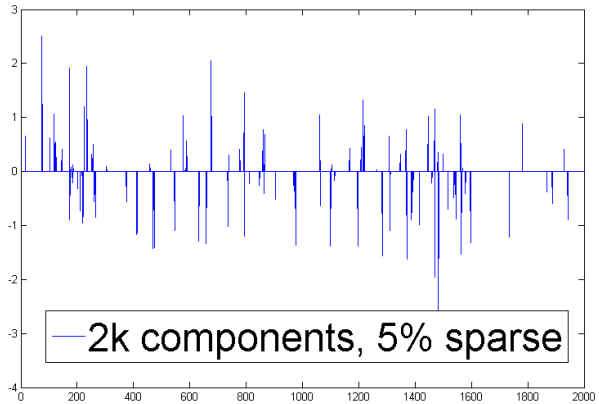
Optimization for Sparse Solutions, A Tutorial

Wotao Yin (Computational and Applied Math, Rice University)

MIIS 2012, Xi'an, China

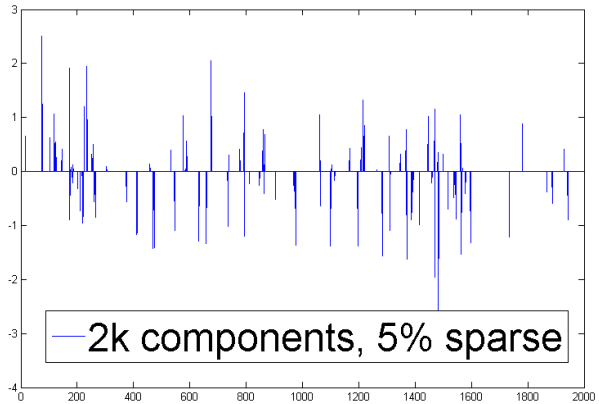
Sparsity means *simplicity*



signals with very few nonzero entries

also, joint sparse signals (share common nonzero positions)

Sparsity means *simplicity*

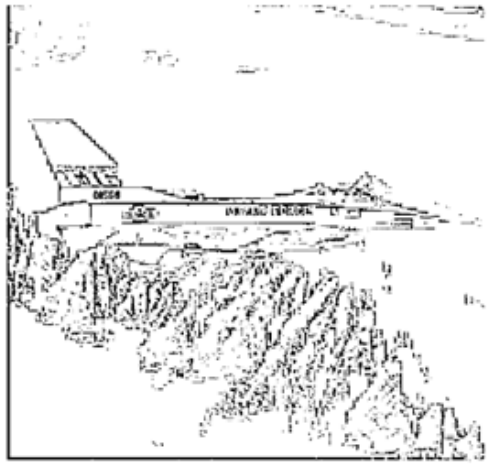


signals with very few nonzero entries

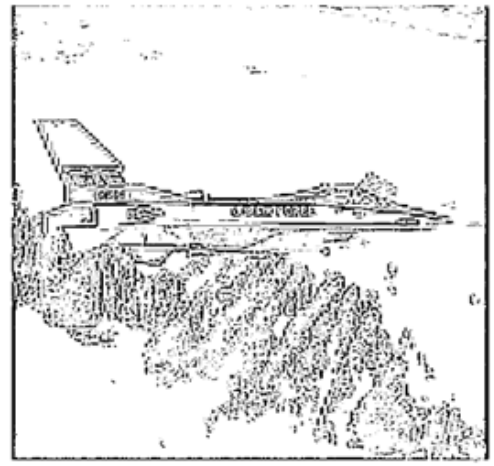
also, joint sparse signals (share common nonzero positions)

Question: How many signals have very few nonzero entries?

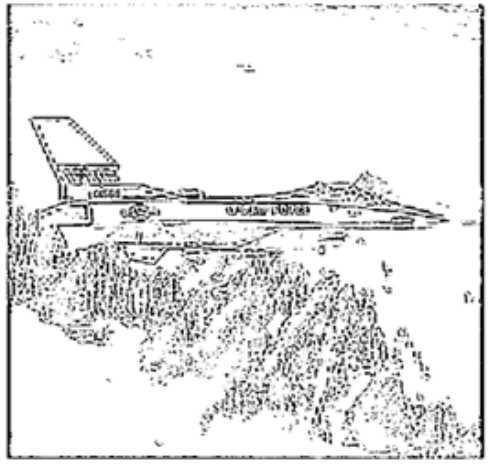
signals with sparse dictionary coefficients (orthogonal bases and frames)



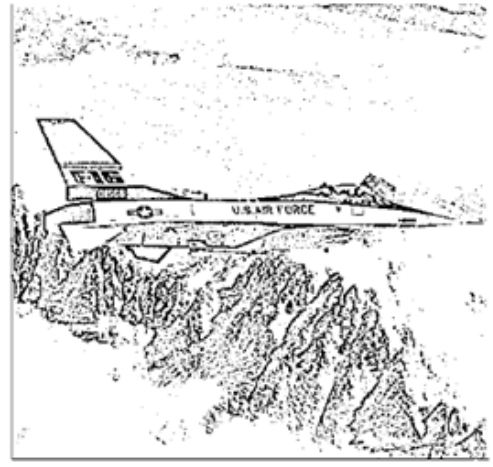
Haar Wavelet



Daubechies Length 6



Biorthogonal Wavelet

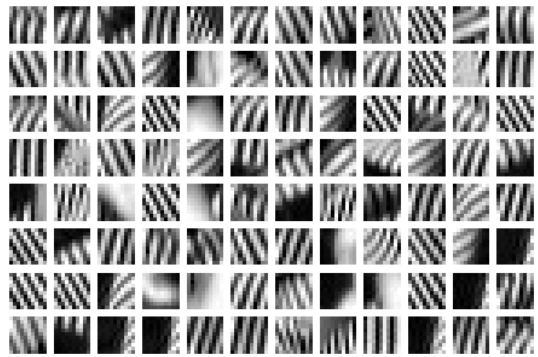


Tight Wavelet Frame

Courtesy of kyunglim Nam'05

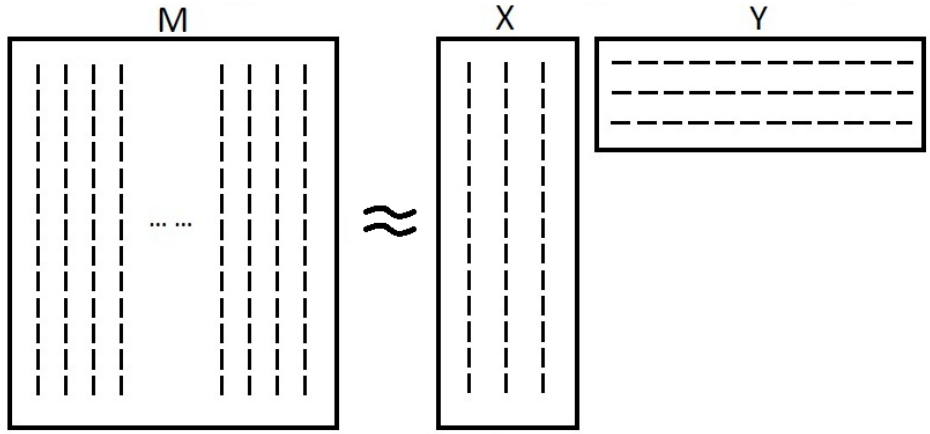



signals transformed to sparse vectors (*gradient field* in the example)



Signal with sparse coefficients under a *learned or trained* dictionary;

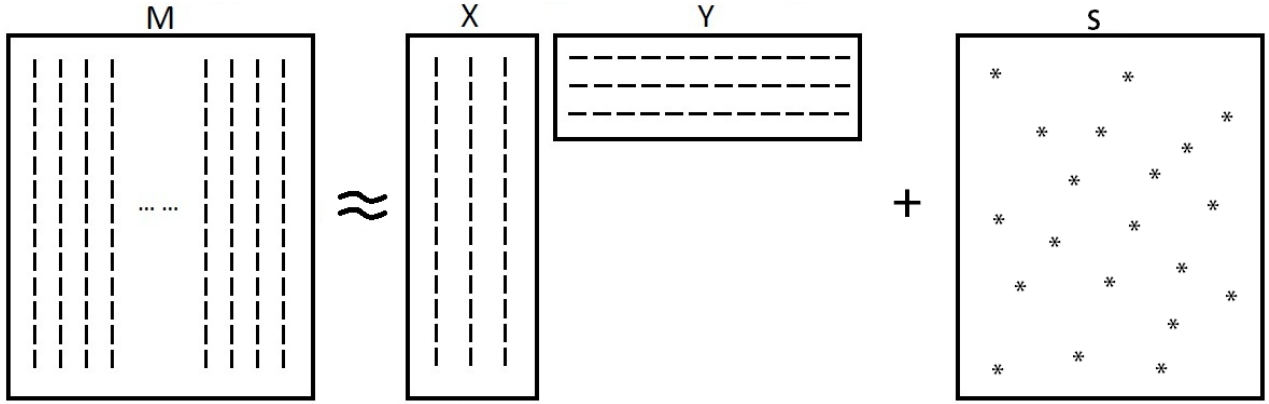
See book *Sparse and Redundant Representations by* M. Elad, Springer.



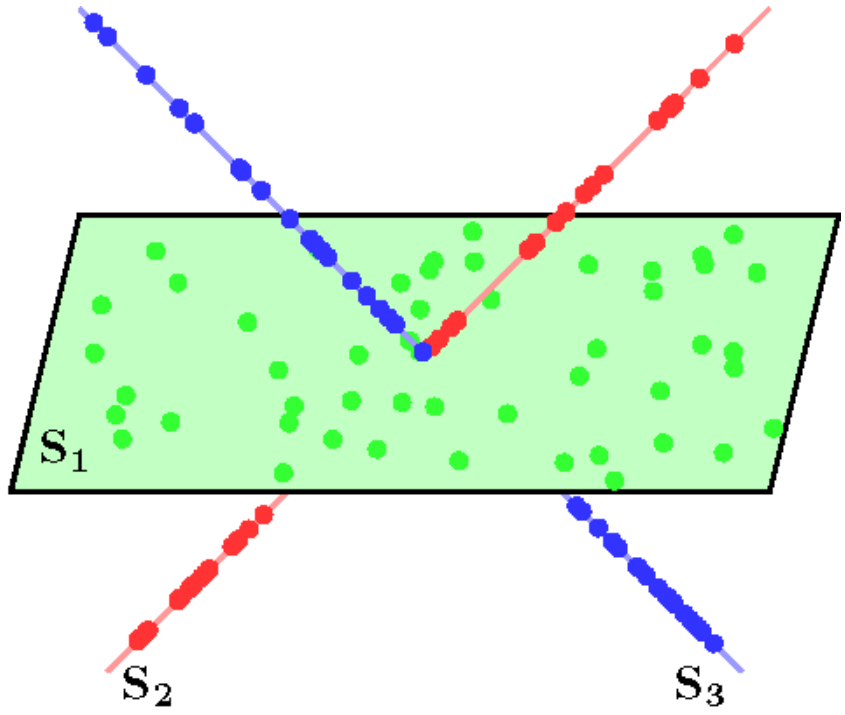
Low rank matrices

Example: if 1 million row square M is approximately rank 20, compare

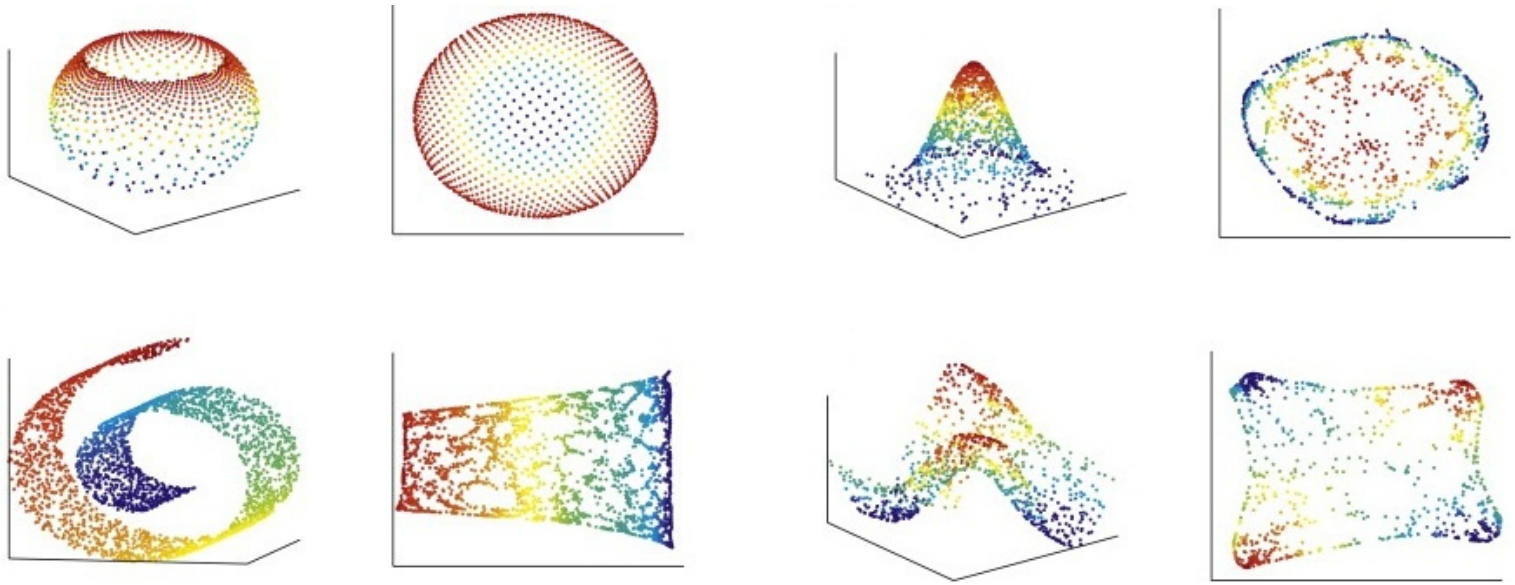
- M: 1m×1m;
- X: 1m×20; Y: 20×1m



$M =$ low rank matrix $+$ sparse matrix



Points on unions of a few subspaces



Points on low dimensional manifolds

Why do care about *sparse signals*?

