Inverse problems and sparse models (4/6)

Rémi Gribonval
INRIA Rennes - Bretagne Atlantique, France
remi.gribonval@inria.fr

Reminder of last sessions

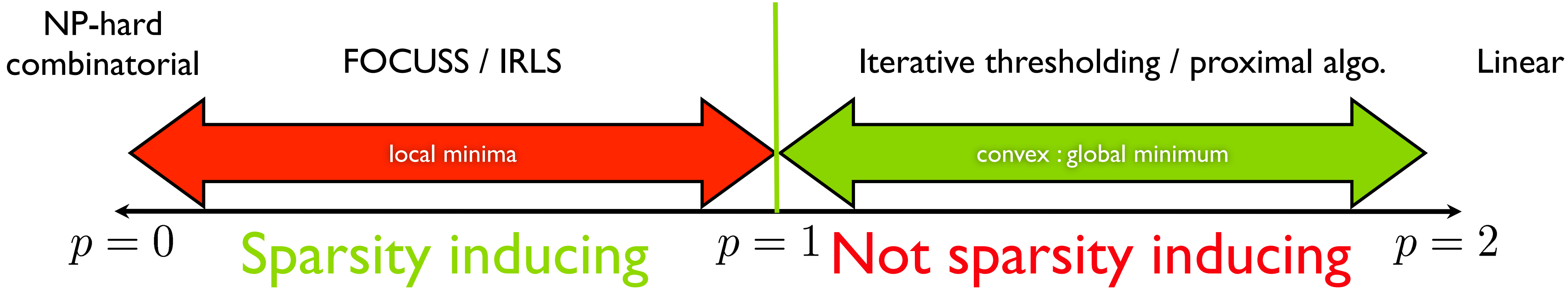
- **Introduction**
  - ✓ sparsity & data compression
  - ✓ inverse problems in signal and image processing
    - ✦ image deblurring, image inpainting,
    - ✦ channel equalization, signal separation,
    - ✦ tomography, MRI
  - ✓ sparsity & under-determined inverse problems
    - ✦ relation to subset selection problem
- **Pursuit Algorithms**
  - ✓ Greedy algorithms: Matching Pursuit & al
  - ✓ L1 minimization principles
  - ✓ L1 minimization algorithms
  - ✓ Complexity of Pursuit Algorithms

Summary

	Global optimization	Iterative greedy algorithms
Principle	$\min_x \frac{1}{2}\|\mathbf{A}x - \mathbf{b}\|_2^2 + \lambda \|x\|_p^p$	iterative decomposition $\mathbf{r}_i = \mathbf{b} - \mathbf{A}x_i$ • select new components • update residual
Tuning quality/sparsity	regularization parameter λ	stopping criterion (nb of iterations, error level, ...) $\|x_i\|_0 \geq k$ $\|\mathbf{r}_i\| \leq \epsilon$
Variants	• choice of sparsity measure p • **optimization algorithm** • initialization	•selection criterion (weak, stagewise ...) •update strategy (orthogonal ...)

Global Optimization : from Principles to Algorithms

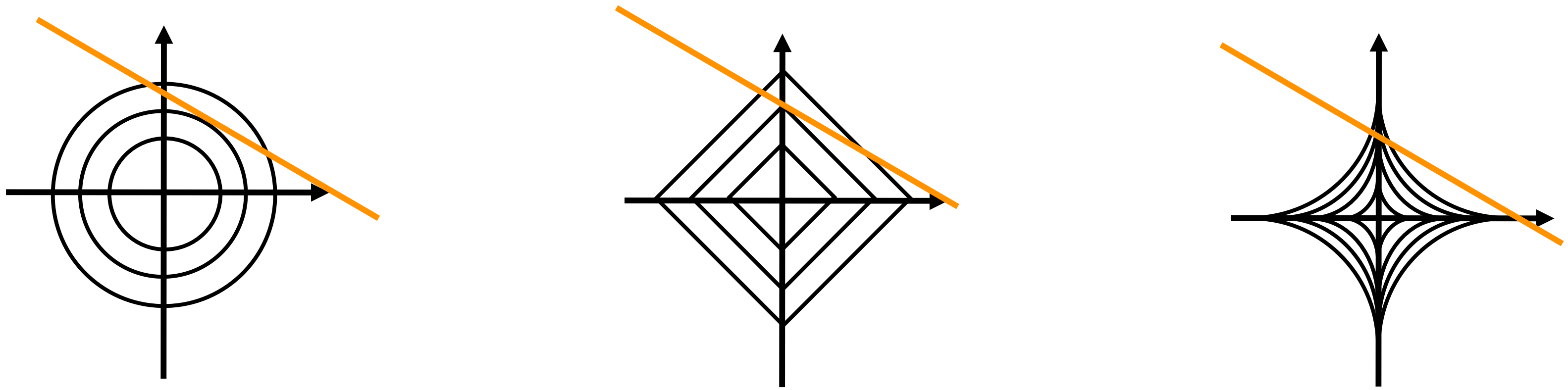
- Optimization principle $$\min_x \frac{1}{2}\|\mathbf{A}x - \mathbf{b}\|_2^2 + \lambda\|x\|_p^p$$
  - ✓ Sparse representation $\lambda \to 0$ $\quad \mathbf{A}x = \mathbf{b}$
  - ✓ Sparse approximation $\lambda > 0$ $\quad \mathbf{A}x \approx \mathbf{b}$



