Practical Applications of the L1 penalty

Aurélie Boisbunon
Data Researcher @Ericsson R&D Center France

StatLearn, April 2022

Outline

Outline

Before we start

Jobs in ML/AI

Source: Les Décodeuses du numérique

Before we start

Jobs in ML/AI

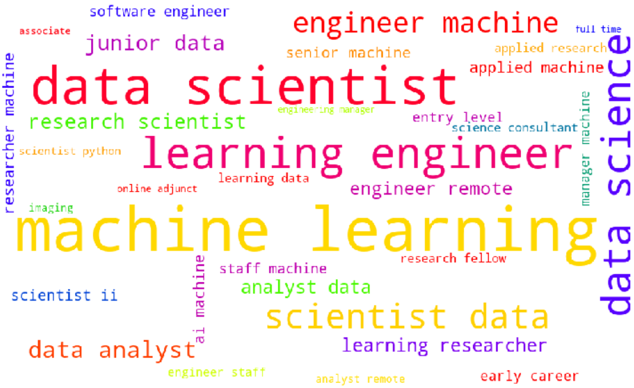
"There is the royal way, getting an AI job in a company and the imperial way, getting an AI job in academia"

Stéphane Canu

Source: Les Décodeuses du numérique

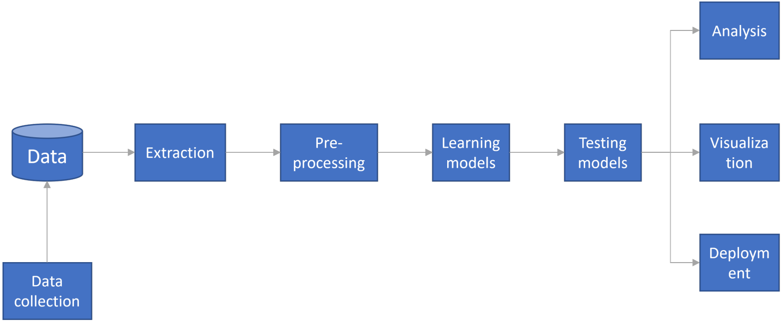
Before we start

Jobs in ML/AI



Before we start

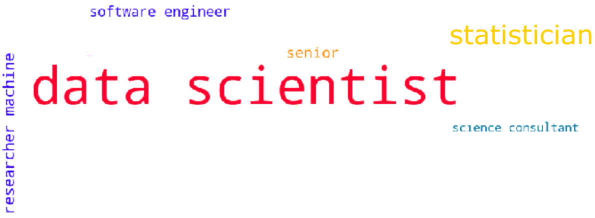
Jobs in ML/AI



Data engineer ML engineer Data Scientist AI researcher Data analyst

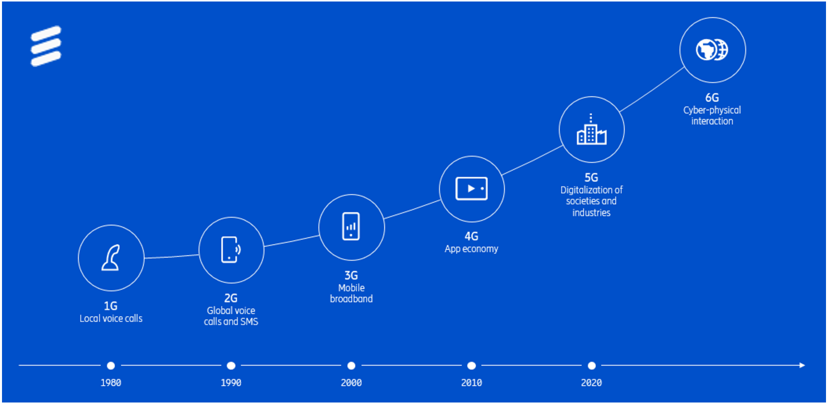
Before we start

My personal experience



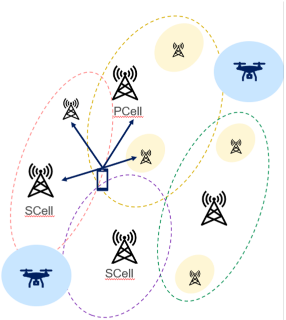
AI in telecom and at Ericsson

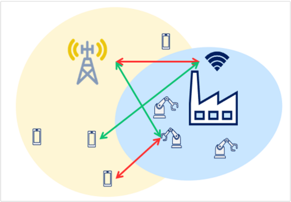
- AI is progressively being integrated into 5G and 6G networks

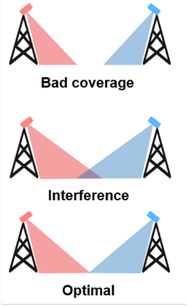


AI in telecom and at Ericsson

- AI is progressively being integrated into 5G and 6G networks
- AI & Systems team started in Paris area (near Saclay) in November
  - Main topics: reinforcement learning, transfer learning, sparse models

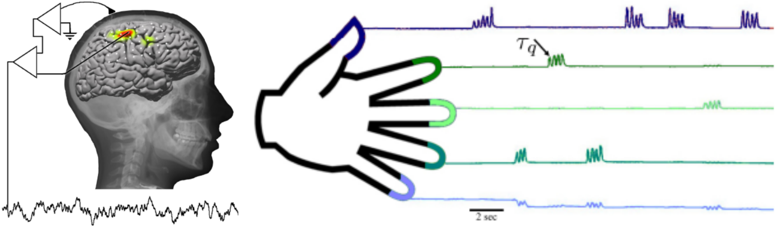






Outline

Motivational example: Brain Computer Interface (BCI)



BCI Competition IV, Dataset 4

- Data: Recordings of ECoG brain signals and of simultaneous finger flexion of a subject (using a glove)
- Objective: predict movement (angle) of the 5 fingers of the subject from its recorded ECoG
- Best performances (at the time!) were obtained using a linear model [Tangermann et al., 2012, Flamary and Rakotomamonjy, 2012]

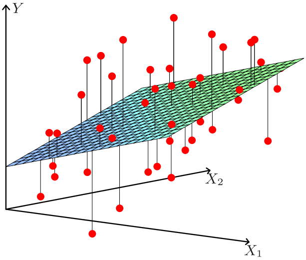
Linear regression

Linear regression model

Find $\mathbf{w} = (w_1, \ldots, w_d) \in \mathbb{R}^d$ such that

$$y_i = \sum_{j=1}^{d} w_j x_{i,j} + \sigma \varepsilon_i, \quad i = 1, \ldots, n$$

$y_i \in \mathbb{R}$
$\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,d})$ fixed, $d < n$
$\varepsilon_i \in \mathbb{R}$, $\mathbb{E}[\varepsilon_i] = 0$, $\mathbb{E}[\varepsilon_i^2] = 1$



Source: [Hastie et al., 2008]

Predictions

Once we estimate the linear coefficient vector, the predictions for a new observation $\mathbf{x}_{new}$ is given by:

$$\hat{y} = \sum_{j=1}^{d} \hat{w}_j x_{new,j} = \mathbf{x}_{new}^\top \hat{\mathbf{w}}$$

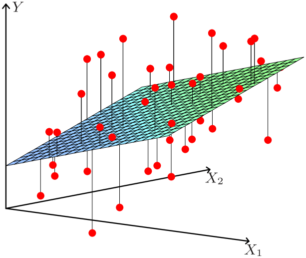
Linear regression

Linear regression model

Find $\mathbf{w} = (w_1, \dots, w_d) \in \mathbb{R}^d$ such that

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \sigma\varepsilon$$

$\mathbf{y} = (y_1, \dots, y_n) \in \mathbb{R}^n$
$\mathbf{X} = (\mathbf{x}^1, \dots, \mathbf{x}^d)$ fixed, $d < n$
$\varepsilon = (\varepsilon_1, \dots, \varepsilon_n) \in \mathbb{R}^n$, $\mathbb{E}[\varepsilon] = 0$, $\mathbb{E}[\varepsilon^\top \varepsilon] = n$.



Source: [Hastie et al., 2008]

Predictions

Once we estimate the linear coefficient vector, the predictions for a new observation $\mathbf{x}_{new}$ is given by:

$$\hat{y} = \sum_{j=1}^{d} \hat{w}_j x_{new,j} = \mathbf{x}_{new}^\top \hat{\mathbf{w}}$$

Notations

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & 1 \\ \mathbf{x}_2 & 1 \\ \vdots & \vdots \\ \mathbf{x}_i & 1 \\ \vdots & \vdots \\ \mathbf{x}_n & 1 \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,j} & \dots & x_{1,d} & 1 \\ x_{2,1} & x_{2,2} & \dots & x_{2,j} & \dots & x_{2,d} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{i,1} & x_{i,2} & \dots & x_{i,j} & \dots & x_{i,d} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,j} & \dots & x_{n,d} & 1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_i \\ \vdots \\ y_n \end{bmatrix}$$

- $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \dots, x_{i,j}, \dots x_{i,d})^\top$ denotes the features for sample i
- $\mathbf{x}^j = (x_{1,j}, x_{2,j}, \dots, x_{i,j}, \dots, x_{n,j})^\top$ denotes variable j

Least Squares solution

Optimization problem

We want to solve

$$\min_{\mathbf{w}} \quad J(\mathbf{w}) \qquad \text{with} \qquad J(\mathbf{w}) = \frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||^2$$

where $J(\mathbf{w})$ is a convex function.

Least Squares solution

Optimization problem

We want to solve

$$\min_{\mathbf{w}} \quad J(\mathbf{w}) \qquad \text{with} \quad J(\mathbf{w}) = \frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||^2$$

where $J(\mathbf{w})$ is a convex function.

$\Rightarrow$ Find the parameter $\mathbf{w}$ that leads to a null gradient:

$$\nabla J(\widehat{\mathbf{w}}) = 0 \quad \Leftrightarrow \quad -\mathbf{X}^\top \mathbf{y} + \mathbf{X}^\top \mathbf{X}\widehat{\mathbf{w}} = \mathbf{0}$$

Least Squares solution

Optimization problem

We want to solve

$$\min_{\mathbf{w}} \quad J(\mathbf{w}) \qquad \text{with} \quad J(\mathbf{w}) = \frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||^2$$

where $J(\mathbf{w})$ is a convex function.

$\Rightarrow$ Find the parameter $\mathbf{w}$ that leads to a null gradient:

$$\nabla J(\widehat{\mathbf{w}}) = 0 \quad \Leftrightarrow \quad -\mathbf{X}^\top \mathbf{y} + \mathbf{X}^\top \mathbf{X}\widehat{\mathbf{w}} = \mathbf{0}$$

The solution for Least Squares is the vector $\widehat{\mathbf{w}}^{ls}$ defined as

$$\widehat{\mathbf{w}}^{ls} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}$$

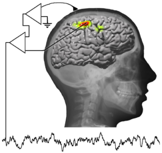
Assumptions

$\mathbf{X}$ is a matrix of rank d (or $d+1$ if bias included) which means that $\mathbf{X}^\top\mathbf{X}$ is invertible.

Sparse linear regression

Issue

What if only a small number of variables are relevant?



Problems

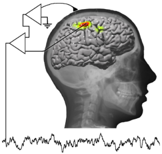
$$\hat{y} = f(\mathbf{x}) = \sum_{j \in J} w_j \mathbf{x}_j$$

- Find a set J of relevant variables
- Estimate the corresponding $\mathbf{w}_J = (w_j)_{j \in J}$

Sparse linear regression

Issue

What if only a small number of variables are relevant?



