{L} : \theta = \frac{1}{2}\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_1$ is convex but **non differentiable**.

Idea: to solve the lasso we restrict ourselves to computing the subdifferential of the function $|.|$

Rem: There exist many options to do this: projected gradient, shooting method, subgradient descent, **coordinate descent** etc.

Sub-gradients / sub-differential

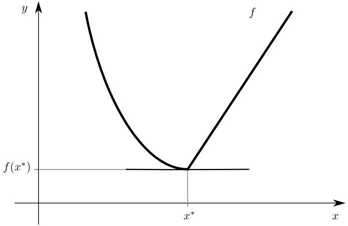
Definitions

For a convex function $f : \mathbb{R}^n \to \mathbb{R}$, $u \in \mathbb{R}^n$ is a **sub-gradient** of f at x^*, if for any $x \in \mathbb{R}^n$,

$$f(x) \geq f(x^*) + \langle u, x - x^* \rangle$$

The **sub-differential** is the set
$\partial f(x^*) = \{u \in \mathbb{R}^n : \forall x \in \mathbb{R}^n, f(x) \geq f(x^*) + \langle u, x - x^* \rangle\}$.

Rem: if the sub-gradient is unique, one recovers the standard gradient



Sub-gradients / sub-differential

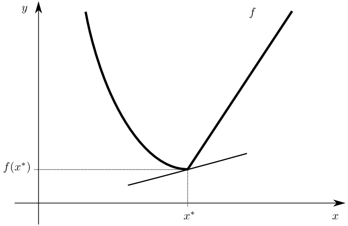
Definitions

For a convex function $f : \mathbb{R}^n \to \mathbb{R}$, $u \in \mathbb{R}^n$ is a **sub-gradient** of f at x^*, if for any $x \in \mathbb{R}^n$,

$$f(x) \geq f(x^*) + \langle u, x - x^* \rangle$$

The **sub-differential** is the set
$\partial f(x^*) = \{u \in \mathbb{R}^n : \forall x \in \mathbb{R}^n, f(x) \geq f(x^*) + \langle u, x - x^* \rangle\}$.

Rem: if the sub-gradient is unique, one recovers the standard gradient



Sub-gradients / sub-differential

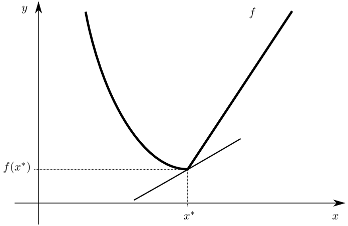
Definitions

For a convex function $f : \mathbb{R}^n \to \mathbb{R}$, $u \in \mathbb{R}^n$ is a **sub-gradient** of f at x^*, if for any $x \in \mathbb{R}^n$,

$$f(x) \geq f(x^*) + \langle u, x - x^* \rangle$$

The **sub-differential** is the set
$\partial f(x^*) = \{u \in \mathbb{R}^n : \forall x \in \mathbb{R}^n, f(x) \geq f(x^*) + \langle u, x - x^* \rangle\}$.

Rem: if the sub-gradient is unique, one recovers the standard gradient



Sub-gradients / sub-differential

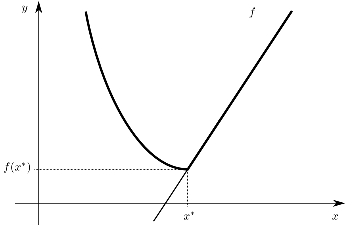
Definitions

For a convex function $f : \mathbb{R}^n \to \mathbb{R}$, $u \in \mathbb{R}^n$ is a **sub-gradient** of f at x^*, if for any $x \in \mathbb{R}^n$,

$$f(x) \geq f(x^*) + \langle u, x - x^* \rangle$$

The **sub-differential** is the set
$\partial f(x^*) = \{u \in \mathbb{R}^n : \forall x \in \mathbb{R}^n, f(x) \geq f(x^*) + \langle u, x - x^* \rangle\}$.

Rem: if the sub-gradient is unique, one recovers the standard gradient



Fermat's Rule

Theorem
A point x^* is a minimum of a convex function $f : \mathbb{R}^n \to \mathbb{R}$ if and only if $0 \in \partial f(x^*)$

Proof: use the sub-gradient definition:

- 0 is a sub-gradient of f at x^* if and only if $\forall x \in \mathbb{R}^n, f(x) \geq f(x^*) + \langle 0, x - x^* \rangle$

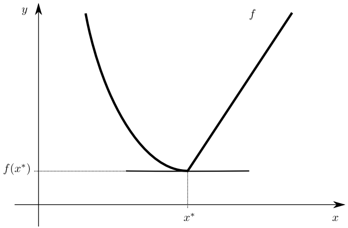
Fermat's Rule

Theorem
A point x^* is a minimum of a convex function $f : \mathbb{R}^n \to \mathbb{R}$ if and only if $0 \in \partial f(x^*)$

Proof: use the sub-gradient definition:

- 0 is a sub-gradient of f at x^* if and only if $\forall x \in \mathbb{R}^n, f(x) \geq f(x^*) + \langle 0, x - x^* \rangle$

