Sparse Optimization

Mário A. T. Figueiredo

[1]Instituto de Telecomunicações,
Instituto Superior Técnico,
Universidade de Lisboa, Portugal

CIMI-ANITI School on Optimisation, Toulouse, 2021

(Based on other tutorials with Stephen Wright, University of Wisconsin, USA)

Outline

- **Part 1:** Sparse Optimization and Applications
- **Part 2:** First-Order Methods
- **Part 3:** Augmented Lagrangian Methods

Part 1: Sparse Optimization and Applications

Mário A. T. Figueiredo

[1]Instituto de Telecomunicações,
Instituto Superior Técnico,
Universidade de Lisboa, Portugal

CIMI-ANITI School on Optimisation, 2021

Inference via Optimization

Many **inference** problems are formulated as **optimization** problems:

- image reconstruction, restoration, denoising, segmentation, ...
- machine learning
  - ✓ supervised learning
  - ✓ unsupervised learning
  - ✓ many other tasks
- statistical inference
- decision theory
- system identification
- ...

Inference via Optimization

Many **inference** problems are formulated as **optimization** problems:

- image reconstruction, restoration, denoising, segmentation, ...
- machine learning
  - ✓ supervised learning
  - ✓ unsupervised learning
  - ✓ many other tasks
- statistical inference
- decision theory
- system identification
- ...

Standard formulation:

- observed data: y
- unknown mathematical object (signal, image, vector, matrix,...): x
- inference criterion:

$$\widehat{x} \in \arg\min_{x} g(x, y)$$

Inference via Optimization

Inference criterion:

$$\widehat{x} \in \arg\min_{x} g(x, y)$$

Question 1: how to build g? Where does it come from?

Answer: from the application domain (machine learning, signal processing, inverse problems, system identification, statistics, computer vision, bioinformatics,...);
... examples ahead.

Inference via Optimization

Inference criterion:

$$\widehat{x} \in \arg\min_{x} g(x, y) = \{x : \ g(x, y) \leq g(z, y), \ \forall_z\}$$

Question 1: how to build g? Where does it come from?

Answer: from the application domain (machine learning, signal processing, inverse problems, system identification, statistics, computer vision, bioinformatics,...);
... examples ahead.

Inference via Optimization

Inference criterion:

$$\widehat{x} \in \arg\min_{x} g(x, y) = \{x : \ g(x, y) \leq g(z, y), \ \forall_z\}$$

Question 1: how to build g? Where does it come from?

Answer: from the application domain (machine learning, signal processing, inverse problems, system identification, statistics, computer vision, bioinformatics,...);
... examples ahead.

Question 2: how to solve the optimization problem?

Answer: the focus of this tutorial.

Regularized Optimization

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

Regularized Optimization

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- $h(x, y) \rightarrow$ how well x "fits" / "explains" the data y;
  (data term, log-likelihood, loss function, observation model,...)

Regularized Optimization

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- $h(x, y) \rightarrow$ how well x "fits" / "explains" the data y; (data term, log-likelihood, loss function, observation model,...)
- $\psi(x) \rightarrow$ knowledge/constraints/structure: the **regularizer**
- $\tau \geq 0$: the **regularization parameter/constant**.

Regularized Optimization

Inference criterion: $\widehat{x} \in \arg\min_x g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- $h(x, y) \rightarrow$ how well x "fits" / "explains" the data y; (data term, log-likelihood, loss function, observation model,...)
- $\psi(x) \rightarrow$ knowledge/constraints/structure: the **regularizer**
- $\tau \geq 0$: the **regularization parameter/constant**.
- Since y is fixed, we often write simply $f(x) = h(x, y)$,

$$\min_x f(x) + \tau\psi(x)$$

Probabilistic/Bayesian Interpretations

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

Probabilistic/Bayesian Interpretations

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- Likelihood (observation model): $p(y|x) = \frac{1}{Z_I} \exp\big(-h(x, y)\big)$

Probabilistic/Bayesian Interpretations

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- Likelihood (observation model): $p(y|x) = \frac{1}{Z_l} \exp\big(-h(x, y)\big)$
- Prior: $p(x) = \frac{1}{Z_p} \exp\big(-\tau\psi(x)\big)$

Probabilistic/Bayesian Interpretations

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- Likelihood (observation model): $p(y|x) = \frac{1}{Z_l} \exp\big(-h(x, y)\big)$
- Prior: $p(x) = \frac{1}{Z_p} \exp\big(-\tau\psi(x)\big)$
- Posterior: $p(x|y) = \frac{p(y|x)\ p(x)}{p(y)}$

Probabilistic/Bayesian Interpretations

Inference criterion: $\widehat{x} \in \arg\min_x g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- Likelihood (observation model): $p(y|x) = \frac{1}{Z_l} \exp\big(-h(x, y)\big)$
- Prior: $p(x) = \frac{1}{Z_p} \exp\big(-\tau\psi(x)\big)$
- Posterior: $p(x|y) = \frac{p(y|x)\ p(x)}{p(y)}$
- Log-posterior: $\log\ p(x|y) = K(y) - h(x, y) - \tau\psi(x)$

Probabilistic/Bayesian Interpretations

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- Likelihood (observation model): $p(y|x) = \frac{1}{Z_l} \exp\big(-h(x, y)\big)$
- Prior: $p(x) = \frac{1}{Z_p} \exp\big(-\tau\psi(x)\big)$
- Posterior: $p(x|y) = \frac{p(y|x)\ p(x)}{p(y)}$
- Log-posterior: $\log\ p(x|y) = K(y) - h(x, y) - \tau\psi(x) = K(y) - g(x, y)$

Probabilistic/Bayesian Interpretations

Inference criterion: $\widehat{x} \in \arg\min_x g(x, y)$

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- Likelihood (observation model): $p(y|x) = \frac{1}{Z_l} \exp\big(-h(x, y)\big)$
- Prior: $p(x) = \frac{1}{Z_p} \exp\big(-\tau\psi(x)\big)$
- Posterior: $p(x|y) = \frac{p(y|x)\ p(x)}{p(y)}$
- Log-posterior: $\log\ p(x|y) = K(y) - h(x, y) - \tau\psi(x) = K(y) - g(x, y)$
- $\widehat{x}$ is a maximum a posteriori (MAP) estimate.

Regularizers

Inference criterion: $\min_x f(x) + \tau\psi(x)$

Typically, the unknown is a vector $x \in \mathbb{R}^n$

or a matrix $x \in \mathbb{R}^{n \times m}$

Regularizers

Inference criterion: $\min_x f(x) + \tau\psi(x)$

Typically, the unknown is a vector $x \in \mathbb{R}^n$
or a matrix $x \in \mathbb{R}^{n \times m}$

Common regularizers impose/encourage one (or a combination of) the following characteristics:

- small norm (vector or matrix)
- sparsity (few nonzeros)
- specific nonzero patterns (*e.g.*, group/tree structure)
- low-rank (matrix)
- smoothness or piece-wise smoothness
- ...

Unconstrained vs Constrained Formulations

- Tikhonov regularization: $\min_x \; f(x) + \tau\psi(x)$

Unconstrained vs Constrained Formulations

- Tikhonov regularization:

$$\min_x \; f(x) + \tau\psi(x)$$

- Morozov regularization:

$$\begin{aligned} \min_x \quad & \psi(x) \\ \text{subject to} \quad & f(x) \leq \varepsilon \end{aligned}$$

- Ivanov regularization:

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & \psi(x) \leq \delta \end{aligned}$$

Unconstrained vs Constrained Formulations

- Tikhonov regularization: $\min_x \; f(x) + \tau\psi(x)$
- Morozov regularization: $\min_x \; \psi(x) \quad \text{subject to} \quad f(x) \leq \varepsilon$
- Ivanov regularization: $\min_x \; f(x) \quad \text{subject to} \quad \psi(x) \leq \delta$

Under mild conditions, these are all *equivalent* (in a precise sense).

Which one is more convenient is problem-dependent.

Example: Under- and Over-Constrained Linear Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

Example: Under- and Over-Constrained Linear Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

- Trivial case, A is invertible: $x = A^{-1}y$

Example: Under- and Over-Constrained Linear Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

- Trivial case, A is invertible: $x = A^{-1}y$
- Over-determined system ($m > n$); least squares solution ($\text{rank}(A) = n$):

$$\widehat{x} = \arg\min_x \sum_{i=1}^{n} (y_i - (Ax)_i)^2 = \arg\min_x \|y - A\,x\|_2^2 = (A^T A)^{-1} A^T y$$

Example: Under- and Over-Constrained Linear Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

- Trivial case, A is invertible: $x = A^{-1}y$
- Over-determined system ($m > n$); least squares solution ($\text{rank}(A) = n$):

$$\widehat{x} = \arg\min_x \sum_{i=1}^{n} (y_i - (Ax)_i)^2 = \arg\min_x \|y - A\,x\|_2^2 = (A^T A)^{-1} A^T y$$

- Under-determined system ($m < n$); minimum norm solution ($\text{rank}(A) = m$):

$$\widehat{x} = \left\{ \begin{array}{l} \arg\min_x \|x\|_2^2 \\ \text{s.t. } Ax = y \end{array} \right\} = A^T (AA^T)^{-1} y$$

Example: Under- and Over-Constrained Linear Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

- Trivial case, A is invertible: $x = A^{-1}y$
- Over-determined system ($m > n$); least squares solution ($\text{rank}(A) = n$):

$$\widehat{x} = \arg\min_x \sum_{i=1}^{n} (y_i - (Ax)_i)^2 = \arg\min_x \|y - A\,x\|_2^2 = (A^T A)^{-1} A^T y$$

- Under-determined system ($m < n$); minimum norm solution ($\text{rank}(A) = m$):

$$\widehat{x} = \left\{ \begin{array}{l} \arg\min_x \|x\|_2^2 \\ \text{s.t. } Ax = y \end{array} \right\} = A^T (AA^T)^{-1} y$$

- Non-trivial cases: resort to optimization and regularization.

Example: Under- and Over-Constrained Linear Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

- Trivial case, A is invertible: $x = A^{-1}y$
- Over-determined system ($m > n$); least squares solution ($\text{rank}(A) = n$):

$$\widehat{x} = \arg\min_x \sum_{i=1}^{n} (y_i - (Ax)_i)^2 = \arg\min_x \|y - A\,x\|_2^2 = (A^T A)^{-1} A^T y$$

- Under-determined system ($m < n$); minimum norm solution ($\text{rank}(A) = m$):

$$\widehat{x} = \left\{ \begin{array}{l} \arg\min_x \|x\|_2^2 \\ \text{s.t. } Ax = y \end{array} \right\} = A^T (AA^T)^{-1} y$$

- Non-trivial cases: resort to optimization and regularization.
- Quadratic (Euclidean norm) losses and regularizers have a long, rich history: Gauss, Legendre, Wiener, Moore-Penrose, Tikhonov, ...

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n \times n}$, ...

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n \times n}$, ...

Some function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_+$ is a norm if it satisfies:

- $\|\alpha x\| = |\alpha| \, \|x\|$, for any $x \in \mathcal{V}$ and $\alpha \in \mathbb{R}$ (homogeneity);

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n\times n}$, ...

Some function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_+$ is a norm if it satisfies:

- $\|\alpha x\| = |\alpha| \, \|x\|$, for any $x \in \mathcal{V}$ and $\alpha \in \mathbb{R}$ (homogeneity);
- $\|x + x'\| \leq \|x\| + \|x'\|$, for any $x, x' \in \mathcal{V}$ (triangle inequality);

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n \times n}$, ...

Some function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_+$ is a norm if it satisfies:

- $\|\alpha x\| = |\alpha| \, \|x\|$, for any $x \in \mathcal{V}$ and $\alpha \in \mathbb{R}$ (homogeneity);
- $\|x + x'\| \leq \|x\| + \|x'\|$, for any $x, x' \in \mathcal{V}$ (triangle inequality);
- $\|x\| = 0 \;\Rightarrow\; x = 0$.

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n\times n}$, ...

Some function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_+$ is a norm if it satisfies:

- $\|\alpha x\| = |\alpha| \, \|x\|$, for any $x \in \mathcal{V}$ and $\alpha \in \mathbb{R}$ (homogeneity);
- $\|x + x'\| \leq \|x\| + \|x'\|$, for any $x, x' \in \mathcal{V}$ (triangle inequality);
- $\|x\| = 0 \;\Rightarrow\; x = 0$.

Examples:

- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_p = \left(\sum_i |x_i|^p\right)^{1/p}$ (called ℓ_p norm, for $p \geq 1$).

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n\times n}$, ...

Some function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_+$ is a norm if it satisfies:

- $\|\alpha x\| = |\alpha| \, \|x\|$, for any $x \in \mathcal{V}$ and $\alpha \in \mathbb{R}$ (homogeneity);
- $\|x + x'\| \leq \|x\| + \|x'\|$, for any $x, x' \in \mathcal{V}$ (triangle inequality);
- $\|x\| = 0 \;\Rightarrow\; x = 0$.

Examples:

- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_p = \left(\sum_i |x_i|^p\right)^{1/p}$ (called ℓ_p norm, for $p \geq 1$).
- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_\infty = \lim_{p\to\infty} \|x\|_p = \max\{|x_1|, ..., |x_n|\}$

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n\times n}$, ...

Some function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_+$ is a norm if it satisfies:

- $\|\alpha x\| = |\alpha| \, \|x\|$, for any $x \in \mathcal{V}$ and $\alpha \in \mathbb{R}$ (homogeneity);
- $\|x + x'\| \leq \|x\| + \|x'\|$, for any $x, x' \in \mathcal{V}$ (triangle inequality);
- $\|x\| = 0 \;\Rightarrow\; x = 0$.

Examples:

- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_p = \left(\sum_i |x_i|^p\right)^{1/p}$ (called ℓ_p norm, for $p \geq 1$).
- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_\infty = \lim_{p\to\infty} \|x\|_p = \max\{|x_1|, ..., |x_n|\}$
- $\mathcal{V} = \mathbb{R}^{n\times n}$, $\|X\|_* = \text{trace}\left(\sqrt{X^T X}\right)$ (matrix nuclear norm)

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n\times n}$, ...

Some function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_+$ is a norm if it satisfies:

- $\|\alpha x\| = |\alpha| \, \|x\|$, for any $x \in \mathcal{V}$ and $\alpha \in \mathbb{R}$ (homogeneity);
- $\|x + x'\| \leq \|x\| + \|x'\|$, for any $x, x' \in \mathcal{V}$ (triangle inequality);
- $\|x\| = 0 \;\Rightarrow\; x = 0$.

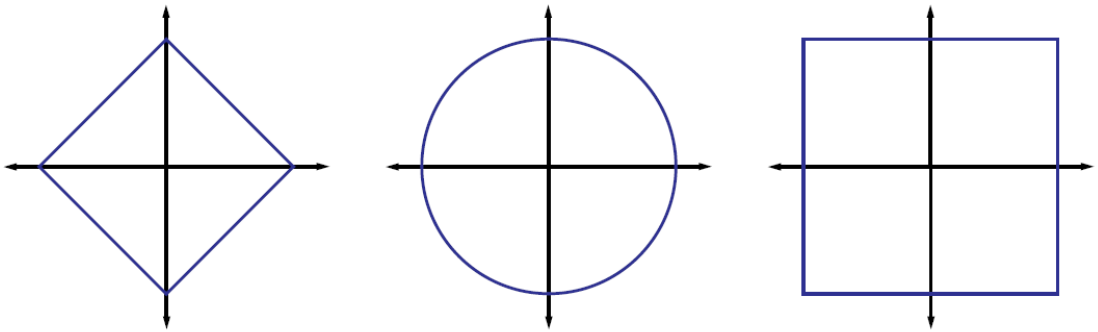
Examples:

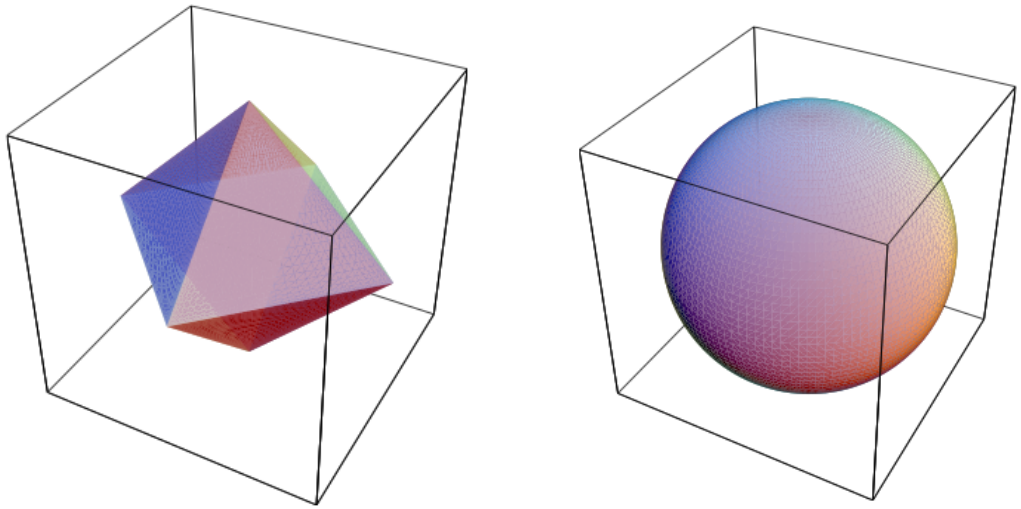
- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_p = \left(\sum_i |x_i|^p\right)^{1/p}$ (called ℓ_p norm, for $p \geq 1$).
- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_\infty = \lim_{p\to\infty} \|x\|_p = \max\{|x_1|, ..., |x_n|\}$
- $\mathcal{V} = \mathbb{R}^{n\times n}$, $\|X\|_* = \text{trace}\left(\sqrt{X^T X}\right)$ (matrix nuclear norm)

Also important (but not a norm): $\|x\|_0 = \lim_{p\to 0} \|x\|_p^p = |\{i : x_i \neq 0\}|$

Norm balls

Radius r ball in ℓ_p norm: $B_p(r) = \{x \in \mathbb{R}^n : \|x\|_p \leq r\}$





$p = 1$ $p = 2$

Examples: Back to Under-Constrained Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

- Under-determined system ($m < n$); minimum norm solution:

$$\widehat{x} = \left\{ \begin{array}{l} \arg\min\limits_{x} \|x\|_2^2 \\ \text{s.t. } Ax = y \end{array} \right\} = A^*(AA^*)^{-1}y$$

Examples: Back to Under-Constrained Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

- Under-determined system ($m < n$); minimum norm solution:

$$\widehat{x} = \left\{ \begin{array}{l} \arg\min\limits_{x} \|x\|_2^2 \\ \text{s.t. } Ax = y \end{array} \right\} = A^*(AA^*)^{-1}y \neq x \ \text{ (in general)}$$

Examples: Back to Under-Constrained Systems

A simple linear inverse problem: from $y = A\,x$, find x $\quad (A \in \mathbb{R}^{m \times n})$

- Under-determined system ($m < n$); minimum norm solution:

$$\widehat{x} = \left\{ \begin{array}{l} \arg\min\limits_{x} \|x\|_2^2 \\ \text{s.t. } Ax = y \end{array} \right\} = A^*(AA^*)^{-1}y \neq x \;\; \text{(in general)}$$

- Can we hope to recover x? Yes! ...if x is sparse enough ($\|x\|_0 < k$) and A satisfies some **conditions**, using

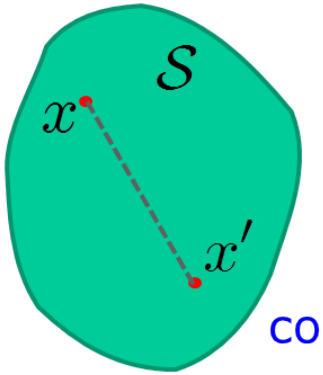
$$\begin{array}{rl} \widehat{x} = & \arg\min\limits_{x} \|x\|_0 \\ & \text{s.t. } Ax = y \end{array}$$

Several proofs, under different conditions (more later).

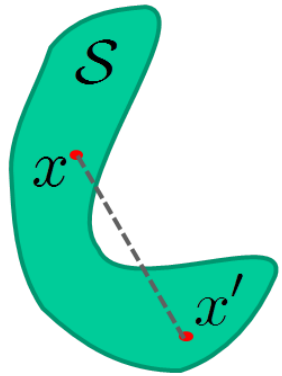
But, this is a hard problem! ℓ_0 "norm" is not **convex**.

Convex and strictly convex sets

$\mathcal{S}$ is convex if $x, x' \in \mathcal{S} \Rightarrow \forall \lambda \in [0,1], \quad \lambda x + (1-\lambda)x' \in \mathcal{S}$

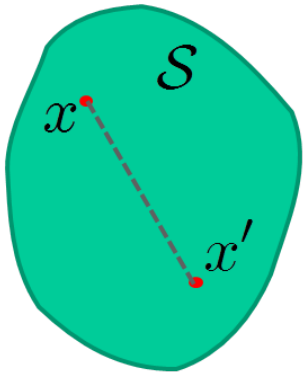


convex

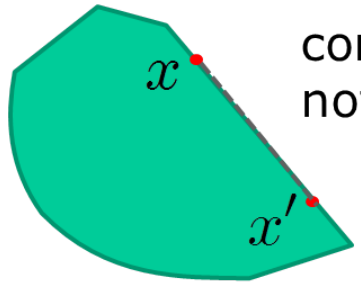


non-convex

$\mathcal{S}$ is strictly convex if $x, x' \in \mathcal{S} \Rightarrow \forall \lambda \in (0,1), \quad \lambda x + (1-\lambda)x' \in \text{int}(\mathcal{S})$



strictly convex



convex, but not strictly

Review of Basics: Convex Functions

Extended real valued function: $f : \mathbb{R}^N \to \bar{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$

Domain: $\mathrm{dom}(f) = \{x : f(x) \neq +\infty\}$

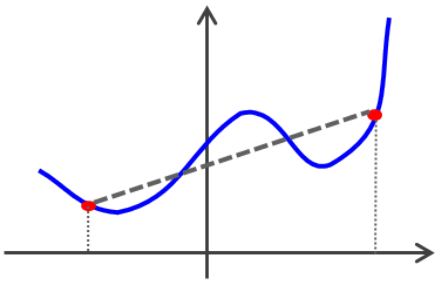
f is proper if $\mathrm{dom}(f) \neq \emptyset$

f is convex if

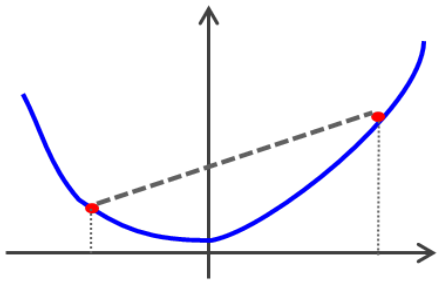
$$\forall \lambda \in [0,1], x, x' \in \mathrm{dom}(f) \quad f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x')$$

f is strictly convex if

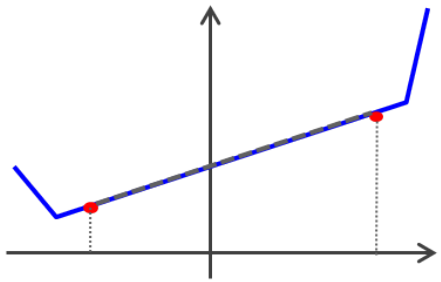
$$\forall \lambda \in (0,1), x, x' \in \mathrm{dom}(f) \quad f(\lambda x + (1-\lambda)x') < \lambda f(x) + (1-\lambda) f(x')$$



non-convex



strictly convex



convex, not strictly

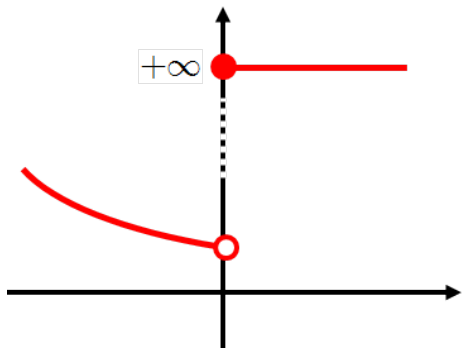
Lower Semi-Continuity: Why Is It Important?

A function $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ is lower semi-continuous (l.s.c.) if

$$\liminf_{x \to x_0} \geq f(x_0), \quad \text{for any } x_0 \in \text{dom}(f)$$

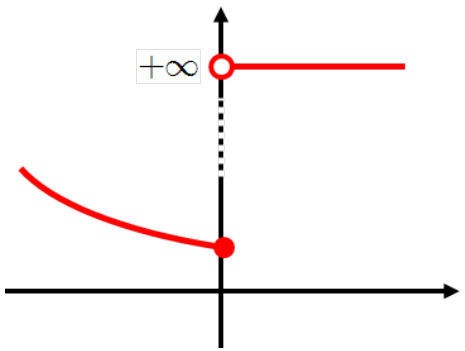
or, equivalently, $\{x : f(x) \leq \alpha\}$ is a closed set, for any $\alpha \in \mathbb{R}$

$$f(x) = \begin{cases} e^{-x}, & \text{if } x < 0 \\ +\infty, & \text{if } x \geq 0 \end{cases}$$



$\text{dom}(f) =]-\infty, 0[, \quad \arg\min_x f(x) = \emptyset$

$$f(x) = \begin{cases} e^{-x}, & \text{if } x \leq 0 \\ +\infty, & \text{if } x > 0 \end{cases}$$



$\text{dom}(f) =]-\infty, 0], \quad \arg\min_x f(x) = \{0\}$

Unless stated otherwise, we only consider l.s.c. functions.