Problems

$$\hat{y} = f(\mathbf{x}) = \sum_{j \in J} w_j \mathbf{x}_j = \sum_{j=1}^{d} w_j \mathbf{x}_j$$

- Find a set J of relevant variables
- Estimate the corresponding $\mathbf{w}_J = (w_j)_{j \in J}$
- For the others: $w_j = 0 \quad \forall j \notin J$

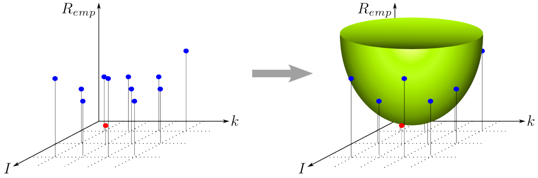
Sparse linear regression

What we would like to do:

$$\begin{cases} \min\limits_{\mathbf{w}\in\mathbb{R}^d} & \dfrac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top \mathbf{w})^2 \\ \text{s.t.} & \#\{w_j \neq 0\} \leq k \end{cases}$$

Issues

- NP-hard problem
- difficult[1] for $d > 40$



[1] One way to avoid the computational burden is to use greedy algorithms

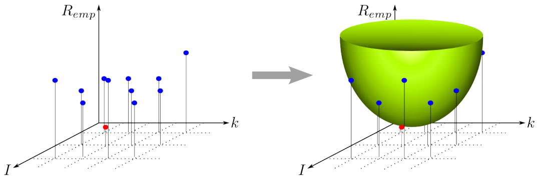
Sparse linear regression

What we would like to do:

$$\begin{cases} \min\limits_{\mathbf{w}\in\mathbb{R}^d} & \dfrac{1}{n}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|^2 \\ \text{s.t.} & \|\mathbf{w}\|_0 = \#\{w_j \neq 0\} \leq k \end{cases}$$

Issues

- NP-hard problem
- difficult[1] for $d > 40$



[1] One way to avoid the computational burden is to use greedy algorithms

Sparse linear regression

First approaches:

Best subset: Leaps and bounds (Furnival and Wilson, 1974), branch and bound
Statistical tests:

- Statistical tests for $\hat{w}_j = 0$ (Z-score)
- Statistical tests for J vs J' (F-tests)

Updating set I by adding/removing a variable:

- Forward/backward/stagewise selection [Efroymson, 1960]

Sparse linear regression

First approaches:

Best subset: Leaps and bounds (Furnival and Wilson, 1974), branch and bound
Statistical tests:

- Statistical tests for $\hat{w}_j = 0$ (Z-score)
- Statistical tests for J vs J' (F-tests)

Updating set I by adding/removing a variable:

- Forward/backward/stagewise selection [Efroymson, 1960]

Seminal works on ℓ_1 penalty

- Linear inversion for seismic data [Santosa and Symes, 1986]
- Soft-thresholding [Donoho, 1995]
- Least Absolute Shrinkage and Selection Operator (Lasso) [Tibshirani, 1996]

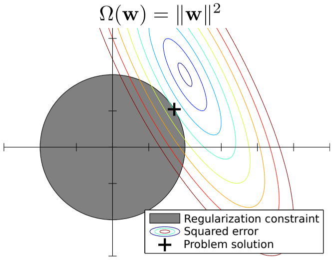
Regularization with the ℓ_2 penalty

Before the ℓ_1 penalty, interesting works had been obtained with the ℓ_2 penalty [Hoerl and Kennard, 1970]

Optimization problems

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top \mathbf{w})^2 \quad s.t. \sum_{j=1}^{d} w_j^2 \leq t$$

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ \frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_2^2 \right\}$$



Source: R. Flamary

- Reduce variance by adding bias

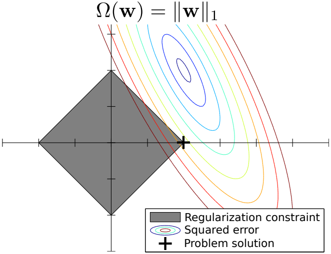
Regularization with the ℓ_1 penalty

Optimization problems

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top \mathbf{w})^2 \quad s.t. \ \sum_{j=1}^{d}|w_j| \le t$$

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ \frac{1}{n}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_1 \right\}$$

- Convex relaxation of the ℓ_0 norm
- Simultaneous selection of variables and estimation
- The ℓ_1 norm promotes sparsity



Source: R. Flamary

Diabetes example (sklearn)

Data

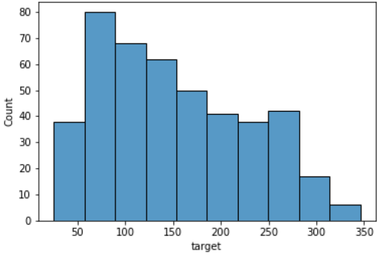
- $n = 442$ diabetes patients
- target = quantitative measure of disease progression one year after baseline
- $d = 10$ input variables: age, sex, body mass index, average blood pressure, and six blood serum measurements

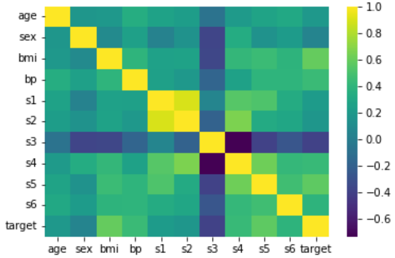
```
from sklearn.datasets import load_diabetes
data = load_diabetes()
X = data.data
y = data.target
features = data.feature_names
```

Diabetes example (sklearn)

Data

- $n = 442$ diabetes patients
- target = quantitative measure of disease progression one year after baseline
- $d = 10$ input variables: age, sex, body mass index, average blood pressure, and six blood serum measurements

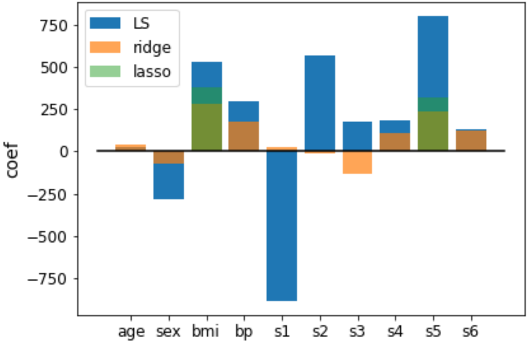




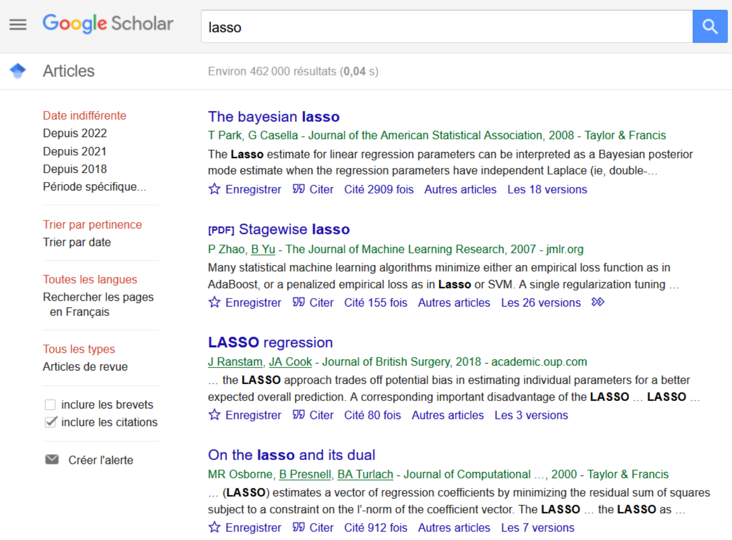
Diabetes example (sklearn)

Estimation of the disease progression

```
from sklearn.linear_model import Lasso
lasso = Lasso(alpha=1)  # default value
lasso.fit(Xtrain, ytrain)
ypred_lasso = lasso.predict(Xtest)
```



Properties of the Lasso



Properties of the Lasso

Special case: $\mathbf{X}$ orthogonal

When $\mathbf{X}$ is orthogonal, we have: $\mathbf{X}^\top \mathbf{X} = I_d$, that is $\mathbf{x}_j^\top \mathbf{x}_j = 1$ and $\mathbf{x}_j^\top \mathbf{x}_l = 0$ for $l \neq j$

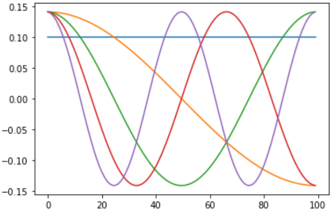
Properties of the Lasso

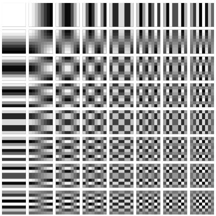
Special case: $\mathbf{X}$ orthogonal

When $\mathbf{X}$ is orthogonal, we have: $\mathbf{X}^\top \mathbf{X} = I_d$, that is $\mathbf{x}_j^\top \mathbf{x}_j = 1$ and $\mathbf{x}_j^\top \mathbf{x}_l = 0$ for $l \neq j$

Example: Basis of discrete cosine

$$\cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)\left(k+\frac{1}{2}\right)\right]$$



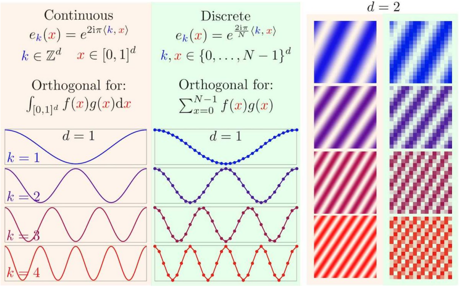


Properties of the Lasso

Special case: $\mathbf{X}$ orthogonal

When $\mathbf{X}$ is orthogonal, we have: $\mathbf{X}^\top\mathbf{X} = I_d$, that is $\mathbf{x}_j^\top\mathbf{x}_j = 1$ and $\mathbf{x}_j^\top\mathbf{x}_l = 0$ for $l \neq j$

Example: Basis of discrete Fourier



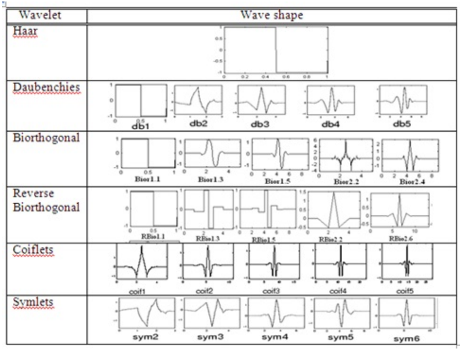
Source: G. Peyré

Properties of the Lasso

Special case: $\mathbf{X}$ orthogonal

When $\mathbf{X}$ is orthogonal, we have: $\mathbf{X}^\top \mathbf{X} = I_d$, that is $\mathbf{x}_j^\top \mathbf{x}_j = 1$ and $\mathbf{x}_j^\top \mathbf{x}_l = 0$ for $l \neq j$

Example: basis of wavelets



Source: [Tarique, 2016]

Properties of the Lasso

Special case: $\mathbf{X}$ orthogonal

- Least squares solution:

$$\hat{\mathbf{w}}^{ls} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \mathbf{y} \qquad \hat{w}_j^{ls} = \mathbf{x}_j^\top \mathbf{y} \qquad \Rightarrow \qquad \hat{y}^{ls} = \mathbf{x}_{new} \mathbf{X}^\top \mathbf{y}$$

Properties of the Lasso

Special case: $\mathbf{X}$ orthogonal

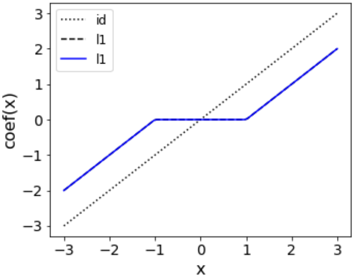
- Least squares solution:

$$\hat{\mathbf{w}}^{ls} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \mathbf{y} \qquad \hat{w}_j^{ls} = \mathbf{x}_j^\top \mathbf{y} \qquad \Rightarrow \qquad \hat{y}^{ls} = \mathbf{x}_{new}\mathbf{X}^\top \mathbf{y}$$

- Lasso solution:

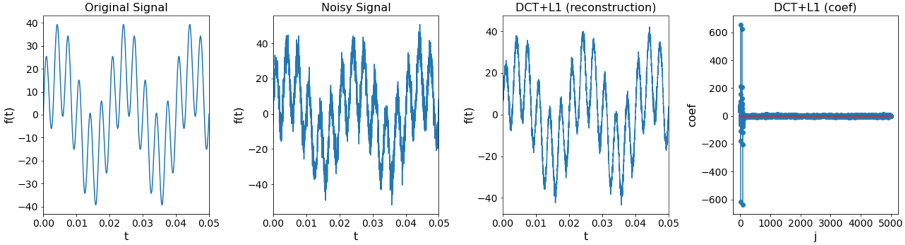
$$\begin{aligned} \hat{w}_j^{lasso} &= (\hat{w}_j^{ls} - \lambda \mathrm{sgn}(\hat{w}_j^{ls}))\mathbb{1}_{\{|\hat{w}_j^{ls}|>\lambda\}} \\ &= \mathrm{sgn}(\hat{w}_j^{ls}) \max\left(|\hat{w}_j^{ls}| - \lambda, 0\right) \end{aligned}$$

$$\hat{y}^{lasso} = \mathbf{x}_{new}^\top \hat{\mathbf{w}}^{lasso}$$



Example with DCT

- data = periodic signal with 2 frequencies (e.g. temperature, tides)
- $n = 5000$ measurements
- noise can come from measurements or external conditions



Solving the Lasso ($\mathbf{X}$ general)

Solving a **differentiable** and **convex** optimization problem is usually performed in 2 steps:

- derive the function to optimize (e.g. in Lagrangian form)
- finding its root by closed form or iteratively e.g. with gradient descent

Solving the Lasso ($\mathbf{X}$ general)

Solving a **differentiable** and **convex** optimization problem is usually performed in 2 steps:

- derive the function to optimize (e.g. in Lagrangian form)
- finding its root by closed form or iteratively e.g. with gradient descent

Convexity of the Lasso

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ J_{lasso} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_1 \right\}$$

- Sum of 2 convex functions = convex function
- $\mathbf{w} \in \mathbb{R}^d$: convex domain

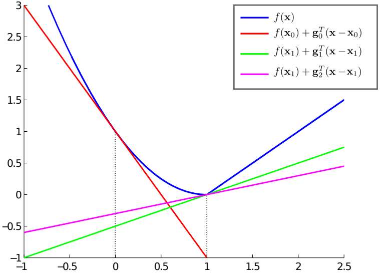
$\Rightarrow$ The problem is therefore convex (but not strictly convex)

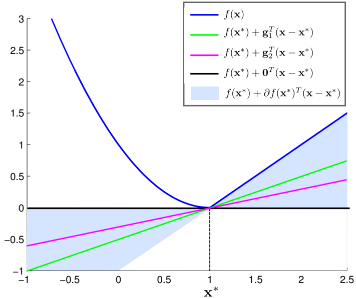
$\Rightarrow$ Any local minimum is also a global minimum

Nondifferentiability of the Lasso

- The absolute value is nondifferentiable in 0

Subgradients and subdifferential



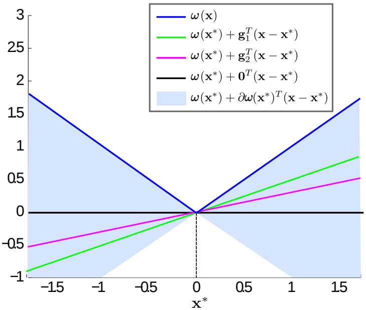


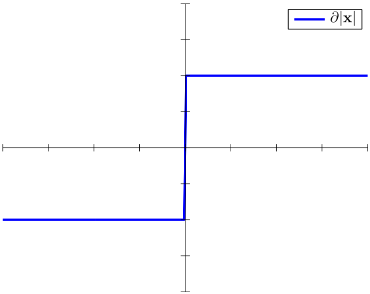
- The notion of gradient can be extended for nondifferentiable functions
- For a convex function $f(\mathbf{x})$, $\mathbf{g}$ is a subgradient of f in $\mathbf{x}_0$ if

$$f(\mathbf{x}) \geq f(\mathbf{x}_0) + \mathbf{g}^\top(\mathbf{x} - \mathbf{x}_0)$$

- The set of all subgradients at $\mathbf{x}_0$ is the subdifferential $\partial f(\mathbf{x}_0)$
- $\mathbf{x}_0$ is a minimum of the convex function f if $\mathbf{0} \in \partial f(\mathbf{x}_0)$

Subdifferential for the ℓ_1 penalty

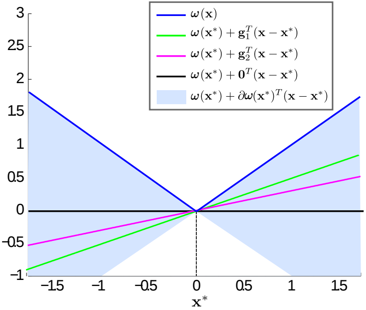


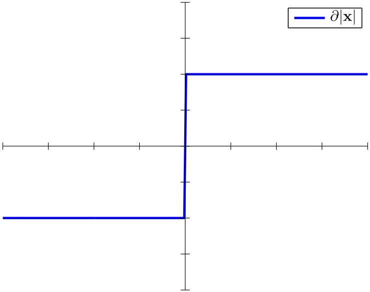


The subdifferential of the absolute value is of the form

$$\partial|x| = \begin{cases} \alpha \in]-1,1[& \text{if } x = 0 \\ sign(x) & \text{if } x \neq 0 \end{cases}$$

Subdifferential for the ℓ_1 penalty





The subdifferential of the ℓ_1 penalty is of the form

$$\partial\|\mathbf{w}\|_1 = (\partial|w_j|)_{j=1}^d = \begin{pmatrix} sign(\mathbf{w}_J) \\ \boldsymbol{\alpha}_{J^c} \end{pmatrix}$$

with J^c the complement pf J *(assuming we know J and the coefficients are reordered)*

Optimality conditions of the Lasso

$\mathbf{w}^\star$ is a solution of the optimization problem if

$$\mathbf{0} \in \partial J_{lasso}(\mathbf{w}^\star) \quad \text{with} \quad J_{lasso}(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|_1$$

This can be reformulated as the following condition

$$-\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\mathbf{w}^\star) + \lambda \mathbf{g} = \mathbf{0} \quad \text{with} \quad \mathbf{g} \in \partial\|\mathbf{w}^\star\|_1$$

Conditions on the components of $\mathbf{w}^\star$

$$\begin{array}{lcr} w_j^\star \neq 0 & \Rightarrow & -\mathbf{x}_j^\top(\mathbf{y} - \mathbf{X}\mathbf{w}^\star) + \lambda \text{sign}(w_j^\star) = 0 \\ w_j^\star = 0 & \Rightarrow & |\mathbf{x}_j^\top(\mathbf{y} - \mathbf{X}\mathbf{w}^\star)| \leq \lambda \end{array}$$

- $\mathbf{x}_j$ is the jth column of $\mathbf{X}$ (feature j).

Lasso with Python

Scikit-Learn

- `sklearn.linear_model.Lasso`: coordinate descent algorithm
- `sklearn.linear_model.SGDRegressor`: stochastic gradient descent
- `sklearn.linear_model.Lars`: least angle regression

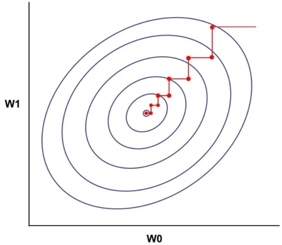
Other Python toolboxes

- `pylops`: ISTA, FISTA and others
- `spams`, `cyanure`: stochastic gradient descent
- `celer.Lasso`, `celer.celer_path`: dual extrapolation

Solvers for Lasso

Coordinate descent (CD)

- Optimize each component of **w** independently until convergence
- Very fast for sparse solutions




```
from sklearn.linear_model import Lasso
lasso = Lasso(alpha=1)  # default value
lasso.fit(Xtrain, ytrain)
ypred_lasso = lasso.predict(Xtest)
```

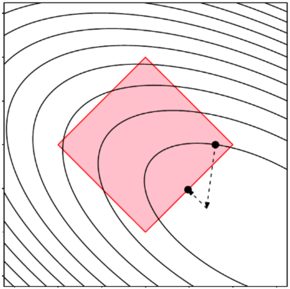
Solvers for Lasso

Proximal gradient descent (PGD)

- Each iteration is a simple soft thresholding of the parameter

$$\mathbf{w}^{(l+1)} = S_\lambda(\mathbf{w}^{(l)} - \gamma \nabla L(\mathbf{y}, \mathbf{X}\mathbf{w}^{(l)}))$$

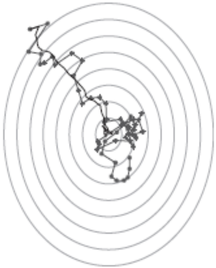
  where $S_\lambda(x) = (x - \lambda)_+$ is the soft-thresholding operator
- (F)ISTA: (Fast) Iterative Soft-Thresholding Algorithm [Daubechies et al., 2010, Beck and Teboulle, 2009]
- Can be coupled with active sets to speedup sparse solutions



Solvers for Lasso

Stochastic gradient descent (SGD)

- Based on proximal algorithms
- Compute the gradient for one sample and optimize for the whole dataset
- Very efficient



```
from sklearn.linear_model import SGDRegressor
lassoSGD = SGDRegressor(penalty='l1', alpha=1)
lassoSGD.fit(Xtrain, ytrain)
ypred_lassoSGD = lassoSGD.predict(Xtest)
```

Regularization path

Aim: find the solution to Lasso for all λ

- Compressed sensing: Basis pursuit denoising [Chen and Donoho, 1994]
- Statistics: Least Angle Regression (LAR) [Efron et al., 2004]

Recall the optimality condition for nonzero coefficients:

$$-\mathbf{X}_J^\top(\mathbf{y} - \mathbf{X}_J\mathbf{w}_J^\star) + \lambda \mathrm{sign}(w_J^\star) = 0$$

Then, assuming we know J and $sign(\mathbf{w}_J)$ (and we can easily compute the inverse matrix):

$$\mathbf{w}_J^\star = (\mathbf{X}_J^\top \mathbf{X}_J)^{-1}\left(\mathbf{X}_J^\top \mathbf{y} - \lambda \mathrm{sign}(\mathbf{w}_J^\star)\right)$$

$\mathbf{w}^*$ is actually linear by parts with respect to λ: we only need to compute it for transition points λ

Regularization path

Aim: find the solution to Lasso for all λ

- Compressed sensing: Basis pursuit denoising [Chen and Donoho, 1994]
- Statistics: Least Angle Regression (LAR) [Efron et al., 2004]

Algorithm

Start: $\mathbf{w}^{(0)} = \mathbf{0}$, $J^{(0)} = \emptyset$, $\lambda^{(0)} = \max_j |\mathbf{x}_j^\top \mathbf{y}|$

Repeat

1. Find vector $\mathbf{x}_j$ most correlated with residual

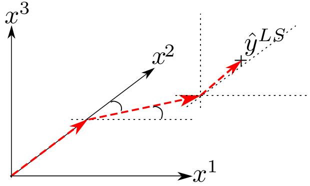
$$\arg\max |\mathbf{x}_j^\top (\mathbf{y} - X_{J^{(l)}} \mathbf{w}^{(l)}_{J^{(l)}})|$$

2. Add it to the set of relevant features

$$J^{(l+1)} \leftarrow J^{(l)} \cup \{j\}$$

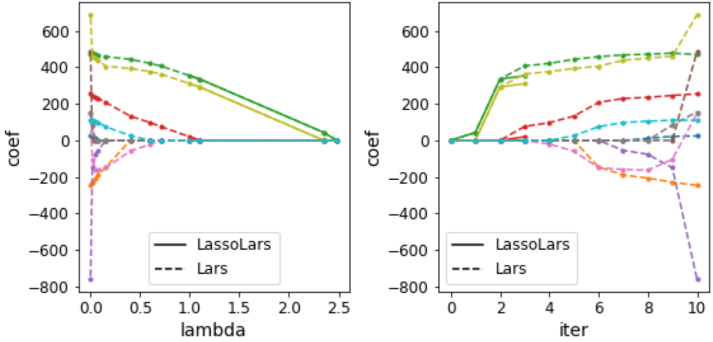
3. Update the coefficients $\mathbf{w}^{(l+1)}_{J^{(l+1)}}$ and $\lambda^{(l+1)}$

until stopping rule.