- cheap to store / transmit

Why do care about *sparse signals*?

- cheap to store / transmit
- sparse coefficients are meaningful. They make more sense.

Why do care about *sparse signals*?

- cheap to store / transmit
- sparse coefficients are meaningful. They make more sense.

Why do care about *sparse signals*?

- cheap to store / transmit
- sparse coefficients are meaningful. They make more sense.
- more robust to errors

Why do care about *sparse signals*?

- cheap to store / transmit
- sparse coefficients are meaningful. They make more sense.
- more robust to errors
- need fewer data to begin with

Why do care about *sparse signals*?

- cheap to store / transmit
- sparse coefficients are meaningful. They make more sense.
- more robust to errors
- need fewer data to begin with
- easy to find by optimization (both speed and storage)

Why do care about *sparse signals*?

- cheap to store / transmit
- sparse coefficients are meaningful. They make more sense.
- more robust to errors
- need fewer data to begin with
- easy to find by optimization (both speed and storage)

In the *Big Data* era, as datasets become larger, it becomes desirable to process the structured information contained within data, rather than data itself.

Questions?

ℓ_1 gives sparse solutions

Minimization ℓ_1 has **a long history** in data processing (geophysical: Claerbout and Muir'73, Santosa and Symes'86), image processing (Rudin, Osher, and Fatemi'92), and statistics (Chen, Donoho, and Saunders'98; Tibshirani'96).

ℓ_1 gives sparse solutions

Minimization ℓ_1 has **a long history** in data processing (geophysical: Claerbout and Muir'73, Santosa and Symes'86), image processing (Rudin, Osher, and Fatemi'92), and statistics (Chen, Donoho, and Saunders'98; Tibshirani'96).

Osher said that Galileo Galilei (1564–1642) used the **median filter**, which for given vector y solves

$$\min_x \sum_i |x - y_i| = \|x\mathbf{1} - \mathbf{y}\|_1.$$

ℓ_1 gives sparse solutions

Minimization ℓ_1 has **a long history** in data processing (geophysical: Claerbout and Muir'73, Santosa and Symes'86), image processing (Rudin, Osher, and Fatemi'92), and statistics (Chen, Donoho, and Saunders'98; Tibshirani'96).

Osher said that Galileo Galilei (1564–1642) used the **median filter**, which for given vector y solves

$$\min_x \sum_i |x - y_i| = \|x\mathbf{1} - \mathbf{y}\|_1.$$

Heuristically, ℓ_1 is known as a **good convex approximate** (convex envelop) of ℓ_0, and ℓ_1 **tends to give sparse solutions**. This is so widely known that many people would just try but not bother with checking the theory.

ℓ_1 gives sparse solutions

Goal: to recover sparse vector x^0 from $b = Ax^0$

- Support $S := \mathrm{supp}(x^0)$
- Zero set $Z := S^C$;
- Sparsity $k = \|x^0\|_0 = |\mathrm{supp}(x^0)|$.

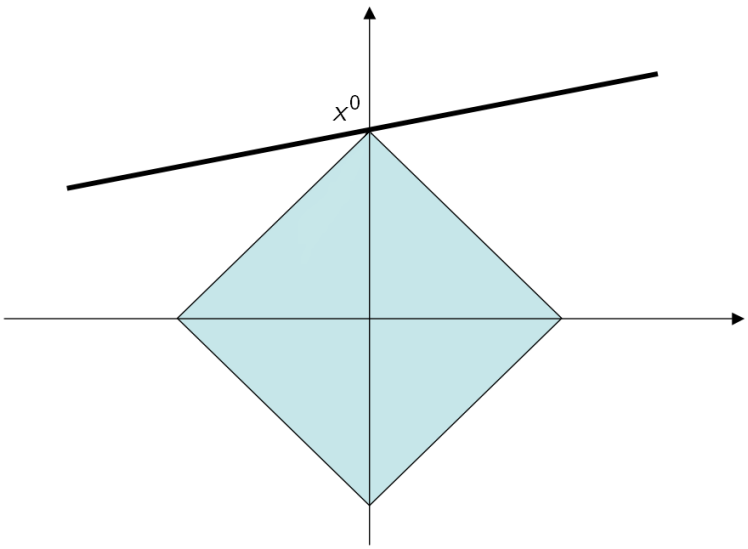
For x^0 to be the unique solution of

$$\min_x \|x\|_1, \text{ subject to } Ax = b,$$

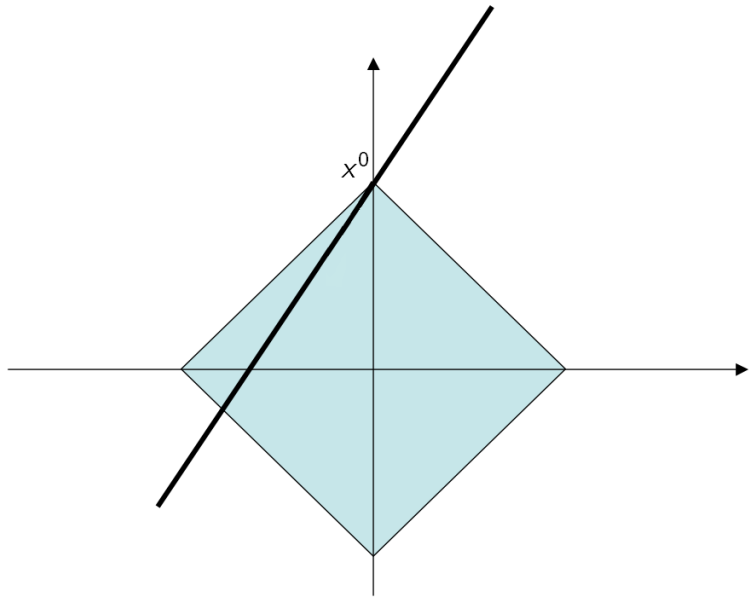
we need

$$\|x^0 + v\|_1 > \|x^0\|_1, \ \forall v \in \mathcal{N}(A), v \neq 0.$$

x^0 is recovered, good



x^0 is not recovered, bad



When the diamond-shaped ℓ_1-ball touches the sparse x^0, we need the affine space $\{x : Ax = b\} = \{x^0 + v : v \in \mathcal{N}(A)\}$ to not cut through the ℓ_1-ball.

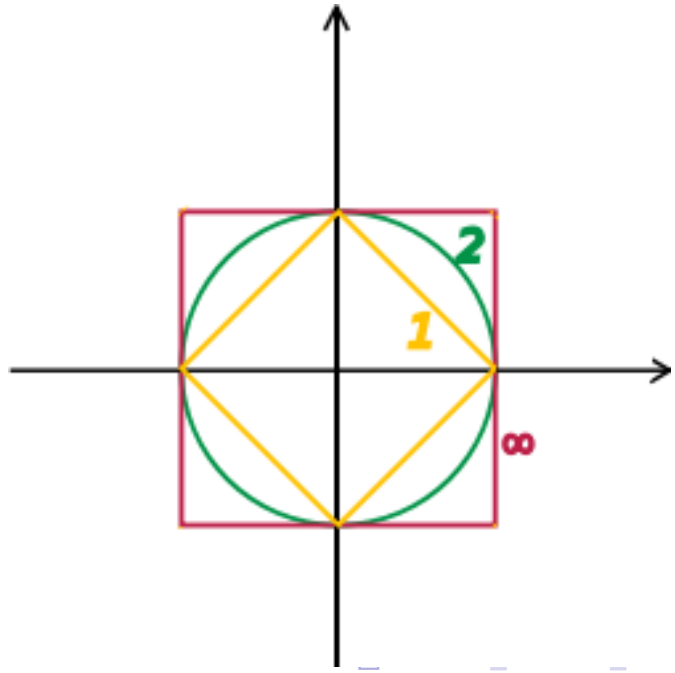
$$\begin{aligned}
\|x^0 + v\|_1 &= \|x^0_S + v_S\|_1 + \|v_Z\|_1 \\
&\geq \|x^0_S\|_1 + \|v_Z\|_1 - \|v_S\|_1 \\
&= \|x^0\|_1 + \|v\|_1 - 2\|v_S\|_1 \\
&\geq \|x^0\|_1 + \|v\|_1 - 2\sqrt{k}\|v\|_2.
\end{aligned}$$

Therefore, x^0 is ℓ_1-recoverable provided

$$\frac{\|v\|_1}{\|v\|_2} > 2\sqrt{k}, \quad \forall v \in \mathcal{N}(A),\ v \neq 0.$$

In general, we only have

$$1 \leq \frac{\|v\|_1}{\|v\|_2} \leq \sqrt{n}.$$



However, $\frac{\|v\|_1}{\|v\|_2}$ is significantly greater than 1 if $\mathcal{V} = \mathcal{N}(A)$ is random.

However, $\frac{\|v\|_1}{\|v\|_2}$ is significantly greater than 1 if $\mathcal{V} = \mathcal{N}(A)$ is random.

[Kashin'77, Garvaev and Gluskin'84] A randomly drawn $(n-m)$-dim subspace $\mathcal{V}$ of $\mathbb{R}^n$ satisfies

$$\frac{\|v\|_1}{\|v\|_2} \geq \frac{c_1\sqrt{m}}{\sqrt{1+\log(n/m)}}, \quad \forall v \in \mathcal{V}, v \neq 0$$

with probability at least $1-\exp(-c_0(n-m))$, where $c_0, c_1 > 0$ are indep. of m and n.

However, $\frac{\|v\|_1}{\|v\|_2}$ is significantly greater than 1 if $\mathcal{V} = \mathcal{N}(A)$ is random.

[Kashin'77, Garvaev and Gluskin'84] A randomly drawn $(n-m)$-dim subspace $\mathcal{V}$ of $\mathbb{R}^n$ satisfies

$$\frac{\|v\|_1}{\|v\|_2} \geq \frac{c_1\sqrt{m}}{\sqrt{1+\log(n/m)}}, \quad \forall v \in \mathcal{V}, v \neq 0$$

with probability at least $1-\exp(-c_0(n-m))$, where $c_0, c_1 > 0$ are indep. of m and n.

Hence, x^0 is ℓ_1-recoverable *with high probability* if A is *random* and

$$k \leq O(m \log^{-1}(n/m)) \quad \text{or} \quad m \geq O(k \log(n/k)).$$

However, $\frac{\|v\|_1}{\|v\|_2}$ is significantly greater than 1 if $\mathcal{V} = \mathcal{N}(A)$ is random.

[Kashin'77, Garvaev and Gluskin'84] A randomly drawn $(n-m)$-dim subspace $\mathcal{V}$ of $\mathbb{R}^n$ satisfies

$$\frac{\|v\|_1}{\|v\|_2} \geq \frac{c_1\sqrt{m}}{\sqrt{1+\log(n/m)}}, \quad \forall v \in \mathcal{V}, v \neq 0$$

with probability at least $1-\exp(-c_0(n-m))$, where $c_0, c_1 > 0$ are indep. of m and n.

Hence, x^0 is ℓ_1-recoverable *with high probability* if A is *random* and

$$k \leq O(m \log^{-1}(n/m)) \quad \text{or} \quad m \geq O(k \log(n/k)).$$

The number of measurements m is a multiple of $k \log(n/k)$ — typically much less than n.

Null Space Property

Definition

Matrix A satisfies the NSP of order k if

$$\|h_S\|_1 < \|h_{S^c}\|_1,$$

holds for $h \in \mathcal{N}(A) \setminus \{0\}$ and $S \subset \{1, 2, \cdots, n\}$ of cardinality k or less.

Theorem (Donoho-Huo-01, Gribonval-Nielsen-03, Zhang-05)

ℓ_1 minimization uniquely recovers all k-sparse vectors x^0 from measurements $b = Ax^0$ if and only if A satisfies the k-NSP.

Null Space Property

Definition

Matrix A satisfies the NSP of order k if

$$\|h_S\|_1 < \|h_{S^c}\|_1,$$

holds for $h \in \mathcal{N}(A) \setminus \{0\}$ and $S \subset \{1, 2, \cdots, n\}$ of cardinality k or less.

Theorem (Donoho-Huo-01, Gribonval-Nielsen-03, Zhang-05)

ℓ_1 minimization uniquely recovers all k-sparse vectors x^0 from measurements $b = Ax^0$ if and only if A satisfies the k-NSP.

Comments:

- No longer necessary if one does not require all k-sparse x^0
- Any decoder that *stably* recovers all nearly sparse signals from $b = Ax^0$ requires A with NSP; see Cohen-Dahmen-DeVore-06.