Coercivity, Convexity, and Minima

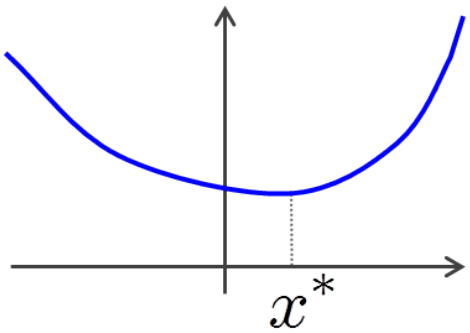
$$f : \mathbb{R}^N \to \bar{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$$

f is coercive if $\lim_{\|x\| \to +\infty} f(x) = +\infty$

if f is coercive, then $G \equiv \arg\min_x f(x)$ is a non-empty set

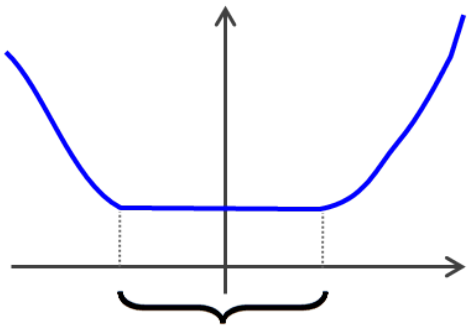
if f is strictly convex, then G has at most one element

coercive and strictly convex



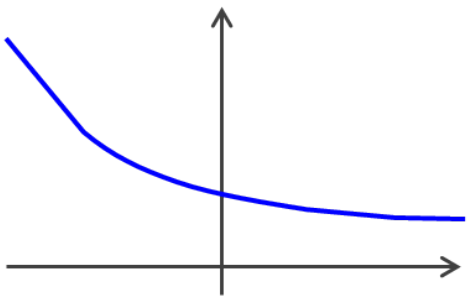
$G = \{x^*\}$

coercive, not strictly convex



G

convex, not coercive

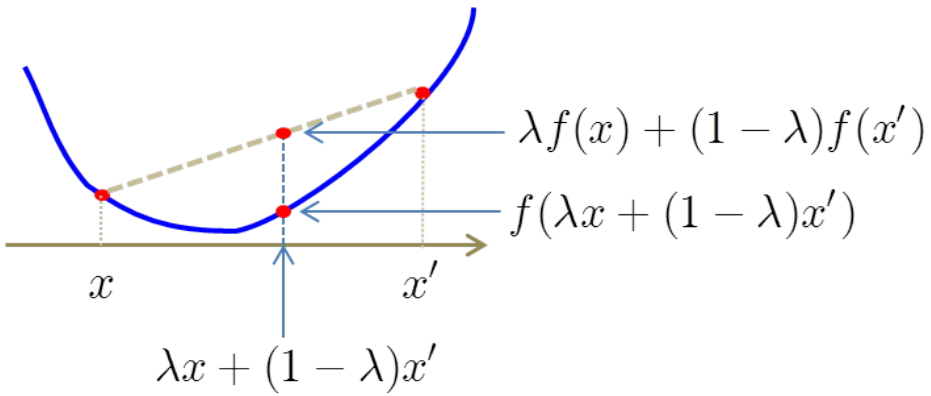


$G = \emptyset$

Another Important Concept: Strong Convexity

Recall the definition of convex function: $\forall \lambda \in [0,1]$,

$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x')$$



convexity

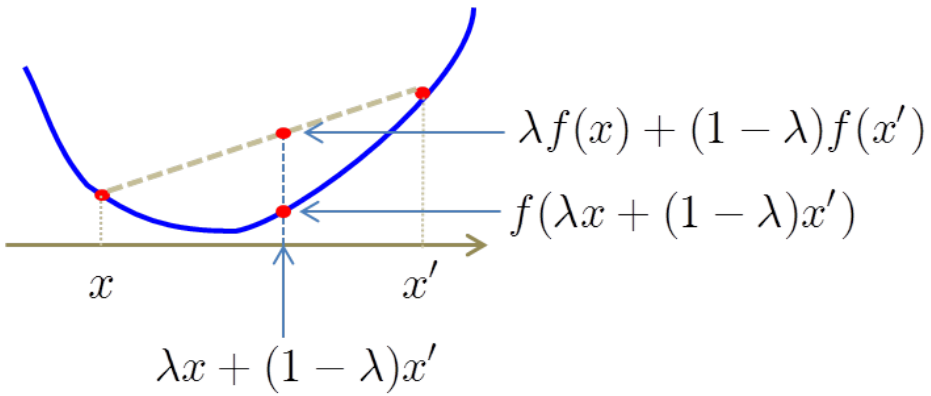
Another Important Concept: Strong Convexity

Recall the definition of convex function: $\forall \lambda \in [0,1]$,

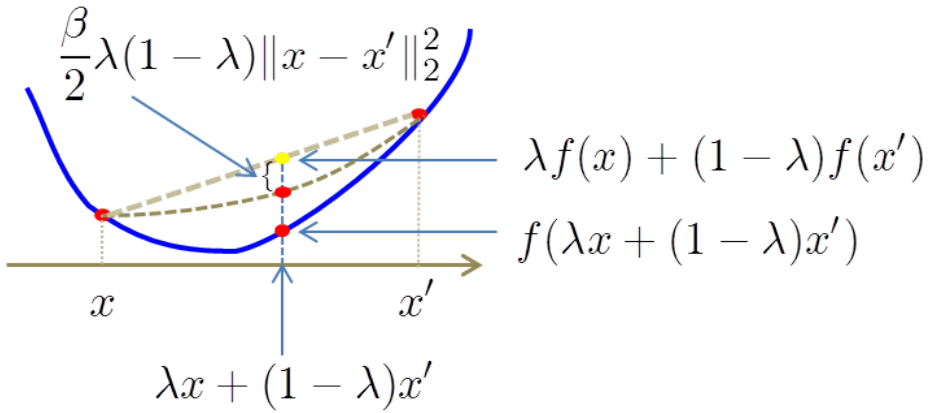
$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x')$$

A $\beta-$strongly convex function satisfies a stronger condition: $\forall \lambda \in [0,1]$

$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x') - \frac{\beta}{2}\lambda(1-\lambda)\|x - x'\|_2^2$$



convexity



strong convexity

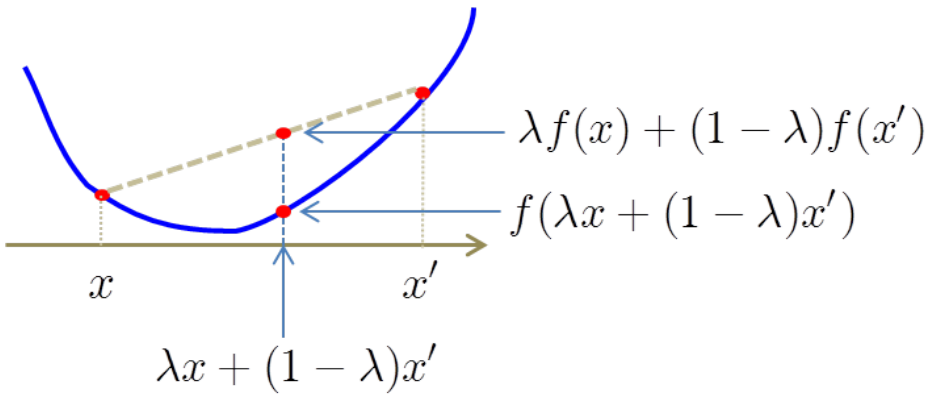
Another Important Concept: Strong Convexity

Recall the definition of convex function: $\forall \lambda \in [0,1]$,

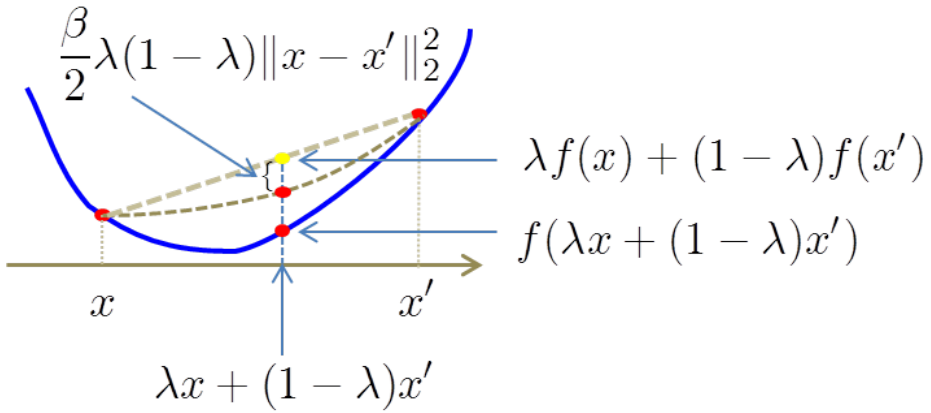
$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x')$$

A $\beta-$strongly convex function satisfies a stronger condition: $\forall \lambda \in [0,1]$

$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x') - \frac{\beta}{2}\lambda(1-\lambda)\|x - x'\|_2^2$$



convexity



strong convexity

Strong convexity $\Rightarrow$ strict convexity.

$\not\Leftarrow$

A Little More on Convex Functions

Let $f_1, ..., f_N : \mathbb{R}^n \to \bar{\mathbb{R}}$ be convex functions. Then

A Little More on Convex Functions

Let $f_1, ..., f_N : \mathbb{R}^n \to \bar{\mathbb{R}}$ be convex functions. Then

- $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $f(x) = \max\{f_1(x), ..., f_N(x)\}$, is convex.

A Little More on Convex Functions

Let $f_1, ..., f_N : \mathbb{R}^n \to \bar{\mathbb{R}}$ be convex functions. Then

- $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $f(x) = \max\{f_1(x), ..., f_N(x)\}$, is convex.
- $g : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $g(x) = f_1(L(x))$, where L is affine, is convex.

  Note: L is affine $\Leftrightarrow$ $L(x) - L(0)$ is linear; e.g. $L(x) = Ax + b$.

A Little More on Convex Functions

Let $f_1, ..., f_N : \mathbb{R}^n \to \bar{\mathbb{R}}$ be convex functions. Then

- $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $f(x) = \max\{f_1(x), ..., f_N(x)\}$, is convex.
- $g : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $g(x) = f_1(L(x))$, where L is affine, is convex.

  Note: L is affine $\Leftrightarrow$ $L(x) - L(0)$ is linear; e.g. $L(x) = Ax + b$.
- $h : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $h(x) = \sum_{j=1}^{N} \alpha_j f_j(x)$, for $\alpha_j > 0$, is convex.

A Little More on Convex Functions

Let $f_1, ..., f_N : \mathbb{R}^n \to \bar{\mathbb{R}}$ be convex functions. Then

- $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $f(x) = \max\{f_1(x), ..., f_N(x)\}$, is convex.
- $g : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $g(x) = f_1(L(x))$, where L is affine, is convex.

  Note: L is affine $\Leftrightarrow$ $L(x) - L(0)$ is linear; e.g. $L(x) = Ax + b$.
- $h : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $h(x) = \sum_{j=1}^{N} \alpha_j f_j(x)$, for $\alpha_j > 0$, is convex.

An important function: the indicator of a set $C \subset \mathbb{R}^n$,

$$\iota_C : \mathbb{R}^n \to \bar{\mathbb{R}}, \quad \iota_C(x) = \begin{cases} 0 & \Leftarrow \quad x \in C \\ +\infty & \Leftarrow \quad x \notin C \end{cases}$$

If C is a closed convex set, ι_C is a l.s.c. convex function.

The Case of Differentiable Functions

Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice differentiable and consider its Hessian matrix at x, denoted $\nabla^2 f(x)$ (or $Hf(x)$):

$$\left(\nabla^2 f(x)\right)_{ij} = \frac{\partial f}{\partial x_i \partial x_j}, \quad \text{for } i, j = 1, ..., n.$$

The Case of Differentiable Functions

Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice differentiable and consider its Hessian matrix at x, denoted $\nabla^2 f(x)$ (or $Hf(x)$):

$$\left(\nabla^2 f(x)\right)_{ij} = \frac{\partial f}{\partial x_i \partial x_j}, \quad \text{for } i, j = 1, ..., n.$$

- f is convex $\Leftrightarrow$ its Hessian $\nabla^2 f(x)$ is positive semidefinite $\forall_x$

The Case of Differentiable Functions

Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice differentiable and consider its Hessian matrix at x, denoted $\nabla^2 f(x)$ (or $Hf(x)$):

$$\left(\nabla^2 f(x)\right)_{ij} = \frac{\partial f}{\partial x_i \partial x_j}, \quad \text{for } i, j = 1, ..., n.$$

- f is convex $\Leftrightarrow$ its Hessian $\nabla^2 f(x)$ is positive semidefinite $\forall_x$
- f is strictly convex $\Leftarrow$ its Hessian $\nabla^2 f(x)$ is positive definite $\forall_x$

The Case of Differentiable Functions

Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice differentiable and consider its Hessian matrix at x, denoted $\nabla^2 f(x)$ (or $Hf(x)$):

$$\left(\nabla^2 f(x)\right)_{ij} = \frac{\partial f}{\partial x_i \partial x_j}, \quad \text{for } i, j = 1, ..., n.$$

- f is convex $\Leftrightarrow$ its Hessian $\nabla^2 f(x)$ is positive semidefinite $\forall_x$
- f is strictly convex $\Leftarrow$ its Hessian $\nabla^2 f(x)$ is positive definite $\forall_x$
- f is β-strongly convex $\Leftrightarrow$ its Hessian $\nabla^2 f(x) \succeq \beta I$, with $\beta > 0$, $\forall_x$.

More on the Relationship Between ℓ_1 and ℓ_0

Finding the sparsest solution is NP-hard (Muthukrishnan, 2005).

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_0 \\ &\quad \text{s. t. } \|Aw - y\|_2^2 \leq \delta \end{aligned}$$

More on the Relationship Between ℓ_1 and ℓ_0

Finding the sparsest solution is NP-hard (Muthukrishnan, 2005).

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_0 \\ &\quad \text{s. t. } \|Aw - y\|_2^2 \leq \delta \end{aligned}$$

The related best subset selection problem is also NP-hard (Amaldi and Kann, 1998; Davis et al., 1997).

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|Aw - y\|_2^2 \\ &\quad \text{s. t. } \|w\|_0 \leq \tau \end{aligned}$$

More on the Relationship Between ℓ_1 and ℓ_0

Finding the sparsest solution is NP-hard (Muthukrishnan, 2005).

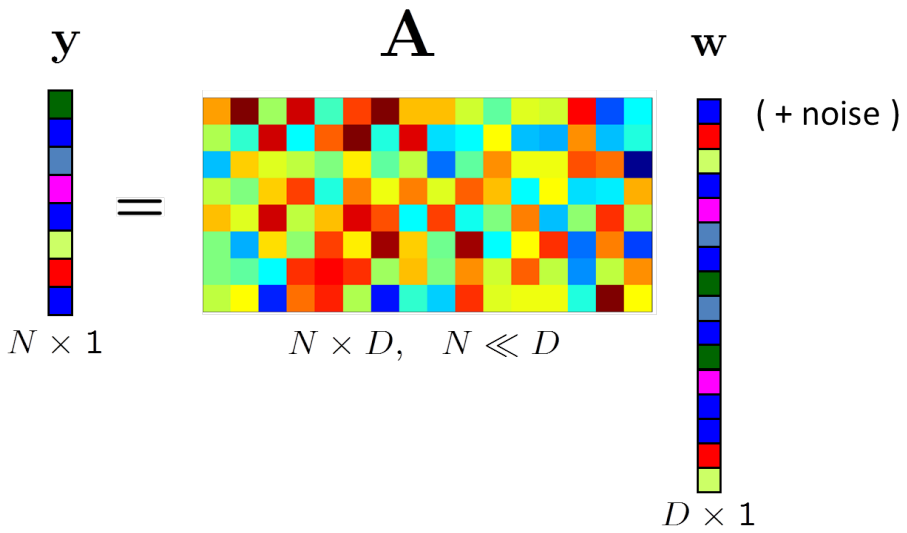
$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_0 \\ &\quad \text{s. t. } \|Aw - y\|_2^2 \leq \delta \end{aligned}$$

The related best subset selection problem is also NP-hard (Amaldi and Kann, 1998; Davis et al., 1997).

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|Aw - y\|_2^2 \\ &\quad \text{s. t. } \|w\|_0 \leq \tau \end{aligned}$$

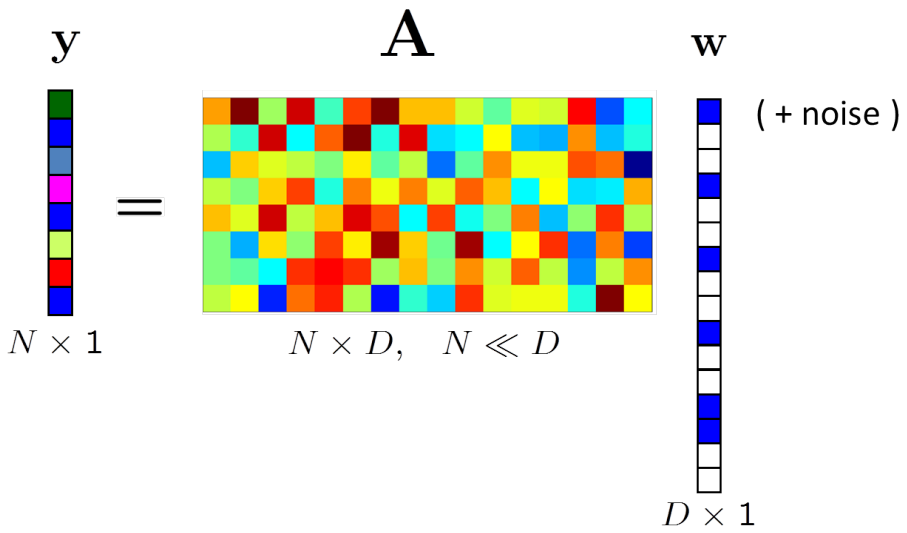
Under conditions, replacing ℓ_0 with ℓ_1 yields "similar" results: central issue in compressive sensing (CS) (Candès et al., 2006a; Donoho, 2006)

Compressive Sensing in a Nutshell



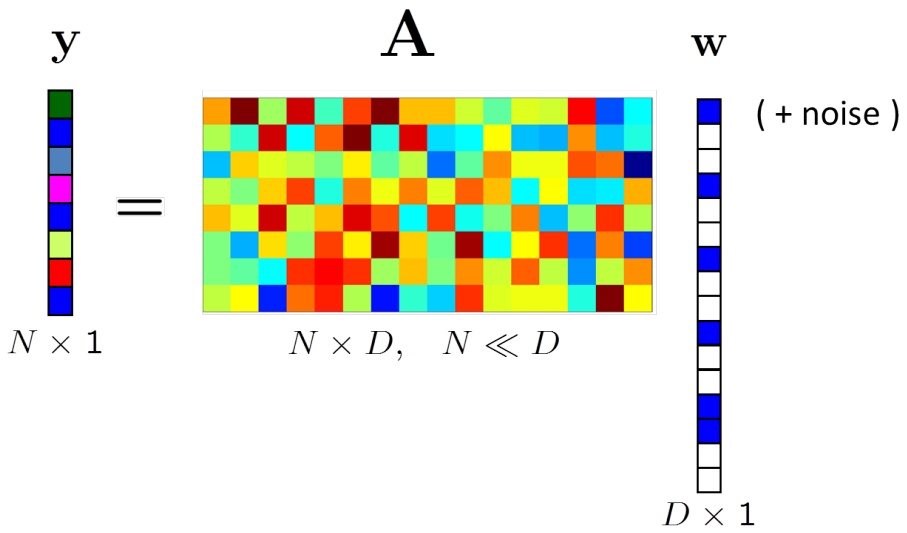
Even in the noiseless case, it seems impossible to recover w from y

Compressive Sensing in a Nutshell



Even in the noiseless case, it seems impossible to recover w from y ...unless, w is sparse and A has some properties.

Compressive Sensing in a Nutshell



Even in the noiseless case, it seems impossible to recover w from y ...unless, w is sparse and A has some properties.

If w is sparse enough and A has certain properties, then w is stably recovered via (Haupt and Nowak, 2006)

$$\begin{aligned} \widehat{w} \quad = \quad & \arg\min_{w} \|w\|_0 \\ & \text{s. t. } \|Aw - y\| \leq \delta \end{aligned}$$

NP-hard!

Compressive Sensing in a Nutshell

Under some conditions on A (e.g., the restricted isometry property (RIP)), ℓ_0 can be replaced with ℓ_1 (Candès et al., 2006b):

$$\begin{aligned} \widehat{w} \quad = \quad & \arg\min_{w} \|w\|_1 \\ & \text{subject to } \|Aw - y\| \leq \delta \end{aligned} \qquad \text{convex problem}$$

Compressive Sensing in a Nutshell

Under some conditions on A (e.g., the restricted isometry property (RIP)), ℓ_0 can be replaced with ℓ_1 (Candès et al., 2006b):

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_1 \\ &\quad \text{subject to } \|Aw - y\| \leq \delta \end{aligned} \qquad \text{convex problem}$$

Matrix A satisfies the RIP of order k, with constant $\delta_k \in (0, 1)$, if

$$\|w\|_0 \leq k \;\Rightarrow\; (1 - \delta_k)\|w\|_2^2 \leq \|Aw\|_2^2 \leq (1 + \delta_k)\|w\|_2^2$$

...i.e., for k-sparse vectors, A is approximately an isometry.

Compressive Sensing in a Nutshell

Under some conditions on A (e.g., the restricted isometry property (RIP)), ℓ_0 can be replaced with ℓ_1 (Candès et al., 2006b):

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_1 \\ &\quad \text{subject to } \|Aw - y\| \leq \delta \qquad \text{convex problem} \end{aligned}$$

Matrix A satisfies the RIP of order k, with constant $\delta_k \in (0, 1)$, if

$$\|w\|_0 \leq k \;\Rightarrow\; (1 - \delta_k)\|w\|_2^2 \leq \|Aw\|_2^2 \leq (1 + \delta_k)\|w\|_2^2$$

...i.e., for k-sparse vectors, A is approximately an isometry.

Other properties (spark and null space property (NSP)) can be used; caveat: checking RIP, NSP, spark is **NP-hard** (Tillmann and Pfetsch, 2012).

Examples: Back to Under-Constrained Systems

Let $\bar{x}$ be the sparsest solution of $Ax = y$, where $A \in \mathbb{R}^{m \times n}$ and $m < n$.

$$\bar{x} = \arg\min \|x\|_0 \ \text{ s.t. } \ Ax = y.$$

Consider the ℓ_1 norm version: $\min_x \|x\|_1 \ \text{ s.t. } \ Ax = y$

Examples: Back to Under-Constrained Systems

Let $\bar{x}$ be the sparsest solution of $Ax = y$, where $A \in \mathbb{R}^{m \times n}$ and $m < n$.

$$\bar{x} = \arg\min \|x\|_0 \ \text{ s.t. } \ Ax = y.$$

Consider the ℓ_1 norm version: $\min_x \|x\|_1 \ \text{ s.t. } \ Ax = y$

Advantage: this is a convex problem! Fact: **all norms are convex**.

Examples: Back to Under-Constrained Systems

Let $\bar{x}$ be the sparsest solution of $Ax = y$, where $A \in \mathbb{R}^{m \times n}$ and $m < n$.

$$\bar{x} = \arg\min \|x\|_0 \ \text{ s.t. } \ Ax = y.$$

Consider the ℓ_1 norm version: $\min_x \|x\|_1 \ \text{ s.t. } \ Ax = y$

Advantage: this is a convex problem! Fact: **all norms are convex**.

Of course, $\bar{x}$ solves this problem too, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \ \ \forall v \in \ker(A)$.

Recall: $\ker(A) = \{x \in \mathbb{R}^n : Ax = 0\}$ is the kernel (a.k.a. null space) of A.

Examples: Back to Under-Constrained Systems

Let $\bar{x}$ be the sparsest solution of $Ax = y$, where $A \in \mathbb{R}^{m \times n}$ and $m < n$.

$$\bar{x} = \arg\min \|x\|_0 \ \text{ s.t. } \ Ax = y.$$

Consider the ℓ_1 norm version: $\min_x \|x\|_1 \ \text{ s.t. } \ Ax = y$

Advantage: this is a convex problem! Fact: **all norms are convex**.

Of course, $\bar{x}$ solves this problem too, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \ \ \forall v \in \ker(A)$.

Recall: $\ker(A) = \{x \in \mathbb{R}^n : Ax = 0\}$ is the kernel (a.k.a. null space) of A.

Next: elementary analysis by Yin and Zhang (2008), based on work by Kashin (1977) and Garnaev and Gluskin (1984).

Equivalence Between ℓ_1 and ℓ_0 Optimization

- Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.
- Minimum ℓ_1 solution(s): $G = \arg\min \|x\|_1$ s.t. $Ax = y$.

Equivalence Between ℓ_1 and ℓ_0 Optimization

- Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.
- Minimum ℓ_1 solution(s): $G = \arg\min \|x\|_1$ s.t. $Ax = y$.
- $\bar{x} \in G$, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1$, $\forall v \in \ker(A)$

Equivalence Between ℓ_1 and ℓ_0 Optimization

- Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.
- Minimum ℓ_1 solution(s): $G = \arg\min \|x\|_1$ s.t. $Ax = y$.
- $\bar{x} \in G$, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \ \forall v \in \ker(A)$
- Let $S = \{i : \bar{x}_i \neq 0\}$ and $Z = \{1, ..., n\} \setminus S$

$$\|\bar{x} + v\|_1 = \|\bar{x}_S + v_S\|_1 + \|v_Z\|_1$$

Equivalence Between ℓ_1 and ℓ_0 Optimization

- Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.
- Minimum ℓ_1 solution(s): $G = \arg\min \|x\|_1$ s.t. $Ax = y$.
- $\bar{x} \in G$, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \ \forall v \in \ker(A)$
- Let $S = \{i : \bar{x}_i \neq 0\}$ and $Z = \{1, ..., n\} \setminus S$

$$\begin{aligned} \|\bar{x} + v\|_1 &= \|\bar{x}_S + v_S\|_1 + \|v_Z\|_1 \\ &\geq \|\bar{x}_S\|_1 + \|v_Z\|_1 - \|v_S\|_1 \qquad (\|a + b\| \geq \|a\| - \|b\|) \end{aligned}$$

Equivalence Between ℓ_1 and ℓ_0 Optimization

- Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.
- Minimum ℓ_1 solution(s): $G = \arg\min \|x\|_1$ s.t. $Ax = y$.
- $\bar{x} \in G$, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \ \forall v \in \ker(A)$
- Let $S = \{i : \bar{x}_i \neq 0\}$ and $Z = \{1, ..., n\} \setminus S$

$$\begin{aligned} \|\bar{x} + v\|_1 &= \|\bar{x}_S + v_S\|_1 + \|v_Z\|_1 \\ &\geq \|\bar{x}_S\|_1 + \|v_Z\|_1 - \|v_S\|_1 \qquad (\|a + b\| \geq \|a\| - \|b\|) \\ &= \|\bar{x}\|_1 + \|v\|_1 - 2\|v_S\|_1 \end{aligned}$$

Equivalence Between ℓ_1 and ℓ_0 Optimization

- Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.
- Minimum ℓ_1 solution(s): $G = \arg\min \|x\|_1$ s.t. $Ax = y$.
- $\bar{x} \in G$, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \ \forall v \in \ker(A)$
- Let $S = \{i : \bar{x}_i \neq 0\}$ and $Z = \{1, ..., n\} \setminus S$

$$
\begin{aligned}
\|\bar{x} + v\|_1 &= \|\bar{x}_S + v_S\|_1 + \|v_Z\|_1 \\
&\geq \|\bar{x}_S\|_1 + \|v_Z\|_1 - \|v_S\|_1 && (\|a + b\| \geq \|a\| - \|b\|) \\
&= \|\bar{x}\|_1 + \|v\|_1 - 2\|v_S\|_1 \\
&\geq \|\bar{x}\|_1 + \|v\|_1 - 2\sqrt{k}\|v\|_2. && (\|a\|_1 \leq \sqrt{n}\,\|a\|_2)
\end{aligned}
$$

Equivalence Between ℓ_1 and ℓ_0 Optimization

- Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.
- Minimum ℓ_1 solution(s): $G = \arg\min \|x\|_1$ s.t. $Ax = y$.
- $\bar{x} \in G$, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \ \forall v \in \ker(A)$
- Let $S = \{i : \bar{x}_i \neq 0\}$ and $Z = \{1, ..., n\} \setminus S$

$$
\begin{aligned}
\|\bar{x} + v\|_1 &= \|\bar{x}_S + v_S\|_1 + \|v_Z\|_1 \\
&\geq \|\bar{x}_S\|_1 + \|v_Z\|_1 - \|v_S\|_1 && (\|a+b\| \geq \|a\| - \|b\|) \\
&= \|\bar{x}\|_1 + \|v\|_1 - 2\|v_S\|_1 \\
&\geq \|\bar{x}\|_1 + \|v\|_1 - 2\sqrt{k}\|v\|_2. && (\|a\|_1 \leq \sqrt{n}\,\|a\|_2)
\end{aligned}
$$

Hence, $\bar{x} \in G$, if $\frac{1}{2}\frac{\|v\|_1}{\|v\|_2} \geq \sqrt{k}, \ \forall v \in \ker(A)$

...but, in general, we have only: $1 \leq \frac{\|v\|_1}{\|v\|_2} \leq \sqrt{n}$

Equivalence Between ℓ_1 and ℓ_0 Optimization

- Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.
- Minimum ℓ_1 solution(s): $G = \arg\min \|x\|_1$ s.t. $Ax = y$.
- $\bar{x} \in G$, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \ \forall v \in \ker(A)$
- Let $S = \{i : \bar{x}_i \neq 0\}$ and $Z = \{1, ..., n\} \setminus S$

$$
\begin{aligned}
\|\bar{x} + v\|_1 &= \|\bar{x}_S + v_S\|_1 + \|v_Z\|_1 \\
&\geq \|\bar{x}_S\|_1 + \|v_Z\|_1 - \|v_S\|_1 && (\|a+b\| \geq \|a\| - \|b\|) \\
&= \|\bar{x}\|_1 + \|v\|_1 - 2\|v_S\|_1 \\
&\geq \|\bar{x}\|_1 + \|v\|_1 - 2\sqrt{k}\|v\|_2. && (\|a\|_1 \leq \sqrt{n}\,\|a\|_2)
\end{aligned}
$$

Hence, $\bar{x} \in G$, if $\frac{1}{2}\frac{\|v\|_1}{\|v\|_2} \geq \sqrt{k}, \ \forall v \in \ker(A)$

...but, in general, we have only: $1 \leq \frac{\|v\|_1}{\|v\|_2} \leq \sqrt{n}$

However, we may have $\frac{\|v\|_1}{\|v\|_2} \gg 1$, if v is restricted to a random subspace.