Diabetes example (sklearn)

```
from sklearn.linear_model import LassoLars, Lars
# Different variants of Lasso regularization path
lasso_lars = LassoLars()  # for full path: set alpha=0
lasso_lars.fit(Xtrain, ytrain)
lars = Lars()
lars.fit(Xtrain, ytrain)
```



Choosing λ

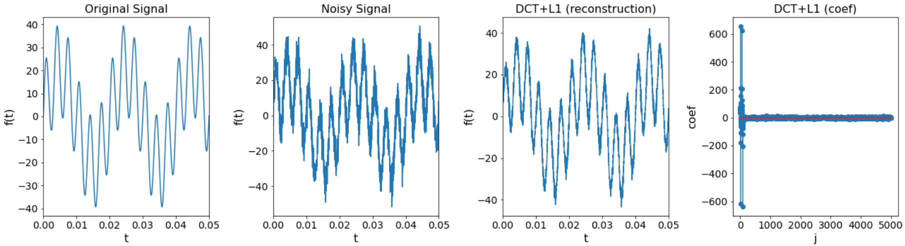
λ tunes the sparsity level:

- $\lambda = 0 \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = \hat{\mathbf{w}}^{ls}$ (all variables are selected)
- $\lambda \to \infty \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = 0$ (no variable is selected)

Choosing λ

λ tunes the sparsity level:

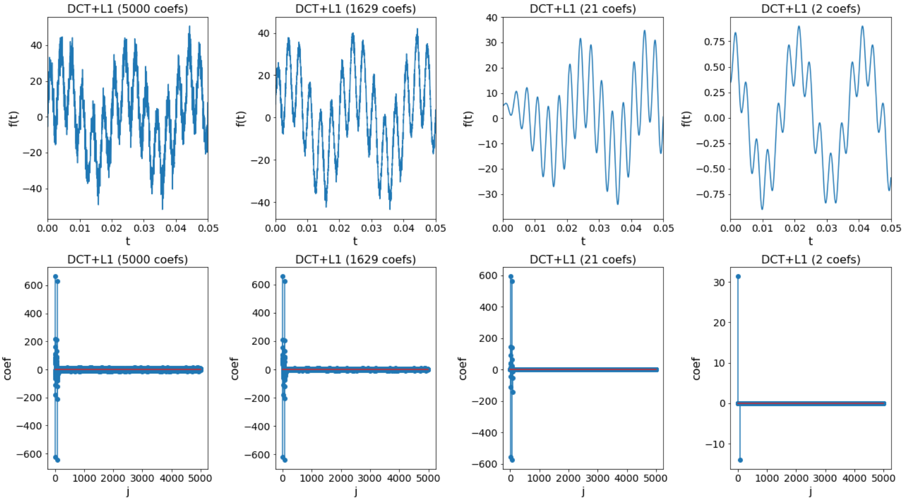
- $\lambda = 0 \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = \hat{\mathbf{w}}^{ls}$ (all variables are selected)
- $\lambda \to \infty \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = 0$ (no variable is selected)



Choosing λ

λ tunes the sparsity level:

- $\lambda = 0 \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = \hat{\mathbf{w}}^{ls}$ (all variables are selected)
- $\lambda \to \infty \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = 0$ (no variable is selected)



Choosing λ

λ tunes the sparsity level:

- $\lambda = 0 \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = \hat{\mathbf{w}}^{ls}$ (all variables are selected)
- $\lambda \to \infty \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = 0$ (no variable is selected)

Measuring the MSE on a different subset

- **Validation**: estimate $\hat{\mathbf{w}}_\lambda$ on train set I_t, find λ minimizing MSE on validation set I_v

$$\lambda^* = \arg\min_{\lambda \geq 0} \frac{1}{n_v} \sum_{i \in I_v} (y_i - \mathbf{x}_i^\top \hat{\mathbf{w}}_\lambda)^2$$

- **Cross-validation**: repeat on K different train/valid partitions

$$\lambda^* = \arg\min_{\lambda \geq 0} \frac{1}{K} \sum_{k=1}^{K} \frac{1}{n_v} \sum_{i \in I_v^{(k)}} (y_i - \mathbf{x}_i^\top \hat{\mathbf{w}}_\lambda)^2$$

Choosing λ

λ tunes the sparsity level:

- $\lambda = 0 \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = \hat{\mathbf{w}}^{ls}$ (all variables are selected)
- $\lambda \to \infty \quad \Rightarrow \quad \hat{\mathbf{w}}^{lasso} = 0$ (no variable is selected)

Information Criteria

Use the same set for both $\hat{\mathbf{w}}_\lambda$ and λ

- **Mallow's C_p/Akaike Information Criterion (AIC)**

$$\lambda^* = \arg\min_{\lambda \geq 0} \frac{1}{n_t} \sum_{i \in I_t} (y_i - \mathbf{x}_i^\top \hat{\mathbf{w}}_\lambda)^2 + 2k_\lambda \hat{\sigma}^2$$

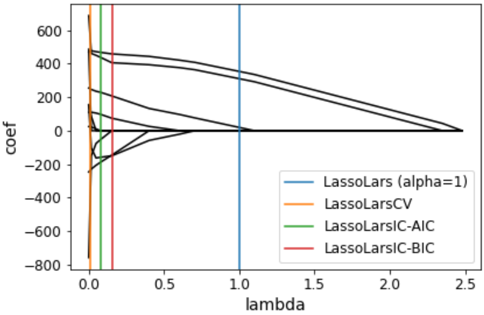
- **Bayes Information Criterion (BIC)**

$$\lambda^* = \arg\min_{\lambda \geq 0} \frac{1}{n_t} \sum_{i \in I_t} (y_i - \mathbf{x}_i^\top \hat{\mathbf{w}}_\lambda)^2 + \log(n_t) k_\lambda \hat{\sigma}^2$$

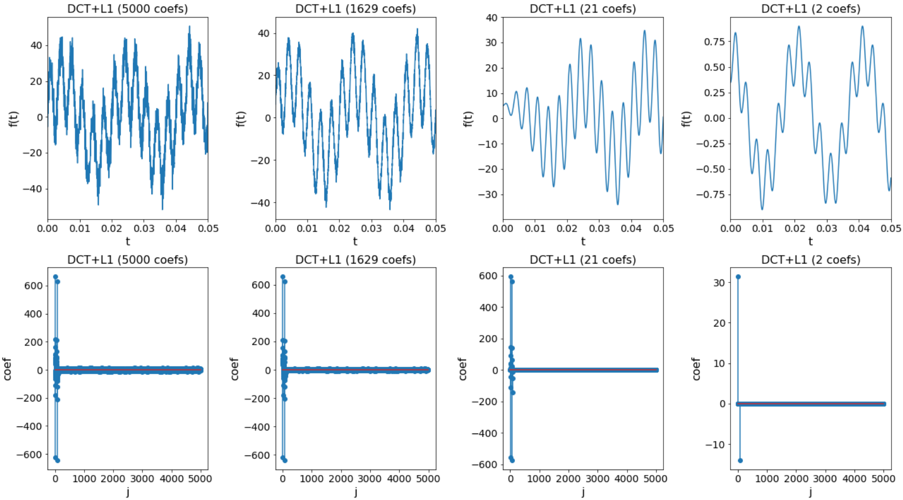
Note: true here because we consider Gaussian errors

Diabetes example (sklearn)

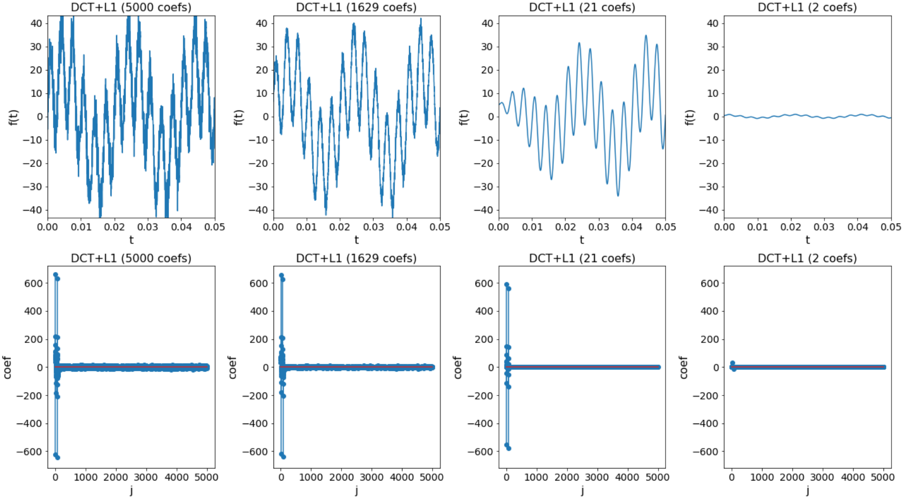
```
from sklearn.linear_model import LassoLars, LassoLarsCV, LassoLarsIC
lasso_lars = LassoLars()
lasso_lars.fit(Xtrain, ytrain)
lasso_larsCV = LassoLarsCV()  # with cross-validation
lasso_larsCV.fit(Xtrain, ytrain)
lasso_larsAIC = LassoLarsIC()  # with AIC or BIC
lasso_larsAIC.fit(Xtrain, ytrain)
lasso_larsBIC = LassoLarsIC(criterion='bic')
lasso_larsBIC.fit(Xtrain, ytrain)
```



Choosing λ



Choosing λ



Issue: Lasso's bias increases with λ

Sparse linear classification

The ℓ_1 penalty can also be applied to classification with other loss functions

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} L(y_i, \mathbf{x}_i\mathbf{w}) + \lambda\|\mathbf{w}\|_1$$

Binary classification

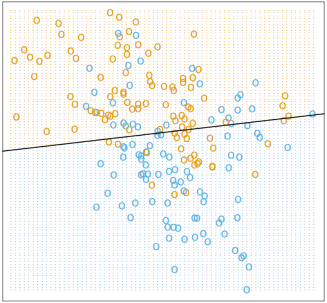
Target takes values: $y_i \in \{-1, 1\}$

- Logistic loss

$$L(y, \mathbf{xw}) = \log(1 + \exp(-y\mathbf{xw}))$$

- Squared hinge loss (SVM type)

$$L(y, \mathbf{xw}) = \max(0, 1 - y\mathbf{xw})^2$$



Source: [Hastie et al., 2008]

Sparse linear classification

The ℓ_1 penalty can also be applied to classification with other loss functions

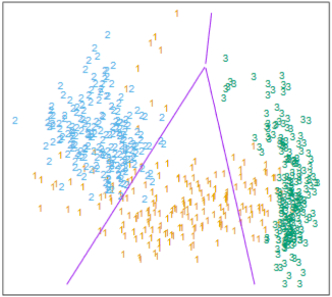
$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} L(y_i, \mathbf{x}_i\mathbf{w}) + \lambda\|\mathbf{w}\|_1$$

Multiclass classification

Target takes values: $y_i \in \{1, \dots, K\}$,
$K =$ no. classes
$\mathbf{W} \in \mathbb{R}^{d\times K}$

- Multinomial logistic regression

$$L(y, \mathbf{x}\mathbf{W}) = -\frac{1}{n}\sum_{i=1}^{n} \log \sum_{k=1}^{K} e^{\mathbf{x}_i^\top(\mathbf{w}^k - \mathbf{w}^{y_i})}$$



Source: [Hastie et al., 2008]

Breast cancer example (sklearn)

Data

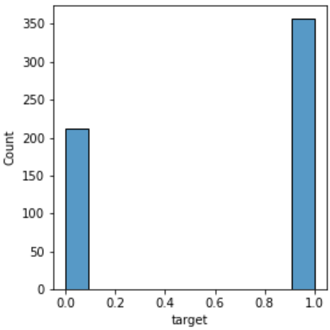
- $n = 569$ patients
- target = detect if mass is malign or benign
- $d = 30$ input variables: mean, std, and "worst" or largest of features (radius, area, texture, perimeter, ...) extracted from images