Other Analyses

- Gorodnitsky, Rao'97: Spark – Smallest number of columns of A that are linearly dependent.
- Donoho, Elad'03: Coherence – Smallest angles between any two columns of A.
- Donoho, Tanner'06: k-neighborly, analysis similar to the above
- Candes, Romberg, and Tao'04 '06: Restricted Isometry Property (RIP) – any $2k$-column submatrices of A must be almost orthogonal.
- RIP$\Rightarrow \ell_1$ recovery: above, Cohen-Dahmen-DeVore, Forcart-Lai, Cai-Wang-Xu, Li-Mo, ...
- Candes and Plan: RIPless analysis
- .....

Spark

Definition (Donoho-Elad-03)

The *spark* of a given matrix A is the smallest number of columns from A that are linearly dependent, written as $\mathrm{spark}(A)$.

Spark

Definition (Donoho-Elad-03)

The *spark* of a given matrix A is the smallest number of columns from A that are linearly dependent, written as $\mathrm{spark}(A)$.

$\mathrm{rank}(A)$ is the largest number of columns from A that are linearly independent. In general, $\mathrm{spark}(A) \neq \mathrm{rank}(A) + 1$; except for many randomly generated matrices.

Spark

Definition (Donoho-Elad-03)

The *spark* of a given matrix A is the smallest number of columns from A that are linearly dependent, written as $\mathrm{spark}(A)$.

$\mathrm{rank}(A)$ is the largest number of columns from A that are linearly independent. In general, $\mathrm{spark}(A) \neq \mathrm{rank}(A) + 1$; except for many randomly generated matrices.

Rank is easy to compute, but spark needs a combinatorial search.

Spark

Theorem (Gorodnitsky-Rao-97)

If $Ax = b$ has a solution x obeying $\|x\|_0 < \mathrm{spark}(A)/2$, then x is the sparsest solution.

Spark

Theorem (Gorodnitsky-Rao-97)

If $Ax = b$ has a solution x obeying $\|x\|_0 < \mathrm{spark}(A)/2$, then x is the sparsest solution.

- Proof idea: any other solution y for $Ay = b$ obeys $A(x - y) = 0$ and $x - y \neq 0$; so

$$\|x\|_0 + \|y\|_0 \geq \|x - y\|_0 \geq \mathrm{spark}(A)$$

or $\|y\|_0 \geq \mathrm{spark}(A) - \|x\|_0 > \mathrm{spark}(A)/2 > \|x\|_0$.

Spark

Theorem (Gorodnitsky-Rao-97)

If $Ax = b$ has a solution x obeying $\|x\|_0 < \mathrm{spark}(A)/2$, then x is the sparsest solution.

- Proof idea: any other solution y for $Ay = b$ obeys $A(x - y) = 0$ and $x - y \neq 0$; so

$$\|x\|_0 + \|y\|_0 \geq \|x - y\|_0 \geq \mathrm{spark}(A)$$

  or $\|y\|_0 \geq \mathrm{spark}(A) - \|x\|_0 > \mathrm{spark}(A)/2 > \|x\|_0$.

- The result does not mean this x can be efficiently found in practice.

Spark

Theorem (Gorodnitsky-Rao-97)

If $Ax = b$ has a solution x obeying $\|x\|_0 < \mathrm{spark}(A)/2$, then x is the sparsest solution.

- Proof idea: any other solution y for $Ay = b$ obeys $A(x - y) = 0$ and $x - y \neq 0$; so

$$\|x\|_0 + \|y\|_0 \geq \|x - y\|_0 \geq \mathrm{spark}(A)$$

  or $\|y\|_0 \geq \mathrm{spark}(A) - \|x\|_0 > \mathrm{spark}(A)/2 > \|x\|_0$.

- The result does not mean this x can be efficiently found in practice.
- For many random matrices $A \in \mathrm{R}^{m \times n}$, the result means that if an algorithm returns x satisfying $\|x\|_0 < (m + 1)/2$, the x is optimal with probability 1.

Coherence

Definition (Mallat-Zhang-93)

The (mutual) coherence of A of a given matrix A is the largest absolution normalized inner product between different columns from A. Suppose $A = [a_1 \ a_2 \ \cdots \ a_n]$. The mutual coherence of A is given by

$$\mu(A) = \max_{k,j,k \neq j} \langle \frac{a_k}{\|a_k\|_2}, \frac{a_j}{\|a_j\|_2} \rangle.$$

Coherence

Definition (Mallat-Zhang-93)

The (mutual) coherence of A of a given matrix A is the largest absolution normalized inner product between different columns from A. Suppose $A = [a_1 \ a_2 \ \cdots \ a_n]$. The mutual coherence of A is given by

$$\mu(A) = \max_{k,j,k \neq j} \langle \frac{a_k}{\|a_k\|_2}, \frac{a_j}{\|a_j\|_2} \rangle.$$

- It characterizes the dependence between columns of A
- For unitary matrices, $\mu(A) = 0$
- For matrices with more columns than rows, $\mu(A) > 0$
- For recovery problems, we desire a small $\mu(A)$

Coherence

Definition (Mallat-Zhang-93)

The (mutual) coherence of A of a given matrix A is the largest absolution normalized inner product between different columns from A. Suppose $A = [a_1 \ a_2 \ \cdots \ a_n]$. The mutual coherence of A is given by

$$\mu(A) = \max_{k,j,k\neq j} \langle \frac{a_k}{\|a_k\|_2}, \frac{a_j}{\|a_j\|_2} \rangle.$$

- It characterizes the dependence between columns of A
- For unitary matrices, $\mu(A) = 0$
- For matrices with more columns than rows, $\mu(A) > 0$
- For recovery problems, we desire a small $\mu(A)$
- For $A = [\Phi \ \Psi]$ where Φ and Ψ are $n \times n$ unitary, it holds $n^{-1/2} \leq \mu(A) \leq 1$

Coherence

Definition (Mallat-Zhang-93)

The (mutual) coherence of A of a given matrix A is the largest absolution normalized inner product between different columns from A. Suppose $A = [a_1 \ a_2 \ \cdots \ a_n]$. The mutual coherence of A is given by

$$\mu(A) = \max_{k,j,k\neq j} \langle \frac{a_k}{\|a_k\|_2}, \frac{a_j}{\|a_j\|_2} \rangle.$$

- It characterizes the dependence between columns of A
- For unitary matrices, $\mu(A) = 0$
- For matrices with more columns than rows, $\mu(A) > 0$
- For recovery problems, we desire a small $\mu(A)$
- For $A = [\Phi \ \Psi]$ where Φ and Ψ are $n \times n$ unitary, it holds $n^{-1/2} \leq \mu(A) \leq 1$
- $\mu(A) = n^{-1/2}$ is achieved with $[I \ F]$, $[I \ \mathrm{Hadamard}]$, etc.

Coherence-based Guarantee

Theorem (Donoho-Elad-03)

$$\mathrm{spark}(A) \geq 1 + \mu^{-1}(A).$$

Corollary

If $Ax = b$ has a solution x obeying $\|x\|_0 < (1 + \mu^{-1}(A))/2$, then x is the unique sparsest solution.

Coherence-based Guarantee

Theorem (Donoho-Elad-03)

$$\mathrm{spark}(A) \geq 1 + \mu^{-1}(A).$$

Corollary

If $Ax = b$ has a solution x obeying $\|x\|_0 < (1 + \mu^{-1}(A))/2$, then x is the unique sparsest solution.

Compare with the previous condition: $\|x\|_0 < \mathrm{spark}(A)/2$

$(1 + \mu^{-1}(A))$ is at most $\sqrt{n} + 1$ but spark can be $n + 1$. However:

Theorem (Donoho-Elad-03,Gribonval-Nielsen-03)

If A has normalized columns and $Ax = b$ has a solution x satisfying

$$\|x\|_0 < \frac{1}{2}\left(1 + \mu^{-1}(A)\right),$$

then this x is the unique minimizer with respect to both ℓ_0 and ℓ_1.

Mutual coherence

Definition

Consider two bases $\Psi = [\phi_1, \ldots, \phi_n]$ and $\Psi = [\psi_1, \ldots, \psi_n]$. Their mutual coherence is

$$\mu(\Phi, \Psi) = \sqrt{n} \max_{1 \leq i,j \leq n} \langle \frac{\phi_i}{\|\phi_i\|_2}, \frac{\psi_i}{\|\psi_i\|_2} \rangle.$$

Mutual coherence

Definition

Consider two bases $\Psi = [\phi_1, \ldots, \phi_n]$ and $\Psi = [\psi_1, \ldots, \psi_n]$. Their mutual coherence is

$$\mu(\Phi, \Psi) = \sqrt{n} \max_{1 \leq i,j \leq n} \langle \frac{\phi_i}{\|\phi_i\|_2}, \frac{\psi_i}{\|\psi_i\|_2} \rangle.$$

Mutual coherence

Definition

Consider two bases $\Psi = [\phi_1, \ldots, \phi_n]$ and $\Psi = [\psi_1, \ldots, \psi_n]$. Their mutual coherence is

$$\mu(\Phi, \Psi) = \sqrt{n} \max_{1 \leq i,j \leq n} \langle \frac{\phi_i}{\|\phi_i\|_2}, \frac{\psi_i}{\|\psi_i\|_2} \rangle.$$

In general, $1 \leq \mu(\Phi, \Psi) \leq \sqrt{n}$. *Low coherence* if $\mu(\Phi, \Psi)$ is much closer to 1 than $\sqrt{n}$.

Mutual coherence

Definition

Consider two bases $\Psi = [\phi_1, \ldots, \phi_n]$ and $\Psi = [\psi_1, \ldots, \psi_n]$. Their mutual coherence is

$$\mu(\Phi, \Psi) = \sqrt{n} \max_{1 \leq i,j \leq n} \langle \frac{\phi_i}{\|\phi_i\|_2}, \frac{\psi_i}{\|\psi_i\|_2} \rangle.$$

In general, $1 \leq \mu(\Phi, \Psi) \leq \sqrt{n}$. *Low coherence* if $\mu(\Phi, \Psi)$ is much closer to 1 than $\sqrt{n}$.

Consider $\Phi^T(\Psi x^o)$. Entry i is $\sum_{1 \leq j \leq n} \langle \phi_i, \psi_j \rangle x_j^o$. Low coherence means every entry i encodes a guaranteed amount of information of every x_j^o.

Mutual coherence

Definition

Consider two bases $\Psi = [\phi_1, \ldots, \phi_n]$ and $\Psi = [\psi_1, \ldots, \psi_n]$. Their mutual coherence is

$$\mu(\Phi, \Psi) = \sqrt{n} \max_{1 \le i,j \le n} \langle \frac{\phi_i}{\|\phi_i\|_2}, \frac{\psi_i}{\|\psi_i\|_2} \rangle.$$

In general, $1 \le \mu(\Phi, \Psi) \le \sqrt{n}$. *Low coherence* if $\mu(\Phi, \Psi)$ is much closer to 1 than $\sqrt{n}$.

Consider $\Phi^T(\Psi x^o)$. Entry i is $\sum_{1 \le j \le n} \langle \phi_i, \psi_j \rangle x_j^o$. Low coherence means every entry i encodes a guaranteed amount of information of every x_j^o.

Theorem (Candes-Romberg-06)

Given signal $u^o = \Psi x^o$ where x^o is k-sparse, choose m entries of $\Phi^T u^o$ uniformly at random to form vector b. As long as

$$m \ge C \cdot \mu^2(\Phi, \Psi) \cdot (k \log n)$$

for some universal constant $C > 0$, the ℓ_1 minimization recovers x^o with overwhelming probability. (The result is shown for nearly all possible sign sequences of x^o.)

Restricted Isometry Principle (RIP)

Definition (Candes-Tao-06)

Matrix A obeys the restricted isometry principle with constant δ_k if

$$(1 - \delta_k)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_k)\|x\|_2^2$$

for all k-sparse vectors x.

Restricted Isometry Principle (RIP)

Definition (Candes-Tao-06)

Matrix A obeys the restricted isometry principle with constant δ_k if

$$(1-\delta_k)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1+\delta_k)\|x\|_2^2$$

for all k-sparse vectors x.

Theorem (Mo-Li-11)

Consider $b = Ax^o + w$. If A satisfies the RIP with $\delta_{2k} \leq 0.4931$, then the solution x^* of

$$\min \|x\|_1 \quad \textit{subject to } \|Ax - b\|_2 \leq \epsilon$$

with ϵ set to $\|w\|_2$ satisfies

$$\|x^* - x^o\|_1 \leq C_1 \cdot \sqrt{k}\|w\|_2 + C_2 \cdot \sigma_{[k]}(x^o),$$
$$\|x^* - x^o\|_2 \leq \bar{C}_1 \cdot \|w\|_2 + \bar{C}_2 \cdot \sigma_{[k]}(x^o)/\sqrt{k},$$

where C_1, C_2, $\bar{C}_1$, and $\bar{C}_2$ are universal constants.

Random matrices with RIPs

Trivial randomly constructed matrices satisfy RIPs with overwhelming probability.

- Gaussian: $A_{ij} \sim N(0, 1/m)$, $\|x\|_0 \leq O(m/\log(n/m))$ whp, proof is based on applying concentration of measures to the singular values of Gaussian matrices (Szarek-91,Davidson-Szarek-01).
- Bernoulli: $A_{ij} \sim \pm 1$ wp $1/2$, $\|x\|_0 \leq O(m/\log(n/m))$ whp, proof is based on applying concentration of measures to the smallest singular value of a subgaussian matrix (Candes-Tao-04,Litvak-Pajor-Rudelson-TomczakJaegermann-04).
- Fourier ensemble: $A \in \mathbb{C}^{m \times n}$ is a randomly chosen submatrix of $F \in \mathbb{C}^{n \times n}$. Candes-Tao shows $\|x\|_0 \leq O(m/\log(n)^6)$ whp; Rudelson-Vershynin shows $\|x\|_0 \leq O(m/\log(n)^4)$; conjectured $\|x\|_0 \leq O(m/\log(n))$.
- .....