Rem:Visually, it corresponds to a horizontal tangent



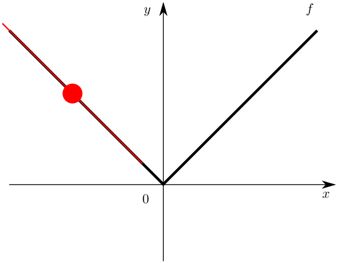
Absolute value sub-differential

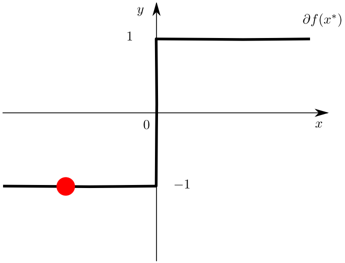
Function (abs):

$$f : \begin{cases} \mathbb{R} & \to \mathbb{R} \\ x & \mapsto |x| \end{cases}$$

Sub-differential (sign)

$$\partial f(x^*) = \begin{cases} \{-1\} & \text{if } x^* \in]-\infty, 0[\\ \{1\} & \text{if } x^* \in]0, \infty[\\ [-1, 1] & \text{if } x^* = 0 \end{cases}$$

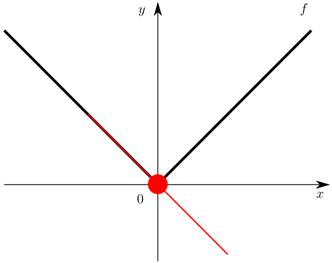




Absolute value sub-differential

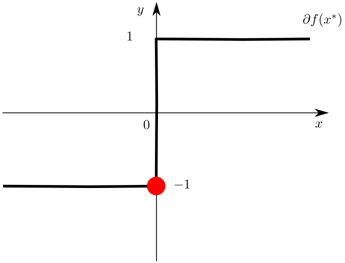
Function (abs):

$$f : \begin{cases} \mathbb{R} & \to \mathbb{R} \\ x & \mapsto |x| \end{cases}$$



Sub-differential (sign)

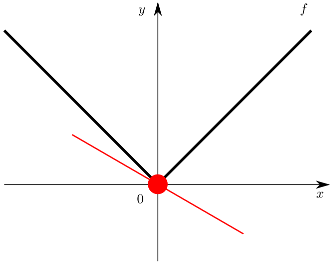
$$\partial f(x^*) = \begin{cases} \{-1\} & \text{if } x^* \in]-\infty, 0[\\ \{1\} & \text{if } x^* \in]0, \infty[\\ [-1, 1] & \text{if } x^* = 0 \end{cases}$$



Absolute value sub-differential

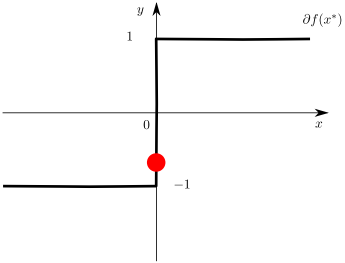
Function (abs):

$$f: \begin{cases} \mathbb{R} & \to \mathbb{R} \\ x & \mapsto |x| \end{cases}$$



Sub-differential (sign)

$$\partial f(x^*) = \begin{cases} \{-1\} & \text{if } x^* \in]-\infty, 0[\\ \{1\} & \text{if } x^* \in]0, \infty[\\ [-1, 1] & \text{if } x^* = 0 \end{cases}$$



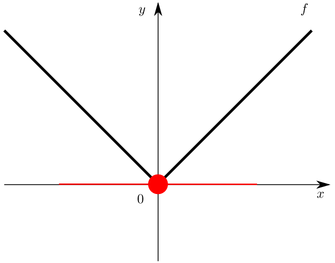
Absolute value sub-differential

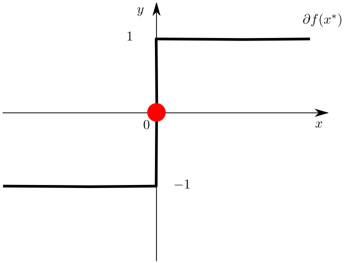
Function (abs):

$$f : \begin{cases} \mathbb{R} & \to \mathbb{R} \\ x & \mapsto |x| \end{cases}$$

Sub-differential (sign)

$$\partial f(x^*) = \begin{cases} \{-1\} & \text{if } x^* \in]-\infty, 0[\\ \{1\} & \text{if } x^* \in]0, \infty[\\ [-1, 1] & \text{if } x^* = 0 \end{cases}$$

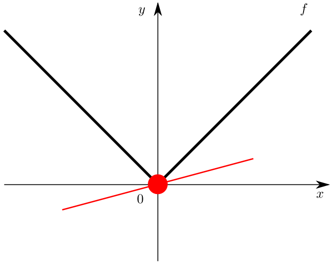




Absolute value sub-differential

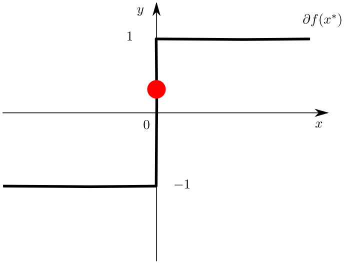
Function (abs):

$$f : \begin{cases} \mathbb{R} & \to \mathbb{R} \\ x & \mapsto |x| \end{cases}$$



Sub-differential (sign)

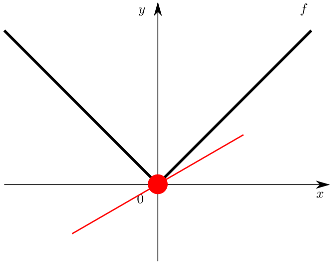
$$\partial f(x^*) = \begin{cases} \{-1\} & \text{if } x^* \in]-\infty, 0[\\ \{1\} & \text{if } x^* \in]0, \infty[\\ [-1, 1] & \text{if } x^* = 0 \end{cases}$$