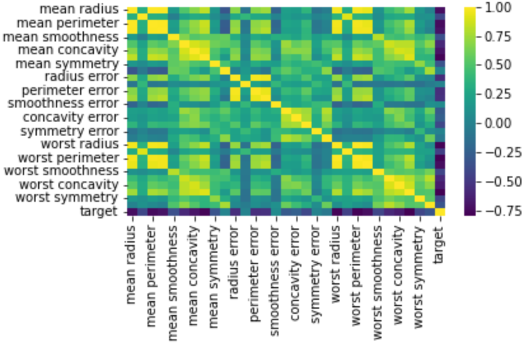
```
from sklearn.datasets import load_breast_cancer
data = load_breast_cancer()
X = data.data
y = data.target
features = data.feature_names
```

Breast cancer example (sklearn)

Data

- $n = 569$ patients
- target = detect if mass is malign or benign
- $d = 30$ input variables: mean, std, and "worst" or largest of features (radius, area, texture, perimeter, ...) extracted from images

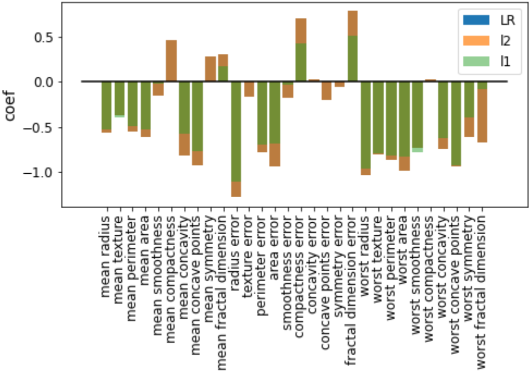




Breast cancer example (sklearn)

Estimation of malignity the mass

```
from sklearn.linear_model import LogisticRegression
l1 = LogisticRegression(penalty='l1', solver='saga', C=1)
l1.fit(Xtrain, ytrain)
ypred_l1 = l1.predict(Xtest)
```



L1 classification with Python

Scikit-Learn

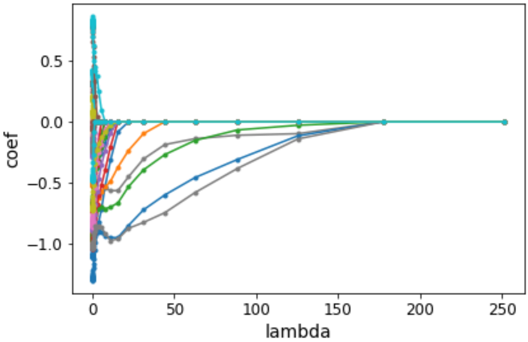
- `sklearn.linear_model.LogisticRegression`: SAGA/liblinear solver
- `sklearn.linear_model.SGDClassifier`: stochastic gradient descent
- `sklearn.svm.LinearSVC(penalty='l1', dual=False)`:
- ✗ regularization path algorithms do not work well here (not piecewise linear): need to compute on a grid

Other Python toolboxes

- `cyanure`: stochastic gradient descent
- `celer.LogisticRegression`, `celer.celer_path`: dual extrapolation

Regularization path

```
from sklearn.linear_model import LogisticRegression
alpha_grid = np.logspace(-5, 2.4, num=50)  # Define a grid
l1_grid = LogisticRegression(penalty='l1', solver='saga', C=1)
coefs_grid = np.zeros((len(l1.coef_[0]), len(alpha_grid)))
for i in range(len(alpha_grid)):
    l1_grid.set_params(C=1/alpha_grid[i])
    l1_grid.fit(Xtrain, ytrain)
    coefs_grid[:, i] = l1_grid.coef_[0]
```



Outline

Sparse penalties

Issue

Lasso is biased, especially for high values of λ (thus very sparse models)

- Can we find better penalties?

Sparse optimization problem

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ \frac{1}{n}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{d} h(|w_j|) \right\}$$

- $h : \mathbb{R}_+ \mapsto \mathbb{R}_+$ is monotously increasing
- $h(|w_j|)$ is nondifferentiable in 0 and assures sparsity
- $h(\cdot)$ does not need to be convex

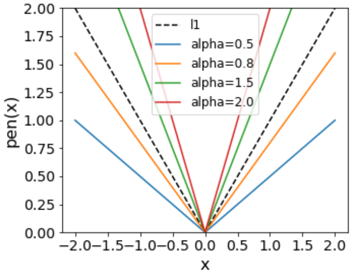
Adaptive Lasso

Convex for fixed weights [Zou, 2006]

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 + \lambda\sum_{j=1}^{d}\alpha_j|w_j|$$

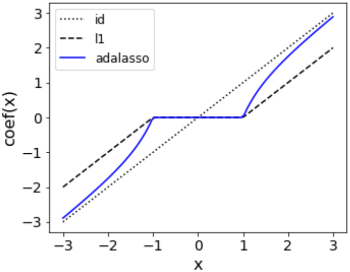
Penalty shape

$$h_j(x) = \alpha_j x \quad \text{with e.g. } \alpha_j = |\hat{w}_j^{ls}|^{-a}$$



Thresholding

$$\hat{w}_j^{adalasso} = \max\left(\hat{w}_j^{ls} - \frac{\lambda \operatorname{sgn}(\hat{w}_j^{ls})}{|\hat{w}_j^{ls}|^a}, 0\right)$$



Reweighted ℓ_1

Weights are not fixed: nonconvex [Candes et al., 2008]

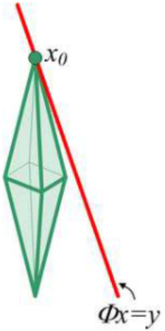
$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{d} \alpha_j |w_j|$$

Algorithm

Iterate between solving a weighted lasso and updating the weights

$$\begin{aligned}
\hat{\mathbf{w}}^{(l)} &= \min_{\mathbf{w}} \|\hat{\boldsymbol{\alpha}}^{(l)}\mathbf{w}\|_1 \quad \text{s.t.} \quad \mathbf{y}=\mathbf{X}\mathbf{w} \\
\hat{\boldsymbol{\alpha}}_j^{(l+1)} &= \frac{1}{|\hat{w}_j^{(l)}|+\epsilon}, \quad \epsilon>0
\end{aligned}$$

- ϵ ensures stability



Source: [Candes et al., 2008]

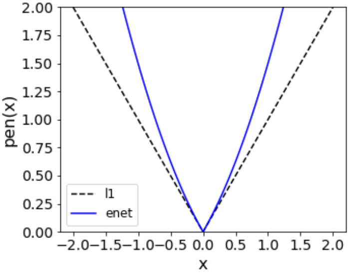
Elastic net ($\ell_1 - \ell_2$)

Strictly convex [Zou and Hastie, 2005]

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda_1 \sum_{j=1}^{d} |w_j| + \lambda_2 \sum_{j=1}^{d} w_j^2$$

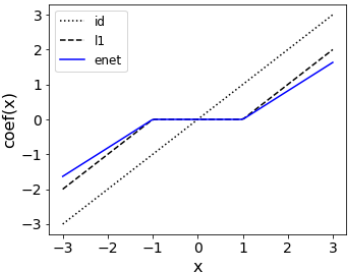
Penalty shape

$$h(x) = x + \rho x^2, \quad \rho = \lambda_2/\lambda_1$$



Thresholding

$$\hat{w}_j^{enet} = \frac{1}{\sqrt{1+\rho\lambda}}\hat{w}_j^{lasso}$$



Adaptive elastic net

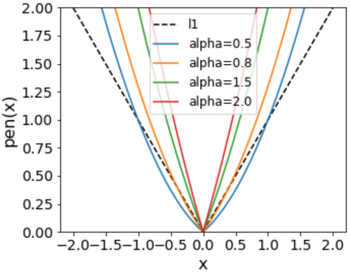
Convex for fixed weights, otherwise nonconvex [Zou and Zhang, 2009]

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda_1 \sum_{j=1}^{d} \alpha_j |w_j| + \lambda_2 \sum_{j=1}^{d} \alpha_j w_j^2$$

Penalty shape

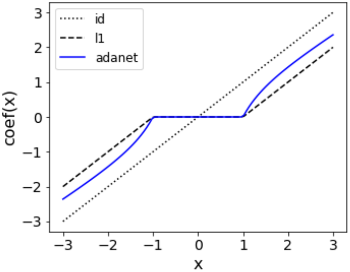
$$h_j(x) = \alpha_j x + \rho(\alpha_j x)^2$$

with e.g. $\alpha_j = |\hat{w}_j^{enet}|^{-a}$



Thresholding

$$\hat{w}_j^{adanet} = \frac{1}{\sqrt{1+\rho\lambda}} \hat{w}_j^{adalasso}$$



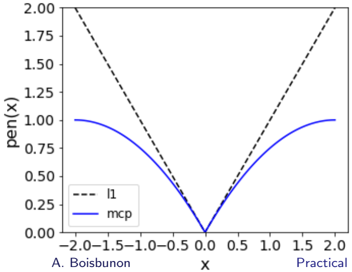
Minimax Concave Penalty (MCP)

Nonconvex penalty [Zhang, 2010], a.k.a semisoft [Gao and Bruce, 1995] or firm shrinkage for $\mathbf{X}$ orthogonal

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{d} h(|w_j|)$$

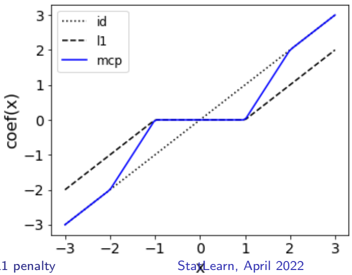
Penalty shape

$$h(x) = \min\left(x - \frac{x^2}{2a\lambda}, \frac{a\lambda}{2}\right),$$
with $a > 1$



Thresholding

$$\hat{w}_j^{mcp} = \frac{a}{a-1}\hat{w}_j^{lasso}\mathbb{1}_{\{|\hat{w}_j^{ls}|\leq a\lambda\}} + \hat{w}_j^{ls}\mathbb{1}_{\{|\hat{w}_j^{ls}|>a\lambda\}}$$



Smoothly Clipped Absolute Deviation (SCAD)

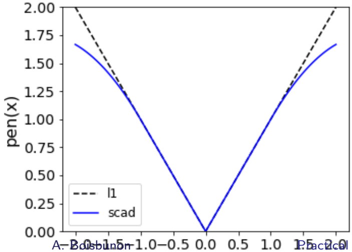
Nonconvex penalty [Fan and Li, 2001]

$$\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{d} h(|w_j|)$$

Penalty shape

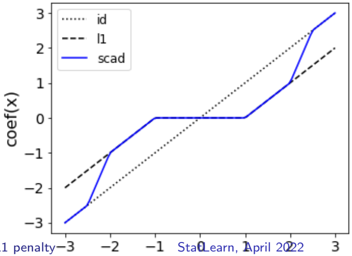
$$\begin{aligned} h(x) &= x \mathbb{1}_{\{x \leq \lambda\}} + \frac{(x-\lambda)^2}{2(a-1)\lambda} \mathbb{1}_{\{\lambda < x \leq a\lambda\}} \\ &+ \frac{(a+1)\lambda}{2} \mathbb{1}_{\{x > a\lambda\}} \end{aligned}$$

with $a > 2$



Thresholding

$$\hat{w}_j^{scad} = \begin{cases} \hat{w}_j^{lasso} & \text{if } |\hat{w}_j^{ls}| \leq 2\lambda \\ \frac{a}{a-2} \hat{w}_j^{lasso} & \text{if } 2\lambda < |\hat{w}_j^{ls}| \leq a\lambda \\ \hat{w}_j^{ls} & \text{if } |\hat{w}_j^{ls}| > a\lambda \end{cases}$$



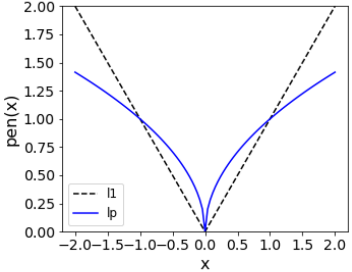
ℓ_p-norm, $0 < p < 1$

[Daubechies et al., 2010]

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{d} |w_j|^p, \qquad 0 < p < 1$$