Applications

See Rice Compressed Sensing Repository: `http://dsp.rice.edu/cs`

- Signal sensing and processing, image processing
- Medical imaging (faster imaging, less radiation, multi-modality)
- (Wireless) communications (joint sparsity, channel estimation)
- Multi/hyperspectral imaging (spectral+spatial structure, data to information)
- Statistics, machine learning (feature selection; LASSO, logistic regression, SVM)
- Geophysical (seismic) data analysis
- Face recognition, video processing (robust PCA: common structure + sparse difference)
- Distributed computation / cloud computing (terabytes of data are spread out, centralized computing is infeasible, sparse recovery at minimal communication cost?)

Many applications require *nonnegativity*, *structured (model) sparsity*, *complex data*.

Discussion time. Questions?

Some uncovered topics:

- Guarantees for stable low-rank matrix recovery
- How to find a *sparsifying* dictionary Ψ?
- How to generate a *good* matrix A for stable recovery?
- What about structured matrices?
- What about structured sensing noise?

Off-the-shelf optimization approaches

Many ℓ_1-like problems can be reformulated as linear programs (LPs) or second-order cone programs (SOCPs)

Many nuclear-norm problems can be reformulated as semi-definite programs (SDPs)

Off-the-shelf optimization approaches

Many ℓ_1-like problems can be reformulated as linear programs (LPs) or second-order cone programs (SOCPs)

Many nuclear-norm problems can be reformulated as semi-definite programs (SDPs)

LPs, SOCPs, SDPs have polynomial time solvers, so why bother with new algorithms?

Off-the-shelf optimization approaches

Many ℓ_1-like problems can be reformulated as linear programs (LPs) or second-order cone programs (SOCPs)

Many nuclear-norm problems can be reformulated as semi-definite programs (SDPs)

LPs, SOCPs, SDPs have polynomial time solvers, so why bother with new algorithms?

Reasons:

- Problem size: millions of constraints, more variables, too many for off-the-shelf simplex and interior-point solvers

Off-the-shelf optimization approaches

Many ℓ_1-like problems can be reformulated as linear programs (LPs) or second-order cone programs (SOCPs)

Many nuclear-norm problems can be reformulated as semi-definite programs (SDPs)

LPs, SOCPs, SDPs have polynomial time solvers, so why bother with new algorithms?

Reasons:

- Problem size: millions of constraints, more variables, too many for off-the-shelf simplex and interior-point solvers
- Sensing operator A: often large but has fast implementations of Ax and A^*y

Off-the-shelf optimization approaches

Many ℓ_1-like problems can be reformulated as linear programs (LPs) or second-order cone programs (SOCPs)

Many nuclear-norm problems can be reformulated as semi-definite programs (SDPs)

LPs, SOCPs, SDPs have polynomial time solvers, so why bother with new algorithms?

Reasons:

- Problem size: millions of constraints, more variables, too many for off-the-shelf simplex and interior-point solvers
- Sensing operator A: often large but has fast implementations of Ax and A^*y
- Solution: existing codes do not take advantages of solution sparsity

Off-the-shelf optimization approaches

Many ℓ_1-like problems can be reformulated as linear programs (LPs) or second-order cone programs (SOCPs)

Many nuclear-norm problems can be reformulated as semi-definite programs (SDPs)

LPs, SOCPs, SDPs have polynomial time solvers, so why bother with new algorithms?

Reasons:

- Problem size: millions of constraints, more variables, too many for off-the-shelf simplex and interior-point solvers
- Sensing operator A: often large but has fast implementations of Ax and A^*y
- Solution: existing codes do not take advantages of solution sparsity

First-order algorithms are easy to program, based on Ax and A^*y, and fast to give sparse solutions.

Off-the-shelf optimization approaches

Many ℓ_1-like problems can be reformulated as linear programs (LPs) or second-order cone programs (SOCPs)

Many nuclear-norm problems can be reformulated as semi-definite programs (SDPs)

LPs, SOCPs, SDPs have polynomial time solvers, so why bother with new algorithms?

Reasons:

- Problem size: millions of constraints, more variables, too many for off-the-shelf simplex and interior-point solvers
- Sensing operator A: often large but has fast implementations of Ax and A^*y
- Solution: existing codes do not take advantages of solution sparsity

First-order algorithms are easy to program, based on Ax and A^*y, and fast to give sparse solutions.

Simplex and interior-point codes are accurate and good benchmarks.

Forward-backward Splitting / Prox-Linear

To solve

$$\min \mu J(x) + F(x),$$

iterate

$$x^{k+1} \leftarrow \mu J(x) + \langle \nabla F(x^k), x \rangle + \frac{1}{2\delta_k} \|x - x^k\|_2{}^2.$$

δ_k : step size. Can be obtained from forward-backward operator splitting (See Combettes-Wajs-05 for theory and examples)

Forward-backward Splitting / Prox-Linear

To solve

$$\min \mu J(x) + F(x),$$

iterate

$$x^{k+1} \leftarrow \mu J(x) + \langle \nabla F(x^k), x \rangle + \frac{1}{2\delta_k} \|x - x^k\|_2{}^2.$$

δ_k : step size. Can be obtained from forward-backward operator splitting (See Combettes-Wajs-05 for theory and examples)

Simple for many choices of $J(x)$: ℓ_1 / TV / $\|\cdot\|_*$ / $\|\cdot\|_{2,1}$, etc.

Forward-backward Splitting / Prox-Linear

To solve

$$\min \mu J(x) + F(x),$$

iterate

$$x^{k+1} \leftarrow \mu J(x) + \langle \nabla F(x^k), x \rangle + \frac{1}{2\delta_k} \|x - x^k\|_2{}^2.$$

δ_k : step size. Can be obtained from forward-backward operator splitting (See Combettes-Wajs-05 for theory and examples)

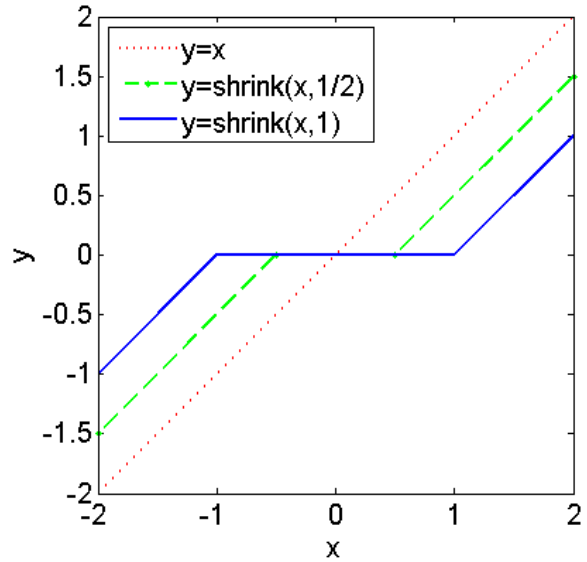
Simple for many choices of $J(x)$: ℓ_1 / TV / $\|\cdot\|_*$ / $\|\cdot\|_{2,1}$, etc.

Properties:

- If $\delta_k < 2/L$ (Lipschitz const of ∇F), $\|x^k - x^*\|$ is non-expansive; a fixed-point is a solution
- Adaptive δ_k: accelerate convergence, Barzilai-Borwein and nonmonotone line search give sufficient descent

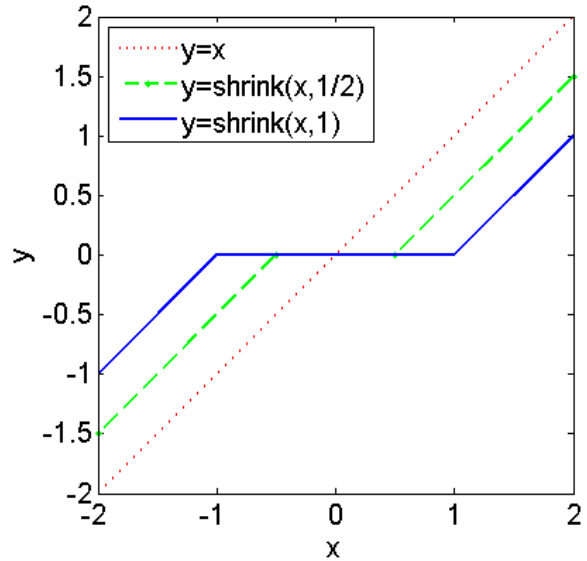
Sparse Vector Recovery

For $J(x) = \|x\|_1$ and given $\nabla F(x^k)$, the subproblem is $O(n)$ and parallel. It is called **shrinkage** or **soft-thresholding**.



Sparse Vector Recovery

For $J(x) = \|x\|_1$ and given $\nabla F(x^k)$, the subproblem is $O(n)$ and parallel. It is called **shrinkage** or **soft-thresholding**.



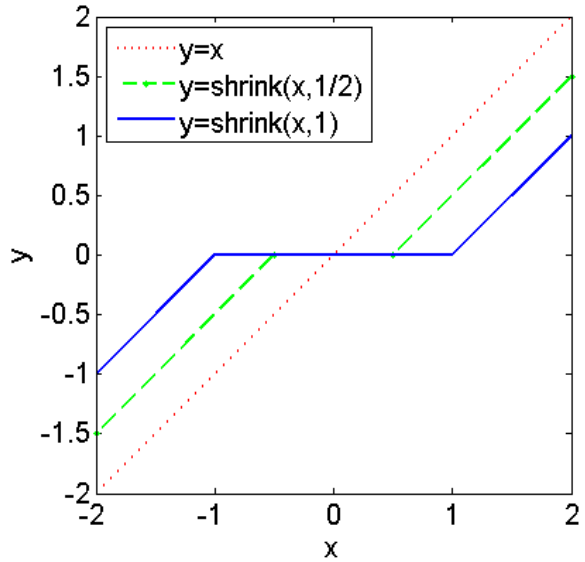
Iteration:

$$x^{k+1} = \text{shrink}(x^k - \delta_k \nabla F(x^k),\ \delta_k \mu)$$

Codes: SpaRSA, FPC

Sparse Vector Recovery

For $J(x) = \|x\|_1$ and given $\nabla F(x^k)$, the subproblem is $O(n)$ and parallel. It is called **shrinkage** or **soft-thresholding**.



Iteration:

$$x^{k+1} = \text{shrink}(x^k - \delta_k \nabla F(x^k),\ \delta_k \mu)$$

Codes: SpaRSA, FPC

$\nabla F(x^k)$ can be demanding in logistic regression and when dealing with non-Gaussian noise. **Code**: LPS

Low-Rank Matrix Recovery

For low-rank matrix recovery, $J(X) = \|X\|_* := \sum_i \sigma_i(X)$. The subproblem is $*$-shrinkage or signular-value soft-thresholding

$$\min_X \mu\|X\|_* + \frac{1}{2}\|X - Y^k\|_F^2$$

Low-Rank Matrix Recovery

For low-rank matrix recovery, $J(X) = \|X\|_* := \sum_i \sigma_i(X)$. The subproblem is $*$-shrinkage or signular-value soft-thresholding

$$\min_X \mu\|X\|_* + \frac{1}{2}\|X - Y^k\|_F^2$$

Since both terms are unitary-invariant, it is solved by SVD and then shrinking the singular values.

Iteration:

$$X^{k+1} = \text{shrink}^*(X^k - \delta_k \nabla F(X^k),\ \delta_k \mu)$$

Codes: FPCA, SVT (linearized Bregman). Apply approximate SVD or compute partial SVD.

Lee et al'10 uses the same framework with the max-norm.

Application to Transform-ℓ_1 and Total Variation

If $J(x) = \|Lx\|_1$ and L is orthogonal, unitary, or tight frame, the subproblem has closed-form solutions.

Application to Transform-ℓ_1 and Total Variation

If $J(x) = \|Lx\|_1$ and L is orthogonal, unitary, or tight frame, the subproblem has closed-form solutions.

If $J(x) = \|\nabla x\|_1$ (total variation or TV), there are graph-cut (computing max-flow/min-cut) algorithms to solve the subproblems.

Application to Transform-ℓ_1 and Total Variation

If $J(x) = \|Lx\|_1$ and L is orthogonal, unitary, or tight frame, the subproblem has closed-form solutions.

If $J(x) = \|\nabla x\|_1$ (total variation or TV), there are graph-cut (computing max-flow/min-cut) algorithms to solve the subproblems.

If $\|Lx\|_1$ and L is non-invertible, the subproblem is less trivial. **Solutions**: operator splitting, smoothing (e.g., Huber norm to replace ℓ_1).

Convergence and Practice

Convergence for convex $J(x)$ is quite standard; also holds with line search and other variants. (e.g., Wright, Figueiredo, Nowak'08).

Convergence and Practice

Convergence for convex $J(x)$ is quite standard; also holds with line search and other variants. (e.g., Wright, Figueiredo, Nowak'08).

Can extend to prox-regular functions (Lewis and Wright'08) and hybrid 1st/2nd-order iterations (Wen, Yin, Zhang, Goldfarb'11)

Convergence and Practice

Convergence for convex $J(x)$ is quite standard; also holds with line search and other variants. (e.g., Wright, Figueiredo, Nowak'08).