Lasso [Tibshirani 1996], *Basis Pursuit (Denoising)* [Chen, Donoho & Saunders, 1999]
Linear/Quadratic programming (interior point, etc.)
Homotopy method [Osborne 2000] / Least Angle Regression [Efron &al 2002]
Iterative / proximal algorithms [Daubechies, de Frise, de Mol 2004, Combettes & Pesquet 2008, Beck & Teboulle 2009 ...]

L_p “norms” level sets

- Strictly convex when $p>1$
- Convex $p=1$
- Nonconvex $p<1$



Observation: *the minimizer is sparse when p<=1*

$\{x \text{ s.t.} \mathbf{b} = \mathbf{A}x\}$

L1 *induces* sparsity (1)

- Real-valued case
  - ✓ **A** = an *m* x *N* real-valued matrix, where m < N
  - ✓ **b** = an *m*-dimensional real-valued vector
  - ✓ *X* = set of all minimum L*1* norm solutions to $\mathbf{A}x = \mathbf{b}$

$$\tilde{x} \in X \Leftrightarrow \|\tilde{x}\|_1 = \min_x \|x\|_1 \text{ s.t. } \mathbf{A}x = \mathbf{b}$$

- **Fact 1**: *X* is convex and contains a “sparse” solution

$$\exists x_0 \in X \subset \mathbb{R}^N, \|x_0\|_0 \leq m < N$$

Proof ? Exercice!

Proof ? Exercice!

- **Convexity of the set of solutions *X*:**
  - ✓ let $x, x' \in X,\ 0 \leq \theta \leq 1$
  - ✓ convexity of constraint

$$\mathbf{A}x = \mathbf{A}x' = \mathbf{A}(\theta x + (1-\theta)x') = \mathbf{b}$$

  - ✓ by definition $\|x\|_1 = \|x'\|_1 = \min \|\tilde{x}\|_1 \text{ s.t. } \mathbf{A}\tilde{x} = \mathbf{b}$
  - ✓ convexity of objective function

$$\|(\theta x + (1-\theta)x')\|_1 \leq \theta\|x\|_1 + (1-\theta)\|x'\|_1 = \|x\|_1$$

  - ✓ hence

$$\theta x + (1-\theta)x' \in X$$

Proof? Exercice!

- **Existence of a sparse solution**
  - ✓ let x satisfy $\mathbf{A}x = \mathbf{b}$ with $\boxed{\|x\|_0 \geq m+1}$
    - ✦ support $I := \text{supp}(x) := \{i, x_i \neq 0\}$ $\quad \ell \geq m+1$
    - ✦ sub-matrix $\underset{m}{} \overset{N}{\boxed{\mathbf{A}}} \longrightarrow \underset{m}{} \boxed{\mathbf{A}_I}$
  - ✓ existence of nontrivial null space vector $\mathbf{A}_I z = 0$
  - ✓ other solution $x' = x + \epsilon z$
  - ✓ for small ϵ

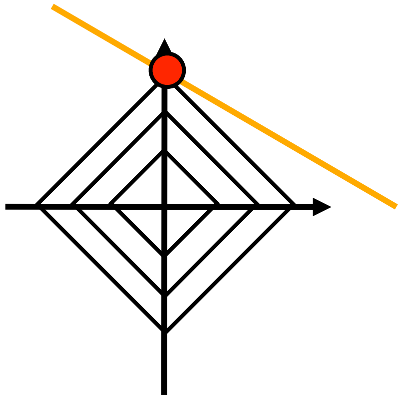
$$\|x+\epsilon z\|_1 = \sum_{i \in I} |x_i + \epsilon z_i|$$

$$= \sum_{i \in I} \text{sign}(x_i)(x_i + \epsilon z_i) = \|x\|_1 + \epsilon \sum_{i \in I} \text{sign}(x_i) z_i$$