Penalty shape

$$h(x) = x^p$$



Proximal operator

$$x = sign(x)\theta \quad \text{s.t. } \theta + p\theta^{p-1} = |x|$$

Ingrid Daubechies

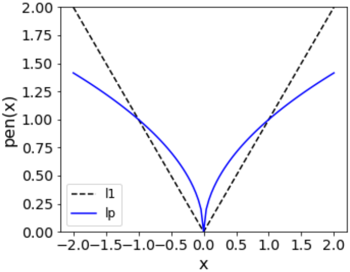
ℓ_p-norm, $0 < p < 1$

[Daubechies et al., 2010]

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{d} |w_j|^p, \qquad 0 < p < 1$$

Penalty shape

$$h(x) = x^p$$



Proximal operator

$$x = sign(x)\theta \quad \text{s.t. } \theta + p\theta^{p-1} = |x|$$

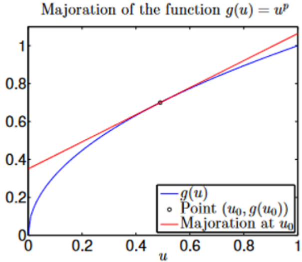


Illustration: [Courty et al., 2014]

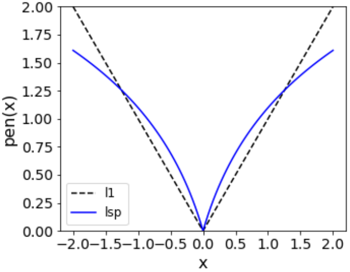
Log-sum penalty (LSP)

[Lobo et al., 2007, Candes et al., 2008]
Proximity operator : [Prater-Bennette et al., 2021]

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{d} \log\left(1+\frac{|w_j|}{\epsilon}\right)$$

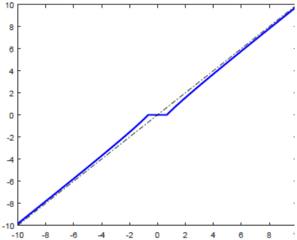
Penalty shape

$h(x) = \log(1 + x/\epsilon)$



Thresholding

$\sqrt{\lambda} \leq \epsilon$



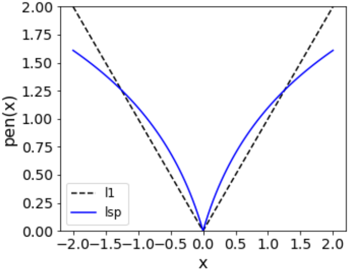
Log-sum penalty (LSP)

[Lobo et al., 2007, Candes et al., 2008]
Proximity operator : [Prater-Bennette et al., 2021]

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 + \lambda \sum_{j=1}^{d} \log\left(1+\frac{|w_j|}{\epsilon}\right)$$

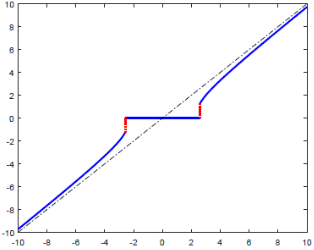
Penalty shape

$h(x) = \log(1 + x/\epsilon)$

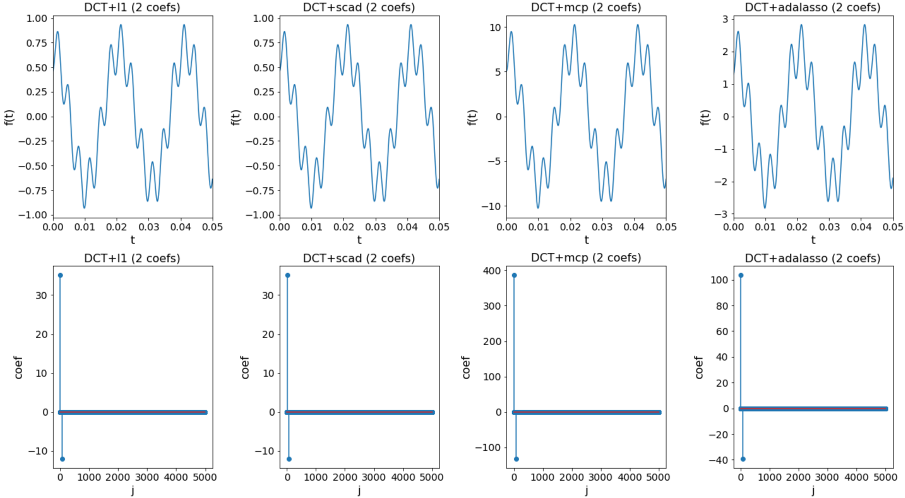


Thresholding

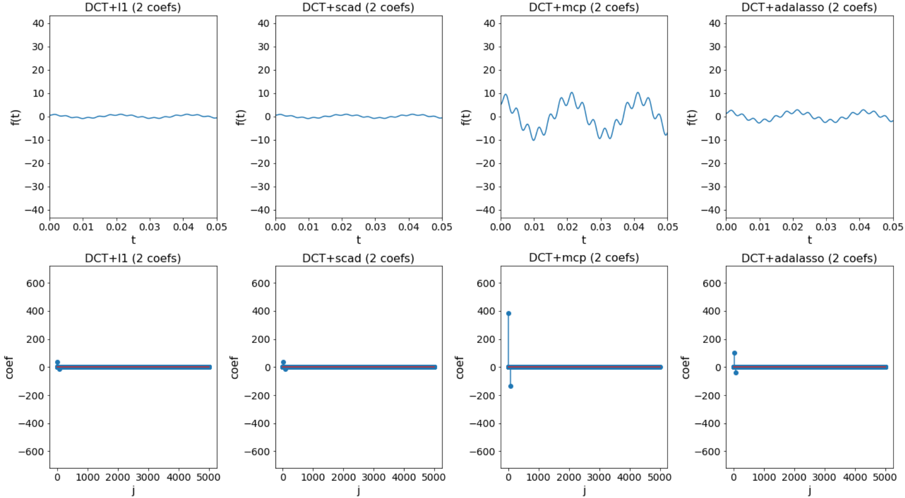
$\sqrt{\lambda} > \epsilon$



Choice of penalty



Choice of penalty



Other interesting sparse penalties

- Group-Lasso [Yuan and Lin, 2006]: promote sparsity on groups of variables

$$\Omega(\mathbf{w}) = \sum_{g=1}^{G} \|\mathbf{w}_{J_g}\|_{K_g} = \sum_{g=1}^{G} (\mathbf{w}_{J_g}^\top K_g \mathbf{w}_{J_g})^{1/2}$$

- Fused lasso [Tibshirani et al., 2005] / Total Variation [Acar and Vogel, 1994]: encourages piecewise constant signals

$$\Omega(\mathbf{w}) = \sum_{j \neq l} |\mathbf{w}_j - \mathbf{w}_l|$$

Optimality conditions

Sparsity for convex penalties

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{\|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda\Omega(\mathbf{w})\right\}, \qquad \Omega(\mathbf{w}) = \sum_{i=1}^{d} \omega(w_i) \text{ convex}$$

- Non-differentiability in $\mathbf{w} = \mathbf{0}$ with subgradient condition

$$\mathbf{0} \in \partial_{\mathbf{0}}\Omega(\mathbf{w}) = \{\mathbf{g} \in \mathbb{R}^n \setminus \Omega(\mathbf{w}) - \Omega(\mathbf{0}) \geq \mathbf{g}^\top(\mathbf{w} - \mathbf{0})\}$$

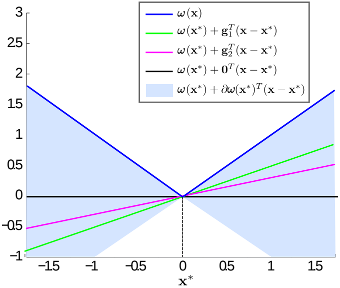
ℓ_1-penalty:

- Subgradient condition

$$\omega(w) = |w| \quad \Rightarrow \quad \partial_0\omega(w) = (-1, 1)$$

- Optimality condition

$$|\left(X^\top(\mathbf{y} - X\mathbf{w})\right)_j| \leq \lambda \quad \forall 1 \leq j \leq p$$



Optimality conditions

Sparsity for non-convex penalties

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ \frac{1}{n}\|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda\Omega(\mathbf{w}) \right\}, \qquad \Omega(\mathbf{w}) = \|\mathbf{w}\|_1 - h(\mathbf{w})$$

- Non-diff. in $\mathbf{w} = \mathbf{0}$ with Difference of Convex (DC) condition

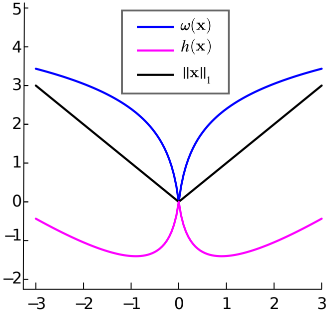
$$\partial_{\mathbf{0}} h(\mathbf{w}) \subset X_J^\top(\mathbf{y} - X\mathbf{w}) + \partial_{\mathbf{0}}\|\mathbf{w}\|_1(\mathbf{w})$$

Log sum penalty (LSP):

$$\omega(w) = \log(1 + |w|/\epsilon)$$

- Optimality condition

$$|\left(X^\top(\mathbf{y} - X\mathbf{w})\right)_j| \leq \lambda/\epsilon \quad \forall 1 \leq j \leq p$$



Sparse penalties with Python

Scikit-Learn

- Elastic net: regression, classification, multitask

Other Python toolboxes

- `cyanure`:
  - Elastic net
  - fused lasso
  - group-lasso
- `celer`:
  - group-lasso (reg)
  - weighted-ℓ_1 (reg/classif)
- `yagml`, `picasso`

Outline

Applications of the ℓ_1 penalty

Many types of problems can be rewritten as a sparse linear problem

Transforming the input variables into new features

- Additive models

$$f(\mathbf{x}) = \sum_{j=1}^{d} w_j \phi_j(x_j)$$

- Modeling interactions between variables

$$f(\mathbf{x}) = \sum_{j'=1}^{d'} w_{j'} \phi_{j'}(\mathbf{x})$$

Applications of the ℓ_1 penalty

Many types of problems can be rewritten as a sparse linear problem

Applying a (nonlinear) transformation to a linear model

- Generalized linear models

$$f(\mathbf{x}) = s\left(\sum_{j=1}^{d} w_j x_j\right)$$

  - Example: activation in a neural network layer

Applications of the ℓ_1 penalty

Many types of problems can be rewritten as a sparse linear problem

Applying sparsity in a different subspace

- Transporting the weights

$$\min_{\mathbf{w}} \|\mathbf{y} - H\mathbf{w}\|_2^2 + \lambda \|\phi^\top \mathbf{w}\|_1$$

Transforming the input variables into new features

Motivation

- linear models are highly interpretable BUT
- Not all data can be estimated/approached with linear regression/classification
- Feature construction is often done "manually" and relies on experts knowledge/ a priori

Random feature generation

- allows to automatically explore possible nonlinearities
- allows to explore features outside prior information

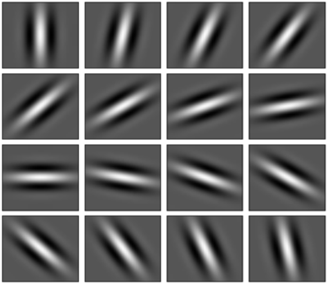
Infinite feature learning

Features generated through Gabor filters, wavelets, Fourier, kernels, etc, with continuous parameter $\theta_j, j = 1, \ldots, d$ [Rakotomamonjy et al., 2013]

$$\mathcal{F} = \{\{\phi_{\theta_j}(\cdot)\}_{j=1}^{d}\}$$

$$f(\mathbf{x}) = \sum_{j=1}^{d} w_j \phi_{\theta_j}(\mathbf{x}) + b_t$$

- Randomly draw d' filters/kernels with different parameters θ_j
- Apply Lasso (or other sparse penalty) to select the most relevant ones
- Repeat step 1 and 2 until convergence