Can extend to prox-regular functions (Lewis and Wright'08) and hybrid 1st/2nd-order iterations (Wen, Yin, Zhang, Goldfarb'11)

How fast?

• For piece-wise linear $J(x)$, e.g., ℓ_1, optimal solution is identified in *finitely many steps* then algorithm converges linearly.

Convergence and Practice

Convergence for convex $J(x)$ is quite standard; also holds with line search and other variants. (e.g., Wright, Figueiredo, Nowak'08).

Can extend to prox-regular functions (Lewis and Wright'08) and hybrid 1st/2nd-order iterations (Wen, Yin, Zhang, Goldfarb'11)

How fast?

- For piece-wise linear $J(x)$, e.g., ℓ_1, optimal solution is identified in *finitely many steps* then algorithm converges linearly.
- In general, objective decreases $\sim O(1/k)$; need $\sim O(L/\epsilon)$ iterations

Convergence and Practice

Convergence for convex $J(x)$ is quite standard; also holds with line search and other variants. (e.g., Wright, Figueiredo, Nowak'08).

Can extend to prox-regular functions (Lewis and Wright'08) and hybrid 1st/2nd-order iterations (Wen, Yin, Zhang, Goldfarb'11)

How fast?

• For piece-wise linear $J(x)$, e.g., ℓ_1, optimal solution is identified in *finitely many steps* then algorithm converges linearly.

• In general, objective decreases $\sim O(1/k)$; need $\sim O(L/\epsilon)$ iterations

Convergence speed depends on solution structure, controlled by μ

- **Larger** μ gives **faster** and **more structured** solution. **Reason** for ℓ_1: smaller optimal support is easier identify and afterward, iterate essentially on a smaller support with a better condition number
- **Smaller** μ gives slower and less structured solution.

Convergence and Practice

Convergence for convex $J(x)$ is quite standard; also holds with line search and other variants. (e.g., Wright, Figueiredo, Nowak'08).

Can extend to prox-regular functions (Lewis and Wright'08) and hybrid 1st/2nd-order iterations (Wen, Yin, Zhang, Goldfarb'11)

How fast?

• For piece-wise linear $J(x)$, e.g., ℓ_1, optimal solution is identified in *finitely many steps* then algorithm converges linearly.

• In general, objective decreases $\sim O(1/k)$; need $\sim O(L/\epsilon)$ iterations

Convergence speed depends on solution structure, controlled by μ

- **Larger** μ gives **faster** and **more structured** solution. **Reason** for ℓ_1: smaller optimal support is easier identify and afterward, iterate essentially on a smaller support with a better condition number
- **Smaller** μ gives slower and less structured solution.

Continuation: start from a large μ and decrease μ sequentially, using previous solution to warm-start the current iteration (e.g., code FPC, SpaRSA).

Techniques Applied with Prox-Linear

- Two-step descents (TwIST)
- Barzilai-Borwein steps, non-monotone line search (Wen et al.'09)
- (Block) coordinate descent: apply descent only to a subset of components. (Tseng and Yun'09, Li and Osher'09, Wright'11).
- Active set: estimate the optimal support and solve a reduced subproblem (Shi et al.'08, Wen et al.'09)
- Use (approximate) second-order information, e.g. in logistic regression (Byrd et al.'10; Shi et al.'08)
- Accelerated first-order methods: given a convex function with L-Lipschitz gradients and no other structures, iterations is reduced from $O(L/\epsilon)$ to $O(L/\sqrt{\epsilon})$ for ϵ-optimal in objective value
  - $\min F(x)$: Nesterov'83;
  - $\min J(x) + F(x)$: Nesterov'04, Bech and Teboulle'08, Tseng'08;
  - ALM: Ma, Scheinberg, Goldfarb'10

Variable Splitting

Consider a linear operator L and

$$\min_x J(Lx) + F(x)$$

Rewrite

$$\min_{x,y} J(y) + F(x), \quad \text{s.t. } y = Lx,$$

which "separates" $J(\cdot)$ from $F(\cdot)$.

Variable Splitting

Consider a linear operator L and

$$\min_x J(Lx) + F(x)$$

Rewrite

$$\min_{x,y} J(y) + F(x), \quad \text{s.t. } y = Lx,$$

which "separates" $J(\cdot)$ from $F(\cdot)$.

Augmented Lagrangian: generates x^k, y^k along with multipliers λ^k

$$\begin{aligned}(x^{k+1}, y^{k+1}) &\leftarrow \min_{x,y} \mathcal{L}(x, y; \lambda^k) = J(y) + F(x) + \langle \lambda, Lx - y \rangle + \frac{c}{2}\|Lx - y\|_2^2, \\ \lambda^{k+1} &\leftarrow \lambda^k + c(Lx^{k+1} - y^{k+1}).\end{aligned}$$

Convergence needs bounded $\lambda^k > 0$ if J and F are convex.

The joint minimization subproblem is still hard to solve.

Variable Splitting

Consider a linear operator L and

$$\min_x J(Lx) + F(x)$$

Introduce

$$\min_{x,y} J(y) + F(x), \quad \text{s.t. } y = Lx,$$

which "separates" $J(\cdot)$ from $F(\cdot)$.

Alternating Direction Method (ADM): alternate x^k and y^k updates

$$\begin{aligned} x^{k+1} &\leftarrow \min_x L(x, y^k; \lambda^k), \\ y^{k+1} &\leftarrow \min_y L(x^{k+1}, y; \lambda^k), \\ \lambda^{k+1} &\leftarrow \lambda^k + \delta c(Lx^{k+1} - y^{k+1}). \end{aligned}$$

Converges if $\delta \in (0, (\sqrt{5}+1)/2)$. Subproblems x/y are decoupled.

Variable Splitting

If a subproblem, say the y-subproblem, is still expensive to solve, consider

Gradient descent:

$$y^{k+1} \leftarrow y^k - \tau^k \nabla_y L(x^{k+1}, y^k; \lambda^k).$$

Proximal descent:

$$y^{k+1} \leftarrow \min_y J(y) + \langle \nabla_y (\langle \lambda, Lx^{k+1} - y \rangle + \frac{c}{2} \|Lx^{k+1} - y\|_2^2), y - y^k \rangle + \frac{1}{2\tau} \|y - y^k\|_2^2.$$

Alternating Direction Method, History

Yin Zhang: "the idea of ADM goes back to **Sun-Tze** (400 BC) and **Caesar** (100 BC):

"Divide and Conquer." — Julius Caesar (100-44 BC)

"远交近攻", "各个击破". — Sun-Tzu (400 BC) "

Back to 1950s–70s, it appears as operator splitting for PDEs and studied by **Douglas, Peaceman, and Rachford**, and then **Glowinsky** et al.'81-89, **Gabay**'83.

Subsequent studies are in the context of variational inequality (**Eckstein and Bertsekes**'92, **He** et al.'02)

Extensions to multiple blocks (**He**, **Yuan**, and collaborators)

Alternating Direction Method, Applications

For sparse optimization, one subproblem is shrinkage or its variant.

The other subproblem is smooth, very often solving a linear system with $(A^\top A + c^k L^\top L)$.

Alternating Direction Method, Applications

For sparse optimization, one subproblem is shrinkage or its variant.

The other subproblem is smooth, very often solving a linear system with $(A^\top A + c^k L^\top L)$.

For $TV(\cdot) = \|\nabla \cdot\|_1$, breaking $L = \nabla$ from $\|\cdot\|_1$. (Wang, Yin, and Zhang'08). The linear system can be diagonalized by DFT for various A (e.g., I and convolution) and suitable boundary cond.

Codes: FTVd, Split Bregman, YALL1, RecPF, SALSA.

Versatile ℓ_1 minimization

Variable splitting lets one apply ADM to many extensions of ℓ_1: transform-ℓ_1, frame-ℓ_1, constrained/penalized, weighted, nonnegative, isotropic/anisotropic TV, complex, group $\ell_{2,1}$, nuclear norm, and even some non-convex models.

Versatile ℓ_1 minimization

Variable splitting lets one apply ADM to many extensions of ℓ_1: transform-ℓ_1, frame-ℓ_1, constrained/penalized, weighted, nonnegative, isotropic/anisotropic TV, complex, group $\ell_{2,1}$, nuclear norm, and even some non-convex models.

For example, codes YALL1 and YALL1-Group solve

$$
\begin{aligned}
\text{BP:} \quad & \min_{x\in\mathbb{C}^n} \ \|Wx\|_{w,1}, \ \text{s.t. } Ax = b \\
\text{L1/L1:} \quad & \min_{x\in\mathbb{C}^n} \ \|Wx\|_{w,1} + \frac{1}{\nu}\|Ax-b\|_1 \\
\text{L1/L2:} \quad & \min_{x\in\mathbb{C}^n} \ \|Wx\|_{w,1} + \frac{1}{2\rho}\|Ax-b\|_2^2 \\
\text{BP+:} \quad & \min_{x\in\mathbb{R}^n} \ \|x\|_{w,1}, \ \text{s.t. } Ax = b, \ x \geq 0 \\
\text{L1/L1+:} \quad & \min_{x\in\mathbb{R}^n} \ \|x\|_{w,1} + \frac{1}{\nu}\|Ax-b\|_1, \ \text{s.t. } x \geq 0 \\
\text{L1/L2+:} \quad & \min_{x\in\mathbb{R}^n} \ \|x\|_{w,1} + \frac{1}{2\rho}\|Ax-b\|_2^2, \ \text{s.t. } x \geq 0
\end{aligned}
$$

and their group/joint sparse version.

Example: Recover Inverse Covariance Matrix (Ma-Xue-Zou'12)

Model

$$\begin{aligned} \min_{R,S,L} \quad & \overbrace{\langle \hat{\Sigma}, R \rangle - \log\det R}^{\text{normal log-likelihood}} + \overbrace{\alpha \|S\|_1}^{\text{sparse}} + \overbrace{\beta \operatorname{Tr}(L)}^{\text{low rank}} \\ \text{subject to} \quad & R = S - L, \ R \succ 0, \ L \succeq 0. \end{aligned}$$

Example: Recover Inverse Covariance Matrix (Ma-Xue-Zou'12)

Model

$$\begin{aligned} \min_{R,S,L} \quad & \overbrace{\langle \hat{\Sigma}, R\rangle - \log\det R}^{\text{normal log-likelihood}} + \overbrace{\alpha\|S\|_1}^{\text{sparse}} + \overbrace{\beta\,\mathrm{Tr}(L)}^{\text{low rank}} \\ \text{subject to} \quad & R = S - L,\ R \succ 0,\ L \succeq 0. \end{aligned}$$

Two exact ADM subproblems:

R-subproblem: $R^{k+1} \leftarrow$ normal log-likelihood(R) + quadratic(R)

(S, L)-subproblem: $(S^{k+1}, L^{k+1}) \leftarrow \alpha\|S\|_1 + \beta\,\mathrm{Tr}(L)$ + quadratic(S, L)

Example: Recover Inverse Covariance Matrix (Ma-Xue-Zou'12)

Model

$$\begin{aligned} \min_{R,S,L} \quad & \overbrace{\langle \hat{\Sigma}, R\rangle - \log\det R}^{\text{normal log-likelihood}} + \overbrace{\alpha\|S\|_1}^{\text{sparse}} + \overbrace{\beta\,\mathrm{Tr}(L)}^{\text{low rank}} \\ \text{subject to} \quad & R = S - L,\ R \succ 0,\ L \succeq 0. \end{aligned}$$

Two exact ADM subproblems:

R-subproblem: $R^{k+1} \leftarrow$ normal log-likelihood(R) + quadratic(R)

(S, L)-subproblem: $(S^{k+1}, L^{k+1}) \leftarrow \alpha\|S\|_1 + \beta\,\mathrm{Tr}(L)$ + quadratic(S, L)

Apply prox-linear approximation:

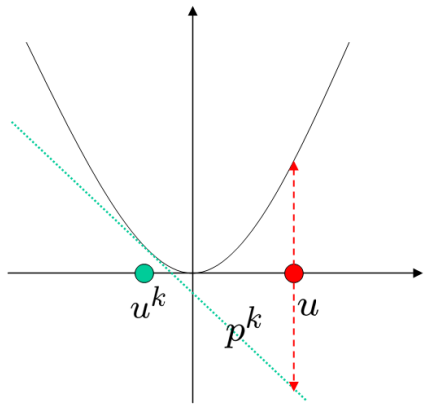
R-subproblem: $R^{k+1} \leftarrow$ closed-form

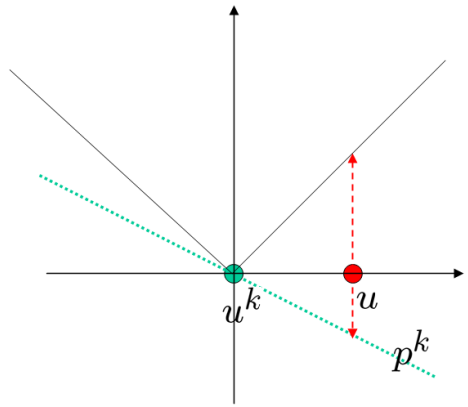
S-subproblem: $S^{k+1} \leftarrow \ell_1$ shrinkage

L-subproblem: $L^{k+1} \leftarrow$ singular-value shrinkage

Related: the Bregman Methods (Osher et al'06, Yin et al'08)

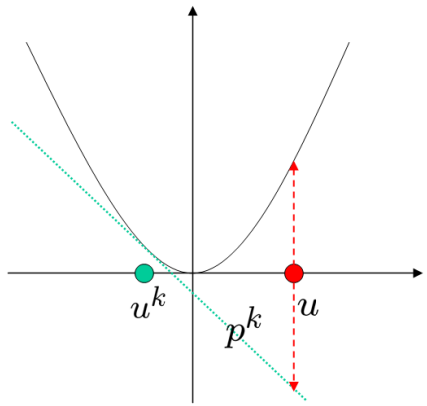
Bregman dist: $D(u, u^k) := J(u) - \big(J(u^k) + \langle p^k, u - u^k \rangle\big)$, $p^k \in \partial J(u^k)$.

