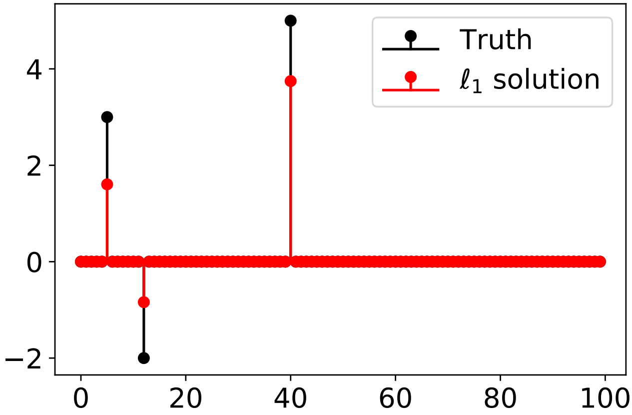
Algorithms for non-convex optimization in ML

Alexandre Gramfort

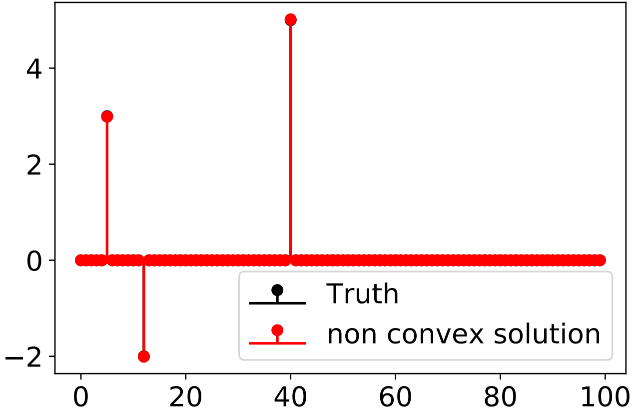
Master 2 Data Science, Univ. Paris Saclay
Optimisation for Data Science

Why non-convexity matters?



ℓ_1 (convex)

Why non-convexity matters?



non-convex

Non-convexity and machine learning

- Sparsity is a way to do feature selection while learning
- ℓ_1 regularization is just a convex surrogate of the ℓ_0 pseudo-norm which is the true quantification of sparsity.
- General non-convex optimization is (too) hard
- but for machine learning, e.g., $F(x) = f(x) + g(x)$ there is hope !
- We'll focus on non-convex regularizations

Non-convex penalties

Use a non-convex separable penalty $g(x) = \sum_i g_i(x^{(i)}) \approx \lambda \|x\|_0$:

$$\hat{x} = \arg\min_{x \in \mathbb{R}^n} \Bigg(\underbrace{f(x)}_{\textbf{data fit}} + \underbrace{\sum_{i=1}^{n} g_i(x^{(i)})}_{\textbf{regularization}} \Bigg)$$

Non-convex penalties

Use a non-convex separable penalty $g(x) = \sum_i g_i(x^{(i)}) \approx \lambda \|x\|_0$:

$$\hat{x} = \arg\min_{x \in \mathbb{R}^n} \Big(\underbrace{f(x)}_{\textbf{data fit}} + \underbrace{\sum_{i=1}^{n} g_i(x^{(i)})}_{\textbf{regularization}} \Big)$$

- Adaptive-Lasso Zou (2006) / ℓ_1 reweighted Candès *et al.* (2008)

$$g_i(t) = \lambda |t|^q \text{ with } 0 < q < 1$$

Non-convex penalties

Use a non-convex separable penalty $g(x) = \sum_i g_i(x^{(i)}) \approx \lambda \|x\|_0$:

$$\hat{x} = \arg\min_{x \in \mathbb{R}^n} \Big(\underbrace{f(x)}_{\textbf{data fit}} + \underbrace{\sum_{i=1}^{n} g_i(x^{(i)})}_{\textbf{regularization}} \Big)$$