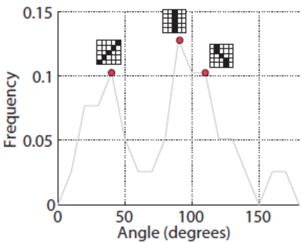
Infinite feature learning

Application: classification of pixels in remote sensing imagery [Tuia et al., 2014]

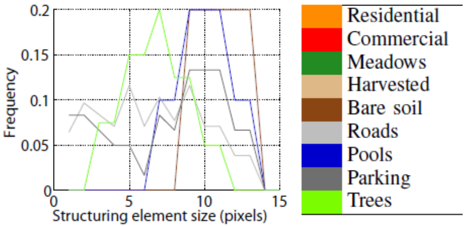
$$\min_{\varphi \in \mathcal{F}} \min_{\mathbf{w}} \frac{1}{n} L(y_i, \mathbf{w}^\top \mathbf{\Phi}_\varphi(\mathbf{x}_i)) + \lambda \|\mathbf{w}\|_1$$

- Classes are types of land cover (one-vs-rest)
- Features generated through Gabor filters
- Different directions and sizes are selected depending on the class

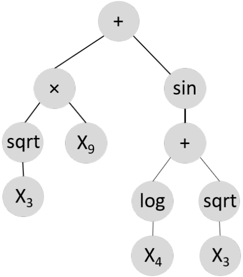
Directions of roads



Size of elements



Symbolic Regression

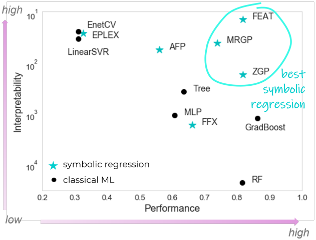


$$\hat{y} = \sqrt{X_3} \cdot X_9 + \sin(\log X_4 + \sqrt{X_3})$$

Search models with analytical form

- Ability to model interactions between variables
- Ability to construct new features
- Good tradeoff between interpretability and flexibility

Symbolic Regression and ℓ_1 penalty



Some SR algorithms randomly build features and then linearly combine them with a sparse penalty to select the most relevant ones

- features = bases in the form $\{op(X_i)\}_{i=1}^p$
  - FFX – Fast Function eXtractor [McConaghy, 2011]
- construct/evolve branches via genetic programming (GP)
  - MRGP – Multiple Regression GP [Arnaldo et al., 2014]
  - FEAT – Feature Engineering Automation Tool [La Cava et al., 2018]
  - ZGP – Zoetrope GP [Boisbunon et al., 2021]

Compressed Sensing with random features

Universal encoding [Candes and Tao, 2006]

Motivation

- Reduce data acquisition, e.g. for Magnetic Resonance Imaging, or for allowing satellite imaging with low transmission rates
- Ensure security in encoder-decoder

Principle

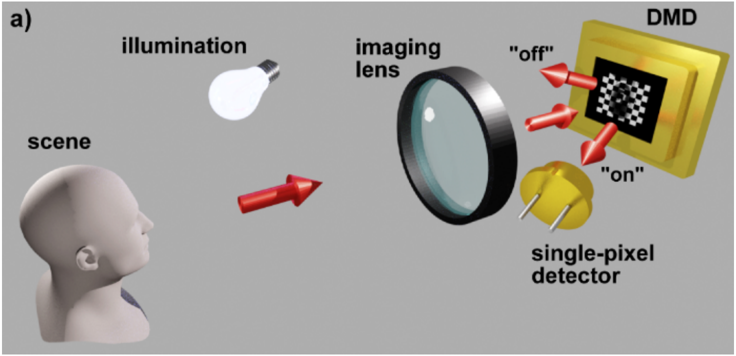
- Generate a collection of random vectors $\mathbf{x}_k$, e.g. random rows of Fourier or random Gaussian vectors
- Share the collection (e.g. send the seed)
- Encoder part: $y_k =< f, \mathbf{x}_k > +$ apply quantization
- Decoder part: Lasso

Compressed Sensing with random features

Single pixel imaging

[Gibson et al., 2020]

- cheaper sensors than traditional sensor arrays (e.g. for infrared)
- ability to detect weak light intensity changes

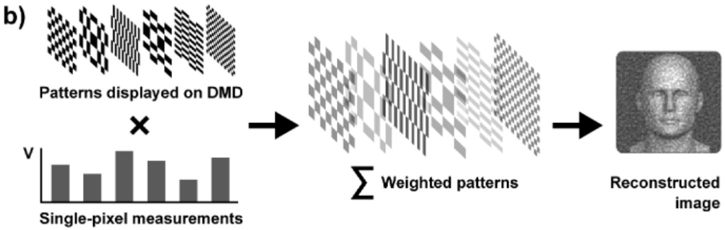


Compressed Sensing with random features

Single pixel imaging

[Gibson et al., 2020]

- cheaper sensors than traditional sensor arrays (e.g. for infrared)
- ability to detect weak light intensity changes



ℓ_1 penalty and Neural Networks

Sparsity in neural networks

- In neural networks, we apply nonlinear transformations to linear models
- The linear layers in a neural network can also be sparsified through ℓ_1
  - limit overfitting
  - concentrate the learning to the most important connexions between neurons

Using ℓ_1 penalty with Keras

```
from tensorflow.keras import layers
from tensorflow.keras import regularizers

tf.keras.regularizers.l1(l1=0.01)
tf.keras.regularizers.l1_l2(l1=0.01, l2=0.01)
```

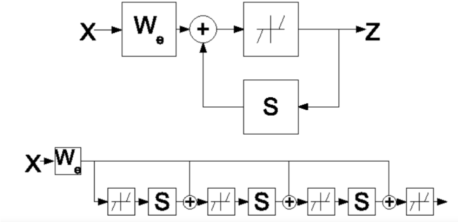
ℓ_1 penalty and Neural Networks

Real-time applications

- Sometimes computing the sparse weights may be too long for real time applications (e.g. in telecom)

Learning ISTA and CD [Gregor and LeCun, 2010]

- Run (F)ISTA/CD or your favorite algorithm on the dataset
- Train a neural network that predicts the results of the (F)ISTA/CD/etc



Signal/image processing

Denoising

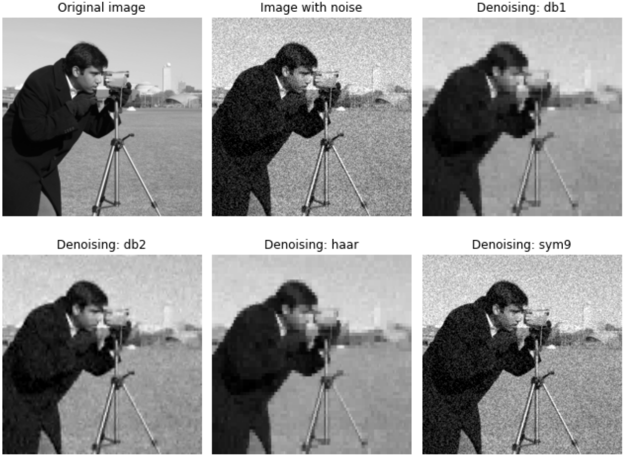
$$\mathbf{y} = \mathbf{x} + \varepsilon$$

- Aim: recover original signal $\mathbf{x} = \mathbf{D}\mathbf{w}$ from noisy observations $\mathbf{y}$
- $\mathbf{D}$ is a (fixed) dictionary
- Regular setting for Lasso

Source: R. Flamary

Denoising with wavelets in Python

```
from skimage.restoration import denoise_wavelet
denoised_img = denoise_wavelet(noisy_img, wavelet='db1',
    mode='soft', method='BayesShrink')
```

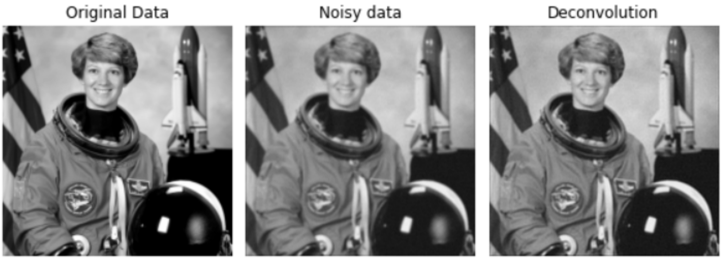


Signal/image processing

Reconstructing a signal

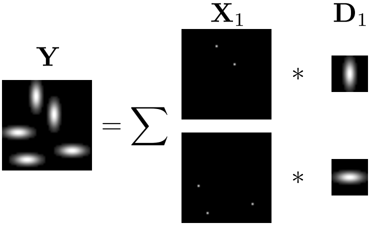
$$\mathbf{y} = \mathbf{H}\mathbf{x} + \varepsilon$$

- Aim: recover original signal $\mathbf{x} = \mathbf{D}\mathbf{w}$ from noisy observations $\mathbf{y}$
- $\mathbf{D}$ is a (fixed) dictionary
- $\mathbf{H}$ is a known linear operator, e.g. convolution or blur operator



Application: object detection

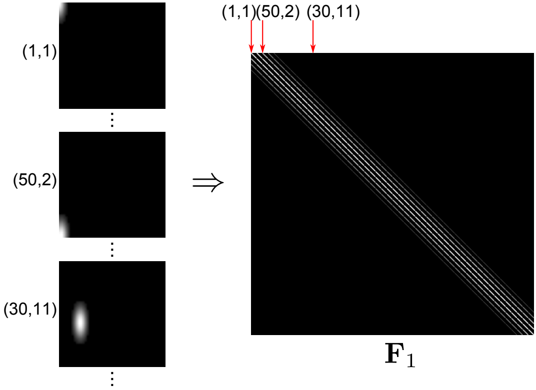
Detection of objects (boats) from satellite images with fixed dictionary [Boisbunon et al., 2014b]



- $\mathbf{Y}$: matrix of size $n \times m$ (image)
- $\mathbf{D}_k$, $k = 1, \ldots, K$: dictionary atoms
- $\mathbf{X}_k$: extremely sparse matrix of size $n \times m$
  - $x_{i,j,k} \neq 0 \Rightarrow$ position (i, j) activated for atom $\mathbf{D}_k$
- Reconstructed image: $\widetilde{\mathbf{Y}} = \sum_{k=1}^{K} \mathbf{X}_k * \mathbf{D}_k$

Application: object detection

Equivalence with linear problem



- Sum of convolutions: $\min_{\mathbf{X}\in\mathbb{R}_+^{n\times m\times K}} \left\{\|\mathbf{Y} - \sum_{k=1}^{K} \mathbf{X}_k * \mathbf{D}_k\|_F^2 + \lambda\Omega(\mathbf{X})\right\}$
- Linear problem: $\min_{\mathbf{x}\geq 0} \left\{\|\mathbf{y} - \mathbf{F}\mathbf{x}\|_2^2 + \lambda\Omega(\mathbf{x})\right\}$

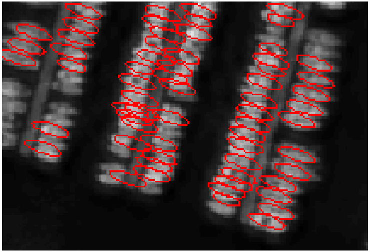
Application: object detection

Algorithm

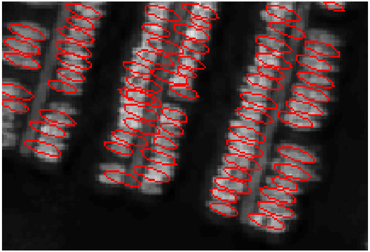
2D Sparse Optimization (2DSO) with active set strategy

- Find the atom most correlated with image $\Rightarrow$ shape + position
- Add the atom to active set
- Solve problem on a small active set[2] (verify optimality conditions) and apply transformation vector $\rightarrow$ matrix

ℓ_1-penalty



Log-sum penalty



[2] [Boisbunon et al., 2014a]

Dictionary learning

Aim: reconstruct $\mathbf{X} = \mathbf{DW}$ with both $\mathbf{D}$ and $\mathbf{W}$ unknown

Optimization problem

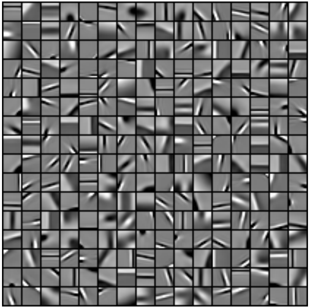
$$\min_{\mathbf{W}\in\mathbb{R}^{d\times m}, \mathbf{D}\in\mathbb{R}^{n\times d}, \|\mathbf{d}\|_j=1} \|\mathbf{Y} - \mathbf{DW}\|_F^2 + \lambda \sum_{j=1}^{d} \|\mathbf{w}_j\|_1$$

Algorithm

Start: $\mathbf{W}^{(0)} = \mathbf{0}$, $\mathbf{D}^{(0)}$

1. Extract patches from image
2. *Repeat*
   - Solve optimization problem for $\mathbf{W}^{(l+1)}$ with $\mathbf{D}^{(l)}$ fixed
   - Solve optimization problem for $\mathbf{D}^{(l+1)}$ with $\mathbf{W}^{(l+1)}$ fixed

until stopping rule.