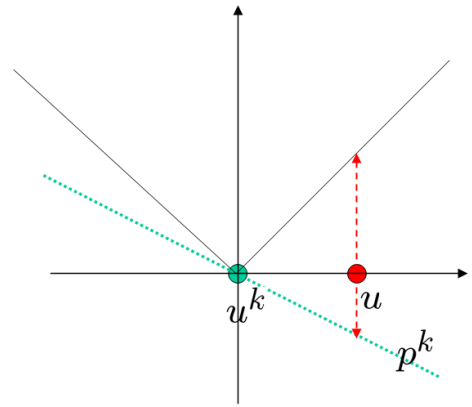




Related: the Bregman Methods (Osher et al'06, Yin et al'08)

Bregman dist: $D(u, u^k) := J(u) - \big(J(u^k) + \langle p^k, u - u^k\rangle\big)$, $p^k \in \partial J(u^k)$.



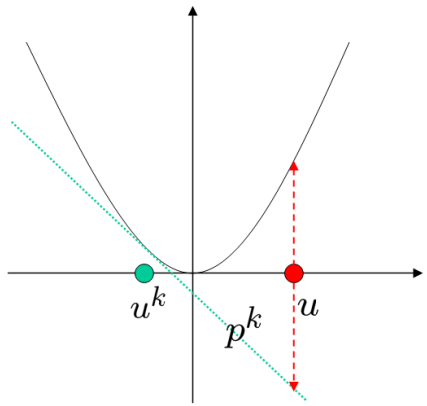


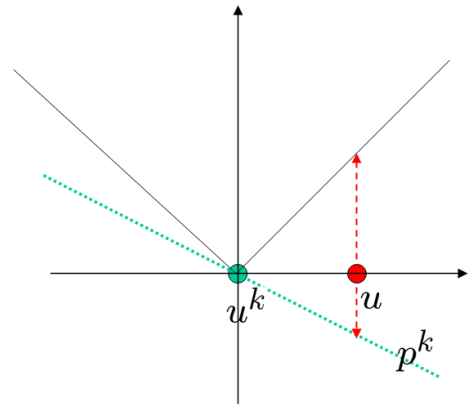
For $\min_x F(x)$ subject to $Ax = b$, Bregman Alg updates x^k and p^k:

$$\begin{aligned} x^{k+1} &\leftarrow \min\ \mu\left[J(x) - \left(J(x^k) + \langle p^k, x - x^k\rangle\right)\right] + \frac{1}{2}\|Ax - b\|_2^2 \\ p^{k+1} &\leftarrow p^k + A^\top(b - Ax^{k+1}). \end{aligned}$$

Related: the Bregman Methods (Osher et al'06, Yin et al'08)

Bregman dist: $D(u, u^k) := J(u) - \big(J(u^k) + \langle p^k, u - u^k\rangle\big)$, $p^k \in \partial J(u^k)$.





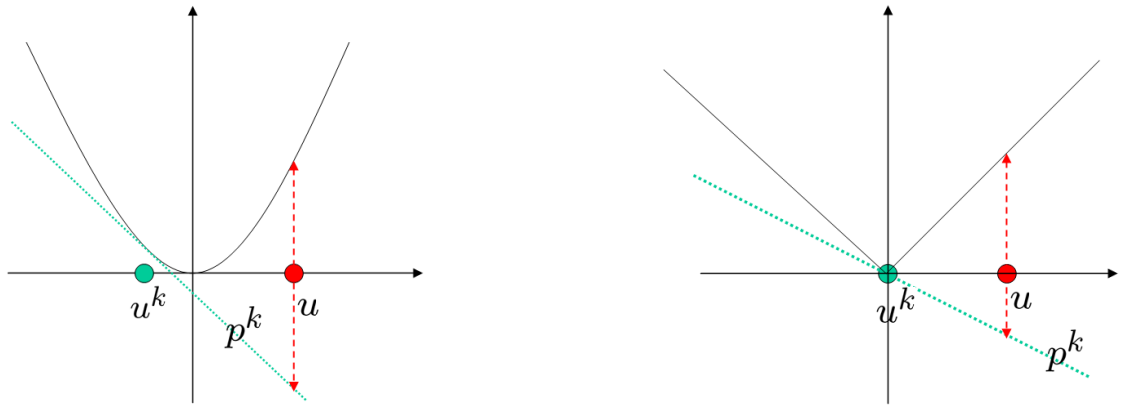
For $\min_x F(x)$ subject to $Ax = b$, Bregman Alg updates x^k and p^k:

$$\begin{aligned} x^{k+1} &\leftarrow \min\ \mu\left[J(x) - (J(x^k) + \langle p^k, x - x^k\rangle)\right] + \frac{1}{2}\|Ax - b\|_2^2 \\ p^{k+1} &\leftarrow p^k + A^\top(b - Ax^{k+1}). \end{aligned}$$

Relation to augmented Lagrangian: $p^k = A^\top \lambda^k$. Split Bregman (applying Bregman to the splitting formulation) is ADM.

Related: the Bregman Methods (Osher et al'06, Yin et al'08)

Bregman dist: $D(u, u^k) := J(u) - \big(J(u^k) + \langle p^k, u - u^k\rangle\big)$, $p^k \in \partial J(u^k)$.



For $\min_x F(x)$ subject to $Ax = b$, Bregman Alg updates x^k and p^k:

$$\begin{aligned} x^{k+1} &\leftarrow \min\ \mu\left[J(x) - (J(x^k) + \langle p^k, x - x^k\rangle)\right] + \frac{1}{2}\|Ax - b\|_2^2 \\ p^{k+1} &\leftarrow p^k + A^\top(b - Ax^{k+1}). \end{aligned}$$

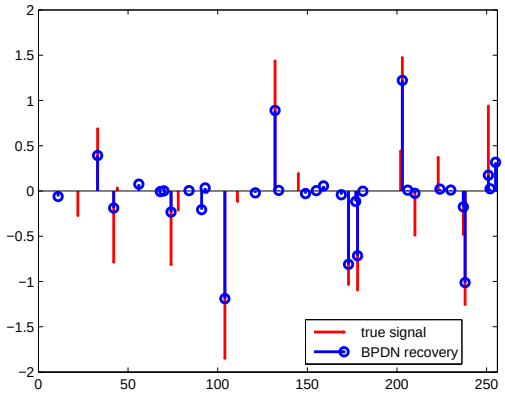
Relation to augmented Lagrangian: $p^k = A^\top \lambda^k$. Split Bregman (applying Bregman to the splitting formulation) is ADM.

Bregman is not novel, but leads to new observations:

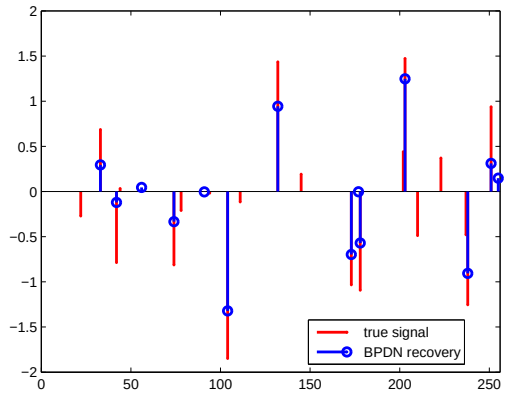
- Successively linearizing $J(\cdot)$ gives rise to $Ax = b$ in the limit
- Better denoising; error forgetting

Example: recover x^0 from **moderate noisy measurements** $b = Ax^0 + \omega$

1. ℓ_1 denoising: $\min \ \mu\|x\|_1 + \frac{1}{2}\|Ax - b\|_2^2$



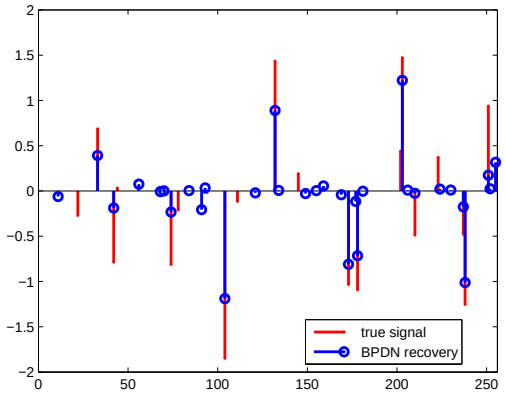
$\mu = 48.5$



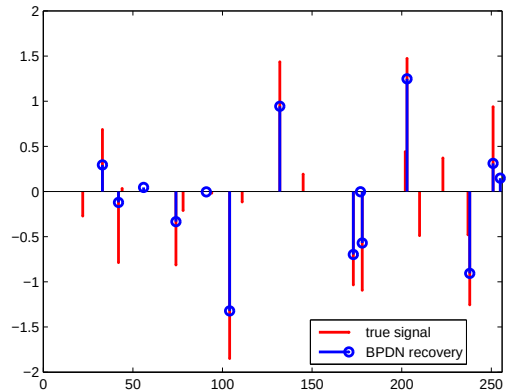
$\mu = 49$

Example: recover x^0 from **moderate noisy measurements** $b = Ax^0 + \omega$

1. ℓ_1 denoising: $\min \ \mu\|x\|_1 + \frac{1}{2}\|Ax - b\|_2^2$

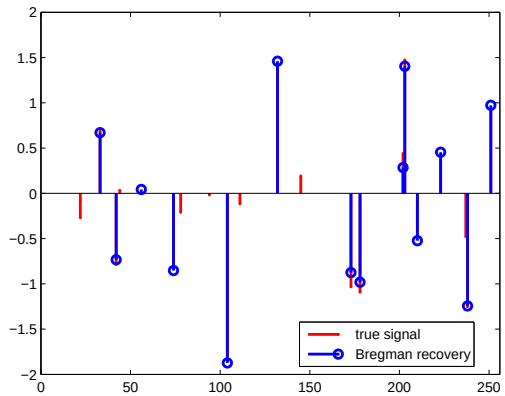


$\mu = 48.5$



$\mu = 49$

2. ℓ_1 Bregman iteration



$\mu = 150$, 5 iterations

Better denoising results are also observed on image reconstruction.

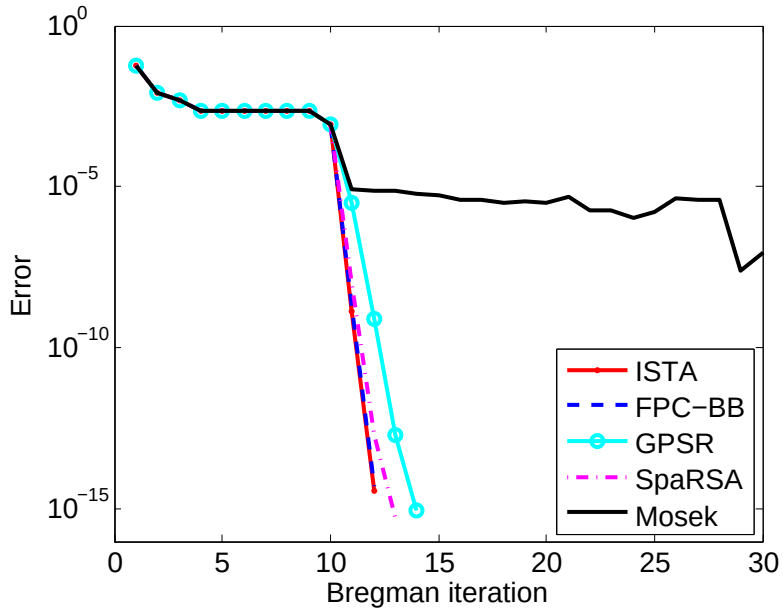
The Add-Back Update, Error Forgetting

After a change of variable, the Bregman update has an **add-back** form:

$$\begin{aligned} x^{k+1} &\leftarrow \min\ \mu J(x) + \frac{1}{2}\|Ax - b^k\|_2^2, \\ b^{k+1} &\leftarrow b + (b^k - Ax^{k+1}). \end{aligned}$$

- Subproblem is in the same form of $\ell_1 + \ell_2^2$
- Subproblem solution errors are iterative cancelled

Simulation: subproblems solved at low accuracy 10^{-6} by ISTA, FPC-BB, GPSR, and SpaRSA, but relative errors drop to machine precision



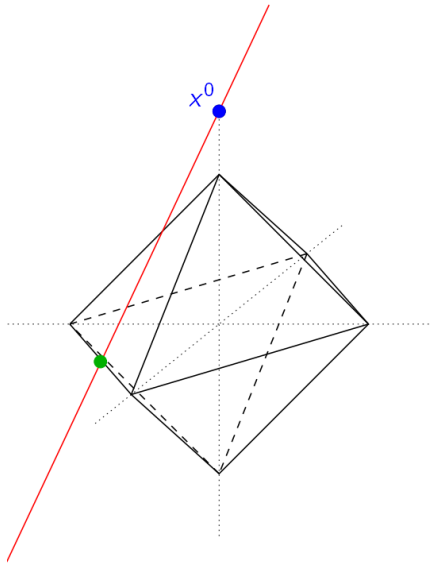
Discussion Time. Questions?