Absolute value sub-differential

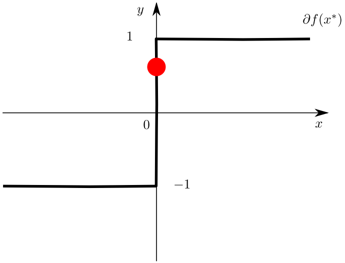
Function (abs):

$$f : \begin{cases} \mathbb{R} & \rightarrow \mathbb{R} \\ x & \mapsto |x| \end{cases}$$



Sub-differential (sign)

$$\partial f(x^*) = \begin{cases} \{-1\} & \text{if } x^* \in]-\infty, 0[\\ \{1\} & \text{if } x^* \in]0, \infty[\\ [-1, 1] & \text{if } x^* = 0 \end{cases}$$



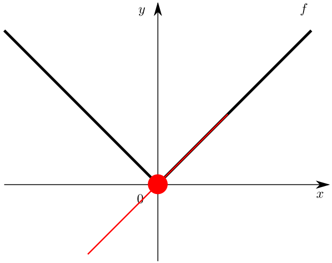
Absolute value sub-differential

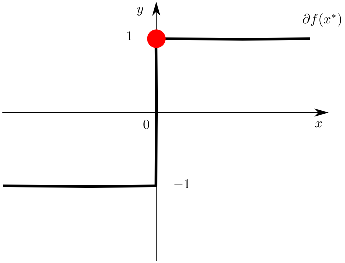
Function (abs):

$$f : \begin{cases} \mathbb{R} & \to \mathbb{R} \\ x & \mapsto |x| \end{cases}$$

Sub-differential (sign)

$$\partial f(x^*) = \begin{cases} \{-1\} & \text{if } x^* \in]-\infty, 0[\\ \{1\} & \text{if } x^* \in]0, \infty[\\ [-1, 1] & \text{if } x^* = 0 \end{cases}$$





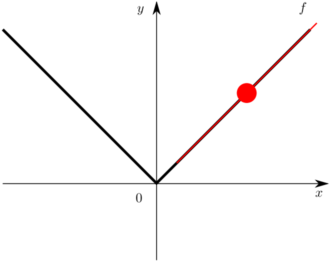
Absolute value sub-differential

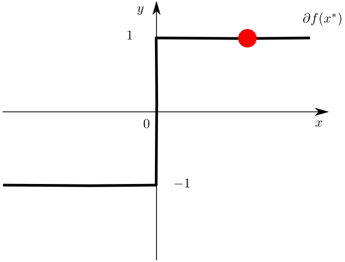
Function (abs):

$$f : \begin{cases} \mathbb{R} & \to \mathbb{R} \\ x & \mapsto |x| \end{cases}$$

Sub-differential (sign)

$$\partial f(x^*) = \begin{cases} \{-1\} & \text{if } x^* \in]-\infty, 0[\\ \{1\} & \text{if } x^* \in]0, \infty[\\ [-1, 1] & \text{if } x^* = 0 \end{cases}$$

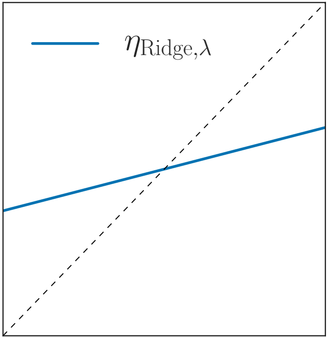




1D Regularization: Ridge

Solve: $\eta_\lambda(z) = \text{argmin}_{x\in\mathbb{R}}\, x \mapsto \frac{1}{2}(z-x)^2 + \frac{\lambda}{2}x^2$

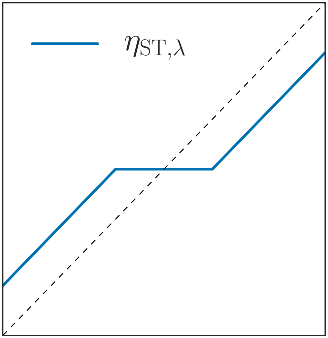
$$\eta_\lambda(z) = \frac{z}{1+\lambda}$$



1D Regularization: Lasso

Solve: $\eta_\lambda(z) = \mathrm{argmin}_{x \in \mathbb{R}}\, x \mapsto \dfrac{1}{2}(z-x)^2 + \lambda |x|$

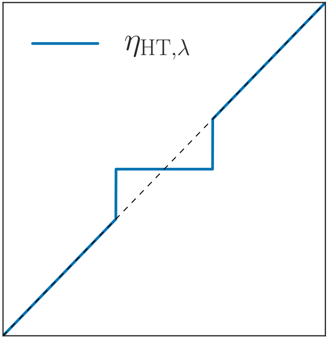
$$\eta_\lambda(z) = \mathrm{sign}(z)(|z| - \lambda)_+ \text{ (Exercise)}$$



1D Regularization: ℓ_0

Solve: $\eta_\lambda(z) = \operatorname{argmin}_{x\in\mathbb{R}}\, x \mapsto \dfrac{1}{2}(z-x)^2 + \lambda\,\mathrm{Id}_{x\neq 0}$

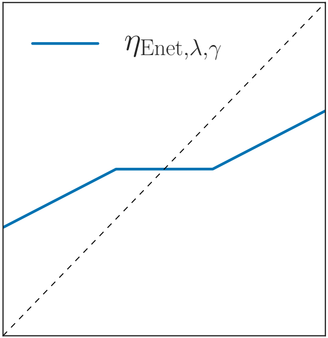
$$\eta_\lambda(z) = z\,\mathrm{Id}_{|z|\geq\sqrt{2\lambda}}$$