Bounding the ℓ_1/ℓ_2 Ratio in Random Matrices

If the elements of $A \in \mathbb{R}^{m \times n}$ are sampled i.i.d. from $\mathcal{N}(0,1)$ (zero mean, unit variance Gaussian), then, with high probability,

$$\frac{\|v\|_1}{\|v\|_2} \geq \frac{C\sqrt{m}}{\sqrt{\log(n/m)}}, \quad \text{for all } v \in \ker(A),$$

for some constant C (based on concentration of measure phenomena).

Thus, with high probability, $\bar{x} \in G$, if

$$m \geq \frac{4}{C^2} k \log n$$

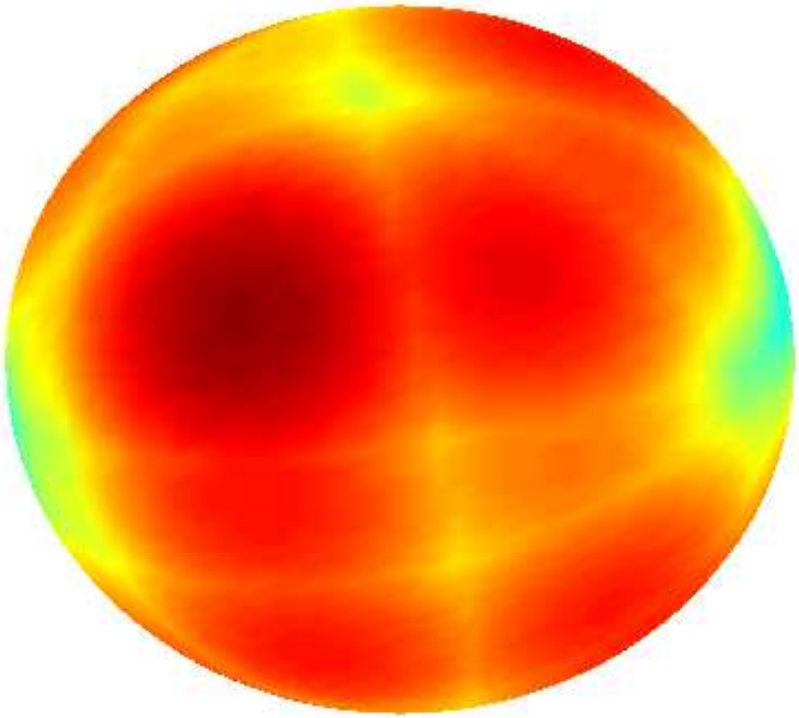
Conclusion: Can solve under-determined system, where A has i.i.d. $\mathcal{N}(0,1)$ elements, by solving

$$\min_x \|x\|_1 \;\; s.t. \;\; Ax = b,$$

(a convex problem), if the solution is sparse enough.

Ratio $\|v\|_1/\|v\|_2$ on Random Null Spaces

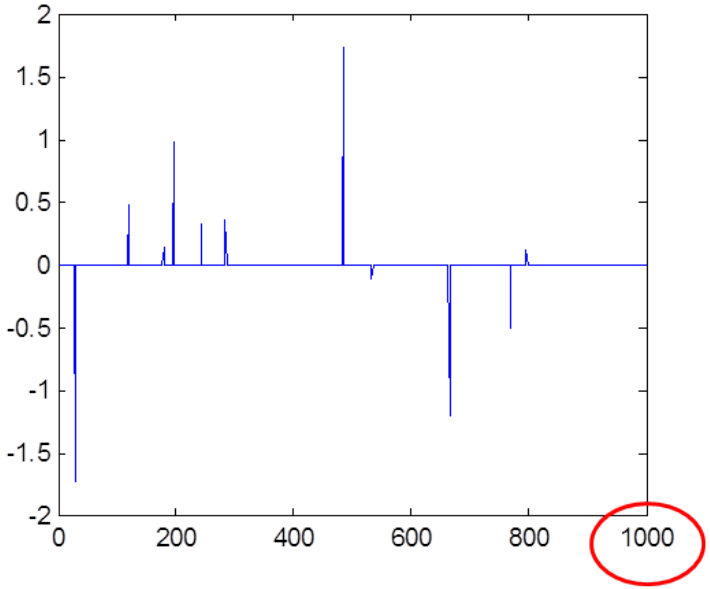
Random $A \in \mathbb{R}^{4\times 7}$, showing ratio $\|v\|_1$ for $v \in \ker(A)$ with $\|v\|_2 = 1$



Blue: $\|v\|_1 \approx 1$. Red: ratio $\approx \sqrt{7}$. Note that $\|v\|_1$ is well away from the lower bound of 1 over the whole nullspace.

When Data is Noisy

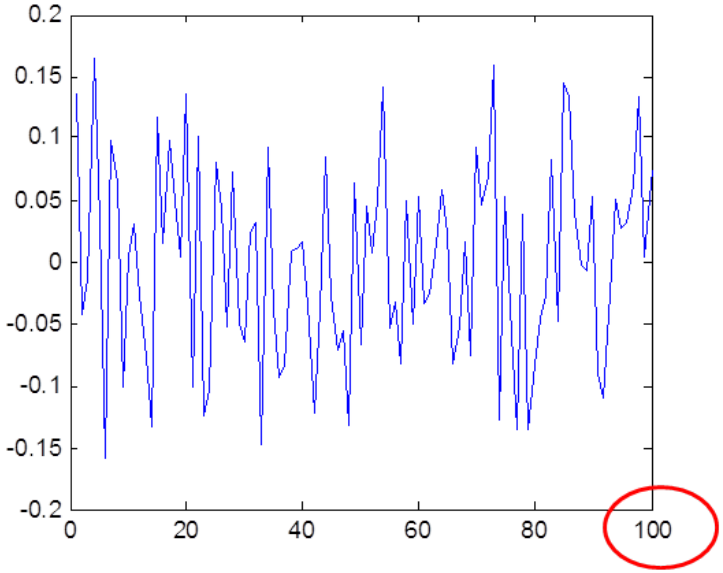
Sparse vector $\mathbf{x}$



$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$$

Random matrix

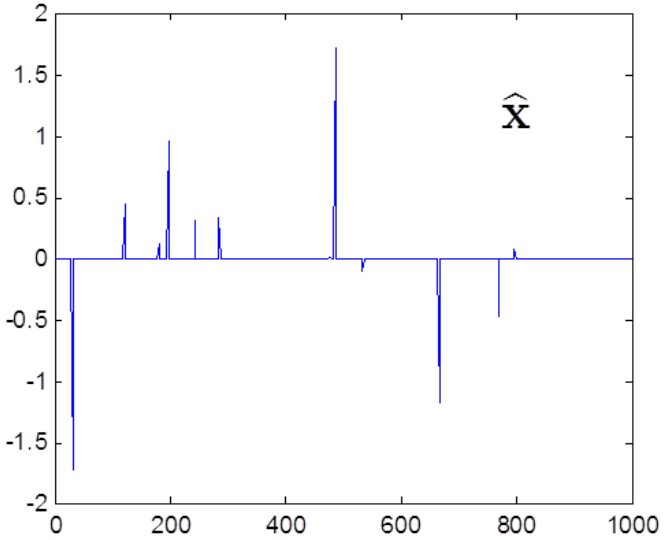
Observed data $\mathbf{y}$



Under certain conditions, "perfect" recovery is possible

$$\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \left\{ \|\mathbf{y} - \mathbf{A}\mathbf{x}\|^2 + 2\,\lambda\,\|\mathbf{x}\|_1 \right\}$$

[Candès, Romberg, Tao, 2004 – 2006]
[Donoho, 2006]



The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

$$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \quad \text{or} \quad \min_x \|Ax - y\|_2^2 \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

or, more generally,

$$\min_x f(x) + \lambda\|x\|_1 \quad \text{or} \quad \min_x f(x) \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

$$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \quad \text{or} \quad \min_x \|Ax - y\|_2^2 \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

  or, more generally,

$$\min_x f(x) + \lambda\|x\|_1 \quad \text{or} \quad \min_x f(x) \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

- Widely used in statistics, signal processing, neural networks, ...

The Ubiquitous

- Lasso (*least ab*) (Tibshirani, 1996)
  a.k.a. *basis pu*

$$\min_x \frac{1}{2}\|Ax \quad \text{s.t.} \quad \|x\|_1 \leq \delta$$

or, more gene

$$\min_x \quad x\|_1 \leq \delta$$

- Widely used in networks, ...

- Geology/geophysics
  - Claerbout and Muir (1973)
  - Taylor et al. (1979)
  - Levy and Fullager (1981)
  - Oldenburg et al. (1983)
  - Santosa and Symes (1988)
- Radio astronomy
  - Högbom (1974)
  - Schwarz (1978)
- Fourier transform spectroscopy
  - Kawata et al. (1983)
  - Mammone (1983)
  - Minami et al. (1985)
- NMR spectroscopy
  - Barkhuijsen (1985)
  - Newman (1988)
- Medical ultrasound
  - Papoulis and Chamzas (1979)

from (Goyal et al, 2010)

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

$$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \quad \text{or} \quad \min_x \|Ax - y\|_2^2 \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

  or, more generally,

$$\min_x f(x) + \lambda\|x\|_1 \quad \text{or} \quad \min_x f(x) \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

- Widely used in statistics, signal processing, neural networks, ...
- Many extensions: namely to express structured sparsity (more later).

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

$$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \quad \text{or} \quad \min_x \|Ax - y\|_2^2 \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

  or, more generally,

$$\min_x f(x) + \lambda\|x\|_1 \quad \text{or} \quad \min_x f(x) \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

- Widely used in statistics, signal processing, neural networks, ...
- Many extensions: namely to express structured sparsity (more later).
- Why does ℓ_1 yield sparse solutions? (next slides)

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

$$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \quad \text{or} \quad \min_x \|Ax - y\|_2^2 \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

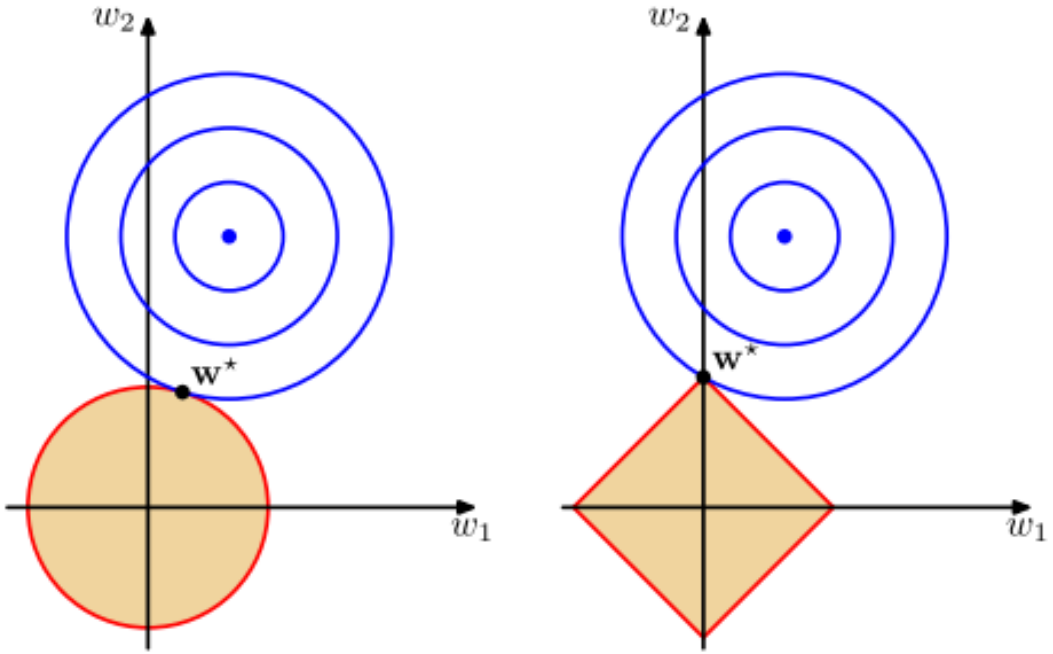
  or, more generally,

$$\min_x f(x) + \lambda\|x\|_1 \quad \text{or} \quad \min_x f(x) \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

- Widely used in statistics, signal processing, neural networks, ...
- Many extensions: namely to express structured sparsity (more later).
- Why does ℓ_1 yield sparse solutions? (next slides)
- How to solve these problems? (this tutorial)

Why ℓ_1 Yields Sparse Solution

$$w^* = \begin{array}{ll} \arg\min_w & \|Aw - y\|_2^2 \\ \text{s.t.} & \|w\|_2 \leq \delta \end{array} \quad \text{vs} \quad w^* = \begin{array}{ll} \arg\min_w & \|Aw - y\|_2^2 \\ \text{s.t.} & \|w\|_1 \leq \delta \end{array}$$



Why ℓ_1 Yields Sparse Solution

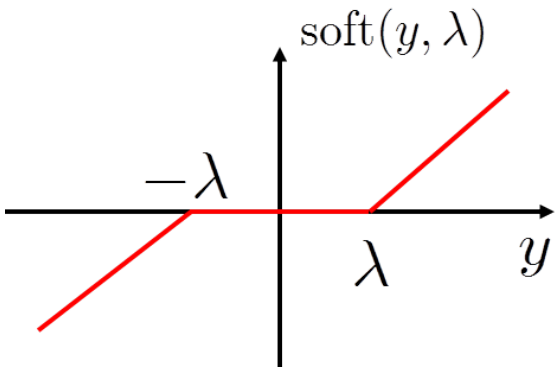
The simplest problem with ℓ_1 regularization

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \lambda|w| = \text{soft}(y,\lambda) = \begin{cases} y-\lambda & \Leftarrow \quad y > \lambda \\ 0 & \Leftarrow \quad |y| \leq \lambda \\ y+\lambda & \Leftarrow \quad y < -\lambda \end{cases}$$

Why ℓ_1 Yields Sparse Solution

The simplest problem with ℓ_1 regularization

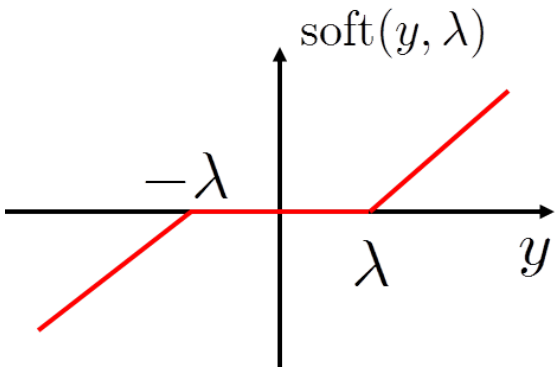
$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \lambda|w| = \text{soft}(y,\lambda) = \begin{cases} y-\lambda & \Leftarrow \quad y > \lambda \\ 0 & \Leftarrow \quad |y| \leq \lambda \\ y+\lambda & \Leftarrow \quad y < -\lambda \end{cases}$$



Why ℓ_1 Yields Sparse Solution

The simplest problem with ℓ_1 regularization

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \lambda|w| = \text{soft}(y,\lambda) = \begin{cases} y-\lambda & \Leftarrow \quad y > \lambda \\ 0 & \Leftarrow \quad |y| \leq \lambda \\ y+\lambda & \Leftarrow \quad y < -\lambda \end{cases}$$

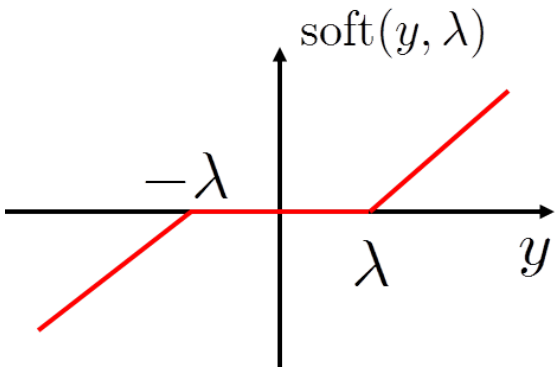


...by the way, how was this solved? (more later).

Why ℓ_1 Yields Sparse Solution

The simplest problem with ℓ_1 regularization

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \lambda|w| = \text{soft}(y,\lambda) = \begin{cases} y-\lambda & \Leftarrow \quad y > \lambda \\ 0 & \Leftarrow \quad |y| \leq \lambda \\ y+\lambda & \Leftarrow \quad y < -\lambda \end{cases}$$



...by the way, how was this solved? (more later).

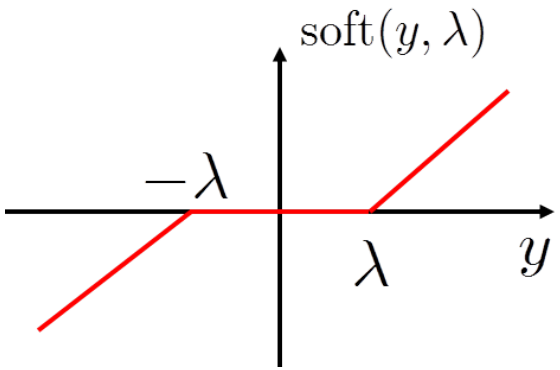
Contrast with the squared ℓ_2 (ridge) regularizer (linear scaling):

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \frac{\lambda}{2}w^2 = \frac{1}{1+\lambda}y$$

Why ℓ_1 Yields Sparse Solution

The simplest problem with ℓ_1 regularization

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \lambda|w| = \text{soft}(y,\lambda) = \begin{cases} y-\lambda & \Leftarrow \quad y > \lambda \\ 0 & \Leftarrow \quad |y| \leq \lambda \\ y+\lambda & \Leftarrow \quad y < -\lambda \end{cases}$$



...by the way, how was this solved? (more later).

Contrast with the squared ℓ_2 (ridge) regularizer (linear scaling):

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \frac{\lambda}{2}w^2 = \frac{1}{1+\lambda}y \qquad \text{(zero iff } y = 0\text{)}$$

More on the Relationship Between ℓ_1 and ℓ_0

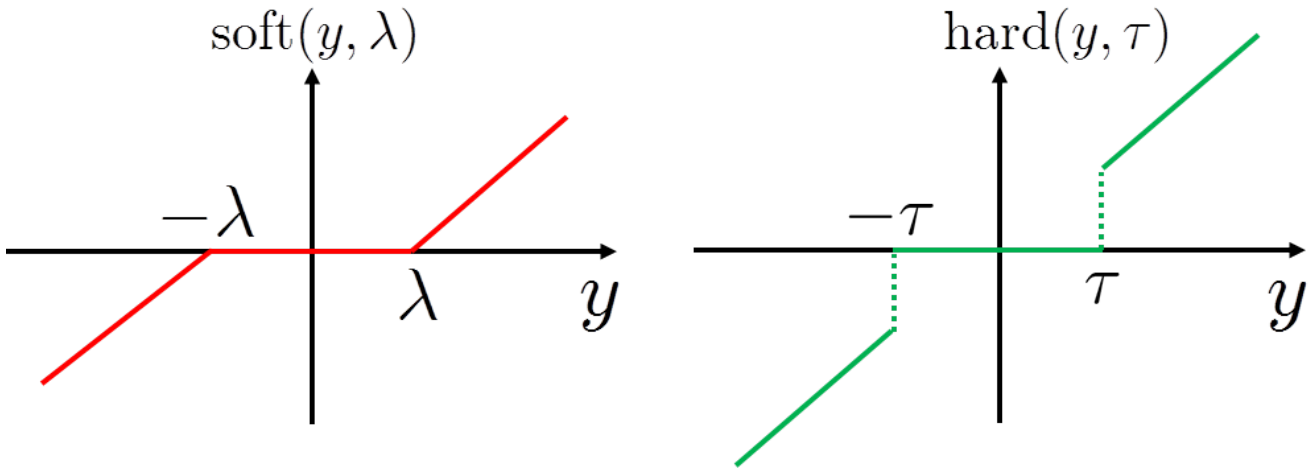
The ℓ_0 "norm" (number of non-zeros): $\|w\|_0 = |\{i : w_i \neq 0\}|$.
Not a norm, not convex, but in the simple case...

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \lambda|w|_0 = \text{hard}(y, \sqrt{2\lambda}) = \begin{cases} y & \Leftarrow \quad |y| > \sqrt{2\lambda} \\ 0 & \Leftarrow \quad |y| \leq \sqrt{2\lambda} \end{cases}$$

More on the Relationship Between ℓ_1 and ℓ_0

The ℓ_0 "norm" (number of non-zeros): $\|w\|_0 = |\{i : w_i \neq 0\}|$.
Not a norm, not convex, but in the simple case...

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \lambda|w|_0 = \text{hard}(y, \sqrt{2\lambda}) = \begin{cases} y & \Leftarrow \quad |y| > \sqrt{2\lambda} \\ 0 & \Leftarrow \quad |y| \leq \sqrt{2\lambda} \end{cases}$$



Another Application: Images

Natural images are well represented by a few coefficients in some bases.

- Images ($N \times M \equiv n$ pixels) are represented by vectors $x \in \mathbb{R}^n$

Another Application: Images

Natural images are well represented by a few coefficients in some bases.

- Images ($N \times M \equiv n$ pixels) are represented by vectors $x \in \mathbb{R}^n$
- Typical images have representations $x = Ww$ that are sparse ($\|w\|_0 \ll n$) on some bases ($W^T W = WW^T = I$), such as wavelets.

Original 1000×1000 image $x \in \mathbb{R}^{10^6}$

...only its 25000 largest coefficients.

Another Application: Images

Natural images are well represented by a few coefficients in some bases.

- Images ($N \times M \equiv n$ pixels) are represented by vectors $x \in \mathbb{R}^n$
- Typical images have representations $x = Ww$ that are sparse ($\|w\|_0 \ll n$) on some bases ($W^T W = WW^T = I$), such as wavelets.

Original 1000×1000 image $x \in \mathbb{R}^{10^6}$

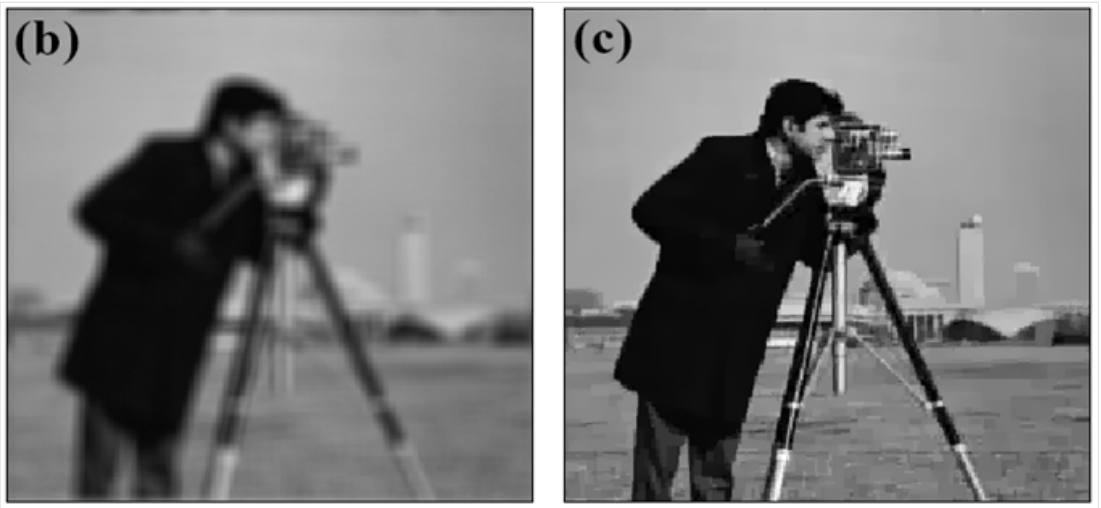
...only its 25000 largest coefficients.

- Also (even more) true with an over-complete tight frame; W is "fat" (more columns than rows) and $WW^T = I$, but $W^T W \neq I$.

Application to Image Deblurring/Deconvolution

blurred

restored



$$\widehat{\mathbf{x}} \in \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \tau \|\mathbf{x}\|_1$$

$$\mathbf{A} = \mathbf{B}\mathbf{W}$$

convolution (blur)

wavelet basis (or tight frame)

Application to Magnetic Resonance Imaging

$$\widehat{\mathbf{x}} \in \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \tau \|\mathbf{x}\|_1$$

$$\mathbf{A} = \mathbf{M}\mathbf{U}\mathbf{W}$$

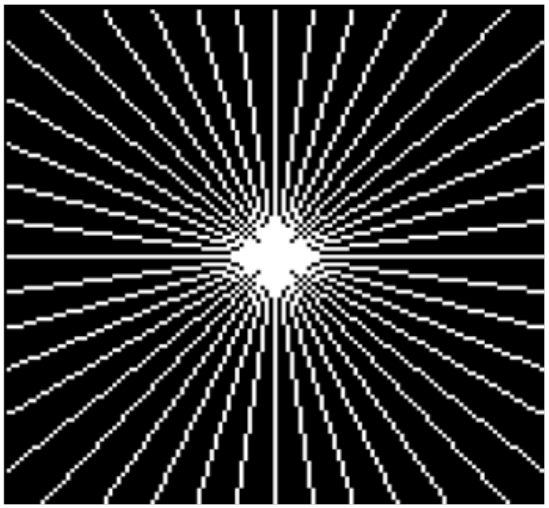
binary mask

discrete Fourier transform

wavelet basis (or tight frame)

original

acquired slices in DFT domain



reconstruction $\mathbf{W}\widehat{\mathbf{x}}$

Machine/Statistical Learning: Linear Regression

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature/variable vectors) and $y_i \in \mathbb{R}$ (outputs).

Goal: find "good" linear function: $\widehat{y} = \sum_{j=1}^{d} w_j x_j + w_{d+1} = [x^T 1] w$

Machine/Statistical Learning: Linear Regression

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature/variable vectors) and $y_i \in \mathbb{R}$ (outputs).

Goal: find "good" linear function: $\widehat{y} = \sum_{j=1}^{d} w_j x_j + w_{d+1} = [x^T 1]w$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Mean squared error: $\min_w \mathbb{E}\big(Y - [X^T 1]w\big)^2$ impossible! $P_{X,Y}$ unknown

Machine/Statistical Learning: Linear Regression

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature/variable vectors) and $y_i \in \mathbb{R}$ (outputs).

Goal: find "good" linear function: $\widehat{y} = \sum_{j=1}^{d} w_j x_j + w_{d+1} = [x^T 1]w$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Mean squared error: $\min_w \mathbb{E}\big(Y - [X^T 1]w\big)^2$ impossible! $P_{X,Y}$ unknown

Empirical error: $\min_w \frac{1}{N} \sum_{i=1}^{N} \big(y_i - [x_i^T 1]w\big)^2 = \min_w \frac{1}{N} \|y - Aw\|_2^2,$

design matrix: $A_{ij} = (x_i)_j$ (j-th component of i-th sample, $A_{i(d+1)} = 1$)

Machine/Statistical Learning: Linear Regression

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature/variable vectors) and $y_i \in \mathbb{R}$ (outputs).