Possible topics:

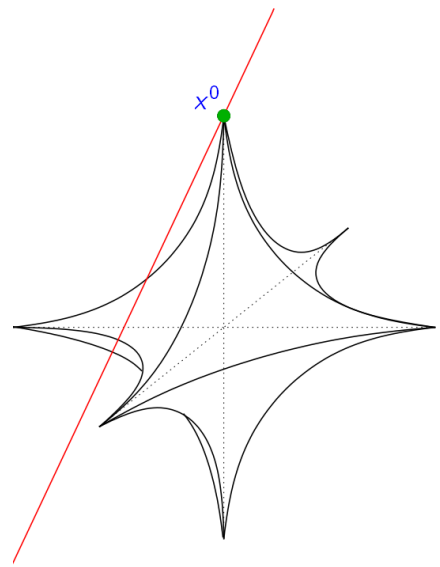
- Initial solution, starting point
- How solution sparsity is taken advantages of?
- When to stop?
- Parameter selection?
- Primal (prox-linear) vs dual (ADM/Bregman)

Non-Convex ℓ_p Minimization, $p < 1$

Pioneered by Rao et al'97; popularized by Boyd et al, Chartrand et al for compressive sensing



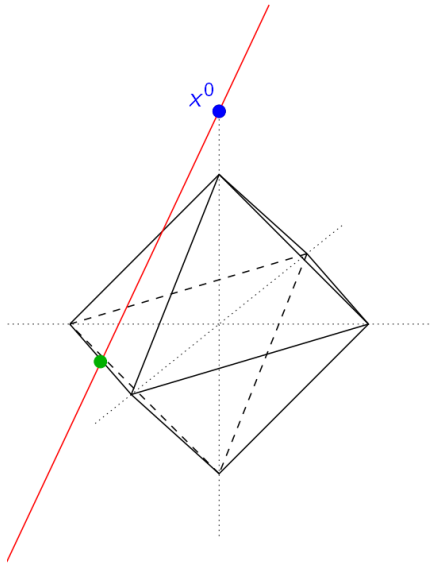
ℓ_1-ball



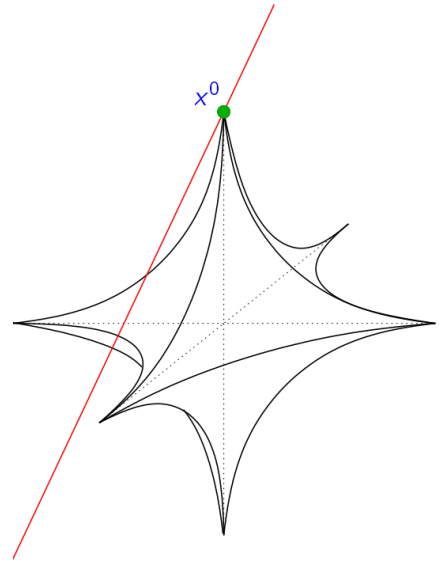
$\ell_{1/2}$-ball

Non-Convex ℓ_p Minimization, $p < 1$

Pioneered by Rao et al'97; popularized by Boyd et al, Chartrand et al for compressive sensing



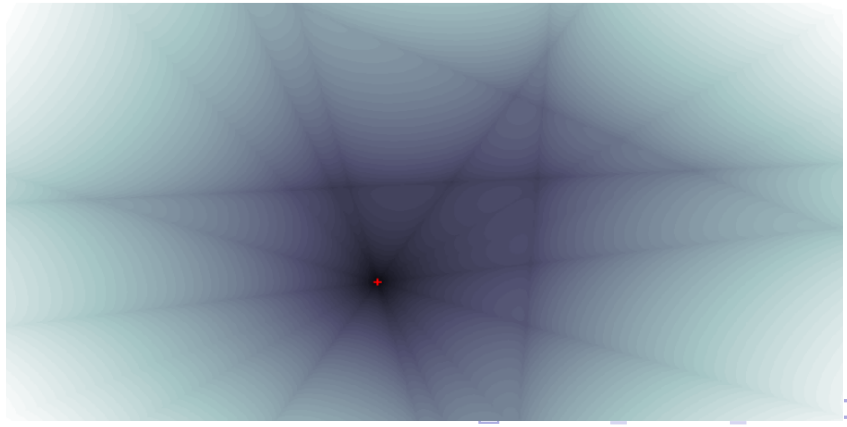
ℓ_1-ball



$\ell_{1/2}$-ball

However, there are **multiple local minima** for $\min_x \|x\|_p^p$, s.t $Ax = b$.

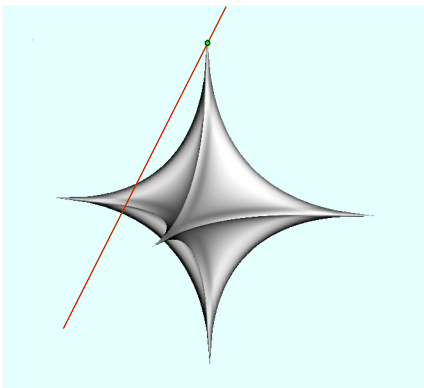
$\|x\|_p^p$ colorized over
$\{x^0 + v : v \in \mathcal{N}(A)\}$



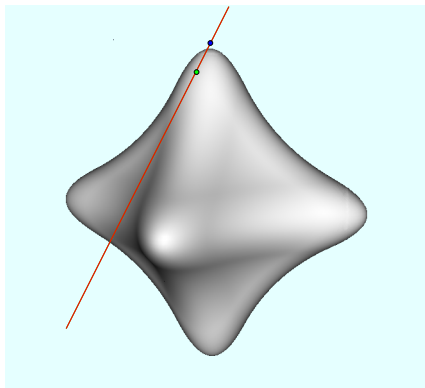
Non-Convex Minimization

Chartrand and Yin'08, Daubechies et al.'09, Lai-Wang'10 smooth out the "small" local minima by

$$\text{replacing} \quad \|x\|_p^p \quad \text{by} \quad \sum_{i=1}^{n} \frac{x_i^2}{(x_i^2+\epsilon)^{(1-p)/2}}.$$



Sharp $\ell_{1/2}$-ball

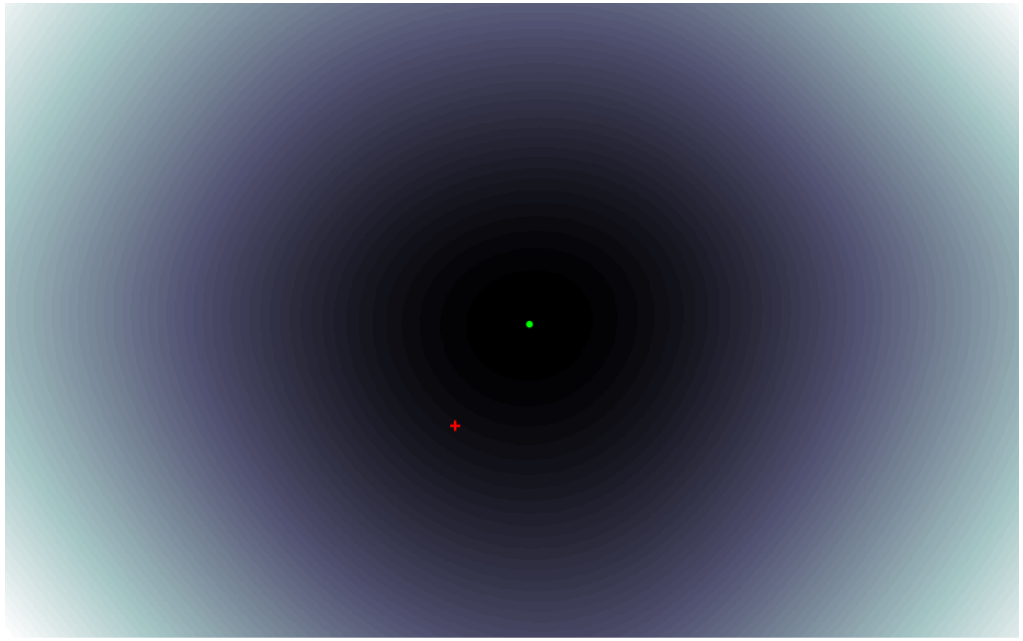


Smoothed $\ell_{1/2}$-ball

Work better than ℓ_1 on fast decaying signals (nonzero entries have a fast decaying distribution)

For fast decaying sparse signals, **smoothing+continuation** help avoid local minima.

Red: sparse solution; Green: smoothed ℓ_p solutions

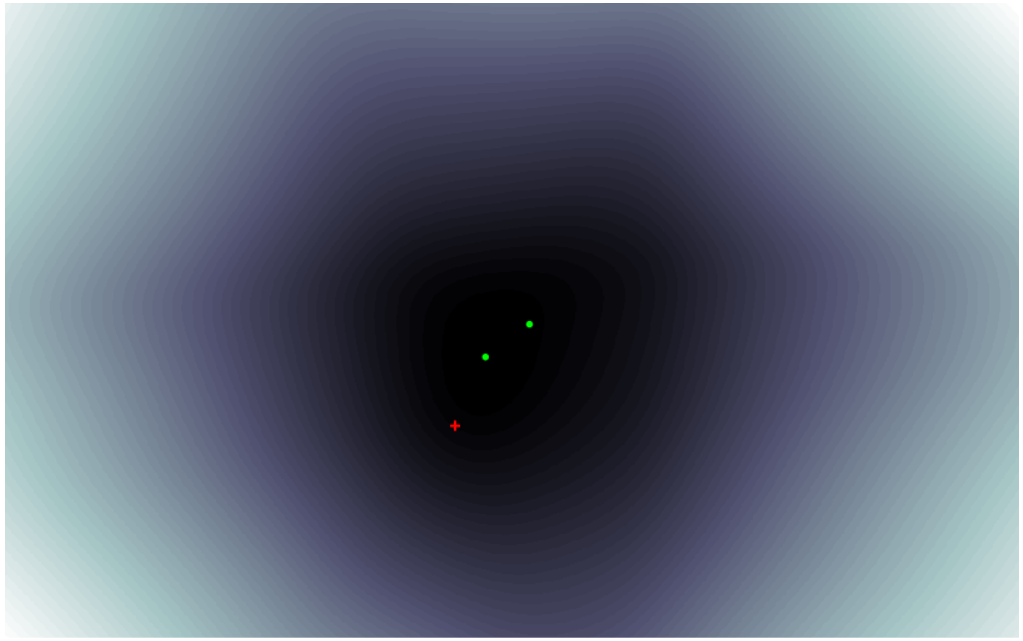


$\epsilon = 1.0$

Courtesy of Rick Chartrand

For fast decaying sparse signals, **smoothing+continuation** help avoid local minima.

Red: sparse solution; Green: smoothed ℓ_p solutions

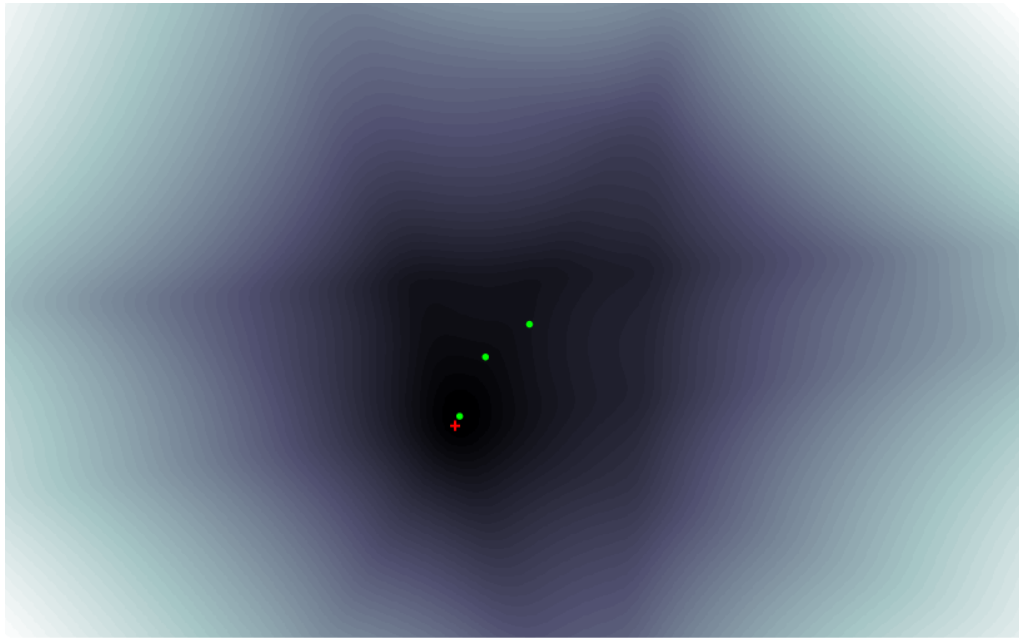


$\epsilon = 0.1$

Courtesy of Rick Chartrand

For fast decaying sparse signals, **smoothing+continuation** help avoid local minima.