1D Regularization: Elastic-Net

Solve: $\eta_\lambda(z) = \text{argmin}_{x \in \mathbb{R}}\, x \mapsto \frac{1}{2}(z-x)^2 + \lambda(\gamma|x| + (1-\gamma)\frac{x^2}{2})$

$$\eta_\lambda(z) = \text{Exercise}$$



Soft thresholding: closed form solution

$$\eta_{\text{Lasso},\lambda}(z) = \begin{cases} z + \lambda & \text{if } \ z \ < -\lambda \\ 0 & \text{if } |z| \leq \lambda \\ z - \lambda & \text{if } \ z \ > \lambda \end{cases}$$

To do: use sub-gradients to prove the previous result

Additional Properties needed for the proof

- $\partial(f(x) + g(x)) = \partial f(x) + \partial g(x)$
- For $\alpha \geq 0, \quad \partial(\alpha f)(x) = \alpha \partial f(x)$

Proof

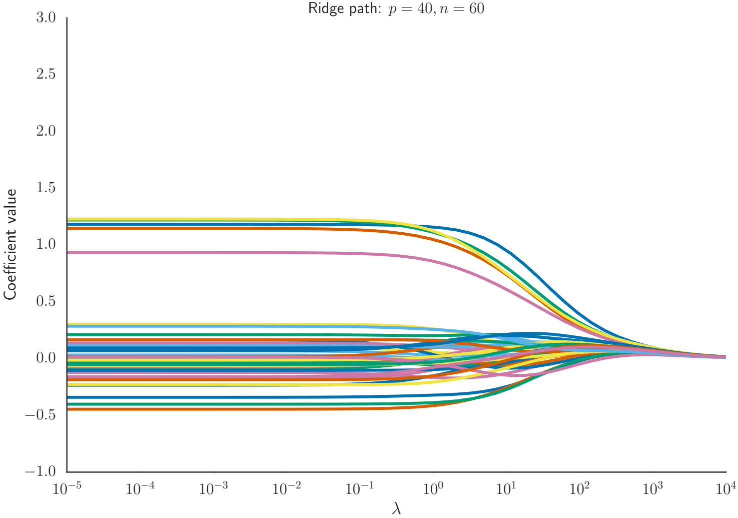
Coordinate Descent Algorithm

Numerical example on simulated data

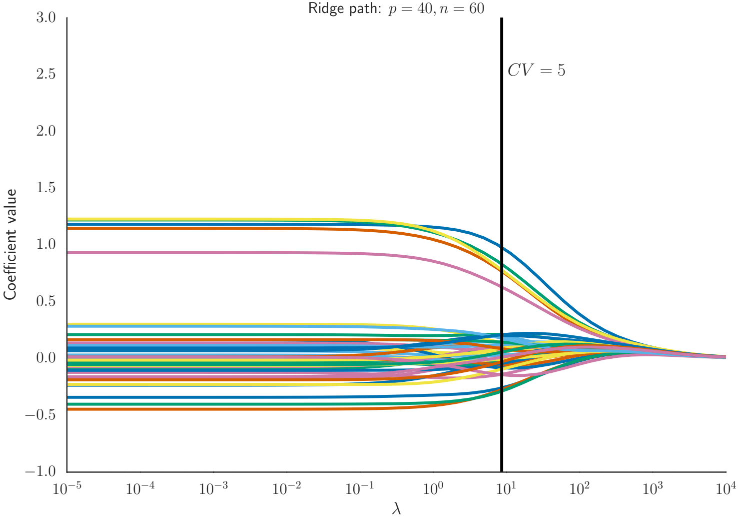
- $\boldsymbol{\theta}^\star = (1, 1, 1, 1, 1, 0, \ldots, 0) \in \mathbb{R}^d$ (5 non-zero coefficients)
- $X \in \mathbb{R}^{n \times d}$ has columns drawn according to a Gaussian distribution
- $y = X\boldsymbol{\theta}^\star + \boldsymbol{\epsilon} \in \mathbb{R}^n$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 \operatorname{Id}_n)$
- We use a grid of 50 λ values