Goal: find "good" linear function: $\widehat{y} = \sum_{j=1}^{d} w_j x_j + w_{d+1} = [x^T 1]w$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Mean squared error: $\min_w \mathbb{E}\big(Y - [X^T 1]w\big)^2$ impossible! $P_{X,Y}$ unknown

Empirical error: $\min_w \frac{1}{N} \sum_{i=1}^{N} \big(y_i - [x_i^T 1]w\big)^2 = \min_w \frac{1}{N} \|y - Aw\|_2^2,$

design matrix: $A_{ij} = (x_i)_j$ (j-th component of i-th sample, $A_{i(d+1)} = 1$)

Regularization: $\min_w \|y - Aw\|_2^2 + \tau\psi(w)$

Machine/Statistical Learning: Linear Classification

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature vectors) and $y_i \in \{-1, +1\}$ (labels).

Goal: find "good" linear classifier (*i.e.*, find the optimal weights):

$$\widehat{y} = \text{sign}([x^T 1]w) = \text{sign}\left(w_{d+1} + \sum_{j=1}^{d} w_j x_j\right)$$

Machine/Statistical Learning: Linear Classification

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature vectors) and $y_i \in \{-1, +1\}$ (labels).

Goal: find "good" linear classifier (*i.e.*, find the optimal weights):

$$\widehat{y} = \text{sign}([x^T 1]w) = \text{sign}\left(w_{d+1} + \sum_{j=1}^{d} w_j x_j\right)$$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Expected error: $\min_{w \in \mathbb{R}^{d+1}} \mathbb{E}\left(1_{Y([X^T 1]w)<0}\right)$ impossible! $P_{X,Y}$ unknown

Machine/Statistical Learning: Linear Classification

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature vectors) and $y_i \in \{-1, +1\}$ (labels).

Goal: find "good" linear classifier (*i.e.*, find the optimal weights):

$$\widehat{y} = \text{sign}([x^T 1]w) = \text{sign}\Big(w_{d+1} + \sum_{j=1}^{d} w_j x_j\Big)$$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Expected error: $\min_{w \in \mathbb{R}^{d+1}} \mathbb{E}\big(1_{Y([X^T 1]w)<0}\big)$ impossible! $P_{X,Y}$ unknown

Empirical error (EE): $\min_w \frac{1}{N} \sum_{i=1}^{N} h\big(\underbrace{y_i\,([x_i^T 1]w)}_{\text{margin}}\big)$, where $h(z) = 1_{z<0}$.

Machine/Statistical Learning: Linear Classification

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature vectors) and $y_i \in \{-1, +1\}$ (labels).

Goal: find "good" linear classifier (*i.e.*, find the optimal weights):

$$\widehat{y} = \text{sign}([x^T 1]w) = \text{sign}\Big(w_{d+1} + \sum_{j=1}^{d} w_j x_j\Big)$$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Expected error: $\min_{w \in \mathbb{R}^{d+1}} \mathbb{E}\big(1_{Y([X^T 1]w)<0}\big)$ impossible! $P_{X,Y}$ unknown

Empirical error (EE): $\min_w \frac{1}{N} \sum_{i=1}^{N} h\big(\underbrace{y_i\,([x_i^T 1]w)}_{\text{margin}}\big)$, where $h(z) = 1_{z<0}$.

Convexification: EE neither convex nor differentiable (NP-hard problem).
Solution: replace $h : \mathbb{R} \to \{0, 1\}$ with convex loss $L : \mathbb{R} \to \mathbb{R}_+$.

Machine/Statistical Learning: Linear Classification

Criterion: $$\min_{w} \underbrace{\sum_{i=1}^{N} L\big(\underbrace{y_i \, (w^T x_i + b)}_{\text{margin}} \big)}_{f(w)} + \tau \psi(w)$$

Convex loss: $L : \mathbb{R} \to \mathbb{R}_+$ is a (preferably convex) loss function.

Machine/Statistical Learning: Linear Classification

Criterion: $$\min_{w} \underbrace{\sum_{i=1}^{N} L\big(\underbrace{y_i\,(w^T x_i + b)}_{\text{margin}}\big)}_{f(w)} + \tau\psi(w)$$

Convex loss: $L : \mathbb{R} \to \mathbb{R}_+$ is a (preferably convex) loss function.

Regularizer: $\psi = \ell_1 \quad \Rightarrow \quad$ encourage sparseness $\quad \Rightarrow$ feature selection

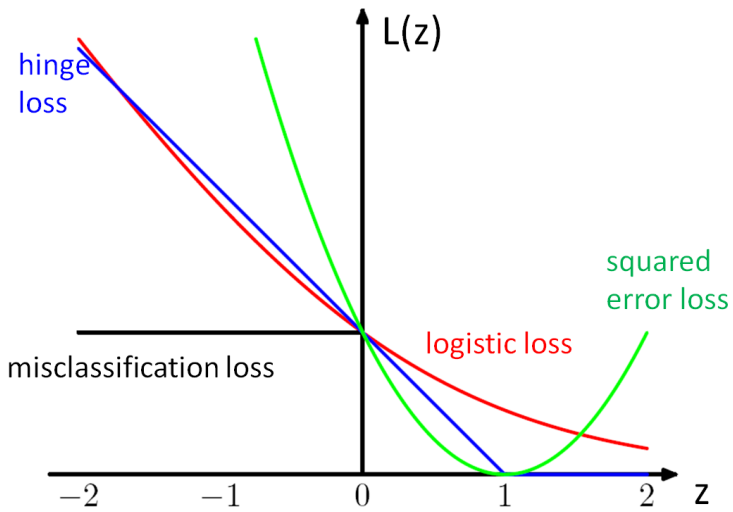
Machine/Statistical Learning: Linear Classification

Criterion: $$\min_w \underbrace{\sum_{i=1}^{N} L\big(\underbrace{y_i\,(w^T x_i + b)}_{\text{margin}}\big) + \tau\psi(w)}_{f(w)}$$

Convex loss: $L : \mathbb{R} \to \mathbb{R}_+$ is a (preferably convex) loss function.

Regularizer: $\psi = \ell_1 \;\Rightarrow\;$ encourage sparseness $\;\Rightarrow$ feature selection

- Misclassification loss: $L(z) = 1_{z<0}$
- Hinge loss: $L(z) = \max\{1 - z, 0\}$
- Logistic loss: $L(z) = \dfrac{\log\big(1+\exp(-z)\big)}{\log 2}$
- Squared loss: $L(z) = (z-1)^2$



Machine/Statistical Learning: Classification

This formulation covers a wide range of linear ML methods:

$$\min_w \underbrace{\sum_{i=1}^{N} L\left(y_i \left([x_i^T \; 1]w\right)\right)}_{f(w)} + \tau\psi(w)$$

- Least squares regression: $L(z) = (z-1)^2, \;\; \psi(w) = 0.$
- Ridge regression: $L(z) = (z-1)^2, \;\; \psi(w) = \|w\|_2^2.$
- Lasso regression: $L(z) = (z-1)^2, \;\; \psi(w) = \|w\|_1$
- Logistic regression: $L(z) = \log(1+\exp(-z))$ (ridge, if $\psi(w) = \|w\|_2^2$
- Sparse logistic regression: $L(z) = \log(1+\exp(-z)), \;\; \psi(w) = \|w\|_1$
- Support vector machines: $L(z) = \max\{1-z, 0\}, \;\; \psi(w) = \|w\|_2^2$
- Boosting: $L(z) = \exp(-z),$
- ...

Machine/Statistical Learning: Nonlinear Problems

What about non-linear functions?

Simply use $\widehat{y} = \phi(x, w) = \sum_{j=1}^{D} w_j \, \phi_j(x)$, where $\phi_j : \mathbb{R}^d \to \mathbb{R}$

Essentially, nothing changes; computationally, a lot may change!

$$\min_w \underbrace{\sum_{i=1}^{N} L\big(y_i \; \phi(x, w)\big)}_{f(w)} + \tau\psi(w)$$

Key feature: $\phi(x, w)$ is still linear with respect to w, thus f inherits the convexity of L.

Examples: polynomials, radial basis functions, wavelets, splines, kernels,...

Recover the linear case, letting $D = d + 1$, $f_j(x) = x_j$, and $f_{d+1} = 1$.

Structured Sparsity

ℓ_1 regularization promotes sparsity

A very simple sparsity pattern: prefer models with small cardinality

Structured Sparsity

ℓ_1 regularization promotes sparsity

A very simple sparsity pattern: prefer models with small cardinality

Can we promote less trivial sparsity patterns? How?

Structured Sparsity

ℓ_1 regularization promotes sparsity

A very simple sparsity pattern: prefer models with small cardinality

Can we promote less trivial sparsity patterns? How?

Structured Sparsity

ℓ_1 regularization promotes sparsity

A very simple sparsity pattern: prefer models with small cardinality

Can we promote less trivial sparsity patterns? How?

Group/structured regularization.

Structured Sparsity and Groups

Main goal: to promote structural patterns, not just penalize cardinality

Structured Sparsity and Groups

Main goal: to promote structural patterns, not just penalize cardinality

Group sparsity: discard/keep entire *groups* of features (Bach et al., 2012)

- density inside each group
- sparsity with respect to the groups which are selected
- choice of groups: prior knowledge about the intended *sparsity patterns*

Structured Sparsity and Groups

Main goal: to promote structural patterns, not just penalize cardinality

Group sparsity: discard/keep entire *groups* of features (Bach et al., 2012)

- density inside each group
- sparsity with respect to the groups which are selected
- choice of groups: prior knowledge about the intended *sparsity patterns*

Yields statistical gains if the assumption is correct (Stojnic et al., 2009)

Structured Sparsity and Groups

Main goal: to promote structural patterns, not just penalize cardinality

Group sparsity: discard/keep entire *groups* of features (Bach et al., 2012)

- density inside each group
- sparsity with respect to the groups which are selected
- choice of groups: prior knowledge about the intended *sparsity patterns*

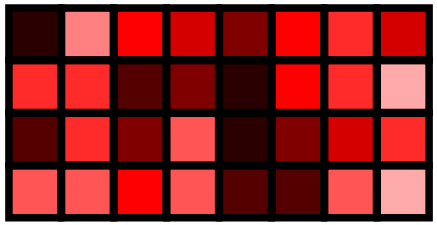
Yields statistical gains if the assumption is correct (Stojnic et al., 2009)

Many applications:

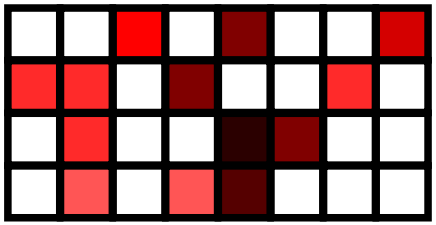
- feature template selection (Martins et al., 2011)
- multi-task learning (Caruana, 1997; Obozinski et al., 2010)
- learning the structure of graphical models (Schmidt and Murphy, 2010)

"Grid" Sparsity

For feature spaces that can be arranged as a grid (examples next)



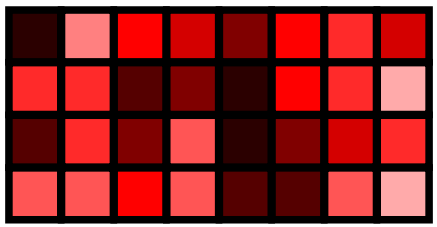
dense



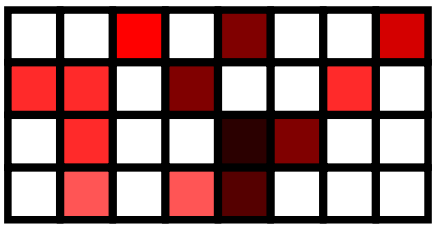
sparse

"Grid" Sparsity

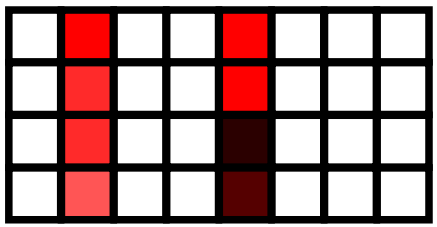
For feature spaces that can be arranged as a grid (examples next)



dense



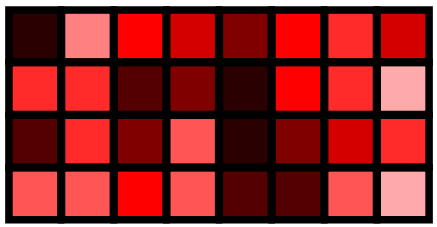
sparse



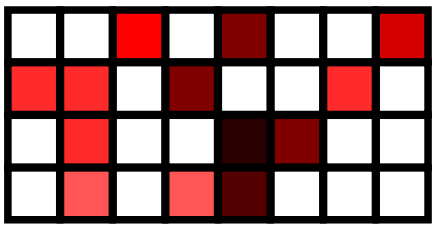
group sparse

"Grid" Sparsity

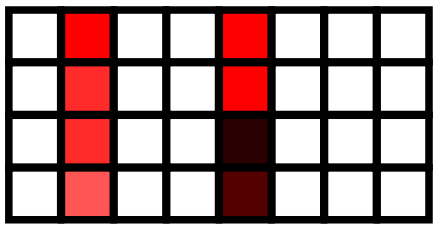
For feature spaces that can be arranged as a grid (examples next)



dense



sparse



group sparse

Goal: push *entire columns* to have zero weights

The groups are the columns of the grid

Example: Sparsity with Multiple Classes

In multi-class (more than just 2 classes) classification, a common formulation is

$$\widehat{y} = \arg \max_{y \in \{1,...,K\}} x^T w_y$$

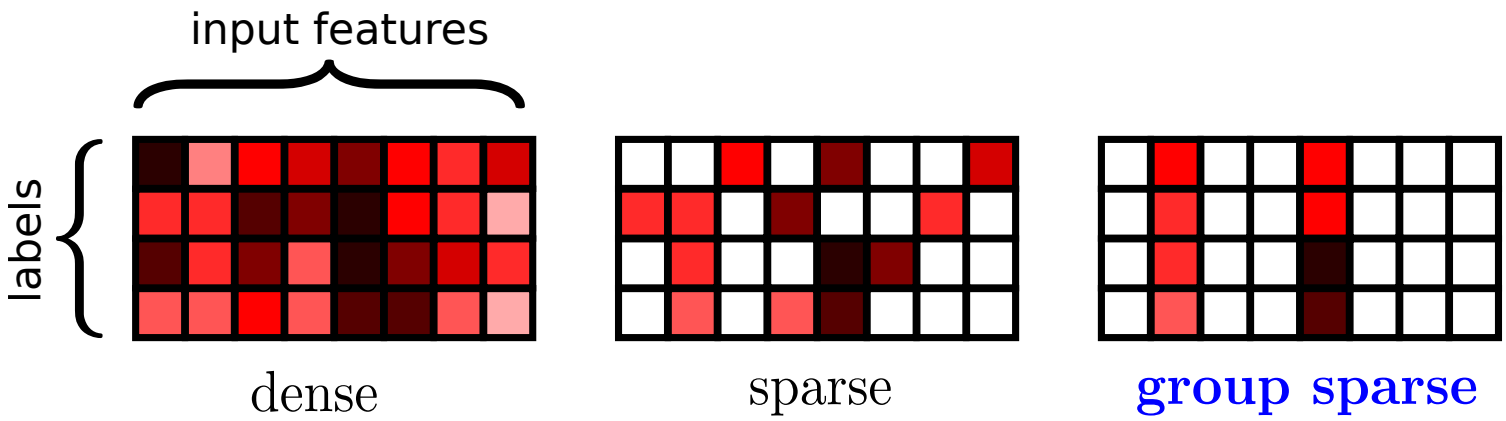
Weight vector $w = (w_1, ..., w_K) \in \mathbb{R}^{Kd}$ has a natural group/grid organization:

Example: Sparsity with Multiple Classes

In multi-class (more than just 2 classes) classification, a common formulation is

$$\widehat{y} = \arg \max_{y \in \{1,\ldots,K\}} x^T w_y$$

Weight vector $w = (w_1, \ldots, w_K) \in \mathbb{R}^{Kd}$ has a natural group/grid organization:



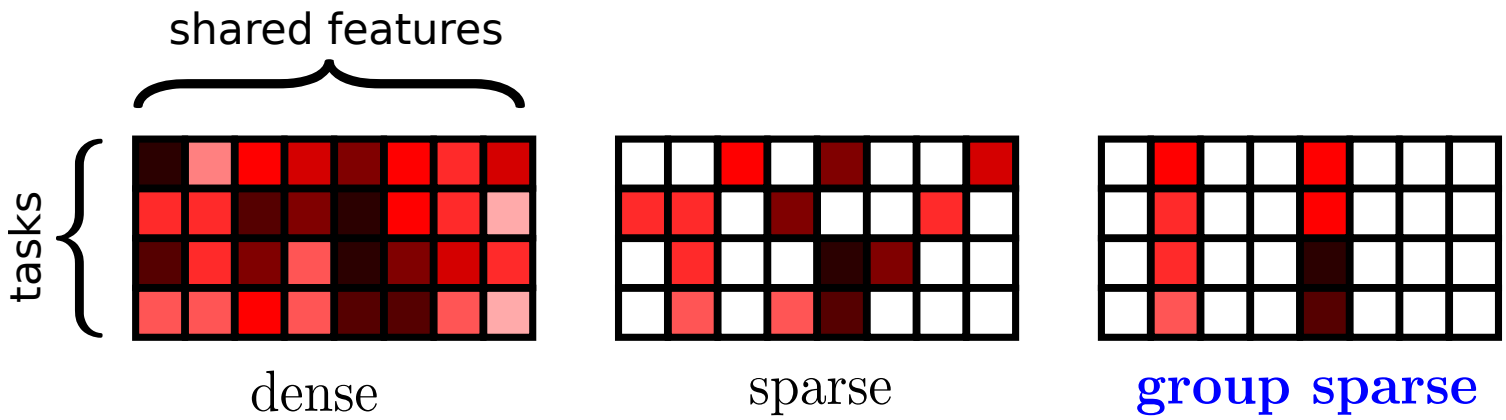
Simple sparsity is wasteful: may still need to keep all the features

Structured sparsity: discard some input features (feature selection)

Example: Multi-Task Learning

Same thing, except now rows are tasks and columns are **features**

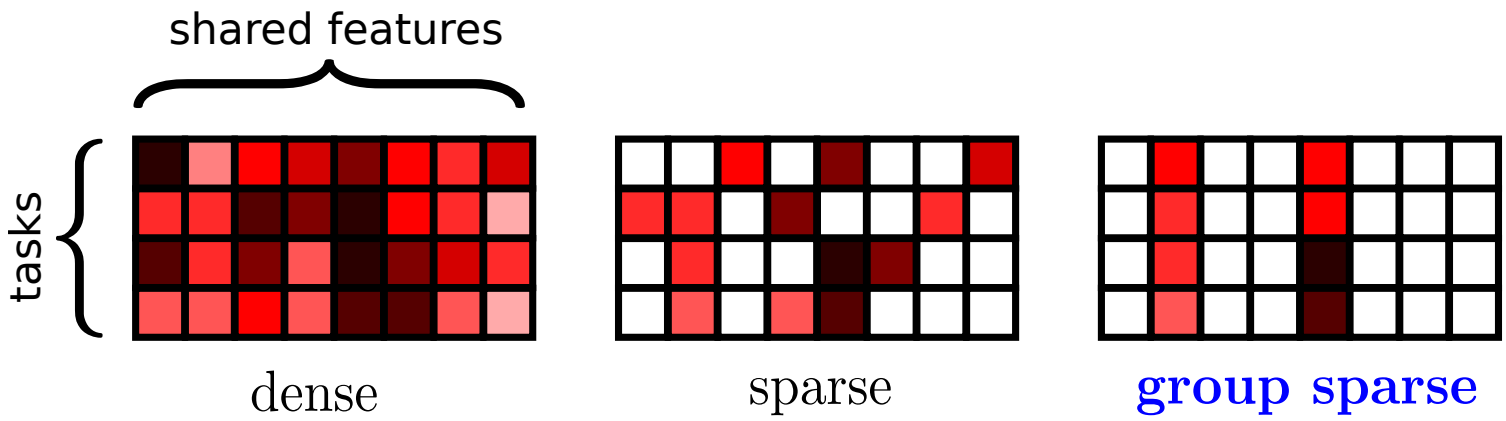
Example: simultaneous regression (seek function into $\mathbb{R}^d \to \mathbb{R}^b$)



Example: Multi-Task Learning

Same thing, except now rows are tasks and columns are **features**

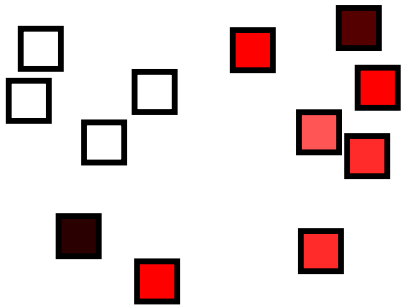
Example: simultaneous regression (seek function into $\mathbb{R}^d \to \mathbb{R}^b$)



Goal: discard features that are irrelevant for *all* tasks

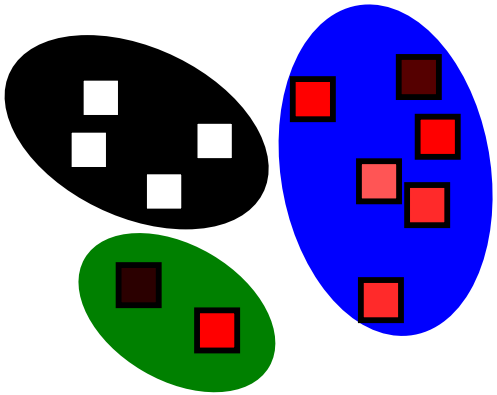
Approach: one group per feature (Caruana, 1997; Obozinski et al., 2010)

Group Sparsity



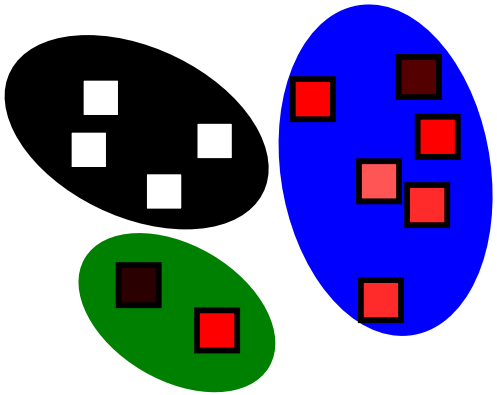
- D features

Group Sparsity



- D features
- M groups $G_1, \ldots, G_M$, each $G_m \subseteq \{1, \ldots, D\}$
- parameter subvectors $x_{G_1}, \ldots, x_{G_M}$

Group Sparsity

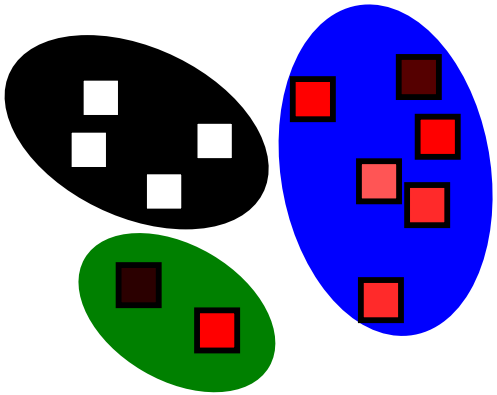


- D features
- M groups $G_1, \ldots, G_M$, each $G_m \subseteq \{1, \ldots, D\}$
- parameter subvectors $x_{G_1}, \ldots, x_{G_M}$

Group-Lasso (Bakin, 1999; Yuan and Lin, 2006):

$$\psi(x) = \sum_{m=1}^{M} \|x_{G_m}\|_2$$

Group Sparsity



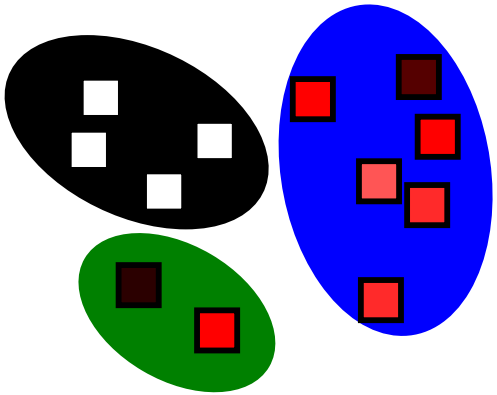
- D features
- M groups $G_1, \ldots, G_M$, each $G_m \subseteq \{1, \ldots, D\}$
- parameter subvectors $x_{G_1}, \ldots, x_{G_M}$

Group-Lasso (Bakin, 1999; Yuan and Lin, 2006):

$$\psi(x) = \sum_{m=1}^{M} \|x_{G_m}\|_2$$

- Intuitively: the ℓ_1 **norm** of the ℓ_2 **norms**
- Technically, still a norm (called a *mixed* norm, denoted $\ell_{2,1}$)

Group Sparsity



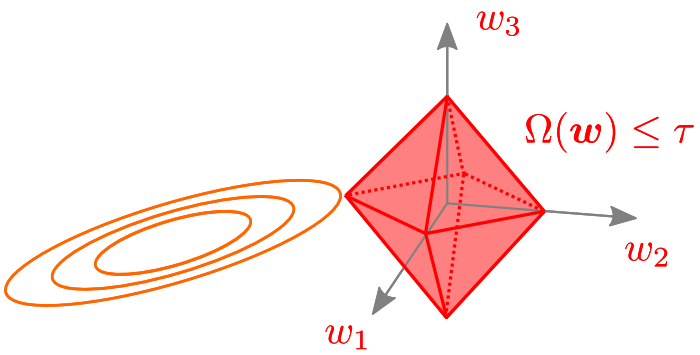
- D features
- M groups $G_1, \ldots, G_M$, each $G_m \subseteq \{1, \ldots, D\}$
- parameter subvectors $x_{G_1}, \ldots, x_{G_M}$

Group-Lasso (Bakin, 1999; Yuan and Lin, 2006):

$$\psi(x) = \sum_{m=1}^{M} \lambda_m \|x_{G_m}\|_2$$

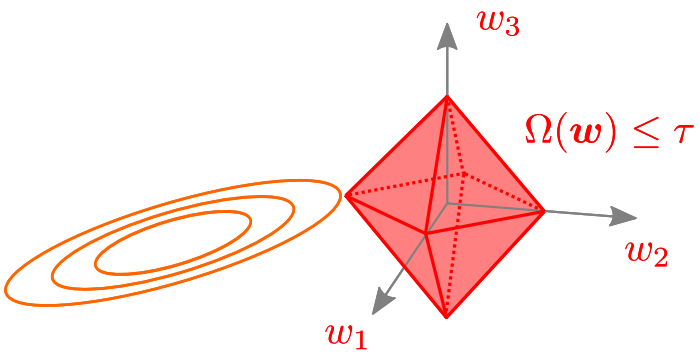
- Intuitively: the ℓ_1 **norm** of the ℓ_2 **norms**
- Technically, still a norm (called a *mixed* norm, denoted $\ell_{2,1}$)
- Weighted version: λ_m are prior weights for groups (groups may have different sizes)