- ℓ_1 reweighted Candès *et al.* (2008)

$$g_i(t) = \lambda \log(1 + |t|/\gamma)$$

Non-convex penalties

Use a non-convex separable penalty $g(x) = \sum_i g_i(x^{(i)}) \approx \lambda \|x\|_0$:

$$\hat{x} = \arg\min_{x \in \mathbb{R}^n} \Big(\underbrace{f(x)}_{\textbf{data fit}} + \underbrace{\sum_{i=1}^{n} g_i(x^{(i)})}_{\textbf{regularization}} \Big)$$

- MCP (*minimax concave penalty*) Zhang (2010) for $\lambda > 0$ and $\gamma > 1$

$$g_i(t) = \begin{cases} \lambda|t| - \frac{t^2}{2\gamma}, & \text{if } |t| \le \gamma\lambda \\ \frac{1}{2}\gamma\lambda^2, & \text{if } |t| > \gamma\lambda \end{cases}$$

Non-convex penalties

Use a non-convex separable penalty $g(x) = \sum_i g_i(x^{(i)}) \approx \lambda \|x\|_0$:

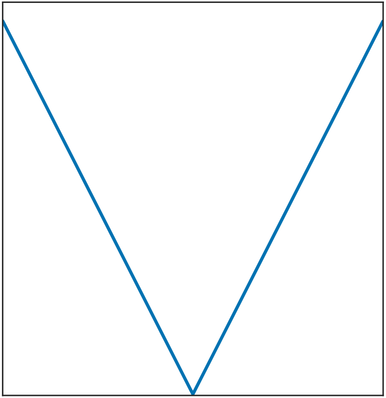
$$\hat{x} = \arg\min_{x \in \mathbb{R}^n} \Big(\underbrace{f(x)}_{\textbf{data fit}} + \underbrace{\sum_{i=1}^{n} g_i(x^{(i)})}_{\textbf{regularization}} \Big)$$

- SCAD (*Smoothly Clipped Absolute Deviation*) Fan et Li (2001) for $\lambda > 0$ and $\gamma > 2$

$$g_i(t) = \begin{cases} \lambda|t|, & \text{if } |t| \leq \lambda \\ \frac{\gamma\lambda|t| - (t^2+\lambda^2)/2}{\gamma-1}, & \text{if } \lambda < |t| \leq \gamma\lambda \\ \frac{\lambda^2(\gamma^2-1)}{2(\gamma-1)}, & \text{if } |t| > \gamma\lambda \end{cases}$$

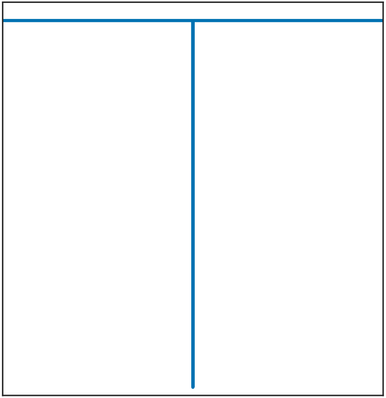
Remark: theoretically and algorithmically difficult (stopping criteria, local minima, etc.)

Classical penalties



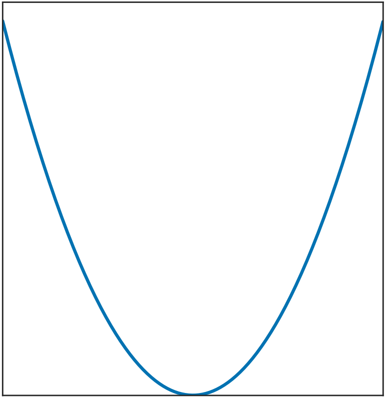
ℓ_1 (convex)