For this example : $n = 60, d = 40, \sigma = 1$

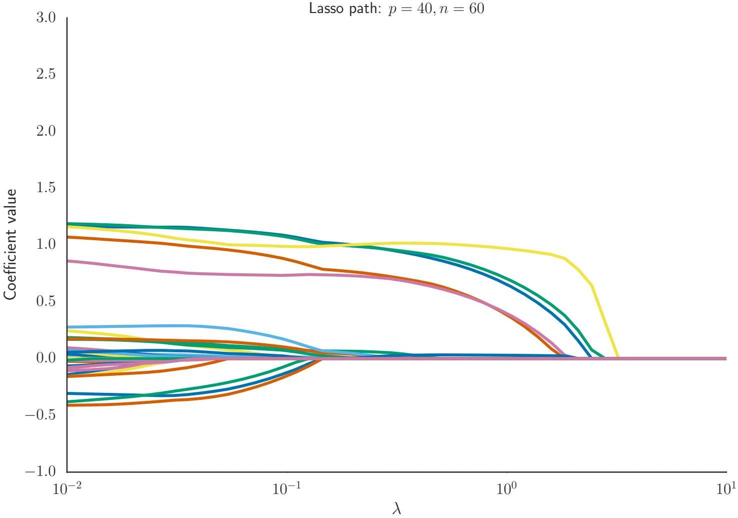
Lasso vs Ridge



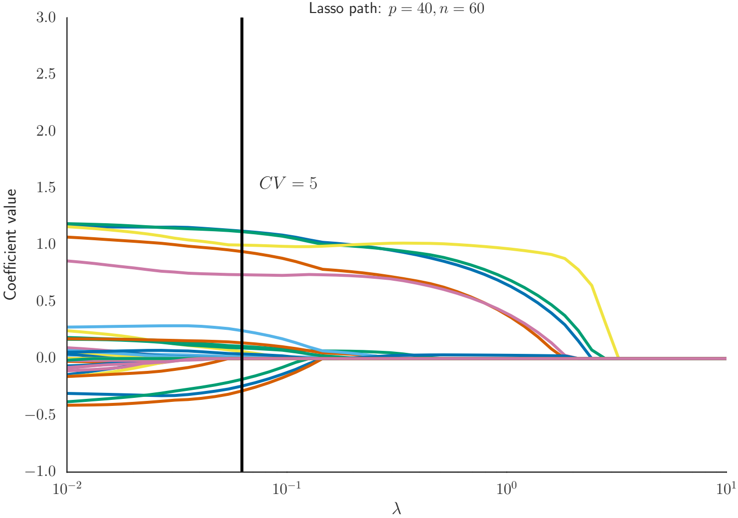
Lasso vs Ridge



Lasso vs Ridge



Lasso vs Ridge



Lasso properties

- Numerical aspect: the Lasso is a **convex** problem
- Variable selection / sparse solutions: $\hat{\boldsymbol{\theta}}_{\lambda}^{\mathrm{Lasso}}$ has potentially many zeroed coefficients. The λ parameter controls the sparsity level: if λ is large, solutions are very sparse.

  Example: We got 17 non-zero coefficients for `LassoCV` in the previous simulated example

  <u>Rem</u>: RidgeCV has no zero coefficients

Improvement and extensions for the Lasso

Elastic-net : ℓ_1/ℓ_2 regularization

The Elastic-Net, introduced by Zou and Hastie (2005) is the (unique) solution of

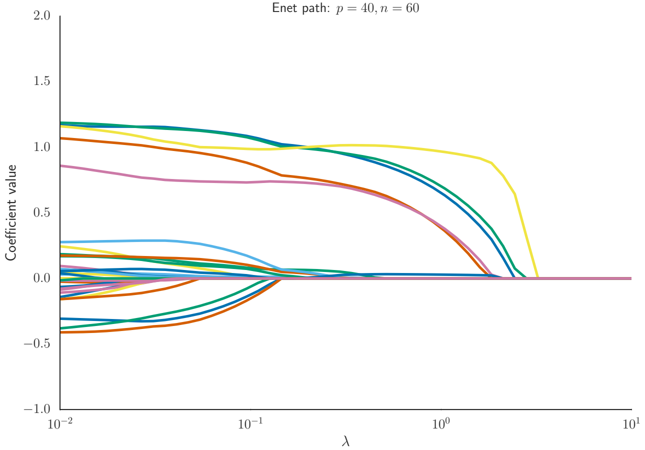
$$\hat{\boldsymbol{\theta}}_\lambda = \mathrm{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^p} \left[\frac{1}{2} \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 + \lambda \left(\gamma \|\boldsymbol{\theta}\|_1 + (1-\gamma) \frac{\|\boldsymbol{\theta}\|_2^2}{2} \right) \right]$$

Motivation: help selecting all relevant but correlated variable (not only one as for the Lasso)

Rem: two parameters needed, one for global regularization, one trading-off Ridge vs. Lasso

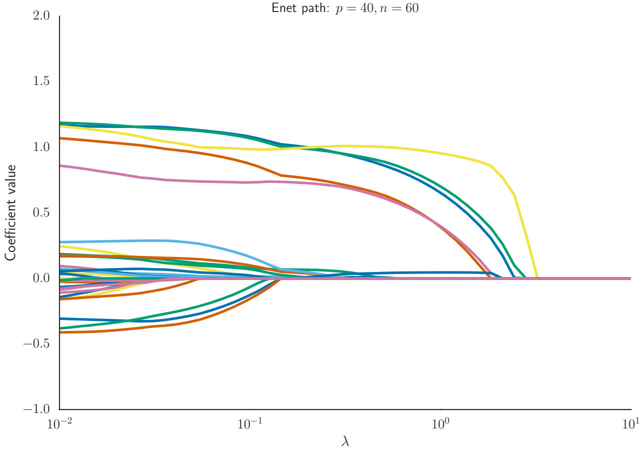
Rem: the solution is unique and the size of the Elastic-Net support is smaller than $\min(n, p)$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



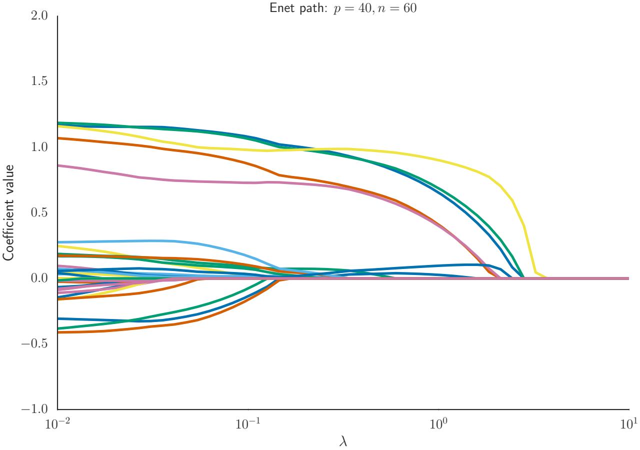
$$\gamma = 1.00$$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