Lasso versus group-Lasso

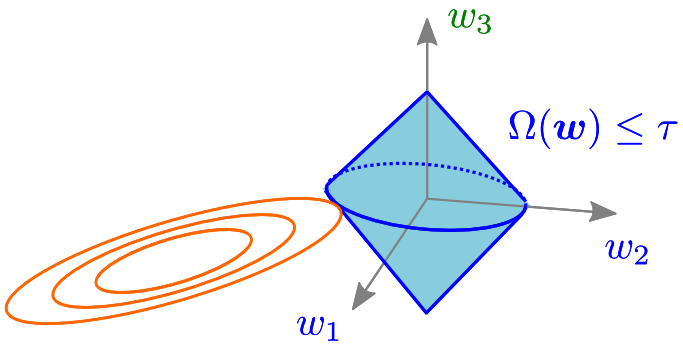


$$\Omega(\boldsymbol{w}) = |w_1| + |w_2| + |w_3|$$

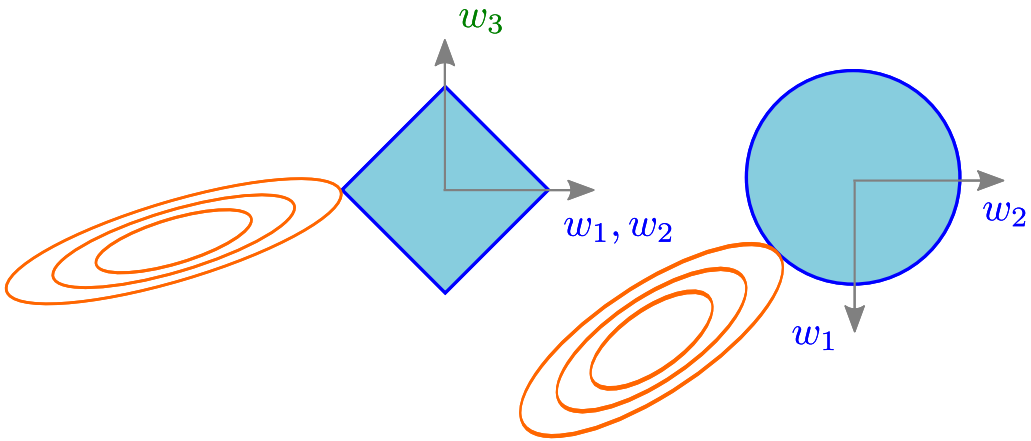
Lasso versus group-Lasso



$$\Omega(\boldsymbol{w}) = |w_1| + |w_2| + |w_3|$$



$$\Omega(\boldsymbol{w}) = \sqrt{w_1^2 + w_2^2} + |w_3|$$



Composite Absolute Penalties (Zhao et al., 2009)

A mixed-norm regularization:

$$\psi(x) = \left(\sum_{m=1}^{M} \|x_m\|_q^r \right)^{1/r}$$

The r-norm of the q-norms ($r \geq 1, q \geq 1$)

Technically, this is also a norm, called a **mixed norm**, denoted $\ell_{q,r}$

Composite Absolute Penalties (Zhao et al., 2009)

A mixed-norm regularization:

$$\psi(x) = \left(\sum_{m=1}^{M} \|x_m\|_q^r \right)^{1/r}$$

The r-norm of the q-norms ($r \geq 1, q \geq 1$)

Technically, this is also a norm, called a **mixed norm**, denoted $\ell_{q,r}$

- The most common choice: $\ell_{2,1}$ norm
- Another frequent choice: $\ell_{\infty,1}$ norm (Quattoni et al., 2009; Graça et al., 2009; Eisenstein et al., 2011; Wright et al., 2009)

Three Scenarios

- Non-overlapping Groups
- Tree-structured Groups
- Graph-structured Groups

Non-overlapping Groups

Assume that $G_1, \ldots, G_M$ (where $G_m \subset \{1, ..., d\}$) constitute a partition:

$$\bigcup_{i=1}^{M} G_m = \{1, ..., d\} \quad \text{and} \quad i \neq j \Rightarrow G_i \cap G_j = \emptyset$$

Non-overlapping Groups

Assume that $G_1, \ldots, G_M$ (where $G_m \subset \{1, ..., d\}$) constitute a partition:

$$\bigcup_{i=1}^{M} G_m = \{1, ..., d\} \quad \text{and} \quad i \neq j \Rightarrow G_i \cap G_j = \emptyset$$

$$\boxed{\psi(x) = \sum_{m=1}^{M} \lambda_m \|x_{G_m}\|_2}$$

Trivial choices of groups recover *unstructured* regularizers:

Non-overlapping Groups

Assume that $G_1, \ldots, G_M$ (where $G_m \subset \{1, ..., d\}$) constitute a partition:

$$\bigcup_{i=1}^{M} G_m = \{1, ..., d\} \quad \text{and} \quad i \neq j \Rightarrow G_i \cap G_j = \emptyset$$

$$\boxed{\psi(x) = \sum_{m=1}^{M} \lambda_m \|x_{G_m}\|_2}$$

Trivial choices of groups recover *unstructured* regularizers:

- ℓ_2-regularization: one large group $G_1 = \{1, \ldots, d\}$
- ℓ_1-regularization: d singleton groups $G_m = \{m\}$

Non-overlapping Groups

Assume that $G_1, \ldots, G_M$ (where $G_m \subset \{1, ..., d\}$) constitute a partition:

$$\bigcup_{i=1}^{M} G_m = \{1, ..., d\} \quad \text{and} \quad i \neq j \Rightarrow G_i \cap G_j = \emptyset$$

$$\boxed{\psi(x) = \sum_{m=1}^{M} \lambda_m \|x_{G_m}\|_2}$$

Trivial choices of groups recover *unstructured* regularizers:

- ℓ_2-regularization: one large group $G_1 = \{1, \ldots, d\}$
- ℓ_1-regularization: d singleton groups $G_m = \{m\}$

Examples of non-trivial groups:

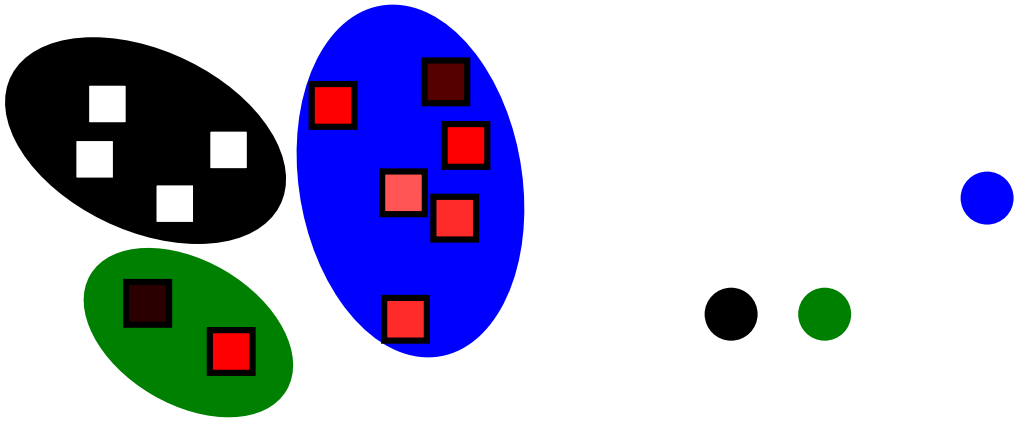
- label-based groups
- task-based groups

Tree-Structured Groups

Assumption: if two groups overlap, one is contained in the other
$\Rightarrow$ hierarchical structure (Kim and Xing, 2010; Mairal et al., 2010)

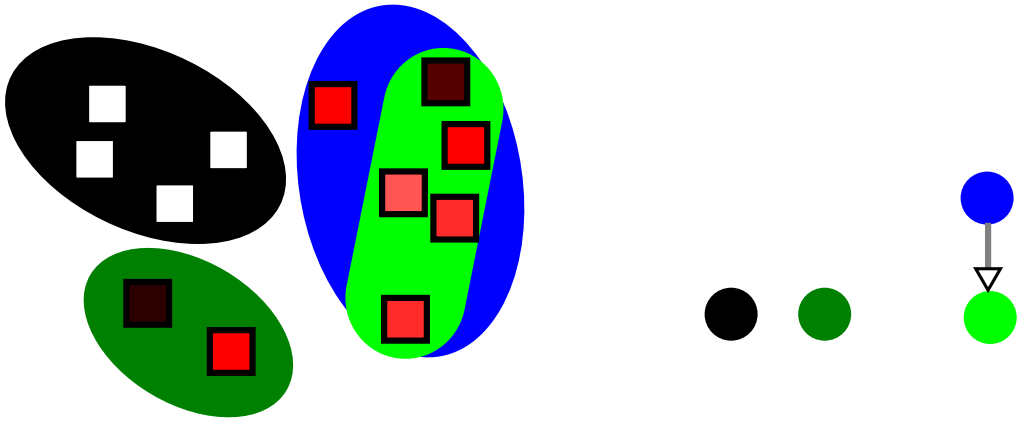
Tree-Structured Groups

Assumption: if two groups overlap, one is contained in the other
$\Rightarrow$ hierarchical structure (Kim and Xing, 2010; Mairal et al., 2010)



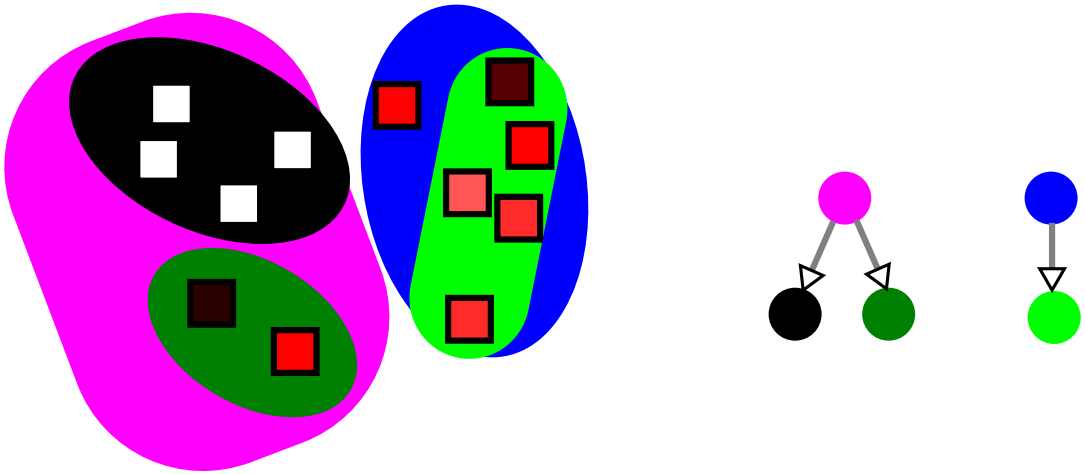
Tree-Structured Groups

Assumption: if two groups overlap, one is contained in the other
$\Rightarrow$ hierarchical structure (Kim and Xing, 2010; Mairal et al., 2010)



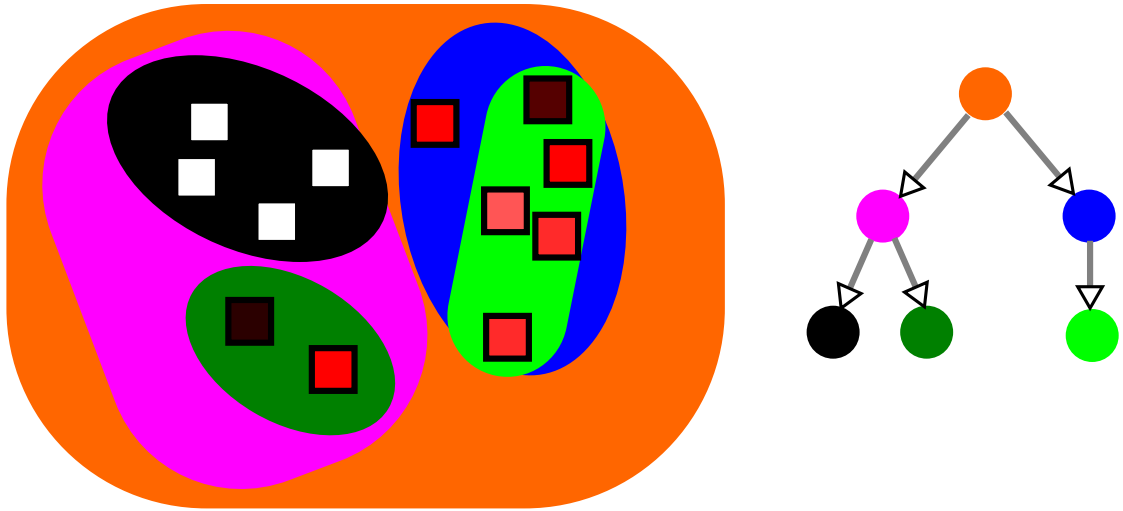
Tree-Structured Groups

Assumption: if two groups overlap, one is contained in the other
$\Rightarrow$ hierarchical structure (Kim and Xing, 2010; Mairal et al., 2010)



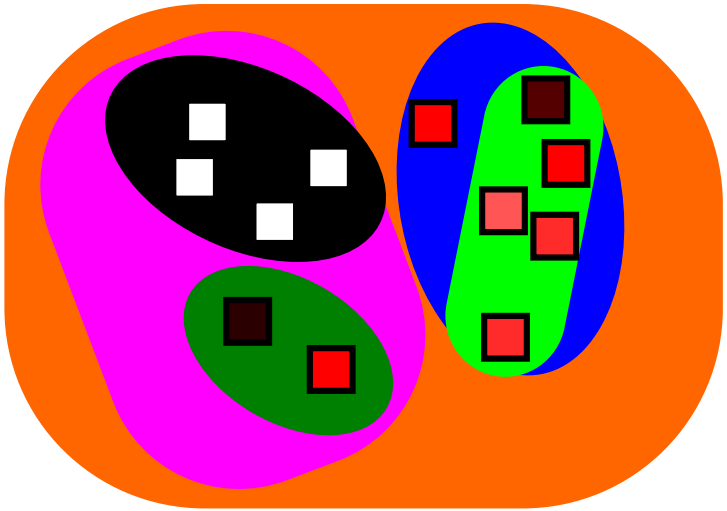
Tree-Structured Groups

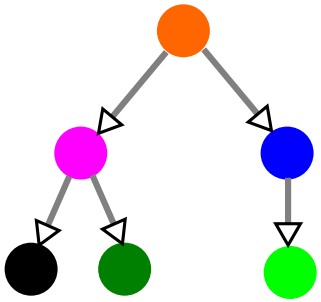
Assumption: if two groups overlap, one is contained in the other
$\Rightarrow$ hierarchical structure (Kim and Xing, 2010; Mairal et al., 2010)



Tree-Structured Groups

Assumption: if two groups overlap, one is contained in the other
$\Rightarrow$ hierarchical structure (Kim and Xing, 2010; Mairal et al., 2010)

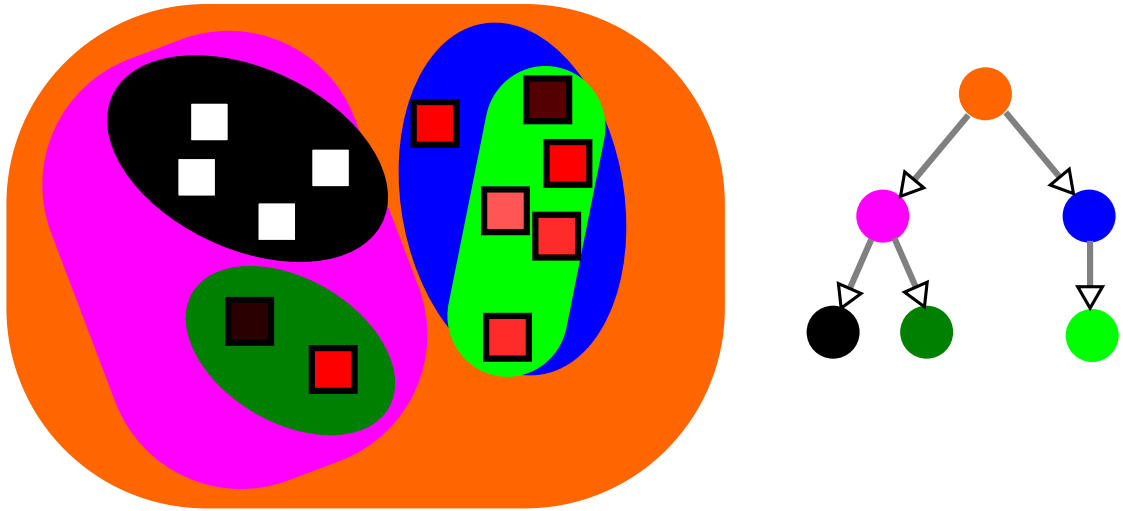




- What is the **sparsity pattern**?

Tree-Structured Groups

Assumption: if two groups overlap, one is contained in the other
$\Rightarrow$ hierarchical structure (Kim and Xing, 2010; Mairal et al., 2010)



- What is the **sparsity pattern**?
- If a group is discarded, all its descendants are also discarded

Matrix Inference Problems

Sparsest solution:

- From $Bx = b \in \mathbb{R}^p$, find $x \in \mathbb{R}^n$ $(p < n)$.
- $\min_x \|x\|_0$ s.t. $Bx = b$
- Yields exact solution, under some conditions.

Matrix Inference Problems

Sparsest solution:

- From $Bx = b \in \mathbb{R}^p$, find $x \in \mathbb{R}^n$ $(p < n)$.
- $\min_x \|x\|_0$ s.t. $Bx = b$
- Yields exact solution, under some conditions.

Lowest rank solution:

- From $\mathcal{B}(X) = b \in \mathbb{R}^p$, find $X \in \mathbb{R}^{m \times n}$ $(p < m\,n)$.
- $\min_X \text{rank}(X)$ s.t. $\mathcal{B}(X) = b$
- Yields exact solution, under some conditions.

Both $NP-$hard (in general); the same is true of noisy versions:

$$\min_{X \in \mathbb{R}^{m \times n}} \text{rank}(X) \text{ s.t. } \|\mathcal{B}(X) - b\|_2^2$$

Matrix Inference Problems

Sparsest solution:

- From $Bx = b \in \mathbb{R}^p$, find $x \in \mathbb{R}^n$ $(p < n)$.
- $\min_x \|x\|_0$ s.t. $Bx = b$
- Yields exact solution, under some conditions.

Lowest rank solution:

- From $\mathcal{B}(X) = b \in \mathbb{R}^p$, find $X \in \mathbb{R}^{m \times n}$ $(p < m\,n)$.
- $\min_X \text{rank}(X)$ s.t. $\mathcal{B}(X) = b$
- Yields exact solution, under some conditions.

Both $NP-$hard (in general); the same is true of noisy versions:

$$\min_{X \in \mathbb{R}^{m \times n}} \text{rank}(X) \text{ s.t. } \|\mathcal{B}(X) - b\|_2^2$$

Under some conditions, the same solution is obtained by replacing $\text{rank}(X)$ by the nuclear norm $\|X\|_*$ (as any norm, it is convex) (Recht et al., 2010)

Matrix Nuclear Norm (and Other Norms)

- Also known as trace norm; the ℓ_1-type norm for matrices $X \in \mathbb{R}^{m \times n}$
- Definition: $\|X\|_* = \text{trace}\left(\sqrt{X^T X}\right) = \sum_{i=1}^{\min\{m,n\}} \sigma_i$,

  the σ_i are the singular values of X.

Matrix Nuclear Norm (and Other Norms)

- Also known as trace norm; the ℓ_1-type norm for matrices $X \in \mathbb{R}^{m \times n}$
- Definition: $\|X\|_* = \text{trace}\left(\sqrt{X^T X}\right) = \sum_{i=1}^{\min\{m,n\}} \sigma_i$,

  the σ_i are the singular values of X.
- Particular case of Schatten q-norm: $\|X\|_q = \left(\sum_{i=1}^{\min\{m,n\}} (\sigma_i)^q \right)^{1/q}$.

Matrix Nuclear Norm (and Other Norms)

- Also known as trace norm; the ℓ_1-type norm for matrices $X \in \mathbb{R}^{m \times n}$
- Definition: $\|X\|_* = \text{trace}\left(\sqrt{X^T X}\right) = \sum_{i=1}^{\min\{m,n\}} \sigma_i$,

  the σ_i are the singular values of X.
- Particular case of Schatten q-norm: $\|X\|_q = \left(\sum_{i=1}^{\min\{m,n\}} (\sigma_i)^q \right)^{1/q}$.
- Two other notable Schatten norms:
  - Frobenius norm: $\|X\|_2 = \|X\|_F = \sqrt{\sum_{i=1}^{\min\{m,n\}} (\sigma_i)^2} = \sqrt{\sum_{i,i} X_{i,j}^2}$
  - Spectral norm: $\|X\|_\infty = \max\left\{\sigma_1, ..., \sigma_{\min\{m,n\}}\right\}$

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau \|X\|_*}_{\tau\psi(X)}$

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau \|X\|_*}_{\tau\psi(X)}$

Linear observations: $\mathcal{B} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$, $\big(\mathcal{B}(X)\big)_i = \langle B_{(i)}, X\rangle$,

$$B_{(i)} \in \mathbb{R}^{m\times n}, \text{ and } \langle B, X\rangle = \sum_{jk} B_{jk}X_{jk} = \text{trace}(B^T X)$$

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau \|X\|_*}_{\tau\psi(X)}$

Linear observations: $\mathcal{B} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$, $\big(\mathcal{B}(X)\big)_i = \langle B_{(i)}, X\rangle$,

$$B_{(i)} \in \mathbb{R}^{m\times n}, \text{ and } \langle B, X\rangle = \sum_{jk} B_{jk}X_{jk} = \text{trace}(B^T X)$$

Matrix completion, each $B_{(i)}$ has one 1 and is 0 everywhere else.

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau \|X\|_*}_{\tau\psi(X)}$

Linear observations: $\mathcal{B} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$, $\big(\mathcal{B}(X)\big)_i = \langle B_{(i)}, X\rangle$,

$$B_{(i)} \in \mathbb{R}^{m\times n}, \text{ and } \langle B, X\rangle = \sum_{jk} B_{jk}X_{jk} = \text{trace}(B^T X)$$

Matrix completion, each $B_{(i)}$ has one 1 and is 0 everywhere else.

Why does the nuclear norm favor low rank solutions?

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau \|X\|_*}_{\tau\psi(X)}$

Linear observations: $\mathcal{B} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$, $\big(\mathcal{B}(X)\big)_i = \langle B_{(i)}, X\rangle$,

$$B_{(i)} \in \mathbb{R}^{m\times n}, \text{ and } \langle B, X\rangle = \sum_{jk} B_{jk}X_{jk} = \text{trace}(B^T X)$$

Matrix completion, each $B_{(i)}$ has one 1 and is 0 everywhere else.

Why does the nuclear norm favor low rank solutions? Let $Y = U\Lambda V^T$ be the singular value decomposition, where $\Lambda = \text{diag}\big(\sigma_1, ..., \sigma_{\min\{m,n\}}\big)$; then

$$\arg\min_X \frac{1}{2}\|Y - X\|_F^2 + \tau\|X\|_* = U \underbrace{\text{soft}(\Lambda, \tau)}_{\text{may yield zeros}} V^T$$

...singular value thresholding (Ma et al., 2011; Cai et al., 2010)

Another Matrix Inference Problem: Inverse Covariance

Consider n samples $y_1, ..., y_n \in \mathbb{R}^d$ of a Gaussian r.v. $Y \sim \mathcal{N}(\mu, C)$; the log-likelihood is

$$L(P) = \log p(y_1, ..., y_n | P) = \log \det(P) - \text{trace}(SP) + \text{constant}$$

where $S = \frac{1}{n} \sum_{i=1}^{n} (y_i - \mu)(y_i - \mu)^T$ and $P = C^{-1}$ (inverse covariance).

Another Matrix Inference Problem: Inverse Covariance

Consider n samples $y_1, ..., y_n \in \mathbb{R}^d$ of a Gaussian r.v. $Y \sim \mathcal{N}(\mu, C)$; the log-likelihood is

$$L(P) = \log p(y_1, ..., y_n|P) = \log \det(P) - \text{trace}(SP) + \text{constant}$$

where $S = \frac{1}{n}\sum_{i=1}^{n}(y_i - \mu)(y_i - \mu)^T$ and $P = C^{-1}$ (inverse covariance).

Zeros in P reveal conditional independencies between components of Y:

$$P_{ij} = 0 \quad \Leftrightarrow \quad Y_i \perp\!\!\!\perp Y_j | \{Y_k,\ k \neq i, j\}$$

...exploited to infer (in)dependencies among Gaussian variables. Widely used in computational biology, neuroscience, (social) network analysis, ...

Another Matrix Inference Problem: Inverse Covariance

Consider n samples $y_1, ..., y_n \in \mathbb{R}^d$ of a Gaussian r.v. $Y \sim \mathcal{N}(\mu, C)$; the log-likelihood is

$$L(P) = \log p(y_1, ..., y_n | P) = \log \det(P) - \text{trace}(SP) + \text{constant}$$

where $S = \frac{1}{n} \sum_{i=1}^{n} (y_i - \mu)(y_i - \mu)^T$ and $P = C^{-1}$ (inverse covariance).

Zeros in P reveal conditional independencies between components of Y:

$$P_{ij} = 0 \quad \Leftrightarrow \quad Y_i \perp\!\!\!\perp Y_j | \{Y_k, \ k \neq i, j\}$$

...exploited to infer (in)dependencies among Gaussian variables. Widely used in computational biology, neuroscience, (social) network analysis, ...

Sparsity (presence of zeros) in P is encouraged by solving

$$\min_{P \succ 0} \underbrace{-\log \det(P) + \text{trace}(SP)}_{f(P)} + \tau \underbrace{\|\text{vect}(P)\|_1}_{\psi(P)}$$

where $\text{vect}(P) = [P_{1,1}, ..., P_{d,d}]^T$.

Atomic-Norm Regularization

Key concept in sparse modeling: synthesize "object" using a few atoms:

$$x = \sum_{i=1}^{|\mathcal{A}|} c_i \, a_i$$

- $\mathcal{A}$ is the set of atoms (the atomic set), or building blocks.
- $c_i \geq 0$ are weights; x is simple/sparse object $\Rightarrow \quad \|c\|_0 \ll |\mathcal{A}|$
- Formally, $\mathcal{A}$ is a compact subset of $\mathbb{R}^n$

Atomic-Norm Regularization

Key concept in sparse modeling: synthesize "object" using a few atoms:

$$x = \sum_{i=1}^{|\mathcal{A}|} c_i \, a_i$$

- $\mathcal{A}$ is the set of atoms (the atomic set), or building blocks.
- $c_i \geq 0$ are weights; x is simple/sparse object $\Rightarrow \quad \|c\|_0 \ll |\mathcal{A}|$
- Formally, $\mathcal{A}$ is a compact subset of $\mathbb{R}^n$

The (Minkowski) gauge of $\mathcal{A}$ is:

$$\|x\|_{\mathcal{A}} = \inf\left\{t > 0 : \; x \in t \, \mathrm{conv}(\mathcal{A})\right\}$$

Assuming that $\mathcal{A}$ centrally symmetry about the origin ($a \in \mathcal{A} \;\Rightarrow\; -a \in \mathcal{A}$), $\|\cdot\|_{\mathcal{A}}$ is a norm, called the atomic norm (Chandrasekaran et al., 2012).

Atomic-Norm Regularization

The atomic norm

$$\|x\|_{\mathcal{A}} = \inf\Big\{t > 0 : \; x \in t\,\mathrm{conv}(\mathcal{A})\Big\}$$

$$= \inf\Big\{\sum_{i=1}^{|\mathcal{A}|} c_i : \; x = \sum_{i=1}^{|\mathcal{A}|} c_i\, a_i, \; c_i \geq 0\Big\}$$

...assuming that the centroid of $\mathcal{A}$ is at the origin.