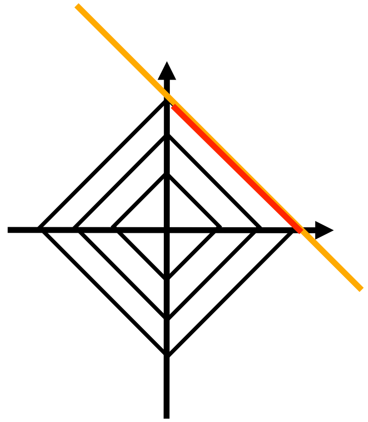
cost function
is not minimum

Convexity of the set of minimizers

- Unique solution



- Non unique solution



L1 *induces* sparsity (2)

- Real-valued case
  - ✓ **A** = an *m* x *N* real-valued matrix, m<N
  - ✓ **b** = an *m*-dimensional real-valued vector
  - ✓ *X* = set of al solutions to regularization problem

$$\mathcal{L}(x) := \frac{1}{2}\|\mathbf{A}x - \mathbf{b}\|_2^2 + \lambda\|x\|_1$$

$$\tilde{x} \in X \Leftrightarrow \mathcal{L}(\tilde{x}) = \min_x \mathcal{L}(x)$$

- **Fact 2**: *X* is a convex set and contains a "sparse" solution

$$\exists x_0 \in X \subset \mathbb{R}^N, \|x_0\|_0 \leq m < N$$

Proof ? Exercice at home!

L*1* induces sparsity

- A word of caution: this **does not hold true in the complex-valued case**
- Counter example: there is a construction where
  - ✓ **A** = a *2* x *3* complex-valued matrix
  - ✓ **b** = a *2*-dimensional complex-valued vector
  - ✓ the minimum L1 norm solution is unique and has 3 nonzero components

    [E.Vincent, Complex Nonconvex Optimization l_p norm minimization for underdetermined source separation, Proc. ICA 2007.]

Convex Pursuit Algorithms

Sparse optimization *principles*

L1 minimization *induces* sparsity

Algorithms for L1 minimization

Algorithms for L1: Linear Programming

- L1 minimization problem of size *m* x *N*

Basis Pursuit (BP)
LASSO

$$\min_x \|x\|_1, \ \text{s.t.} \ \mathbf{A}x = \mathbf{b}$$

- Equivalent **linear program** of size *m* x 2*N*

$$\min_{z \geq 0} \mathbf{c}^T z, \ \text{s.t.} \ [\mathbf{A}, -\mathbf{A}]z = \mathbf{b}$$

$$\mathbf{c} = (c_i), \ c_i = 1, \forall i$$

L1 regularization: Quadratic Programming

- L1 minimization problem of size *m* x *N*

Basis Pursuit Denoising (BPDN)

$$\min_x \frac{1}{2}\|\mathbf{b} - \mathbf{A}x\|_2^2 + \lambda\|x\|_1$$

- Equivalent **quadratic program** of size *m* x 2*N*

$$\min_{z \geq 0} \frac{1}{2}\|\mathbf{b} - [\mathbf{A}, -\mathbf{A}]z\|_2^2 + \mathbf{c}^T z$$

$$\mathbf{c} = (c_i), \; c_i = 1, \forall i$$

Generic approaches *vs* specific algorithms

- Many algorithms for linear / quadratic programming
- Matlab Optimization Toolbox: `linprog` /`qp`
- But ...
  - ✓ The problem size is “doubled”
  - ✓ Specific structures of the matrix **A** can help solve BP and BPDN more efficiently
  - ✓ More efficient toolboxes have been developed
- CVX package (Michael Grant & Stephen Boyd):
  - ✓ http://www.stanford.edu/~boyd/cvx/

Example of CVX program

- Matlab code

```
m=100;
N=1000;
A = randn(m,N);
b = randn(m,1);
cvx_begin
        variable x(N)
        minimize ( norm(x,1) )
        subject to
          A*x = b
cvx_end
```

- How is it implemented? SDPT3 or SeDuMi packages ...