Classical penalties



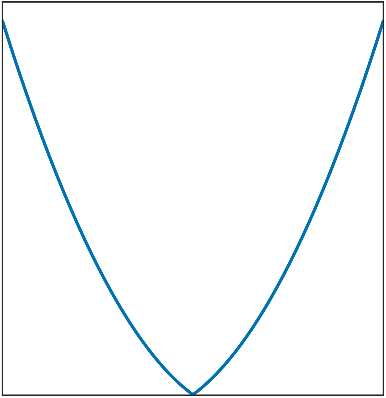
ℓ_0 (non-convex)

Classical penalties



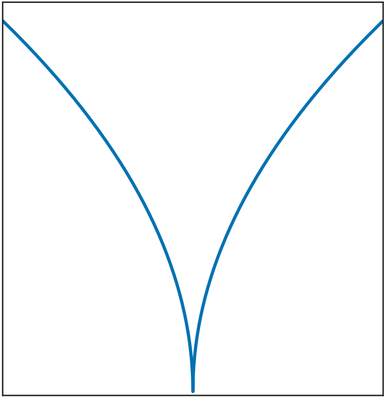
ℓ_2 (convex)

Classical penalties



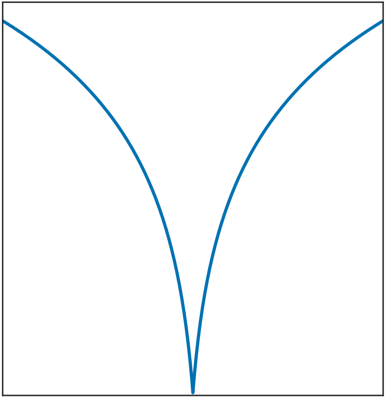
enet (convex)

Classical penalties



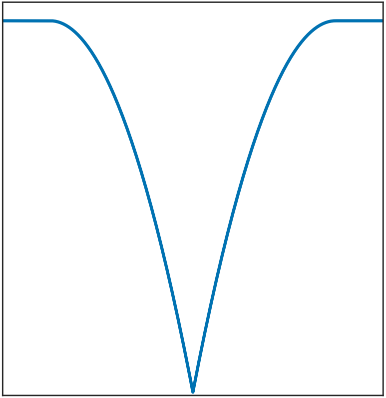
sqrt (non-convex)

Classical penalties



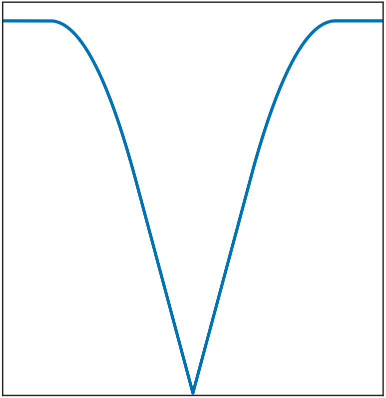
log (non-convex)

Classical penalties



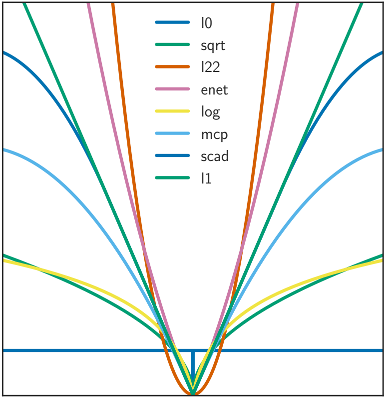
mcp (non-convex)

Classical penalties



scad (non-convex)

Classical penalties



CD for composite separable problem

We consider:

$$F(x) = f(x) + \sum_{i=1}^{n} g_i(x^{(i)}) \quad ,$$

with

- f convex, differentiable
- $g(x) = \sum_i g_i(x^{(i)})$ separable
- each g_i convex or non-convex

Proximal coordinate descent

Parameters: $\gamma_1, \ldots, \gamma_n > 0$

Algorithm:

Choose $i_{k+1} \in \{1, \ldots, n\}$

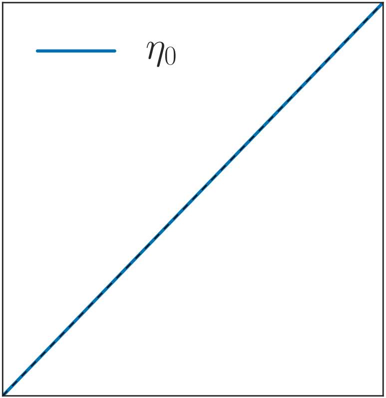
$$\begin{cases} x_{k+1}^{(i)} = \eta_{\gamma_i g_i}\left(x_k^{(i)} - \gamma_i \nabla_i f(x_k)\right) & \text{if } i = i_{k+1} \\ x_{k+1}^{(i)} = x_k^{(i)} & \text{if } i \neq i_{k+1} \end{cases}$$

$$\eta_{\gamma_i g_i}(z) = \arg\min_{x \in \mathbb{R}} g_i(x) + \frac{1}{2\gamma_i}(x - z)^2 \qquad \text{(Prox. operator)}$$

Remark: In non-convex case no guarantee to find a global minimum.

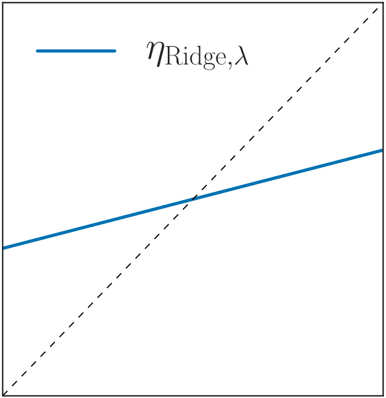
Regularization (1D): No $g_i(z) = 0$

$\eta_0(z) = z$



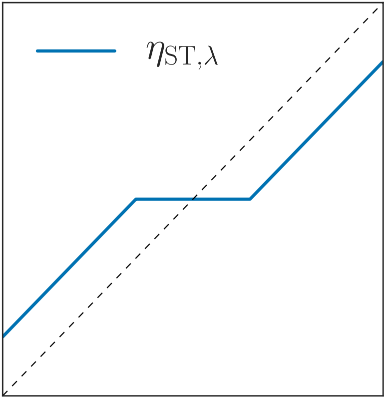
Regularization (1D): Ridge $g_i(z) = z^2$

$$\eta_{\mathrm{Ridge},\lambda}(z) = \frac{z}{1 + 2\lambda}$$



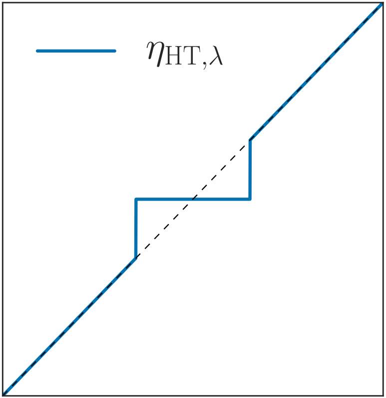
Regularization (1D): Lasso $g_i(z) = |z|$

$$\eta_{\mathrm{Lasso},\lambda}(z) = \mathrm{sign}(z)(|z| - \lambda)_+ \qquad \text{(Soft thresholding)}$$



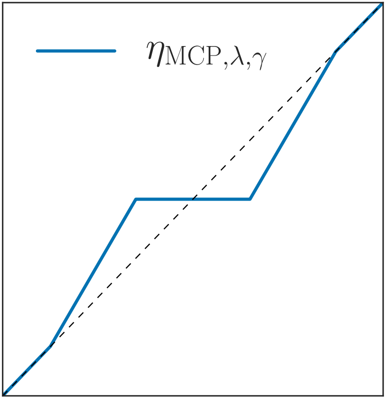
Regularization (1D): ℓ_0 $g_i(z) = \mathbf{1}_{z \neq 0}$

$$\eta_{\ell_0,\lambda}(z) = z\mathbf{1}_{|z| \geq \sqrt{2\lambda}} \qquad \text{(Hard thresholding)}$$