Atomic-Norm Regularization

The atomic norm

$$\|x\|_{\mathcal{A}} = \inf\left\{t > 0 : \; x \in t\,\mathrm{conv}(\mathcal{A})\right\}$$
$$= \inf\left\{\sum_{i=1}^{|\mathcal{A}|} c_i : \; x = \sum_{i=1}^{|\mathcal{A}|} c_i\, a_i, \; c_i \geq 0\right\}$$

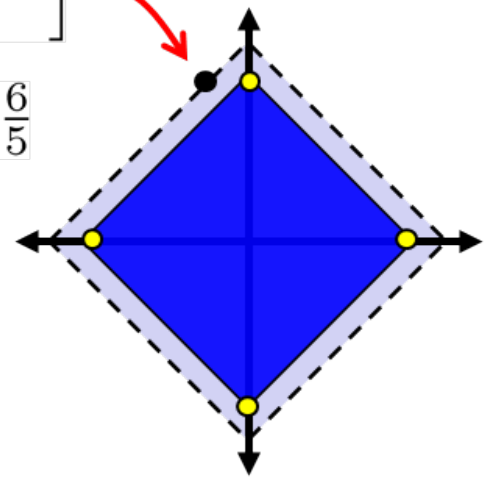
...assuming that the centroid of $\mathcal{A}$ is at the origin.

Example: the ℓ_1 norm as an atomic norm

- $\mathcal{A} = \left\{ \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \end{bmatrix} \right\}$
- $\mathrm{conv}(\mathcal{A}) = B_1(1)$ (ℓ_1 unit ball).
- $\|x\|_{\mathcal{A}} = \inf\left\{t > 0 : \; x \in t\, B_1(1)\right\}$
  $= \|x\|_1$

$$x = \begin{bmatrix} -1/5 \\ 1 \end{bmatrix}$$

$$\|x\|_{\mathcal{A}} = \tfrac{6}{5}$$



Atomic Norms: More Examples

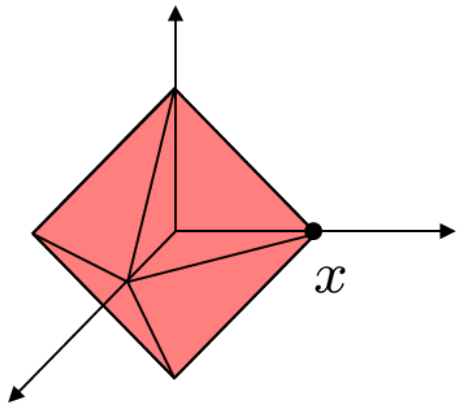
Examples with easy forms:

- *sparse vectors*

$$\mathcal{A} = \{\pm e_i\}_{i=1}^N$$

$$\text{conv}(\mathcal{A}) = \text{cross-polytope}$$

$$\|x\|_{\mathcal{A}} = \|x\|_1$$

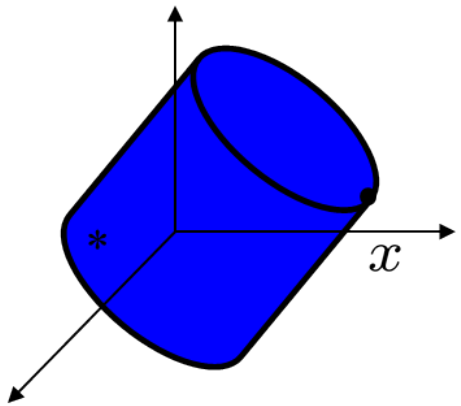


- *low-rank matrices*

$$\mathcal{A} = \{A : \text{rank}(A) = 1, \|A\|_F = 1\}$$

$$\text{conv}(\mathcal{A}) = \text{nuclear norm ball}$$

$$\|x\|_{\mathcal{A}} = \|x\|_\star$$



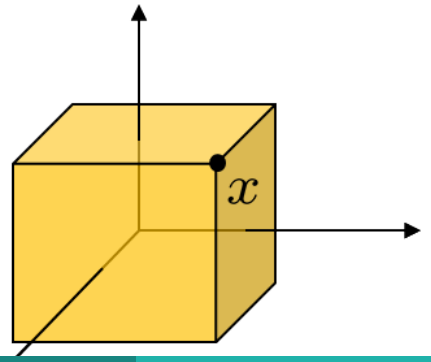
*symmetric matrices

- *binary vectors*

$$\mathcal{A} = \{\pm 1\}^N$$

$$\text{conv}(\mathcal{A}) = \text{hypercube}$$

$$\|x\|_{\mathcal{A}} = \|x\|_\infty$$



Atomic-Norm Regularization

Given an atomic set $\mathcal{A}$, we can adopt an Ivanov formulation

$$\min f(x) \text{ s.t. } \|x\|_{\mathcal{A}} \leq \delta$$

(for some $\delta > 0$) tends to recover x with sparse atomic representation.

Atomic-Norm Regularization

Given an atomic set $\mathcal{A}$, we can adopt an Ivanov formulation

$$\min f(x) \text{ s.t. } \|x\|_{\mathcal{A}} \leq \delta$$

(for some $\delta > 0$) tends to recover x with sparse atomic representation.

Can formulate algorithms for the various special cases — but is a general approach available for this formulation?

Atomic-Norm Regularization

Given an atomic set $\mathcal{A}$, we can adopt an Ivanov formulation

$$\min f(x) \text{ s.t. } \|x\|_{\mathcal{A}} \leq \delta$$

(for some $\delta > 0$) tends to recover x with sparse atomic representation.

Can formulate algorithms for the various special cases — but is a general approach available for this formulation?

Yes! The conditional gradient (more later.)

Summary

- Many inference, learning, signal/image processing problems can be formulated as optimization problems.
- Sparsity-inducing regularizers play an important role in these problems
- There are several way to induce sparsity
- It is possible to formulate structured sparsity
- It is possible to extend the sparsity rationale to other objects, namely matrices
- Atomic norms provide a unified framework for sparsity/simplicity regularization

References I

Amaldi, E. and Kann, V. (1998). On the approximation of minimizing non zero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209:237–260.

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. (2012). Structured sparsity through convex optimization. *Statistical Science*, 27:450–468.

Bakin, S. (1999). *Adaptive regression and model selection in data mining problems*. PhD thesis, Australian National University.

Cai, J.-F., Candès, E., and Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM JOurnal on Optimization*, 20:1956–1982.

Candès, E., Romberg, J., and Tao, T. (2006a). Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509.

Candès, E., Romberg, J., and Tao, T. (2006b). Stable signal recovery from incomplete and inaccurate measurements. *Communications in Pure and Applied Mathematics*, 59:1207–1223.

Caruana, R. (1997). Multitask learning. *Machine Learning*, 28(1):41–75.

Chandrasekaran, V., Recht, B., Parrilo, P., and Willsky, A. (2012). The convex geometry of linear inverse problems. *Foundations of Computational Mathematics*, 12:805–849.

Chen, S., Donoho, D., and Saunders, M. (1995). Atomic decomposition by basis pursuit. Technical report, Department of Statistics, Stanford University.

Davis, G., Mallat, S., and Avellaneda, M. (1997). Greedy adaptive approximation. *Journal of Constructive Approximation*, 13:57–98.

References II

Donoho, D. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, 52:1289–1306.

Eisenstein, J., Smith, N. A., and Xing, E. P. (2011). Discovering sociolinguistic associations with structured sparsity. In *Proc. of ACL*.

Garnaev, A. and Gluskin, E. (1984). The widths of an Euclidean ball. *Doklady Akademii Nauk*, 277:1048–1052.

Graça, J., Ganchev, K., Taskar, B., and Pereira, F. (2009). Posterior vs. parameter sparsity in latent variable models. *Advances in Neural Information Processing Systems*.

Haupt, J. and Nowak, R. (2006). Signal reconstruction from noisy random projections. *IEEE Transactions on Information Theory*, 52:4036–4048.

Kashin, B. (1977). Diameters of certain finite-dimensional sets in classes of smooth functions. *Izvestiya Akademii Nauk. SSSR: Seriya Matematicheskaya*, 41:334–351.

Kim, S. and Xing, E. (2010). Tree-guided group lasso for multi-task regression with structured sparsity. In *Proc. of ICML*.

Ma, S., Goldfarb, D., and Chen, L. (2011). Fixed point and Bregman iterative methods for matrix rank minimization. *Mathematical Programming (Series A)*, 128:321–353.

Mairal, J., Jenatton, R., Obozinski, G., and Bach, F. (2010). Network flow algorithms for structured sparsity. In *Advances in Neural Information Processing Systems*.

References III

Martins, A. F. T., Smith, N. A., Aguiar, P. M. Q., and Figueiredo, M. A. T. (2011). Structured Sparsity in Structured Prediction. In *Proc. of Empirical Methods for Natural Language Processing*.

Muthukrishnan, S. (2005). *Data Streams: Algorithms and Applications*. Now Publishers, Boston, MA.

Obozinski, G., Taskar, B., and Jordan, M. (2010). Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20(2):231–252.

Quattoni, A., Carreras, X., Collins, M., and Darrell, T. (2009). An efficient projection for $l_{1,\infty}$ regularization. In *Proc. of ICML*.

Recht, B., Fazel, M., and Parrilo, P. (2010). Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52:471–501.

Schmidt, M. and Murphy, K. (2010). Convex structure learning in log-linear models: Beyond pairwise potentials. In *Proc. of AISTATS*.

Stojnic, M., Parvaresh, F., and Hassibi, B. (2009). On the reconstruction of block-sparse signals with an optimal number of measurements. *Signal Processing, IEEE Transactions on*, 57(8):3075–3085.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B.*, pages 267–288.

Tillmann, A. and Pfetsch, M. (2012). The computational complexity of RIP, NSP, and related concepts in compressed sensing. Technical report, arXiv/1205.2081.

References IV

Wright, S., Nowak, R., and Figueiredo, M. (2009). Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57:2479–2493.

Yin, W. and Zhang, Y. (2008). Extracting salient features from less data via ℓ_1-minimization authors. *SIAG/OPT Views-and-News*, 19:11–19.

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

Zhao, P., Rocha, G., and Yu, B. (2009). Grouped and hierarchical model selection through composite absolute penalties. *Annals of Statistics*, 37(6A):3468–3497.

Part 2: First-Order Methods

Mário A. T. Figueiredo

[1]Instituto de Telecomunicações,
Instituto Superior Técnico,
Universidade de Lisboa, Portugal

CIMI-ANITI School on Optimisation, 2021

Focus (Initially) on Smooth Convex Functions

Consider $\min_{x \in \mathbb{R}^n} f(x)$, with f smooth and convex.

Usually assume $\mu I \preceq \nabla^2 f(x) \preceq LI, \ \forall_x$, with $0 \leq \mu \leq L$
(L is a Lipschitz constant of ∇f).

Focus (Initially) on Smooth Convex Functions

Consider $\min_{x \in \mathbb{R}^n} f(x)$, with f smooth and convex.

Usually assume $\mu I \preceq \nabla^2 f(x) \preceq LI, \ \forall_x$, with $0 \leq \mu \leq L$
(L is a Lipschitz constant of ∇f).

If $\mu > 0$, then f is μ-strongly convex (as seen in Part 1) and

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{\mu}{2} \|y - x\|_2^2.$$

Focus (Initially) on Smooth Convex Functions

Consider $\min_{x \in \mathbb{R}^n} f(x)$, with f smooth and convex.

Usually assume $\mu I \preceq \nabla^2 f(x) \preceq LI, \ \ \forall_x$, with $0 \leq \mu \leq L$
(L is a Lipschitz constant of ∇f).

If $\mu > 0$, then f is μ-strongly convex (as seen in Part 1) and

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{\mu}{2} \|y - x\|_2^2.$$

Define conditioning (or condition number) as $\kappa := L/\mu$.

Focus (Initially) on Smooth Convex Functions

Consider $\min_{x \in \mathbb{R}^n} f(x)$, with f smooth and convex.

Usually assume $\mu I \preceq \nabla^2 f(x) \preceq LI, \ \ \forall_x$, with $0 \leq \mu \leq L$
(L is a Lipschitz constant of ∇f).

If $\mu > 0$, then f is μ-strongly convex (as seen in Part 1) and

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{\mu}{2} \|y - x\|_2^2.$$

Define conditioning (or condition number) as $\kappa := L/\mu$.

We are often interested in convex quadratics:

$$f(x) = \frac{1}{2} x^T A x, \qquad \mu I \preceq A \preceq LI \ \text{ or}$$

$$f(x) = \frac{1}{2} \|Bx - b\|_2^2, \quad \mu I \preceq B^T B \preceq LI$$

What's the Setup?

We consider iterative algorithms

$$x_{k+1} = \Phi(x_k), \quad \text{or} \quad x_{k+1} = \Phi(x_k, x_{k-1})$$

Assume we can evaluate $f(x_t)$ and $\nabla f(x_t)$ at each iteration.

What's the Setup?

We consider iterative algorithms

$$x_{k+1} = \Phi(x_k), \quad \text{or} \quad x_{k+1} = \Phi(x_k, x_{k-1})$$

Assume we can evaluate $f(x_t)$ and $\nabla f(x_t)$ at each iteration.

Later, we look at broader classes of problems:

- nonsmooth regularization; *i.e.*, instead of just $f(x)$, minimize $f(x) + \tau\psi(x)$;
- nonsmooth f;
- f not available (or too expensive to evaluate exactly);
- only an *estimate* of the gradient is available;
- a constraint $x \in \Omega$, usually for a simple Ω (e.g. ball, box, simplex).

We focus on algorithms that can be adapted to those scenarios.

Steepest Descent

Steepest descent (a.k.a. gradient descent):

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \text{for some } \alpha_k > 0.$$

Different ways to select an appropriate α_k.

1. Hard: interpolating scheme with safeguarding to identify an approximate minimizing α_k.

Steepest Descent

Steepest descent (a.k.a. gradient descent):

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \text{for some } \alpha_k > 0.$$

Different ways to select an appropriate α_k.

1. Hard: interpolating scheme with safeguarding to identify an approximate minimizing α_k.
2. Easy: backtracking. $\bar{\alpha}$, $\frac{1}{2}\bar{\alpha}$, $\frac{1}{4}\bar{\alpha}$, $\frac{1}{8}\bar{\alpha}$, ... until sufficient decrease in f is obtained.

Steepest Descent

Steepest descent (a.k.a. gradient descent):

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \text{for some } \alpha_k > 0.$$

Different ways to select an appropriate α_k.

1. Hard: interpolating scheme with safeguarding to identify an approximate minimizing α_k.
2. Easy: backtracking. $\bar{\alpha}$, $\frac{1}{2}\bar{\alpha}$, $\frac{1}{4}\bar{\alpha}$, $\frac{1}{8}\bar{\alpha}$, ... until sufficient decrease in f is obtained.
3. Trivial: don't test for function decrease; use rules based on L and μ.

Steepest Descent

Steepest descent (a.k.a. gradient descent):

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \text{for some } \alpha_k > 0.$$

Different ways to select an appropriate α_k.

1. Hard: interpolating scheme with safeguarding to identify an approximate minimizing α_k.
2. Easy: backtracking. $\bar{\alpha}$, $\frac{1}{2}\bar{\alpha}$, $\frac{1}{4}\bar{\alpha}$, $\frac{1}{8}\bar{\alpha}$, ... until sufficient decrease in f is obtained.
3. Trivial: don't test for function decrease; use rules based on L and μ.

Analysis of 1 and 2 yields global convergence at unspecified rate, but not applicable to non-smooth problems.

Analysis of 3 focuses on convergence rate, and leads to accelerated multi-step methods.

Constant (Short) Steplength

By elementary use of Taylor's theorem, and since $\nabla^2 f(x) \preceq LI$,

$$f(x_{k+1}) \leq f(x_k) - \alpha_k \left(1 - \frac{\alpha_k}{2} L\right) \|\nabla f(x_k)\|_2^2$$

For $\alpha_k \equiv 1/L$, $\quad f(x_{k+1}) \leq f(x_k) - \frac{1}{2L} \|\nabla f(x_k)\|_2^2$,

thus $\quad \|\nabla f(x_k)\|^2 \leq 2L[f(x_k) - f(x_{k+1})]$

Summing for $k = 0, 1, \ldots, N$, and telescoping the sum,

$$\sum_{k=0}^{N} \|\nabla f(x_k)\|^2 \leq 2L[f(x_0) - f(x_{N+1})].$$

It follows that $\nabla f(x_k) \to 0$ if f is bounded below.

Rate Analysis

Suppose that the minimizer x^* is unique.

Another elementary use of Taylor's theorem shows that

$$\|x_{k+1} - x^*\|^2 \leq \|x_k - x^*\|^2 - \alpha_k \left(\frac{2}{L} - \alpha_k\right) \|\nabla f(x_k)\|^2,$$

so that $\{\|x_k - x^*\|\}$ is decreasing.

Define for convenience: $\Delta_k := f(x_k) - f(x^*)$. By convexity, have

$$\Delta_k \leq \nabla f(x_k)^T (x_k - x^*) \leq \|\nabla f(x_k)\| \, \|x_k - x^*\| \leq \|\nabla f(x_k)\| \, \|x_0 - x^*\|.$$

From previous page (subtracting $f(x^*)$ from both sides of the inequality), and using the inequality above, we have

$$\Delta_{k+1} \leq \Delta_k - (1/2L)\|\nabla f(x_k)\|^2 \leq \Delta_k - \frac{1}{2L\|x_0 - x^*\|^2} \Delta_k^2.$$

Weakly convex: $1/k$ sublinear; Strongly convex: linear

Take reciprocal of both sides and manipulate (using $(1-\epsilon)^{-1} \geq 1+\epsilon$):

$$\frac{1}{\Delta_{k+1}} \geq \frac{1}{\Delta_k} + \frac{1}{2L\|x_0 - x^*\|^2} \geq \frac{1}{\Delta_0} + \frac{k+1}{2L\|x_0 - x^*\|^2},$$

which yields

$$f(x_{k+1}) - f(x^*) \leq \frac{2L\|x_0 - x\|^2}{k+1}.$$

The classic $1/k$ convergence rate!

By assuming $\mu > 0$, can set $\alpha_k \equiv 2/(\mu + L)$ and get a linear (geometric) rate: Much better than sublinear, in the long run

$$\|x_k - x^*\|^2 \leq \left(\frac{L-\mu}{L+\mu}\right)^{2k} \|x_0 - x^*\|^2 = \left(1 - \frac{2}{\kappa+1}\right)^{2k} \|x_0 - x^*\|^2.$$

Weakly convex: $1/k$ sublinear; Strongly convex: linear

Since by Taylor's theorem we have

$$\Delta_k = f(x_k) - f(x^*) \leq (L/2)\|x_k - x^*\|^2,$$

it follows immediately that

$$f(x_k) - f(x^*) \leq \frac{L}{2}\left(1 - \frac{2}{\kappa + 1}\right)^{2k} \|x_0 - x^*\|^2.$$

Note: A geometric/linear rate is generally better than almost any sublinear ($1/k$ or $1/k^2$) rate.

Detour: Convergence Rates

Sequence t_k, for $k = 1, 2, \ldots$, that converges to zero, $\lim_{k \to \infty} t_k = 0$

Detour: Convergence Rates

Sequence t_k, for $k = 1, 2, \ldots$, that converges to zero, $\lim_{k \to \infty} t_k = 0$

Sublinear: $\frac{t_{k+1}}{t_k} \to 1$ (for example, $t_k = K/k$)

Detour: Convergence Rates

Sequence t_k, for $k = 1, 2, \ldots$, that converges to zero, $\lim_{k\to\infty} t_k = 0$

Sublinear: $\frac{t_{k+1}}{t_k} \to 1$ (for example, $t_k = K/k$)

Linear: $\frac{t_{k+1}}{t_k} \leq \gamma$, for $\gamma \in (0, 1)$ (for example, $t_k = \gamma^k$)

Detour: Convergence Rates

Sequence t_k, for $k = 1, 2, ...$, that converges to zero, $\lim_{k\to\infty} t_k = 0$

Sublinear: $\frac{t_{k+1}}{t_k} \to 1$ (for example, $t_k = K/k$)

Linear: $\frac{t_{k+1}}{t_k} \leq \gamma$, for $\gamma \in (0, 1)$ (for example, $t_k = \gamma^k$)

Superlinear: $\frac{t_{k+1}}{t_k} \to 0$ (for example, $t_k = \gamma^{k^2}$)

Detour: Convergence Rates

Sequence t_k, for $k = 1, 2, ...$, that converges to zero, $\lim_{k\to\infty} t_k = 0$

Sublinear: $\frac{t_{k+1}}{t_k} \to 1$ (for example, $t_k = K/k$)

Linear: $\frac{t_{k+1}}{t_k} \leq \gamma$, for $\gamma \in (0, 1)$ (for example, $t_k = \gamma^k$)

Superlinear: $\frac{t_{k+1}}{t_k} \to 0$ (for example, $t_k = \gamma^{k^2}$)

Quadratic: $\frac{t_{k+1}}{(t_k)^2} \to B < \infty$ (for example, $t_k = \gamma^{2^k}$, with $\gamma \in (0, 1)$)

Detour: Convergence Rates

Sequence t_k, for $k = 1, 2, ...$, that converges to zero, $\lim_{k\to\infty} t_k = 0$

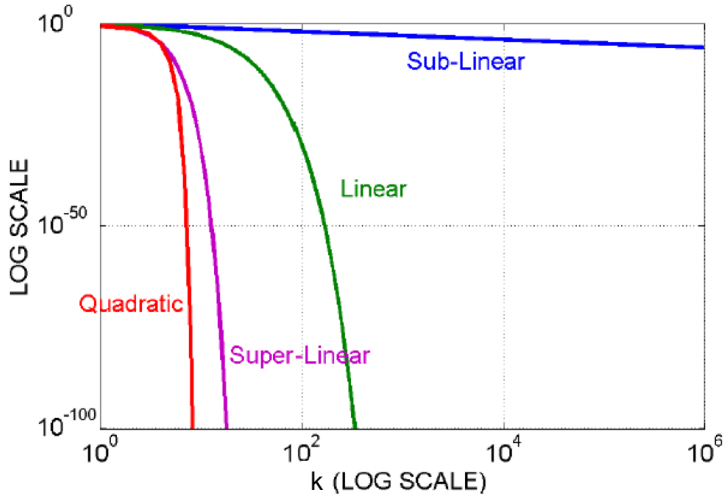
Sublinear: $\frac{t_{k+1}}{t_k} \to 1$ (for example, $t_k = K/k$)

Linear: $\frac{t_{k+1}}{t_k} \leq \gamma$, for $\gamma \in (0, 1)$ (for example, $t_k = \gamma^k$)

Superlinear: $\frac{t_{k+1}}{t_k} \to 0$ (for example, $t_k = \gamma^{k^2}$)

Quadratic: $\frac{t_{k+1}}{(t_k)^2} \to B < \infty$ (for example, $t_k = \gamma^{2^k}$, with $\gamma \in (0, 1)$)

The above sequences with $\gamma = 0.5$



Multistep Methods: The Heavy-Ball

Enhanced search direction with a contribution from the previous step.
(known as heavy ball, momentum, or two-step)

Consider first a constant step length α, and a second parameter β for the "momentum" term:

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$$

Multistep Methods: The Heavy-Ball

Enhanced search direction with a contribution from the previous step. (known as heavy ball, momentum, or two-step)

Consider first a constant step length α, and a second parameter β for the "momentum" term:

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$$

Analyze by defining a composite iterate vector:

$$w_k := \begin{bmatrix} x_k - x^* \\ x_{k-1} - x^* \end{bmatrix}.$$

Thus

$$w_{k+1} = B w_k + o(\|w_k\|), \qquad B := \begin{bmatrix} -\alpha \nabla^2 f(x^*) + (1+\beta) I & -\beta I \\ I & 0 \end{bmatrix}.$$

Multistep Methods: The Heavy-Ball

Matrix B has same eigenvalues as

$$\begin{bmatrix} -\alpha\Lambda + (1+\beta)I & -\beta I \\ I & 0 \end{bmatrix}, \qquad \Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n),$$

where λ_i are the eigenvalues of $\nabla^2 f(x^*)$.

Multistep Methods: The Heavy-Ball

Matrix B has same eigenvalues as

$$\begin{bmatrix} -\alpha\Lambda + (1+\beta)I & -\beta I \\ I & 0 \end{bmatrix}, \qquad \Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n),$$

where λ_i are the eigenvalues of $\nabla^2 f(x^*)$.

Choose α, β to explicitly minimize the max eigenvalue of B, obtain

$$\alpha = \frac{4}{L}\frac{1}{(1+1/\sqrt{\kappa})^2}, \quad \beta = \left(1 - \frac{2}{\sqrt{\kappa}+1}\right)^2.$$

Multistep Methods: The Heavy-Ball

Matrix B has same eigenvalues as

$$\begin{bmatrix} -\alpha\Lambda + (1+\beta)I & -\beta I \\ I & 0 \end{bmatrix}, \qquad \Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n),$$

where λ_i are the eigenvalues of $\nabla^2 f(x^*)$.

Choose α, β to explicitly minimize the max eigenvalue of B, obtain

$$\alpha = \frac{4}{L}\frac{1}{(1+1/\sqrt{\kappa})^2}, \quad \beta = \left(1 - \frac{2}{\sqrt{\kappa}+1}\right)^2.$$

Leads to linear convergence for $\|x_k - x^*\|$ with rate approximately

$$\left(1 - \frac{2}{\sqrt{\kappa}+1}\right).$$

Summary: Linear Convergence, Strictly Convex f

- Steepest descent: Linear rate approx $\left(1 - \frac{2}{\kappa}\right)$;
- Heavy-ball: Linear rate approx $\left(1 - \frac{2}{\sqrt{\kappa}}\right)$.

Summary: Linear Convergence, Strictly Convex f

- Steepest descent: Linear rate approx $\left(1 - \frac{2}{\kappa}\right)$;
- Heavy-ball: Linear rate approx $\left(1 - \frac{2}{\sqrt{\kappa}}\right)$.

Big difference! To reduce $\|x_k - x^*\|$ by a factor ϵ, need k large enough that

$$\left(1 - \frac{2}{\kappa}\right)^k \leq \epsilon \quad \Leftarrow \quad k \geq \frac{\kappa}{2}|\log \epsilon| \quad \text{(steepest descent)}$$

$$\left(1 - \frac{2}{\sqrt{\kappa}}\right)^k \leq \epsilon \quad \Leftarrow \quad k \geq \frac{\sqrt{\kappa}}{2}|\log \epsilon| \quad \text{(heavy-ball)}$$

A factor of $\sqrt{\kappa}$ difference; e.g. if $\kappa = 1000$ (not at all uncommon in inverse problems), need ~ 30 times fewer steps.

Accelerated First-Order Methods

Accelerate the rate to $1/k^2$ for weakly convex, while retaining the linear rate (related to $\sqrt{\kappa}$) for strongly convex case.

Accelerated First-Order Methods

Accelerate the rate to $1/k^2$ for weakly convex, while retaining the linear rate (related to $\sqrt{\kappa}$) for strongly convex case.

Nesterov (1983) describes a method that requires κ.

Initialize: Choose x_0, $\alpha_0 \in (0,1)$; set $y_0 \leftarrow x_0$.

Iterate: $x_{k+1} \leftarrow y_k - \frac{1}{L}\nabla f(y_k)$; (*short-step*)

find $\alpha_{k+1} \in (0,1)$: $\alpha_{k+1}^2 = (1-\alpha_{k+1})\alpha_k^2 + \frac{\alpha_{k+1}}{\kappa}$;

set $\beta_k = \dfrac{\alpha_k(1-\alpha_k)}{\alpha_k^2 + \alpha_{k+1}}$;

set $y_{k+1} \leftarrow x_{k+1} + \beta_k(x_{k+1} - x_k)$.