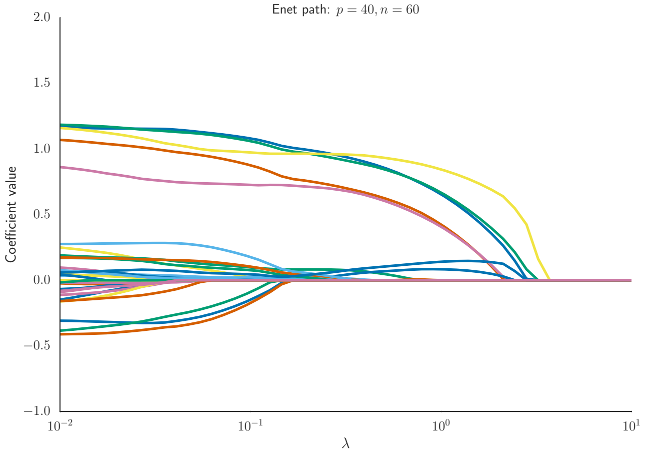
$\gamma = 0.99$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



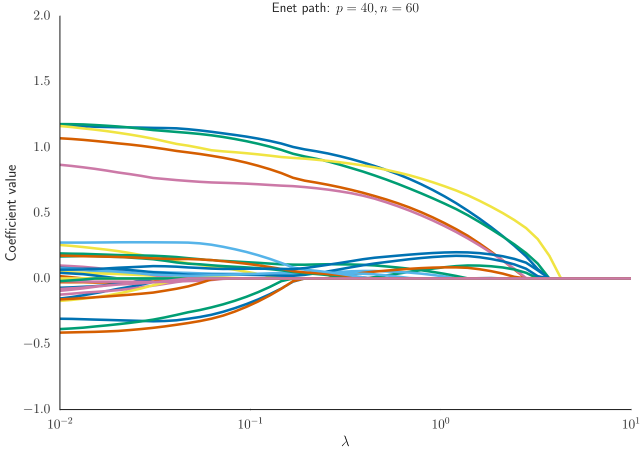
$\gamma = 0.95$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



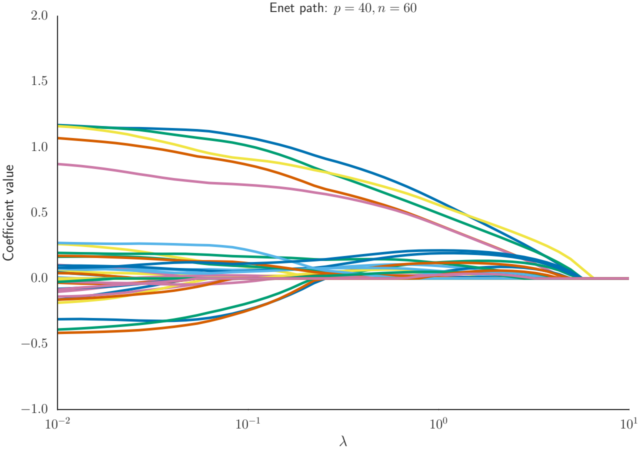
$\gamma = 0.90$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



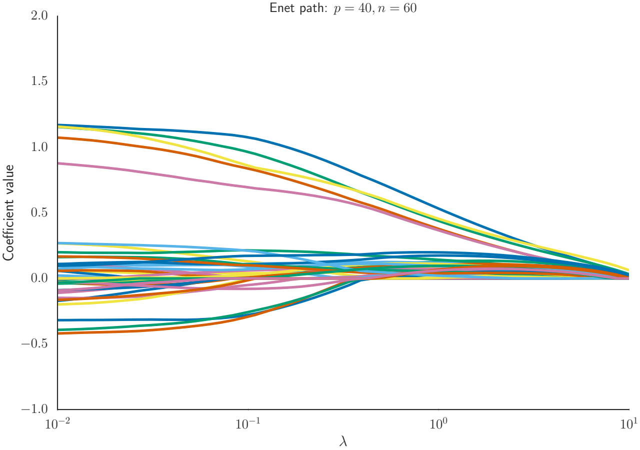
$\gamma = 0.75$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



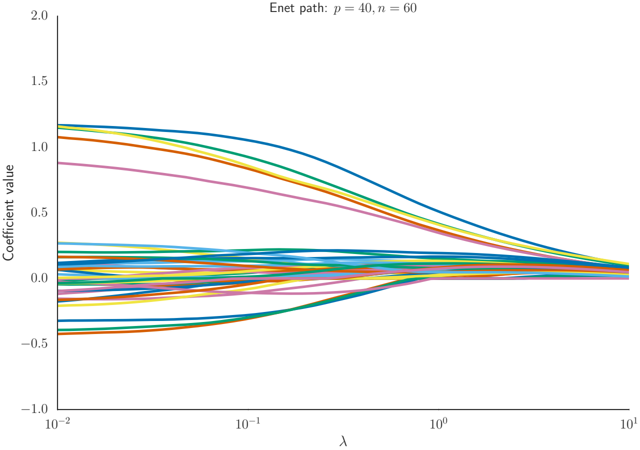
$\gamma = 0.50$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