Convex Pursuit Algorithms

Sparse optimization *principles*

L1 minimization *induces* sparsity

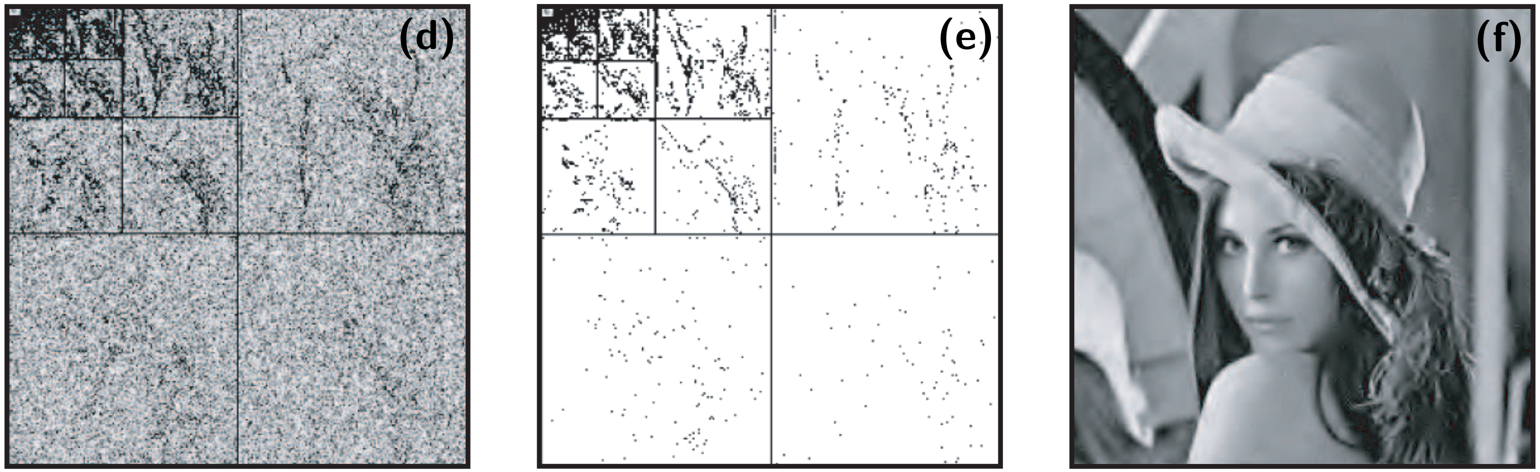
Algorithms for L1 minimization

Do it yourself!

Wavelet Domain Denoising

Courtesy: G. Peyré, Ceremade, Université Paris 9 Dauphine

Original Noisy . Smoothed .



Coefficients Thresholded coefficients Denoised

Denoising problem

- Original NxN image is corrupted by noise

$$\mathbf{b} = \mathbf{y} + \mathbf{e}$$

- Original image is sparse in wavelet basis

$$\mathbf{b} = \mathbf{\Phi} x + \mathbf{e} \qquad x = \mathbf{\Phi}^T \mathbf{y} \qquad \|x\|_0 \ll N \times N$$

- Wavelet basis is an **orthonormal basis**

$$\mathbf{\Phi}\mathbf{\Phi}^T = \mathbf{Id} \qquad \mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{Id}$$

- Idealized denoising problem:

$$\hat{x} := \arg\min_x \frac{1}{2}\|\mathbf{b} - \mathbf{\Phi} x\|_2^2 + \lambda \|x\|_0$$

Exploiting the fact that **A** is orthonormal

- Assumption : *m*=*N* and **A** is *orthonormal*

$$\mathbf{A}^T\mathbf{A} = \mathbf{A}\mathbf{A}^T = \mathbf{Id}_N$$
$$\|\mathbf{b} - \mathbf{A}x\|_2^2 = \|\mathbf{A}^T\mathbf{b} - x\|_2^2$$

- Expression of BPDN criterion to be minimized

$$\sum_n \frac{1}{2}\left((\mathbf{A}^T\mathbf{b})_n - x_n\right)^2 + \lambda|x_n|^p$$