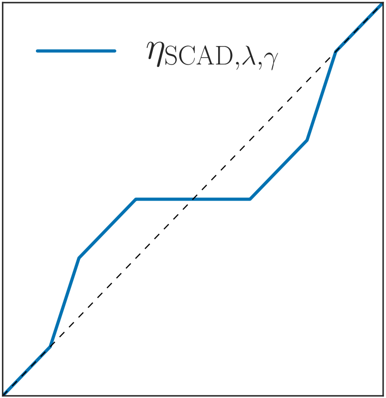
Regularization (1D): MCP

$$\eta_{\mathrm{MCP},\lambda,\gamma}(z) = \begin{cases} \mathrm{sign}(z)(|z| - \lambda)_+/(1 - 1/\gamma) & \text{if } |z| \leq \gamma\lambda \\ z & \text{if } |z| > \gamma\lambda \end{cases}$$



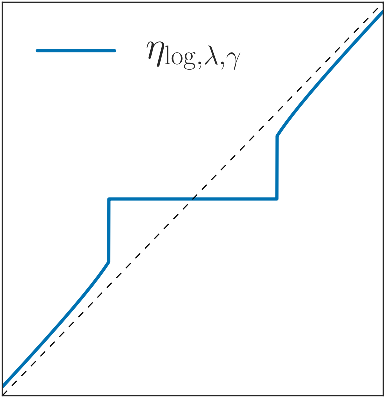
Regularization (1D): SCAD

$$\eta_{\mathrm{SCAD},\lambda,\gamma}(z) = \begin{cases} \mathrm{sign}(z)(|z|-\lambda)_+/(1-1/\gamma) & \text{if } |z| \leq 2\lambda \\ ([\gamma-1)z - \mathrm{sign}(z)\gamma\lambda]/(\gamma-2) & \text{if } 2\lambda \leq |z| \leq \gamma\lambda \\ z & \text{if } |z| > \gamma\lambda \end{cases}$$



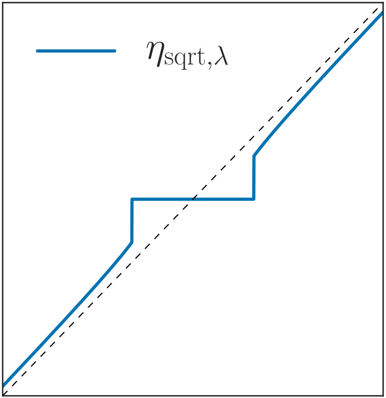
Regularization (1D): log $g_i(z) = \log(\varepsilon + |z|)$

$$\eta_{\log,\lambda}(z) = \ldots$$



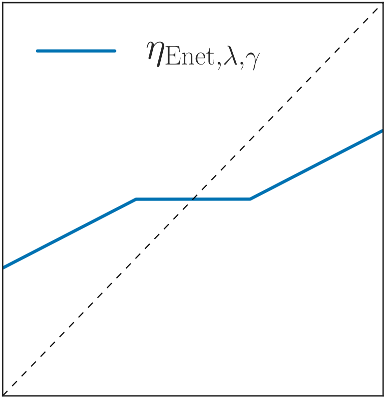
Regularization (1D): sqrt $g_i(z) = \sqrt{|z|}$