Red: sparse solution; Green: smoothed ℓ_p solutions

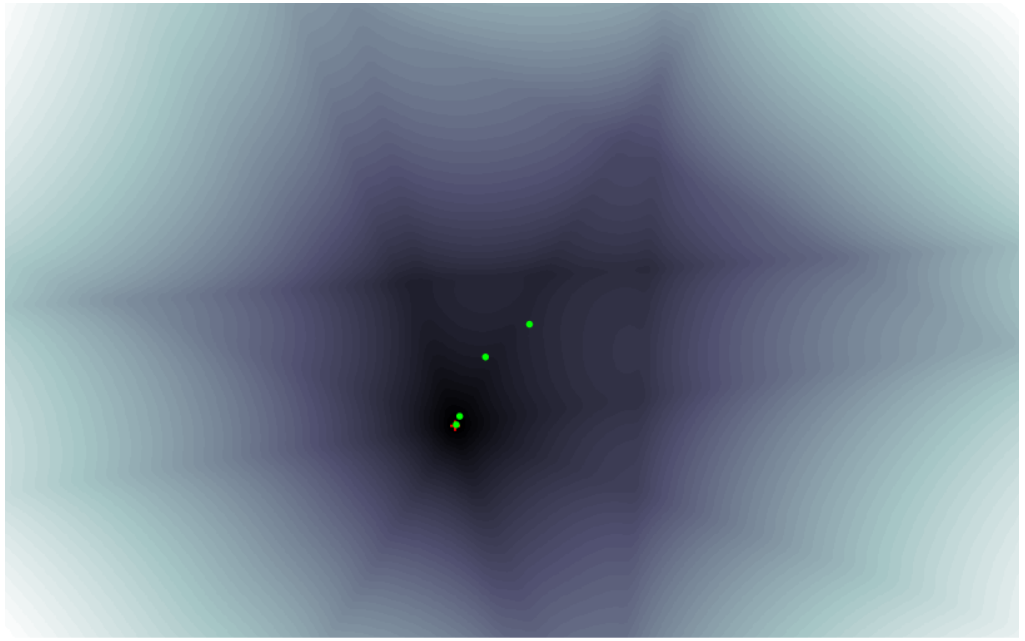


$\epsilon = 0.01$

Courtesy of Rick Chartrand

For fast decaying sparse signals, **smoothing+continuation** help avoid local minima.

Red: sparse solution; Green: smoothed ℓ_p solutions



$\epsilon = 0.001$

Courtesy of Rick Chartrand

Other non-convex techniques

- Boyd, Candes, Wakin '08 uses $\sum_{i=1}^{n} |x_i|/(|x_i| + \epsilon)^{(1-p)}$.
- Wang and Yin'10 uses $\sum_i w_i |x_i|$ with binary w_i.
  Code: Threshold–ISD
- Z. Xu uses $\ell_{1/2}$, closed-form subproblem solution
- Mohimani, Babaie-Zadeh, Jutten use other approximations.
  Code: SL0
- Extensions to joint sparse vector and low-rank matrix recovery .....

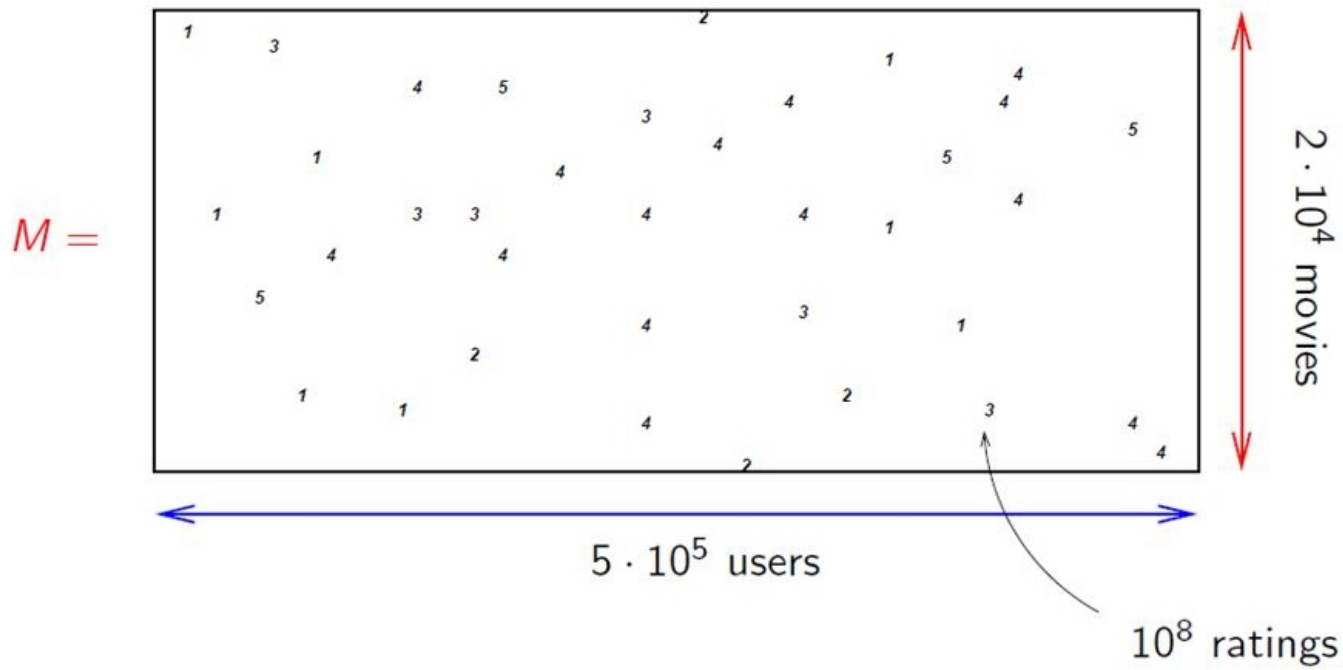
On **fast decaying signals**, results are significantly better ℓ_1.

Matrix Completion

(Candes and Recht'09; Recht, Fazel, and Parrilo'10) A matrix M can be exactly recovered from enough yet few random subsamples with overwhelming probability, provided

- Low rank
- Row and column subspaces are incoherent

Motivating example: low rank due to only a few factors contributing to users' preferences



Matrix Completion Model

Given samples in Ω, recover X by

$$\min_x \ \mu J(X) + \frac{1}{2}\|X_\Omega - M_\Omega\|_F^2$$

Choices of $J(X)$:

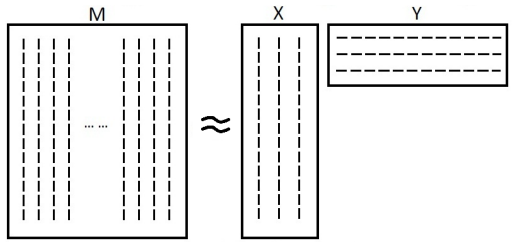
- $\mathrm{rank}(X)$: computationally intractable
- $\|X\|_*$: convex envelope of $\mathrm{rank}(X)$
- $\|X\|_{\max}$: another convex regularizer of $\mathrm{rank}(X)$
- Nonconvex ℓ_p semi-norm of singular values of X

Codes: SVT, FPCA, APGL, ...

Also, $\mathrm{trace}(X^\top X)^{p/2}$: equals $\|X\|_*$ when $p = 1$; it is non-convex when $p < 1$ and is the ℓ_p-norm of singular values

On terabyte data, SVDs become almost forbidden.

Code LMaFit avoids SVDs by exploiting low-rank factorization $M \approx XY$



and solve

$$\min_{X,Y} \|\mathcal{P}_\Omega(XY - M)\|_F$$

Given an approximate rank r, X is m-by-r and Y is r-by-n.

Drawbacks:

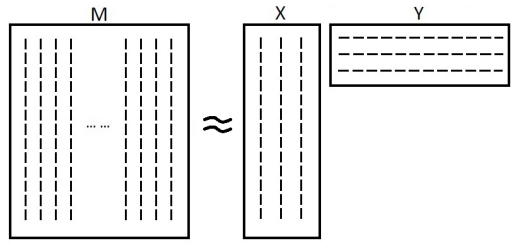
- non-convex, no theoretical guarantees known yet
- need rank estimate or dynamic rank update

Advantages:

- cheap (alternating minimization)
- does not store any full matrix

On terabyte data, SVDs become almost forbidden.

Code LMaFit avoids SVDs by exploiting low-rank factorization $M \approx XY$



and solve

$$\min_{X,Y} \|\mathcal{P}_\Omega(XY - M)\|_F$$

Given an approximate rank r, X is m-by-r and Y is r-by-n.

Drawbacks:

- non-convex, no theoretical guarantees known yet
- need rank estimate or dynamic rank update

Advantages:

- cheap (alternating minimization)
- does not store any full matrix

Other models and codes:

- OptSpace (descent on the Grassmanian manifold)
- Low-rank+Sparse, splitting method: Yuan, Yang'09, Lin et al.'10, etc.

Optimization/Algorithmic Techniques Used, Part 1

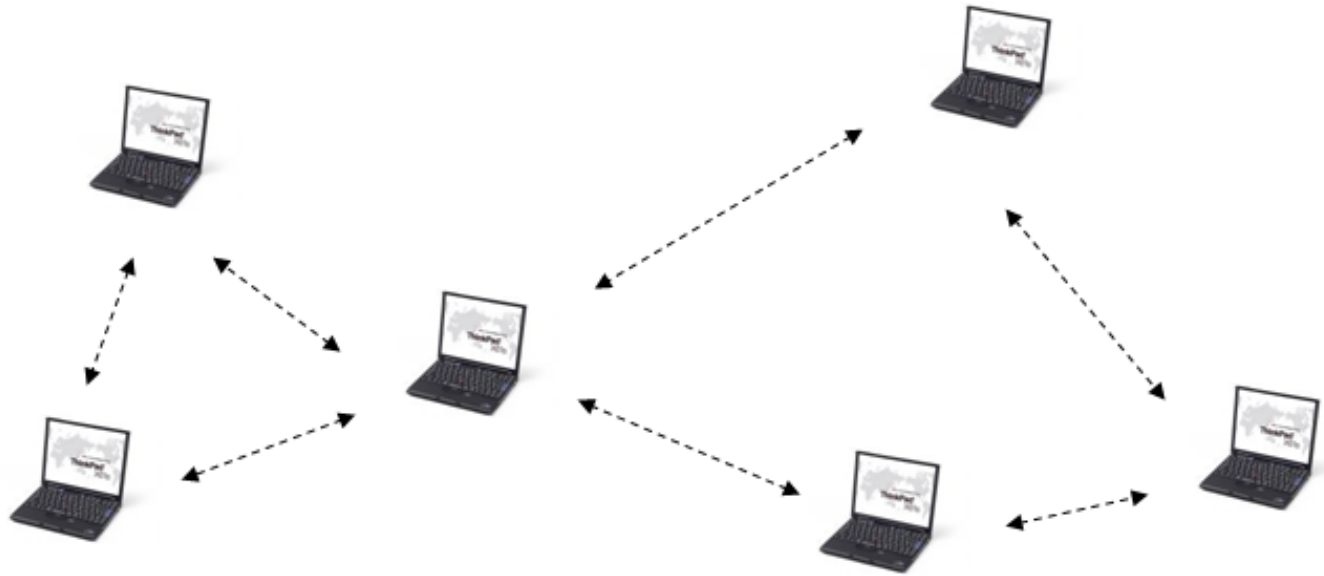
- First-order: (sub)gradient descent, gradient projection, continuation, coordinate descents, prox-point, conjugate gradient, forward backward splitting, accelerated first-order
- High-order: interior-point, semismooth Newton, hybrid 1st and 2nd order, BFGS, L-BFGS
- Final debiasing
- Duality: augmented Lagrangian, alternating direction, Bregman (original, linearized, split), primal-dual methods. They are applied to both primal and dual formulations.
- Homotopy: parametric quadratic programming, parameter continuation

Optimization/Algorithmic Techniques Used, Part 2

- Non-convex approximations to ℓ_0, ℓ_q minimization, reweighted itr
- Stochastic approximation (gradient, Hessian), sample-average approx
- Distributed and decentralized reconstruction
- Parallel and GPU-friendly algorithms
- Numerical linear algebra exploitation
- Heuristics. Solution structure may be domain specific.
- Greedy algorithms (OMP family), support detection
- Combinatorial algorithms, some running in sublinear time

Sources: Optimization Online and Rice CS Repository. We pick a few to discuss...

Distributed sparse optimization?



Consider

$$\min_X \mu\|X\|_{2,1} + \frac{1}{2}\|\mathcal{A}(X) - B\|_F^2$$

or

$$\min_X \mu\|X\|_* + \frac{1}{2}\|\mathcal{A}(X) - B\|_F^2.$$

$X = [x^1, x^2, \ldots, x^L]$ and $B = [b^1, b^2, \ldots, b^L]$ are all over the network.

Due to the large amount of data or security reasons, data and/or solutions are *not* allowed to be shared.

To finish up

Some thoughts:

- The work of finding structured solutions is interdisciplinary
- Recognizing the structure are important
- It is a common ground for existing and novel algorithms
- This field grows quickly, and its tools becoming standard techniques in computational mathematics and engineering

To finish up

Some thoughts:

- The work of finding structured solutions is interdisciplinary
- Recognizing the structure are important
- It is a common ground for existing and novel algorithms
- This field grows quickly, and its tools becoming standard techniques in computational mathematics and engineering

Benefits of structured solutions:

- cheap to store / transmit
- more meaningful, easy to understand, easy to use
- robust to errors
- easy to find

Acknowledgements: R. Chartrand for some nice figures, and Y. Xu for some slides; MIIS 2012 organizers; NSF, ONR, DARPA.

Discussion time. Questions?

Despite of many sparse optimization solvers, we still look for

- solvers that are both fast and stable
- solvers for distributed / decentralized large-scale data
- solvers that quickly produce solutions corresponding to a sequence of parameters
- ADM-kind solvers that deal with non-convex functions