Still works for weakly convex ($\kappa = \infty$).

Convergence Results: Nesterov

If $\alpha_0 \geq 1/\sqrt{\kappa}$, have

$$f(x_k) - f(x^*) \leq c_1 \min\left(\left(1 - \frac{1}{\sqrt{\kappa}}\right)^k, \frac{4L}{(\sqrt{L} + c_2 k)^2}\right),$$

where constants c_1 and c_2 depend on x_0, α_0, L.

- Linear convergence "heavy-ball" rate for strongly convex f;
- $1/k^2$ sublinear rate otherwise.

In the special case of $\alpha_0 = 1/\sqrt{\kappa}$, this scheme yields

$$\alpha_k \equiv \frac{1}{\sqrt{\kappa}}, \qquad \beta_k \equiv 1 - \frac{2}{\sqrt{\kappa} + 1}.$$

Beck and Teboulle

Beck and Teboulle (2009) propose a similar algorithm.

Initialize: Choose x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate:
$$x_k \leftarrow y_k - \tfrac{1}{L}\nabla f(y_k);$$
$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$
$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

Beck and Teboulle

Beck and Teboulle (2009) propose a similar algorithm.

Initialize: Choose x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate:
$$x_k \leftarrow y_k - \tfrac{1}{L}\nabla f(y_k);$$
$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$
$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

For (weakly) convex f, converges with $f(x_k) - f(x^*) \sim 1/k^2$.

Beck and Teboulle

Beck and Teboulle (2009) propose a similar algorithm.

Initialize: Choose x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate:
$$x_k \leftarrow y_k - \tfrac{1}{L}\nabla f(y_k);$$
$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$
$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

For (weakly) convex f, converges with $f(x_k) - f(x^*) \sim 1/k^2$.

When L is not known, increase an estimate of L until it's big enough.

Beck and Teboulle

Beck and Teboulle (2009) propose a similar algorithm.

Initialize: Choose x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate:
$$x_k \leftarrow y_k - \tfrac{1}{L}\nabla f(y_k);$$
$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$
$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

For (weakly) convex f, converges with $f(x_k) - f(x^*) \sim 1/k^2$.

When L is not known, increase an estimate of L until it's big enough.

Beck and Teboulle (2009) do the convergence analysis in 2-3 pages; elementary, but not intuitive.

A Non-Monotone Gradient Method: Barzilai-Borwein

Barzilai and Borwein (1988) (BB) proposed an unusual choice of α_k.
Allows f to increase (sometimes a lot) on some steps: non-monotone.

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \alpha_k := \arg\min_{\alpha} \|s_k - \alpha z_k\|^2,$$

where

$$s_k := x_k - x_{k-1}, \qquad z_k := \nabla f(x_k) - \nabla f(x_{k-1}).$$

A Non-Monotone Gradient Method: Barzilai-Borwein

Barzilai and Borwein (1988) (BB) proposed an unusual choice of α_k.
Allows f to increase (sometimes a lot) on some steps: non-monotone.

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \alpha_k := \arg\min_{\alpha} \|s_k - \alpha z_k\|^2,$$

where

$$s_k := x_k - x_{k-1}, \qquad z_k := \nabla f(x_k) - \nabla f(x_{k-1}).$$

Explicitly, we have

$$\alpha_k = \frac{s_k^T z_k}{z_k^T z_k}.$$

A Non-Monotone Gradient Method: Barzilai-Borwein

Barzilai and Borwein (1988) (BB) proposed an unusual choice of α_k.
Allows f to increase (sometimes a lot) on some steps: non-monotone.

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \alpha_k := \arg\min_{\alpha} \|s_k - \alpha z_k\|^2,$$

where

$$s_k := x_k - x_{k-1}, \qquad z_k := \nabla f(x_k) - \nabla f(x_{k-1}).$$

Explicitly, we have

$$\alpha_k = \frac{s_k^T z_k}{z_k^T z_k}.$$

Note that for $f(x) = \frac{1}{2} x^T A x$, we have

$$\alpha_k = \frac{s_k^T A s_k}{s_k^T A^2 s_k} \in \left[\frac{1}{L}, \frac{1}{\mu}\right].$$

A Non-Monotone Gradient Method: Barzilai-Borwein

Barzilai and Borwein (1988) (BB) proposed an unusual choice of α_k.
Allows f to increase (sometimes a lot) on some steps: non-monotone.

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \alpha_k := \arg\min_{\alpha} \|s_k - \alpha z_k\|^2,$$

where

$$s_k := x_k - x_{k-1}, \qquad z_k := \nabla f(x_k) - \nabla f(x_{k-1}).$$

Explicitly, we have

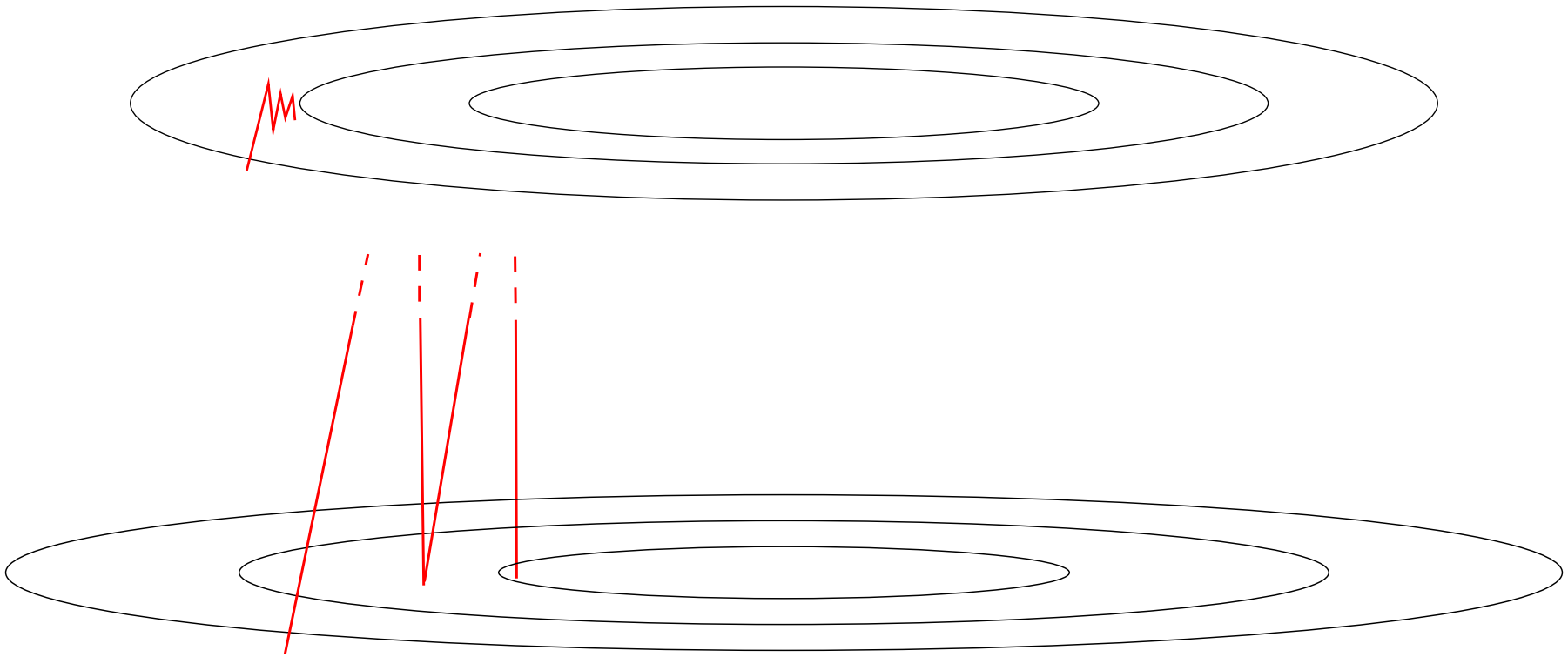
$$\alpha_k = \frac{s_k^T z_k}{z_k^T z_k}.$$

Note that for $f(x) = \frac{1}{2} x^T A x$, we have

$$\alpha_k = \frac{s_k^T A s_k}{s_k^T A^2 s_k} \in \left[\frac{1}{L}, \frac{1}{\mu}\right].$$

BB can be seen as a quasi-Newton method, with Hessian $\simeq \alpha_k^{-1} I$.

Comparison: BB vs Greedy Steepest Descent



Extending to the Constrained Case: $x \in \Omega$

How to change these methods to handle the constraint $x \in \Omega$?

(Ω is a closed convex set)

Extending to the Constrained Case: $x \in \Omega$

How to change these methods to handle the constraint $x \in \Omega$?

(Ω is a closed convex set)

Some algorithms and theory stay much the same,

...if we can involve the constraint $x \in \Omega$ explicity in the subproblems.

Extending to the Constrained Case: $x \in \Omega$

How to change these methods to handle the constraint $x \in \Omega$?

(Ω is a closed convex set)

Some algorithms and theory stay much the same,

...if we can involve the constraint $x \in \Omega$ explicity in the subproblems.

Example: Nesterov's constant step scheme requires just one calculation to be changed from the unconstrained version.

Initialize: Choose x_0, $\alpha_0 \in (0,1)$; set $y_0 \leftarrow x_0$.

Iterate: $x_{k+1} \leftarrow \arg\min_{y \in \Omega} \frac{1}{2}\|y - [y_k - \frac{1}{L}\nabla f(y_k)]\|_2^2$;

find $\alpha_{k+1} \in (0,1)$: $\alpha_{k+1}^2 = (1-\alpha_{k+1})\alpha_k^2 + \frac{\alpha_{k+1}}{\kappa}$;

set $\beta_k = \frac{\alpha_k(1-\alpha_k)}{\alpha_k^2+\alpha_{k+1}}$;

set $y_{k+1} \leftarrow x_{k+1} + \beta_k(x_{k+1} - x_k)$.

Convergence theory is unchanged.

Regularized Optimization

How to change these methods to handle regularizers?

$$\min_x \ f(x) + \tau\psi(x),$$

where f is convex and smooth, while ψ is convex but usually **nonsmooth**.

Regularized Optimization

How to change these methods to handle regularizers?

$$\min_x \ f(x) + \tau\psi(x),$$

where f is convex and smooth, while ψ is convex but usually **nonsmooth**.

Often, all that is needed is to change the update step to

$$x_{k+1} = \arg\min_x \|x - \Phi(x_k)\|_2^2 + \lambda\psi(x).$$

where $\Phi(x_k)$ is gradient descent step, or something more complicated (such as heavy ball, with $\Phi(x_k, x_{k-1})$, or some other accelerated method).

Regularized Optimization

How to change these methods to handle regularizers?

$$\min_x \ f(x) + \tau\psi(x),$$

where f is convex and smooth, while ψ is convex but usually **nonsmooth**.

Often, all that is needed is to change the update step to

$$x_{k+1} = \arg\min_x \|x - \Phi(x_k)\|_2^2 + \lambda\psi(x).$$

where $\Phi(x_k)$ is gradient descent step, or something more complicated (such as heavy ball, with $\Phi(x_k, x_{k-1})$, or some other accelerated method).

This is the shrinkage/tresholding step; how to solve it with a nonsmooth ψ? That is the topic of the following slides.

Handling Nonsmoothness (e.g. ℓ_1 Norm)

Convexity $\Rightarrow$ continuity (on the domain of the function).

Convexity $\not\Rightarrow$ differentiability (e.g., $\psi(x) = \|x\|_1$).

Subgradients generalize gradients for general convex functions:

Handling Nonsmoothness (e.g. ℓ_1 Norm)

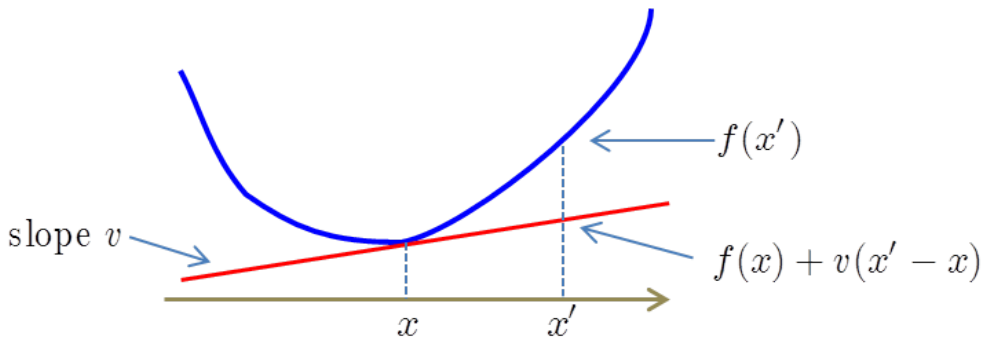
Convexity $\Rightarrow$ continuity (on the domain of the function).

Convexity $\not\Rightarrow$ differentiability (e.g., $\psi(x) = \|x\|_1$).

Subgradients generalize gradients for general convex functions:

v is a subgradient of f at x if $f(x') \geq f(x) + v^T(x' - x)$

Subdifferential: $\partial f(x) = \{\text{all subgradients of } f \text{ at } x\}$



linear lower bound

Handling Nonsmoothness (e.g. ℓ_1 Norm)

Convexity $\Rightarrow$ continuity (on the domain of the function).

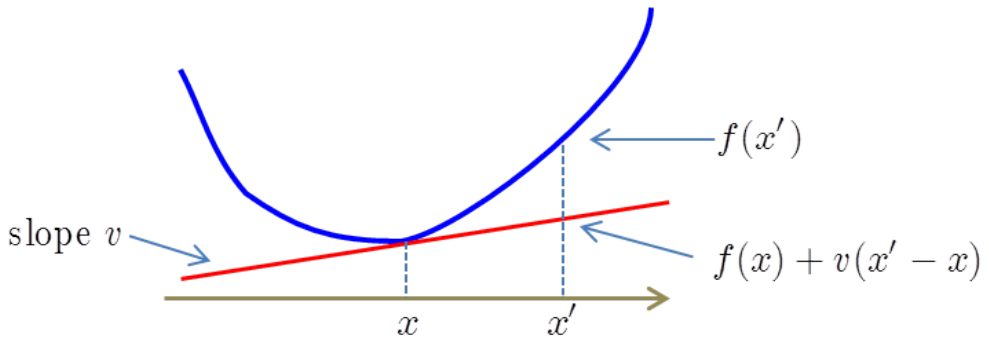
Convexity $\not\Rightarrow$ differentiability (e.g., $\psi(x) = \|x\|_1$).

Subgradients generalize gradients for general convex functions:

v is a subgradient of f at x if $f(x') \geq f(x) + v^T(x' - x)$

Subdifferential: $\partial f(x) = \{\text{all subgradients of } f \text{ at } x\}$

If f is differentiable, $\partial f(x) = \{\nabla f(x)\}$



linear lower bound

Handling Nonsmoothness (e.g. ℓ_1 Norm)

Convexity $\Rightarrow$ continuity (on the domain of the function).

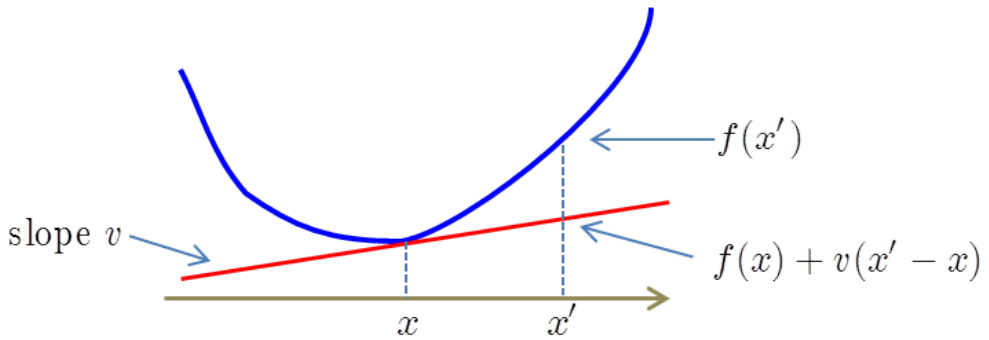
Convexity $\not\Rightarrow$ differentiability (e.g., $\psi(x) = \|x\|_1$).

Subgradients generalize gradients for general convex functions:

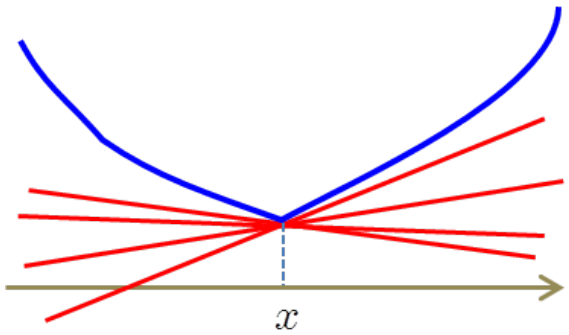
v is a subgradient of f at x if $f(x') \geq f(x) + v^T(x' - x)$

Subdifferential: $\partial f(x) = \{\text{all subgradients of } f \text{ at } x\}$

If f is differentiable, $\partial f(x) = \{\nabla f(x)\}$



linear lower bound



nondifferentiable case

More on Subgradients and Subdifferentials

The subdifferential is a set-valued function:

$$f : \mathbb{R}^d \to \mathbb{R} \quad \Rightarrow \quad \partial f : \mathbb{R}^d \to 2^{\mathbb{R}^d} \quad \text{(power set of } \mathbb{R}^d\text{)}$$

More on Subgradients and Subdifferentials

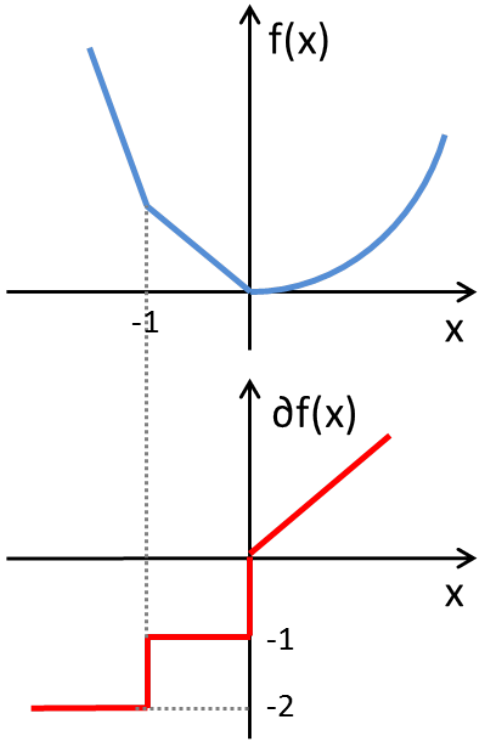
The subdifferential is a set-valued function:

$$f : \mathbb{R}^d \to \mathbb{R} \quad \Rightarrow \quad \partial f : \mathbb{R}^d \to 2^{\mathbb{R}^d} \quad \text{(power set of } \mathbb{R}^d\text{)}$$

Example:

$$f(x) = \begin{cases} -2x - 1, & x \leq -1 \\ -x, & -1 < x \leq 0 \\ x^2/2, & x > 0 \end{cases}$$

$$\partial f(x) = \begin{cases} \{-2\}, & x < -1 \\ [-2, -1], & x = -1 \\ \{-1\}, & -1 < x < 0 \\ [-1, 0], & x = 0 \\ \{x\}, & x > 0 \end{cases}$$



More on Subgradients and Subdifferentials

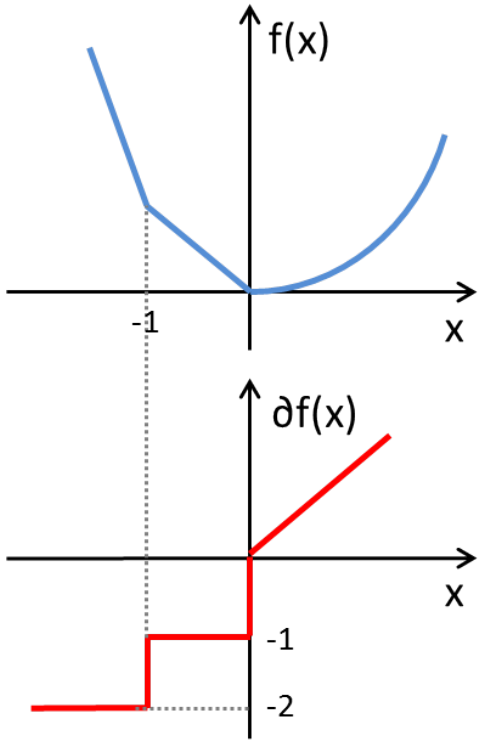
The subdifferential is a set-valued function:

$$f : \mathbb{R}^d \to \mathbb{R} \quad \Rightarrow \quad \partial f : \mathbb{R}^d \to 2^{\mathbb{R}^d} \quad \text{(power set of } \mathbb{R}^d\text{)}$$

Example:

$$f(x) = \begin{cases} -2x - 1, & x \leq -1 \\ -x, & -1 < x \leq 0 \\ x^2/2, & x > 0 \end{cases}$$

$$\partial f(x) = \begin{cases} \{-2\}, & x < -1 \\ [-2, -1], & x = -1 \\ \{-1\}, & -1 < x < 0 \\ [-1, 0], & x = 0 \\ \{x\}, & x > 0 \end{cases}$$



Fermat's Rule: $x \in \arg\min_x f(x) \Leftrightarrow 0 \in \partial f(x)$

A Key Tool: Moreau's Proximity Operators

Moreau (1962) proximity operator

$$\boxed{\widehat{x} \in \arg\min_x \frac{1}{2}\|x - y\|_2^2 + \psi(x) =: \text{prox}_\psi(y)}$$

...well defined for convex ψ, since $\|\cdot - y\|_2^2$ is coercive and strictly convex.

A Key Tool: Moreau's Proximity Operators

Moreau (1962) proximity operator

$$\boxed{\widehat{x} \in \arg\min_x \frac{1}{2}\|x-y\|_2^2 + \psi(x) =: \text{prox}_\psi(y)}$$

...well defined for convex ψ, since $\|\cdot - y\|_2^2$ is coercive and strictly convex.

Example: (seen above) $\text{prox}_{\tau|\cdot|}(y) = \text{soft}(y,\tau) = \text{sign}(y)\max\{|y|-\tau, 0\}$

A Key Tool: Moreau's Proximity Operators

Moreau (1962) proximity operator

$$\boxed{\widehat{x} \in \arg\min_x \frac{1}{2}\|x-y\|_2^2 + \psi(x) =: \text{prox}_\psi(y)}$$

...well defined for convex ψ, since $\|\cdot - y\|_2^2$ is coercive and strictly convex.

Example: (seen above) $\text{prox}_{\tau|\cdot|}(y) = \text{soft}(y,\tau) = \text{sign}(y)\max\{|y|-\tau, 0\}$

Block separability: $x = (x_1, ..., x_N)$ (a partition of the components of x)

$$\psi(x) = \sum_i \psi_i(x_i) \;\Rightarrow\; (\text{prox}_\psi(y))_i = \text{prox}_{\psi_i}(y_i)$$

Relationship with subdifferential: $z = \text{prox}_\psi(y) \;\Leftrightarrow\; z - y \in \partial\psi(z)$

A Key Tool: Moreau's Proximity Operators

Moreau (1962) proximity operator

$$\boxed{\widehat{x} \in \arg\min_x \frac{1}{2}\|x-y\|_2^2 + \psi(x) =: \text{prox}_\psi(y)}$$

...well defined for convex ψ, since $\|\cdot - y\|_2^2$ is coercive and strictly convex.

Example: (seen above) $\text{prox}_{\tau|\cdot|}(y) = \text{soft}(y,\tau) = \text{sign}(y)\max\{|y|-\tau, 0\}$

Block separability: $x = (x_1, ..., x_N)$ (a partition of the components of x)

$$\psi(x) = \sum_i \psi_i(x_i) \;\Rightarrow\; (\text{prox}_\psi(y))_i = \text{prox}_{\psi_i}(y_i)$$

Relationship with subdifferential: $z = \text{prox}_\psi(y) \;\Leftrightarrow\; z - y \in \partial\psi(z)$

Resolvent: $z = \text{prox}_\psi(y) \;\Leftrightarrow\; 0 \in \partial\psi(z) + (z-y) \;\Leftrightarrow\; y \in (\partial\psi + I)z$

$$\text{prox}_\psi(y) = (\partial\psi + I)^{-1} y$$

Important Proximity Operators

- Soft-thresholding is the proximity operator of the ℓ_1 norm.

Important Proximity Operators

- Soft-thresholding is the proximity operator of the ℓ_1 norm.
- Consider the indicator $\iota_{\mathcal{S}}$ of a convex set $\mathcal{S}$;

$$\text{prox}_{\iota_{\mathcal{S}}}(u) = \arg\min_x \frac{1}{2}\|x-u\|_2^2 + \iota_{\mathcal{S}}(x) = \arg\min_{x\in\mathcal{S}} \frac{1}{2}\|x-y\|_2^2 = P_{\mathcal{S}}(u)$$

...the Euclidean projection on $\mathcal{S}$.

Important Proximity Operators

- Soft-thresholding is the proximity operator of the ℓ_1 norm.
- Consider the indicator $\iota_{\mathcal{S}}$ of a convex set $\mathcal{S}$;

$$\text{prox}_{\iota_{\mathcal{S}}}(u) = \arg\min_x \frac{1}{2}\|x-u\|_2^2 + \iota_{\mathcal{S}}(x) = \arg\min_{x\in\mathcal{S}} \frac{1}{2}\|x-y\|_2^2 = P_{\mathcal{S}}(u)$$

...the Euclidean projection on $\mathcal{S}$.

- Squared Euclidean norm (separable, smooth):

$$\text{prox}_{\tau\|\cdot\|_2^2}(y) = \arg\min_x \|x-y\|_2^2 + \tau\|x\|_2^2 = \frac{y}{1+\tau}$$

Important Proximity Operators

- Soft-thresholding is the proximity operator of the ℓ_1 norm.
- Consider the indicator $\iota_{\mathcal{S}}$ of a convex set $\mathcal{S}$;

$$\text{prox}_{\iota_{\mathcal{S}}}(u) = \arg\min_x \frac{1}{2}\|x-u\|_2^2 + \iota_{\mathcal{S}}(x) = \arg\min_{x\in\mathcal{S}} \frac{1}{2}\|x-y\|_2^2 = P_{\mathcal{S}}(u)$$

...the Euclidean projection on $\mathcal{S}$.