- Minimization can be done coordinate-wise

$$\min_{x_n} \frac{1}{2}\left(c_n - x_n\right)^2 + \lambda|x_n|^p$$

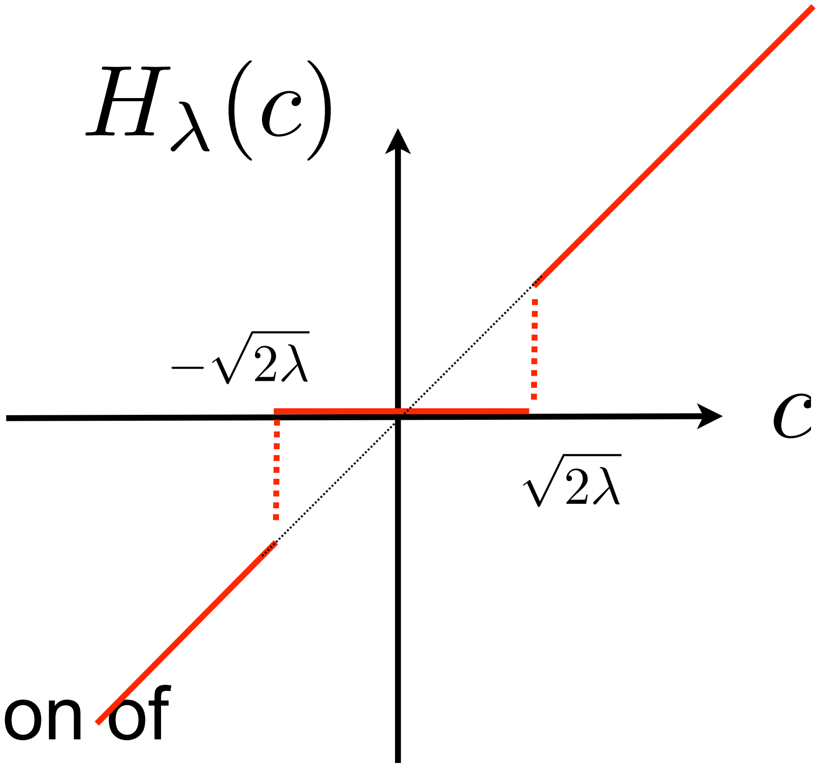
Exercice

- Given c, find the solution to

$$\min_{x_n} \frac{1}{2}\left(c_n - x_n\right)^2 + \lambda |x_n|^p$$

  - ✓ p=0
  - ✓ p=1

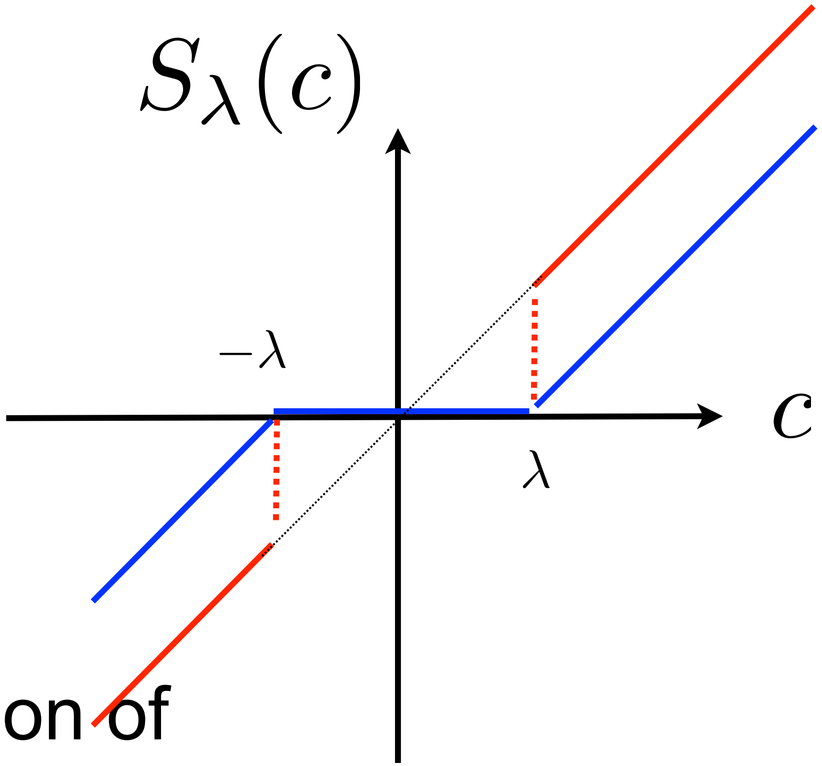
Hard-thresholding (p=0)



- Solution of

$$\min_x \frac{1}{2}(c-x)^2 + \lambda \cdot |x|^0$$

Soft-thresholding (p=1)



- Solution of

$$\min_x \frac{1}{2}(c-x)^2 + \lambda \cdot |x|$$

Matlab code ?