Source: [Bach et al., 2011]

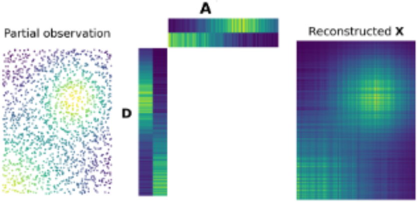
Application: inpainting

Dictionary learning with a mask

[Mairal et al., 2008]

$$\min_{\mathbf{W}\in\mathbb{R}^{d\times m},\mathbf{D}\in\mathbb{R}^{n\times d},\|\mathbf{d}\|_j=1} \|\mathbf{M}\odot(\mathbf{Y}-\mathbf{DW})\|_F^2+\lambda\sum_{j=1}^{d}\|\mathbf{w}_j\|_1$$

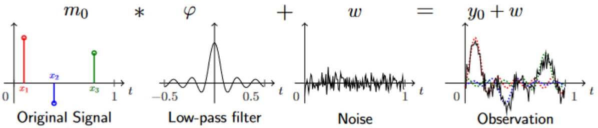
- $\mathbf{M}$ = binary mask of pixels we wish to recover
- $\odot$ is the pointwise multiplication



Beurling Lasso

Input = Blurred observations from measuring devices

- Beurling allows to explore Diracs in continuous space



Reconstruct a discrete measure from noisy samples [Azais et al., 2015]

$$\min_{\mu} \| \int \phi d\mu - \mathbf{y} \|^2 + \lambda \|\mu\|_{TV}$$

$$\min_{\mu} \|\phi\mu - \mathbf{y}\|^2 + \lambda |\mu|(\mathcal{X})$$

Image courtesy of S. Ladjal

Outline

Other interesting works with sparse penalties

Some applications of ℓ_1 I did not mention but are very interesting too:

- Selecting the k best singular value for matrix factorization, e.g. in recommandation systems
- Analysis of spike trains in the brain with Hawkes processes [Reynaud-Bouret et al., 2013]
- Sparse subspace clustering [Elhamifar and Vidal, 2013]
- Multitask learning
- Unbalanced optimal transport [Chapel et al., 2021]

and many more!

Take-home messages

- ℓ_1 penalty is everywhere!
- always try simple/classical approaches first (baseline)
- research is not always about new ideas, it can also be about how to adapt it in a new framework/context

Outline

References I

Acar, R. and Vogel, C. R. (1994).
Analysis of bounded variation penalty methods for ill-posed problems.
Inverse problems, 10(6):1217.

Arnaldo, I., Krawiec, K., and O'Reilly, U.-M. (2014).
Multiple regression genetic programming.
In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 879–886.

Azais, J.-M., De Castro, Y., and Gamboa, F. (2015).
Spike detection from inaccurate samplings.
Applied and Computational Harmonic Analysis, 38(2):177–195.

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. (2011).
Optimization for Machine Learning, chapter Convex optimization with sparsity-inducing norms, pages 19–54.
MIT Press.

Beck, A. and Teboulle, M. (2009).
A fast iterative shrinkage-thresholding algorithm for linear inverse problems.
SIAM Journal on Imaging Sciences, 2(1):183–202.

Boisbunon, A., Fanara, C., Grenet, I., Daeden, J., Vighi, A., and Schoenauer, M. (2021).
Zoetrope genetic programming for regression.
In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 776–784.

Boisbunon, A., Flamary, R., and Rakotomamonjy, A. (2014a).
Active set strategy for high-dimensional non-convex sparse optimization problems.
In *International Conference on Acoustic, Speech and Signal Processing (ICASSP)*.

Boisbunon, A., Flamary, R., Rakotomamonjy, A., Giros, A., and Zerubia, J. (2014b).
Large scale sparse optimization for object detection in high resolution images.
In *MLSP-24th IEEE Workshop on Machine Learning for Signal Processing*.

References II

Candes, E. J. and Tao, T. (2006).
Near-optimal signal recovery from random projections: Universal encoding strategies?
IEEE transactions on information theory, 52(12):5406–5425.

Candes, E. J., Wakin, M. B., and Boyd, S. P. (2008).
Enhancing sparsity by reweighted l1 minimization.
Journal of Fourier analysis and applications, 14(5):877–905.

Chapel, L., Flamary, R., Wu, H., Févotte, C., and Gasso, G. (2021).
Unbalanced optimal transport through non-negative penalized linear regression.
Advances in Neural Information Processing Systems, 34.

Chen, S. and Donoho, D. (1994).
Basis pursuit.
In *Proceedings of 1994 28th Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 41–44. IEEE.

Courty, N., Flamary, R., and Tuia, D. (2014).
Domain adaptation with regularized optimal transport.
In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 274–289. Springer.

Daubechies, I., DeVore, R., Fornasier, M., and Güntürk, C. S. (2010).
Iteratively reweighted least squares minimization for sparse recovery.
Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences, 63(1):1–38.

Donoho, D. L. (1995).
De-noising by soft-thresholding.
IEEE transactions on information theory, 41(3):613–627.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004).
Least angle regression.
The Annals of statistics, 32(2):407–499.

References III

Efroymson, M. A. (1960).
Multiple regression analysis.
Mathematical methods for digital computers, pages 191–203.

Elhamifar, E. and Vidal, R. (2013).
Sparse subspace clustering: Algorithm, theory, and applications.
IEEE transactions on pattern analysis and machine intelligence, 35(11):2765–2781.

Fan, J. and Li, R. (2001).
Variable selection via nonconcave penalized likelihood and its oracle properties.
Journal of the American statistical Association, 96(456):1348–1360.

Flamary, R. and Rakotomamonjy, A. (2012).
Decoding finger movements from ecog signals using switching linear models.
Frontiers in neuroscience, 6:29.

Gao, H.-Y. and Bruce, A. G. (1995).
Waveshrink and semisoft shrinkage.
In *StatSci Division of MathSoft, Inc.* Citeseer.

Gibson, G. M., Johnson, S. D., and Padgett, M. J. (2020).
Single-pixel imaging 12 years on: a review.
Optics Express, 28(19):28190–28208.

Gregor, K. and LeCun, Y. (2010).
Learning fast approximations of sparse coding.
In *Proceedings of the 27th international conference on international conference on machine learning*, pages 399–406.

Hastie, T., Tibshirani, R., and Friedman, J. (2008).
The Elements of Statistical Learning: Data Mining, Inference and Prediction (2nd Edition), volume 1.
Springer Series in Statistics.

References IV

Hoerl, A. E. and Kennard, R. W. (1970).
Ridge regression: Biased estimation for nonorthogonal problems.
Technometrics, 12(1):55–67.

La Cava, W., Singh, T. R., Taggart, J., Suri, S., and Moore, J. H. (2018).
Learning concise representations for regression by evolving networks of trees.
In *International Conference on Learning Representations*.

Lobo, M. S., Fazel, M., and Boyd, S. (2007).
Portfolio optimization with linear and fixed transaction costs.
Annals of Operations Research, 152(1):341–365.

Mairal, J., Ponce, J., Sapiro, G., Zisserman, A., and Bach, F. R. (2008).
Supervised dictionary learning.
In *Advances in Neural Information Processing Systems*, pages 1033–1040.

Mallat, S. and Zhang, Z. (1993).
Matching pursuits with time-frequency dictionaries.
IEEE Trans. Signal Process., 41:3397–3415.

McConaghy, T. (2011).
Ffx: Fast, scalable, deterministic symbolic regression technology.
In *Genetic Programming Theory and Practice IX*, pages 235–260. Springer.

Prater-Bennette, A., Shen, L., and Tripp, E. E. (2021).
The proximity operator of the log-sum penalty.
arXiv preprint arXiv:2103.02681.

Rakotomamonjy, A., Flamary, R., and Yger, F. (2013).
Learning with infinitely many features.
Machine Learning, 91(1):43–66.

References V

Reynaud-Bouret, P., Rivoirard, V., and Tuleau-Malot, C. (2013).
Inference of functional connectivity in neurosciences via hawkes processes.
In *2013 IEEE global conference on signal and information processing*, pages 317–320. IEEE.

Santosa, F. and Symes, W. W. (1986).
Linear inversion of band-limited reflection seismograms.
SIAM Journal on Scientific and Statistical Computing, 7(4):1307–1330.

Tangermann, M., Müller, K.-R., Aertsen, A., Birbaumer, N., Braun, C., Brunner, C., Leeb, R., Mehring, C., Miller, K. J., Mueller-Putz, G., et al. (2012).
Review of the bci competition iv.
Frontiers in neuroscience, page 55.

Tarique, M. (2016).
Performances of orthogonal wavelet division multiplex (owdm) system under awgn, rayleigh, and ricean channel conditions.
International Journal of Computer Networks & Communications (IJCNC), 8(3).

Tibshirani, R. (1996).
Regression shrinkage and selection via the lasso.
Journal of the Royal Statistical Society. Series B (Methodological), 58(1):267–288.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005).
Sparsity and smoothness via the fused lasso.
Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67(1):91–108.

Tuia, D., Volpi, M., Dalla Mura, M., Rakotomamonjy, A., and Flamary, R. (2014).
Automatic feature learning for spatio-spectral image classification with sparse svm.
IEEE Transactions on Geoscience and Remote Sensing, 52(10):6062–6074.

References VI

Yuan, M. and Lin, Y. (2006).
Model selection and estimation in regression with grouped variables.
Journal of the Royal Statistical Society: Series B (Statistical Methodology), 68(1):49–67.

Zhang, C. (2010).
Nearly unbiased variable selection under minimax concave penalty.
Annals of Statistics, 38(2):894–942.

Zou, H. (2006).
The adaptive lasso and its oracle properties.
Journal of the American statistical association, 101(476):1418–1429.

Zou, H. and Hastie, T. (2005).
Regularization and variable selection via the elastic net.
Journal of the royal statistical society: series B (statistical methodology), 67(2):301–320.

Zou, H. and Zhang, H. H. (2009).
On the adaptive elastic-net with a diverging number of parameters.
Annals of statistics, 37(4):1733.

Pursuit algorithms

X = overcomplete dictionary of p atoms (wavelets, Gabor, Fourier), $p > n$

Matching pursuit [Mallat and Zhang, 1993]

$$\min_{\mathbf{w}} \|\mathbf{y} - X\mathbf{w}\|_2^2 \qquad s.t. \quad \|\mathbf{w}\|_0 \leq p^*,$$

regularization path where the coefficients are updated with $w_j = \mathbf{x}_j^\top \mathbf{y}$

Basis pursuit

$$\min_{\mathbf{w}} \|\mathbf{w}\|_1 \qquad s.t. \quad \mathbf{y} = X\mathbf{w}$$

Basis pursuit denoising [Chen and Donoho, 1994]

$$\min_{\mathbf{w}} \|\mathbf{w}\|_1 \qquad s.t. \quad \|\mathbf{y} - X\mathbf{w}\|_2^2 \leq t_\lambda$$

with $t_\lambda = \sigma\sqrt{2\log(p)}$ Jump to reg path slide