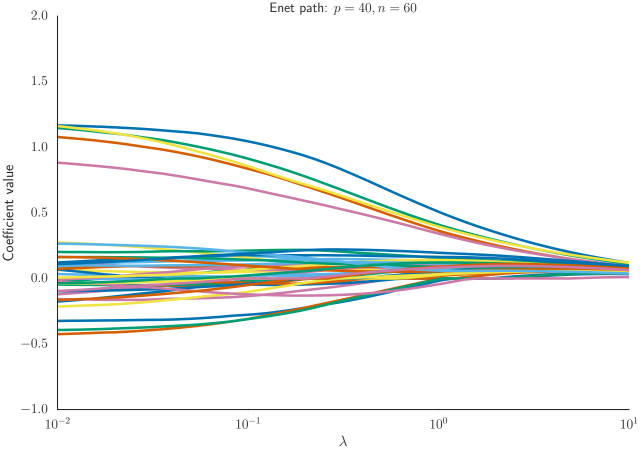
$\gamma = 0.25$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



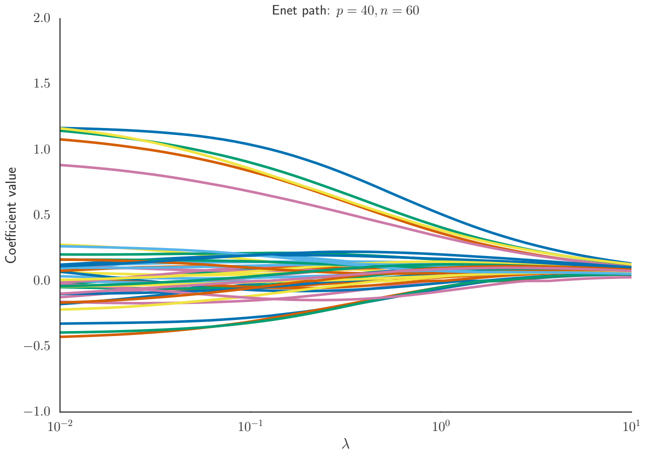
$$\gamma = 0.1$$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



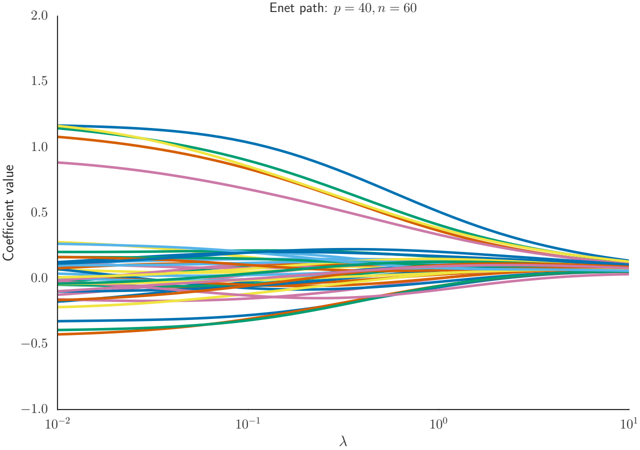
$\gamma = 0.05$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



$$\gamma = 0.01$$

Elastic-Net: $\gamma\|\boldsymbol{\theta}\|_1 + (1-\gamma)\|\boldsymbol{\theta}\|_2^2/2$



$\gamma = 0.00$

The Lasso bias

The Lasso is biased: it shrinks large coefficients towards 0

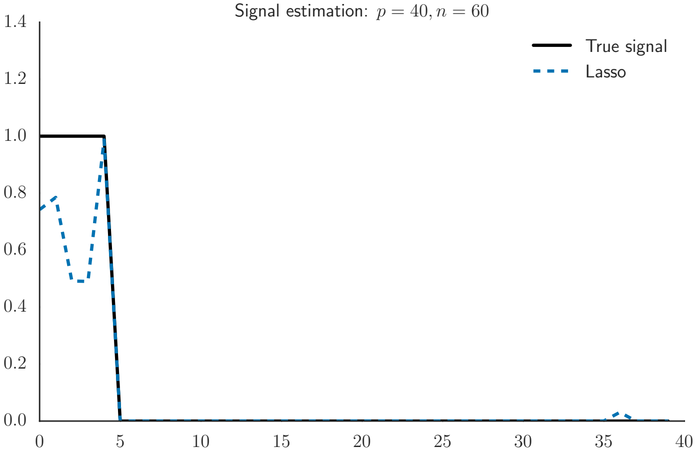


Illustration over the previous example

The Lasso bias

The Lasso is biased: it shrinks large coefficients towards 0

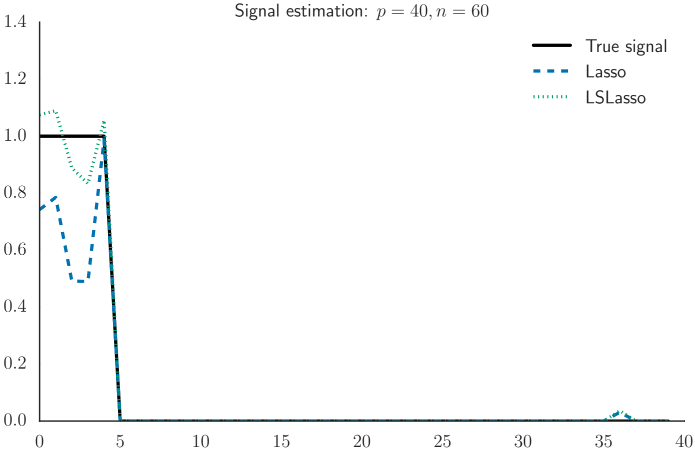


Illustration over the previous example

The Lasso bias: a simple remedy

How to rescale shrunk coefficients?