$\eta_{\mathrm{sqrt},\lambda}(z) = \ldots$

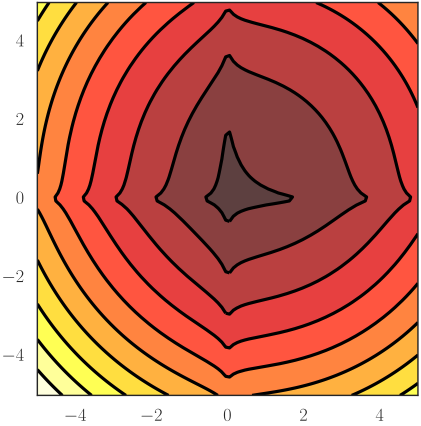


Regularization (1D): Enet $g_i(z) = \rho|z| + (1-\rho)z^2$

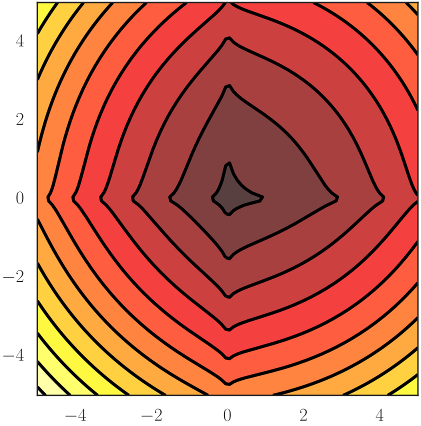
$$\eta_{Enet,\lambda,\rho}(z) = \dots$$



Level lines for log



Level lines for sqrt



Prox. CD with squared loss

Let $f(x) = \frac{1}{2}\|y - Ax\|^2$, where $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{m\times n}$ is the design matrix with columns $A_1, \ldots, A_n$ (one per feature)

Consider minimizing over $x^{(i)}$, with all $x^{(j)}$, $j \neq i$ fixed.

We obtain:

$$x^{(i)} \leftarrow \eta_{\frac{1}{\|A_i\|^2} g_i}\left(x^{(i)} + \frac{A_i^\top r}{\|A_i\|^2}\right)$$

where $r = y - Ax$ is the current *residual*.

Repeat these updates by cycling or random pass over coordinates.

$\rightarrow$ notebook

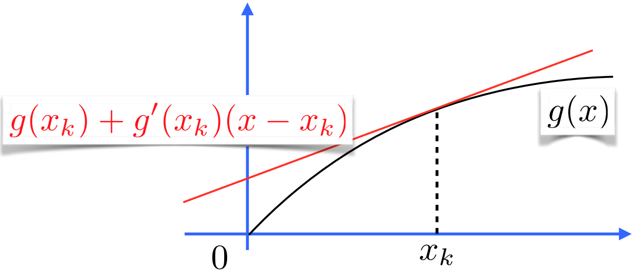
Adaptive-Lasso

Many names for the same idea:

- Adaptive-Lasso Zou (2006)
- ℓ_1 reweighted Candès *et al.* (2008)
- DC-programming (for *Difference of Convex Programming*) Gasso *et al.* (2008)

Intuition for adaptive-Lasso & Majorization-Minimization

A non-convex concave function can be upper bounded by its tangent:



The idea of Majorization-Minimization (MM) is to minimize convex majorant functions iteratively.

Adaptive Lasso (for $q = 1/2$)

Example : take g_j concave e.g. $g_j(t) = \lambda |t|^q$ with $q = 1/2$

Require: $X, \mathbf{y}$, number of iterations K, regularization λ

1: Initialization: $\hat{w} \leftarrow (1, \ldots, 1)^\top$
2: **for** $k = 1, \ldots, K$ **do**
3: $\quad \hat{\boldsymbol{\theta}} \leftarrow \arg\min_{\boldsymbol{\theta}} \left(\frac{\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2}{2} + \lambda \sum_{j=1}^{p} \hat{w}_j |\theta_j| \right)$
4: $\quad \hat{w}_j \leftarrow g_j'(\hat{\theta}_j), \forall j \in [\![1, p]\!]$
5: **end for**
6: **return** $\hat{\boldsymbol{\theta}}$

Remark: in practice no need to do many iterations (5 iterations)

Remark: use a Lasso solver to compute $\hat{\boldsymbol{\theta}}$

$\rightarrow$ notebook