- Soft thresholding
- `@softthresh(c,lambda)(sign(c).*max(abs(c)-lambda,0))`
- `x = softthresh(c,lambda);`

- Hard-thresholding
- `@hardthresh(c,lambda)(c.*(abs(c)>=sqrt(2*lambda)))`
- `x = hardthresh(c,lambda);`
-

Iterative thresholding

- Definition: **proximity operator**

$$\Theta^p_\lambda(c) = \arg\min_x \frac{1}{2}(x-c)^2 + \lambda |x|^p$$

- Goal = compute

$$\arg\min_x \frac{1}{2}\|\mathbf{A}x - \mathbf{b}\|_2^2 + \lambda \|x\|_p^p$$

- **Iterative algorithm:**
  - ✓ gradient descent on fidelity term

$$x^{(i+1/2)} := x^{(i)} + \alpha^{(i)} \mathbf{A}^T (\mathbf{b} - \mathbf{A}x^{(i)})$$

  - ✓ thresholding

$$x^{(i+1)} := \Theta^p_{\lambda^{(i)}}(x^{(i+1/2)})$$

Iterative Thresholding

- **Theorem :** [Daubechies, de Mol, Defrise 2004, Combettes & Pesquet 2008]
  - ✓ consider the iterates $x^{(i+1)} = f(x^{(i)})$ defined by the thresholding function, with $p \geq 1$

$$f(x) = \Theta^p_{\alpha\lambda}(x + \alpha\mathbf{A}^T(\mathbf{b} - \mathbf{A}x))$$

  - ✓ assume that $\forall x,\ \|\mathbf{A}x\|_2^2 \leq c\|x\|_2^2$ and $\alpha < 2/c$
  - ✓ then, the iterates converge strongly to a limit $x^\star$

$$\|x^{(i)} - x^\star\|_2 \rightarrow_{i\rightarrow\infty} 0$$

  - ✓ the limit $x^\star$ is a global minimum of $\frac{1}{2}\|\mathbf{A}x - \mathbf{b}\|_2^2 + \lambda\|x\|_p^p$
  - ✓ if *p*>1, or if **A** is invertible, $x^\star$ is the *unique* minimum

Iterative Thresholding: convex penalties

- Strong convergence to global minimum
- Accelerated convergence:
  - ✓ Nesterov schemes
  - ✓ see e.g. Beck & Teboulle 2009;
- Many variants of iterative thresholding
  - ✓ depends on properties of penalty terms
    - ✦ smoothness
    - ✦ strong convexity
    - ✦ etc.

Iterative Thresholding: nonconvex penalties

- Example: Iterative Hard Thresholding for L0
  - ✓ keep components above threshold
  - ✓ *or rather* **keep *k* largest components**
    - ✦ [IHT: Blumensath & Davies 2009]
- More generally, with *nonconvex* cost functions
  - ✓ Possible 'spurious' local minima
  - ✓ Convergence: fixed point, under certain assumptions
  - ✓ Limit = global min: under certain assumptions (RIP)
- Pruning strategies:
  - ✓ ex: keep 2k components, project, keep k components
    - ✦ ex: CoSAMP [Needell &Tropp 2008], ALPS [Cevher 2011], ...

Code for Iterative Thresholding?

- Proximal operator (or *prox*)

$$\mathrm{prox}_f(\mathbf{c}) := \arg\min_x \left\{ \frac{1}{2}\|x - \mathbf{c}\|_2^2 + f(x) \right\}$$

- Prox of the absolute value = soft-thresholding

```
@prox(c,lambda)(sign(c).*max(abs(c)-lambda,0))
```

- Iterative thresholding with general prox

```
function xhat = iterate_thresh(b,A,prox,step,niter)
  xhat = 0;
  for i=1:niter
    xhat =prox(xhat+ step * A'*(b-A*xhat))
  end
```

Exercice at home

- Write Matlab code for MP
- Idem for OMP
- Idem for L1 minimization with CVX
- Idem for Iterative Hard Thresholding

Exercice: Matlab code for (O)MP

- Full clean code would include some checking (column normalization, dimension checking, etc.)

```
function [x res] = mp(b,A,k)
% explain here what the function should do
....
end

function [x res] = omp(b,A,k)
% explain here what the function should do
....
end
```