LSLasso (Least Square Lasso)

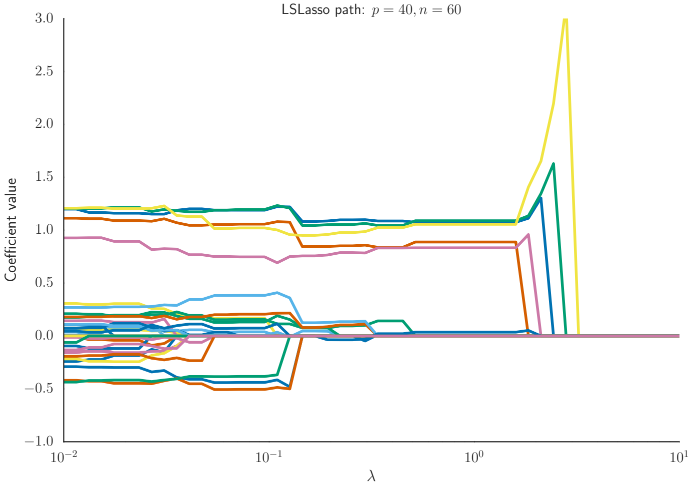
1. Lasso : compute $\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{Lasso}}$
2. Perform least squares over selected variables: $\mathrm{Supp}(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{Lasso}})$

$$\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{LSLasso}} = \operatorname*{argmin}_{\substack{\boldsymbol{\theta}\in\mathbb{R}^p \\ \mathrm{Supp}(\boldsymbol{\theta})=\mathrm{Supp}(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{Lasso}})}} \frac{1}{2}\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2$$

<u>Rem</u>: perform CV for the double step procedure; choosing λ by LassoCV and then performing OLS keeps too many variables

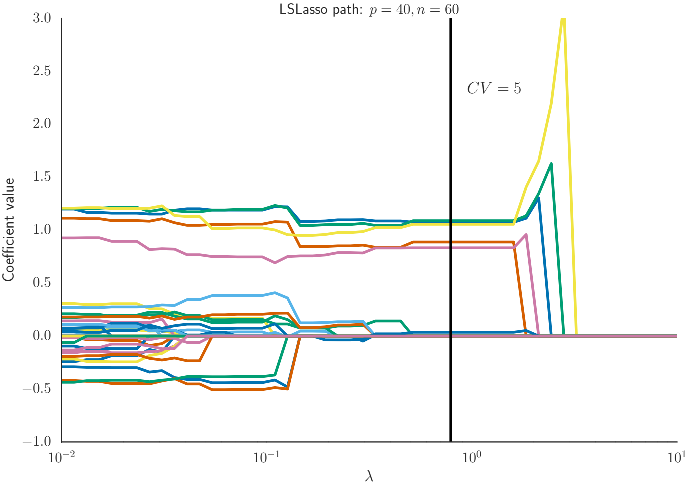
<u>Rem</u>: LSLasso is not coded in standard packages

De-biasing



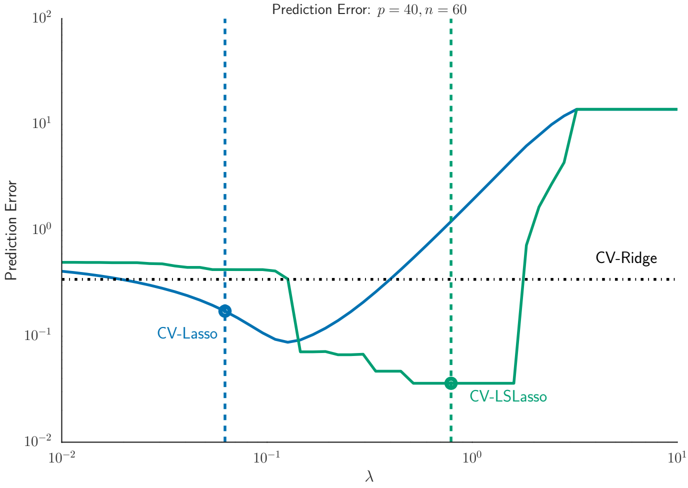
LSLasso path: $p = 40, n = 60$
Coefficient value
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
10^{-2}
10^{-1}
10^{0}
10^{1}
λ

De-biasing



LSLasso path: $p = 40, n = 60$
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
Coefficient value
$CV = 5$
10^{-2}
10^{-1}
10^{0}
10^{1}
λ

Prediction: Lasso vs. LSLasso



Summary

- Regularisation in Machine Learning is (almost always) necessary to prevents **complex models** from overfitting
- There are mainly three types of regularisation that are used
  - ℓ_2 regularisation aka Ridge
  - ℓ_1 regularisation aka Lasso
  - ℓ_1/ℓ_2 regularisation aka ElasticNet
- Ridge allows to reduce the impact of multicolinearity
- Lasso is particularly adapted in high dimension (in-built variable selection)
- ElasticNet takes the best of both world

Summary

- Regularisation in Machine Learning is (almost always) necessary to prevents **complex models** from overfitting
- There are mainly three types of regularisation that are used
  - ℓ_2 regularisation aka Ridge
  - ℓ_1 regularisation aka Lasso
  - ℓ_1/ℓ_2 regularisation aka ElasticNet
- Ridge allows to reduce the impact of multicolinearity
- Lasso is particularly adapted in high dimension (in-built variable selection)
- ElasticNet takes the best of both world

These penalizations can be added to **any parametric machine learning** models (logistic regression, SVM, Deep Neural Networks)