- Squared Euclidean norm (separable, smooth):

$$\text{prox}_{\tau\|\cdot\|_2^2}(y) = \arg\min_x \|x-y\|_2^2 + \tau\|x\|_2^2 = \frac{y}{1+\tau}$$

- Euclidean norm (not separable, nonsmooth):

$$\text{prox}_{\tau\|\cdot\|_2}(y) = \begin{cases} \frac{x}{\|x\|_2}(\|x\|_2 - \tau), & \text{if } \|x\|_2 > \tau \\ 0 & \text{if } \|x\|_2 \leq \tau \end{cases}$$

More Proximity Operators

	$\phi(x)$	$\operatorname{prox}_{\phi} x$
i	$\iota_{[\underline{\omega},\overline{\omega}]}(x)$	$P_{[\underline{\omega},\overline{\omega}]}\, x$
ii	$\sigma_{[\underline{\omega},\overline{\omega}]}(x) = \begin{cases} \underline{\omega} x & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ \overline{\omega} x & \text{otherwise} \end{cases}$	$\operatorname{soft}_{[\underline{\omega},\overline{\omega}]}(x) = \begin{cases} x - \underline{\omega} & \text{if } x < \underline{\omega} \\ 0 & \text{if } x \in [\underline{\omega},\overline{\omega}] \\ x - \overline{\omega} & \text{if } x > \overline{\omega} \end{cases}$
iii	$\psi(x) + \sigma_{[\underline{\omega},\overline{\omega}]}(x)$ $\psi \in \Gamma_0(\mathbb{R})$ differentiable at 0 $\psi'(0) = 0$	$\operatorname{prox}_{\psi}(\operatorname{soft}_{[\underline{\omega},\overline{\omega}]}(x))$
iv	$\max\{\lvert x\rvert - \omega, 0\}$	$\begin{cases} x & \text{if } \lvert x\rvert < \omega \\ \operatorname{sign}(x)\omega & \text{if } \omega \leq \lvert x\rvert \leq 2\omega \\ \operatorname{sign}(x)(\lvert x\rvert - \omega) & \text{if } \lvert x\rvert > 2\omega \end{cases}$
v	$\kappa\lvert x\rvert^q$	$\operatorname{sign}(x)p$, where $p \geq 0$ and $p + q\kappa p^{q-1} = \lvert x\rvert$
vi	$\begin{cases} \kappa x^2 & \text{if } \lvert x\rvert \leq \omega/\sqrt{2\kappa} \\ \omega\sqrt{2\kappa}\lvert x\rvert - \omega^2/2 & \text{otherwise} \end{cases}$	$\begin{cases} x/(2\kappa+1) & \text{if } \lvert x\rvert \leq \omega(2\kappa+1)/\sqrt{2\kappa} \\ x - \omega\sqrt{2\kappa}\operatorname{sign}(x) & \text{otherwise} \end{cases}$
vii	$\omega\lvert x\rvert + \tau\lvert x\rvert^2 + \kappa\lvert x\rvert^q$	$\operatorname{sign}(x)\operatorname{prox}_{\kappa\lvert\cdot\rvert^q/(2\tau+1)}\dfrac{\max\{\lvert x\rvert - \omega, 0\}}{2\tau+1}$
viii	$\omega\lvert x\rvert - \ln(1+\omega\lvert x\rvert)$	$(2\omega)^{-1}\operatorname{sign}(x)\Big(\omega\lvert x\rvert - \omega^2 - 1 + \sqrt{\lvert\omega\lvert x\rvert - \omega^2 - 1\rvert^2 + 4\omega\lvert x\rvert}\Big)$
ix	$\begin{cases} \text{if } x \geq 0 \\ +\infty \quad \text{otherwise} \end{cases}$	$\begin{cases} x - \omega & \text{if } x \\ & \text{otherwise} \end{cases}$
x	$\begin{cases} & x \\ & \text{otherwise} \end{cases}$	p where p and p
xi	$\begin{cases} \omega x^{-q} & \text{if } x > 0 \\ +\infty & \text{otherwise} \end{cases}$	$p > 0$ such that $p^{q+2} - xp^{q+1} = \omega q$
xii	$\begin{cases} x\ln(x) & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ +\infty & \text{otherwise} \end{cases}$	$W(e^{x-1})$, where W is the Lambert W-function
xiii	$\begin{cases} -\ln(x-\underline{\omega}) + \ln(-\underline{\omega}) & \text{if } x \in]\underline{\omega}, 0] \\ -\ln(\overline{\omega} - x) + \ln(\overline{\omega}) & \text{if } x \in]0, \overline{\omega}[\\ +\infty & \text{otherwise} \end{cases}$ $\underline{\omega} < 0 < \overline{\omega}$	$\begin{cases} \frac{1}{2}\Big(x + \underline{\omega} + \sqrt{\lvert x-\underline{\omega}\rvert^2 + 4}\Big) & \text{if } x < 1/\underline{\omega} \\ \frac{1}{2}\Big(x + \overline{\omega} - \sqrt{\lvert x-\overline{\omega}\rvert^2 + 4}\Big) & \text{if } x > 1/\overline{\omega} \\ 0 & \text{otherwise} \end{cases}$ (see Figure 1)
xiv	$\begin{cases} -\kappa\ln(x) + \tau x^2/2 + \alpha x & \text{if } x > 0 \\ +\infty & \text{otherwise} \end{cases}$	$\dfrac{1}{2(1+\tau)}\Big(x - \alpha + \sqrt{\lvert x-\alpha\rvert^2 + 4\kappa(1+\tau)}\Big)$
xv	$\begin{cases} -\kappa\ln(x) + \alpha x + \omega x^{-1} & \text{if } x > 0 \\ +\infty & \text{otherwise} \end{cases}$	$p > 0$ such that $p^3 + (\alpha - x)p^2 - \kappa p = \omega$
xvi	$\begin{cases} -\kappa\ln(x) + \omega x^q & \text{if } x > 0 \\ +\infty & \text{otherwise} \end{cases}$	$p > 0$ such that $q\omega p^q + p^2 - xp = \kappa$
xvii	$\begin{cases} -\underline{\kappa}\ln(x-\underline{\omega}) - \overline{\kappa}\ln(\overline{\omega} - x) & \\ \qquad \text{if } x \in]\underline{\omega}, \overline{\omega}[& \\ +\infty \quad \text{otherwise} & \end{cases}$	$p \in]\underline{\omega}, \overline{\omega}[$ such that $p^3 - (\underline{\omega} + \overline{\omega} + x)p^2 +$ $(\underline{\omega}\overline{\omega} - \underline{\kappa} - \overline{\kappa} + (\underline{\omega} + \overline{\omega})x)p = \underline{\omega}\overline{\omega}x - \underline{\omega}\overline{\kappa} - \overline{\omega}\underline{\kappa}$

Many others!

(Combettes and Pesquet, 2011)

Another Key Tool: Fenchel-Legendre Conjugates

The Fenchel-Legendre conjugate of a proper convex function f — denoted by $f^* : \mathbb{R}^n \to \bar{\mathbb{R}}$ — is defined by

$$\boxed{f^*(u) = \sup_x x^T u - f(x)}$$

Another Key Tool: Fenchel-Legendre Conjugates

The Fenchel-Legendre conjugate of a proper convex function f — denoted by $f^* : \mathbb{R}^n \to \bar{\mathbb{R}}$ — is defined by

$$\boxed{f^*(u) = \sup_x x^T u - f(x)}$$

Main properties and relationship with proximity operators:

- Biconjugation: if f is convex and proper, $f^{**} = f$.

Another Key Tool: Fenchel-Legendre Conjugates

The Fenchel-Legendre conjugate of a proper convex function f — denoted by $f^* : \mathbb{R}^n \to \bar{\mathbb{R}}$ — is defined by

$$\boxed{f^*(u) = \sup_x x^T u - f(x)}$$

Main properties and relationship with proximity operators:

- Biconjugation: if f is convex and proper, $f^{**} = f$.
- Moreau's decomposition: $\text{prox}_f(u) + \text{prox}_{f^*}(u) = u$

  ...meaning that, if you know prox_f, you know prox_{f^*}, and vice-versa.

Another Key Tool: Fenchel-Legendre Conjugates

The Fenchel-Legendre conjugate of a proper convex function f — denoted by $f^* : \mathbb{R}^n \to \bar{\mathbb{R}}$ — is defined by

$$\boxed{f^*(u) = \sup_x x^T u - f(x)}$$

Main properties and relationship with proximity operators:

- Biconjugation: if f is convex and proper, $f^{**} = f$.
- Moreau's decomposition: $\text{prox}_f(u) + \text{prox}_{f^*}(u) = u$

  ...meaning that, if you know prox_f, you know prox_{f^*}, and vice-versa.
- Conjugate of indicator: if $f(x) = \iota_C(x)$, where C is a convex set,

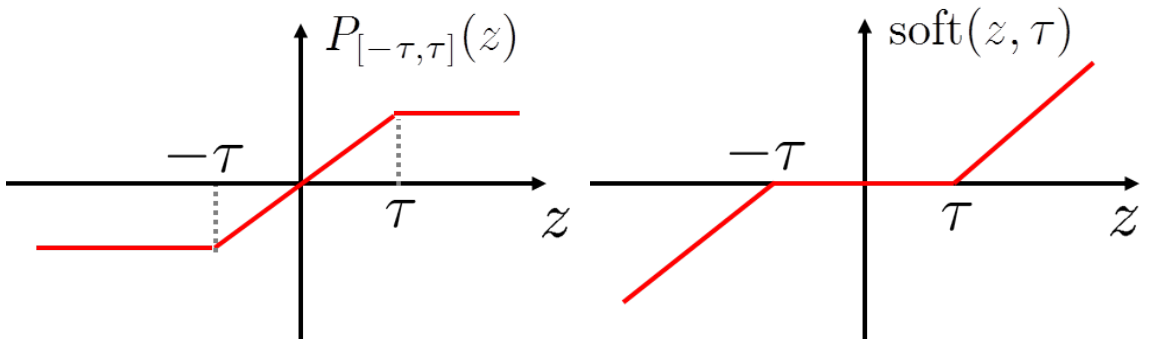
  $f^*(u) = \sup_x x^T u - \iota_C(x) = \sup_{x \in C} x^T u \equiv \sigma_C(u)$ (support function of C).

From Conjugates to Proximity Operators

Notice that $|u| = \sup_{x \in [-1,1]} x^T u = \sigma_{[-1,1]}(u)$, thus $|\cdot|^* = \iota_{[-1,1]}$.

Using Moreau's decomposition, we easily derive the soft-threshold:

$$\text{prox}_{\tau|\cdot|} = 1 - \text{prox}_{\iota_{[-\tau,\tau]}} = 1 - P_{[-\tau,\tau]} = \text{soft}(\cdot, \tau)$$



Conjugate of a norm: if $f(x) = \tau\|x\|_p$ then $f^* = \iota_{\{x:\|x\|_q \leq \tau\}}$,

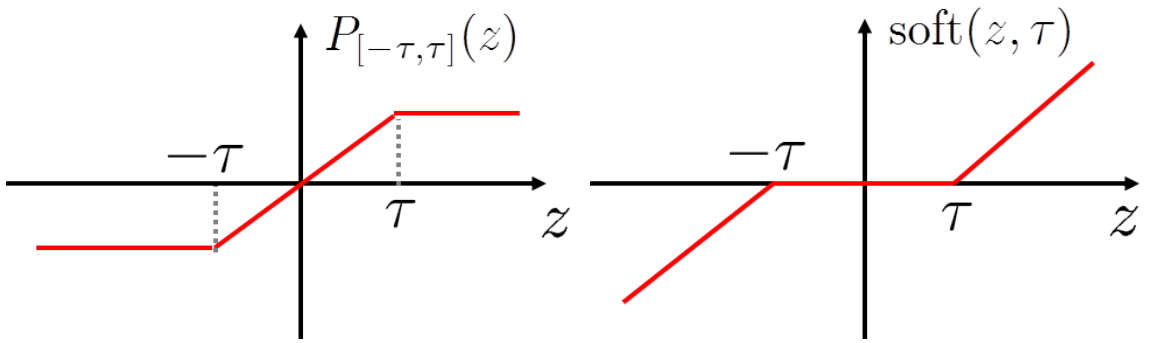
where $\frac{1}{q} + \frac{1}{p} = 1$ (a Hölder pair, or Hölder conjugates).

From Conjugates to Proximity Operators

Notice that $|u| = \sup_{x \in [-1,1]} x^T u = \sigma_{[-1,1]}(u)$, thus $|\cdot|^* = \iota_{[-1,1]}$.

Using Moreau's decomposition, we easily derive the soft-threshold:

$$\text{prox}_{\tau|\cdot|} = 1 - \text{prox}_{\iota_{[-\tau,\tau]}} = 1 - P_{[-\tau,\tau]} = \text{soft}(\cdot, \tau)$$



Conjugate of a norm: if $f(x) = \tau\|x\|_p$ then $f^* = \iota_{\{x:\|x\|_q \leq \tau\}}$,

where $\frac{1}{q} + \frac{1}{p} = 1$ (a Hölder pair, or Hölder conjugates).

That is, $\|\cdot\|_p$ and $\|\cdot\|_q$ are dual norms:

$$\|z\|_q = \sup\{x^T z : \|x\|_p \leq 1\} = \sup_{x \in B_p(1)} x^T z = \sigma_{B_p(1)}(z)$$

From Conjugates to Proximity Operators

- Proximity of norm:

$$\boxed{\text{prox}_{\tau\|\cdot\|_p} = I - P_{B_q(\tau)}}$$

where $B_q(\tau) = \{x : \ \|x\|_q \leq \tau\}$ and $\frac{1}{q} + \frac{1}{p} = 1$.

From Conjugates to Proximity Operators

- Proximity of norm:

$$\boxed{\text{prox}_{\tau\|\cdot\|_p} = I - P_{B_q(\tau)}}$$

where $B_q(\tau) = \{x : \ \|x\|_q \leq \tau\}$ and $\frac{1}{q} + \frac{1}{p} = 1$.

- Example: computing $\text{prox}_{\|\cdot\|_\infty}$ (notice ℓ_∞ is not separable):

Since $\frac{1}{\infty} + \frac{1}{1} = 1$,

$$\text{prox}_{\tau\|\cdot\|_\infty} = I - P_{B_1(\tau)}$$

... the proximity operator of ℓ_∞ norm is the residual of the projection on an ℓ_1 ball.

From Conjugates to Proximity Operators

- Proximity of norm:

$$\boxed{\text{prox}_{\tau\|\cdot\|_p} = I - P_{B_q(\tau)}}$$

where $B_q(\tau) = \{x : \ \|x\|_q \leq \tau\}$ and $\frac{1}{q} + \frac{1}{p} = 1$.

- Example: computing $\text{prox}_{\|\cdot\|_\infty}$ (notice ℓ_∞ is not separable):

Since $\frac{1}{\infty} + \frac{1}{1} = 1$,

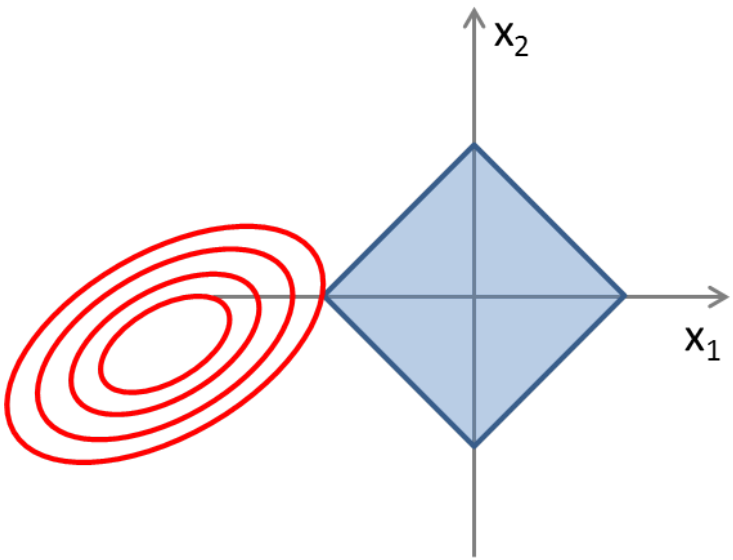
$$\text{prox}_{\tau\|\cdot\|_\infty} = I - P_{B_1(\tau)}$$

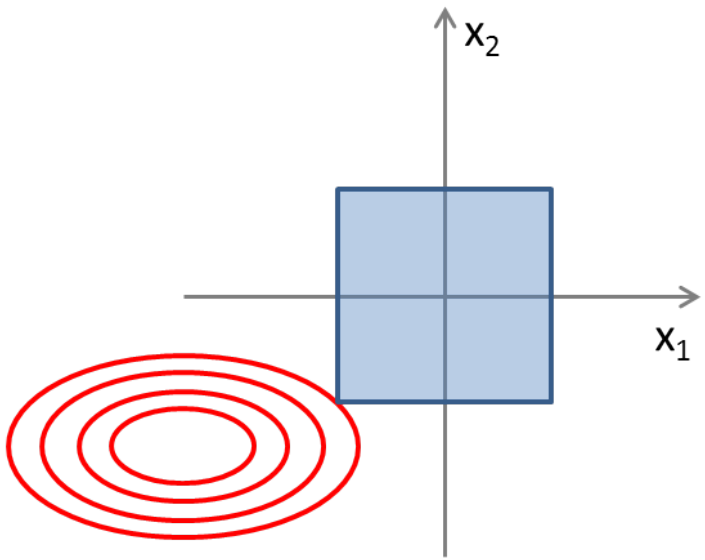
... the proximity operator of ℓ_∞ norm is the residual of the projection on an ℓ_1 ball.

- Projection on ℓ_1 ball has no closed form, but there are efficient (linear cost) algorithms (Brucker, 1984), (Maculan and de Paula, 1989).

Geometry and Effect of $\text{prox}_{\ell_\infty}$

Whereas ℓ_1 promotes sparsity, ℓ_∞ promotes equality (in absolute value).

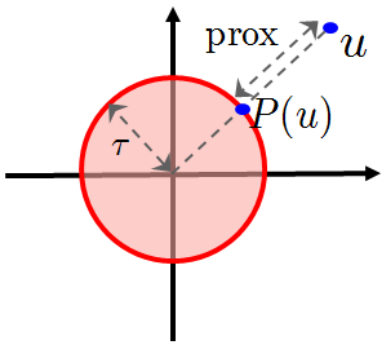




From Conjugates to Proximity Operators

The dual of the ℓ_2 norm is the ℓ_2 norm.

$$\text{prox}_{\tau\|\cdot\|_2}(u) = u - P_{\{x:\|x\|_2\leq\tau\}}(u)$$



$$= u - \begin{cases} u & \Leftarrow \quad \|u\|_2 \leq \tau \\ \tau\, u/\|u\|_2 & \Leftarrow \quad \|u\|_2 > \tau \end{cases}$$

$$= \frac{u}{\|u\|_2} \max\{0, \|u\|_2 - \tau\}$$

vector soft thresholding

Matrix Nuclear Norm and its Prox Operator

- Recall the trace/nuclear norm: $\|X\|_* = \sum_{i=1}^{\min\{m,n\}} \sigma_i$.
- The dual of a Schatten p-norm is a Schatten q-norm, with $\frac{1}{q} + \frac{1}{p} = 1$. Thus, the dual of the nuclear norm is the spectral norm:

$$\|X\|_\infty = \max\left\{\sigma_1, ..., \sigma_{\min\{m,n\}}\right\}.$$

- If $Y = U\Lambda V^T$ is the SVD of Y, we have

$$\begin{aligned}\text{prox}_{\tau\|\cdot\|_*}(Y) &= U\Lambda V^T - P_{\{X:\max\{\sigma_1,...,\sigma_{\min\{m,n\}}\}\leq\tau\}}(U\Lambda V^T) \\ &= U\,\text{soft}\big(\Lambda,\tau\big)V^T.\end{aligned}$$

Atomic Norms: A Unified View

vectors			matrices		
norm	prox	atomic set	norm	prox	atomic set
ℓ_1 $\|x\|_1$	component soft thresholding	$\mathcal{A} = \{\pm e_i\}$ $\|\mathcal{A}\| = 2N$	nuclear $\|X\|_*$	singular value thresholding	$\mathcal{A}$ = set of all rank 1, norm 1 matrices
ℓ_∞ $\|x\|_\infty$	residual of projection on ℓ_1 ball	$\mathcal{A} = \{\pm 1\}^N$ $\|\mathcal{A}\| = 2^N$	spectral $\|X\|_2$	residual of s.v. proj. on ℓ_1 ball	$\mathcal{A}$ = set of all orthogonal matrices
ℓ_2 $\|x\|_2$	vector soft thresholding	$\mathcal{A}$ = set of all vectors with norm 1 $\|\mathcal{A}\| = \infty$	Frobenius $\|X\|_F$	matrix soft threshold.	$\mathcal{A}$ = all matrices of unit Frobenius norm.

Another Use of Fenchel-Legendre Conjugates

- The original problem $\min_x f(x) + \psi(x)$
- ... often has the form: $\min_x g(Ax) + \psi(x)$

Another Use of Fenchel-Legendre Conjugates

- The original problem $\min_x f(x) + \psi(x)$
- ... often has the form: $\min_x g(Ax) + \psi(x)$
- Using the definition of conjugate $g(Ax) = \sup_u u^T Ax - g^*(u)$

$$\begin{aligned}
\min_x g(Ax) + \psi(x) &= \inf_x \sup_u u^T Ax - g^*(u) + \psi(x) \\
&= \sup_u (-g^*(u) + \inf_x u^T Ax + \psi(x)) \\
&= \sup_u (-g^*(u) - \underbrace{\sup_x -x^T A^T u - \psi(x)}_{\psi^*(-A^T u)}) \\
&= -\inf_u g^*(u) + \psi^*(-A^T u)
\end{aligned}$$

Another Use of Fenchel-Legendre Conjugates

- The original problem $\min_x f(x) + \psi(x)$
- ... often has the form: $\min_x g(Ax) + \psi(x)$
- Using the definition of conjugate $g(Ax) = \sup_u u^T Ax - g^*(u)$

$$\begin{aligned}
\min_x g(Ax) + \psi(x) &= \inf_x \sup_u u^T Ax - g^*(u) + \psi(x) \\
&= \sup_u (-g^*(u) + \inf_x u^T Ax + \psi(x)) \\
&= \sup_u (-g^*(u) - \underbrace{\sup_x -x^T A^T u - \psi(x)}_{\psi^*(-A^T u)}) \\
&= -\inf_u g^*(u) + \psi^*(-A^T u)
\end{aligned}$$

- The problem $\inf_u g^*(u) + \psi^*(-A^T u)$ is sometimes easier to handle.

Basic Proximal-Gradient Algorithm

Use basic structure:

$$x_k = \arg\min_x \|x - \Phi(x_k)\|_2^2 + \psi(x).$$

with $\Phi(x_k)$ a simple gradient descent step, thus

$$\boxed{x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k)\big)}$$

Basic Proximal-Gradient Algorithm

Use basic structure:

$$x_k = \arg\min_x \|x - \Phi(x_k)\|_2^2 + \psi(x).$$

with $\Phi(x_k)$ a simple gradient descent step, thus

$$\boxed{x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k)\big)}$$

This approach goes by many names, such as

- "proximal gradient algorithm" (PGA),
- "iterative shrinkage/thresholding" (IST),
- "forward-backward splitting" (FBS)

Basic Proximal-Gradient Algorithm

Use basic structure:

$$x_k = \arg\min_x \|x - \Phi(x_k)\|_2^2 + \psi(x).$$

with $\Phi(x_k)$ a simple gradient descent step, thus

$$\boxed{x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k)\big)}$$

This approach goes by many names, such as

- "proximal gradient algorithm" (PGA),
- "iterative shrinkage/thresholding" (IST),
- "forward-backward splitting" (FBS)

Reinvented several times in different communities: optimization, PDEs, convex analysis, signal processing, machine learning.

Convergence of the Proximal-Gradient Algorithm

- Basic algorithm: $x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k)\big)$

Convergence of the Proximal-Gradient Algorithm

- Basic algorithm: $x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k)\big)$
- Generalized (possibly inexact) version:

$$x_{k+1} = (1 - \lambda_k) x_k + \lambda_k \Big(\text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k) + b_k\big) + a_k \Big)$$

where a_k and b_k are "errors" in computing the prox and the gradient; λ_k is an over-relaxation parameter.

Convergence of the Proximal-Gradient Algorithm

- Basic algorithm: $x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k)\big)$
- Generalized (possibly inexact) version:

$$x_{k+1} = (1 - \lambda_k) x_k + \lambda_k \Big(\text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k) + b_k\big) + a_k \Big)$$

  where a_k and b_k are "errors" in computing the prox and the gradient; λ_k is an over-relaxation parameter.

- Convergence is guaranteed (Combettes and Wajs, 2006) if
  - ✓ $0 < \inf \alpha_k \leq \sup \alpha_k < \frac{2}{L}$
  - ✓ $\lambda_k \in (0, 1]$, with $\inf \lambda_k > 0$
  - ✓ $\sum_k^\infty \|a_k\| < \infty$ and $\sum_k^\infty \|b_k\| < \infty$

Proximal-Gradient Algorithm: Quadratic Case

- Consider the quadratic case (of great interest): $f(x) = \frac{1}{2}\|Bx - b\|_2^2$.

Proximal-Gradient Algorithm: Quadratic Case

- Consider the quadratic case (of great interest): $f(x) = \frac{1}{2}\|Bx - b\|_2^2$.
- Here, $\nabla f(x) = B^T(Bx - b)$ and the IST/PGA/FBS algorithm is

$$x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k B^T(Bx - b)\big)$$

requires only matrix-vector multiplications with B and B^T.

Proximal-Gradient Algorithm: Quadratic Case

- Consider the quadratic case (of great interest): $f(x) = \frac{1}{2}\|Bx - b\|_2^2$.
- Here, $\nabla f(x) = B^T(Bx - b)$ and the IST/PGA/FBS algorithm is

$$x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k B^T(Bx - b)\big)$$

  requires only matrix-vector multiplications with B and B^T.
- Very important in large-scale applications, *e.g.*, image processing.
- Often, fast algorithms exist for computing these products (e.g. fast Fourier transforms or wavelet transforms), but these matrices cannot be formed and stored explicitly.

Proximal-Gradient Algorithm: Quadratic Case

- Consider the quadratic case (of great interest): $f(x) = \frac{1}{2}\|Bx - b\|_2^2$.
- Here, $\nabla f(x) = B^T(Bx - b)$ and the IST/PGA/FBS algorithm is

$$x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k B^T(Bx - b)\big)$$

  requires only matrix-vector multiplications with B and B^T.
- Very important in large-scale applications, *e.g.*, image processing.
- Often, fast algorithms exist for computing these products (e.g. fast Fourier transforms or wavelet transforms), but these matrices cannot be formed and stored explicitly.
- In this case, some more refined convergence results are available.

Proximal-Gradient Algorithm: Quadratic Case

- Consider the quadratic case (of great interest): $f(x) = \frac{1}{2}\|Bx - b\|_2^2$.
- Here, $\nabla f(x) = B^T(Bx - b)$ and the IST/PGA/FBS algorithm is

$$x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k B^T(Bx - b)\big)$$

  requires only matrix-vector multiplications with B and B^T.
- Very important in large-scale applications, *e.g.*, image processing.
- Often, fast algorithms exist for computing these products (e.g. fast Fourier transforms or wavelet transforms), but these matrices cannot be formed and stored explicitly.
- In this case, some more refined convergence results are available.
- Even more refined results are available if $\psi(x) = \tau\|x\|_1$

More on IST/FBS/PGA for the ℓ_2-ℓ_1 Case

- Problem: $\widehat{x} \in G = \arg\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$ (recall $B^T B \preceq LI$)
- IST/FBS/PGA becomes $\boxed{x_{k+1} = \text{soft}\big(x_k - \alpha B^T(Bx - b), \alpha\tau\big)}$ with $\alpha < 2/L$.

More on IST/FBS/PGA for the ℓ_2-ℓ_1 Case

- Problem: $\widehat{x} \in G = \arg\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$ (recall $B^T B \preceq LI$)
- IST/FBS/PGA becomes $\boxed{x_{k+1} = \text{soft}\big(x_k - \alpha B^T(Bx - b), \alpha\tau\big)}$ with $\alpha < 2/L$.
- The zero set: $\mathcal{Z} \subseteq \{1, ..., n\}$: $\widehat{x} \in G \Rightarrow \widehat{x}_{\mathcal{Z}} = 0$
- Zeros are found in a finite number of iterations (Hale et al., 2008): after a finite number of iterations $(x_k)_{\mathcal{Z}} = 0$.

More on IST/FBS/PGA for the ℓ_2-ℓ_1 Case

- Problem: $\widehat{x} \in G = \arg\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$ (recall $B^T B \preceq LI$)
- IST/FBS/PGA becomes $\boxed{x_{k+1} = \text{soft}\big(x_k - \alpha B^T(Bx - b), \alpha\tau\big)}$ with $\alpha < 2/L$.
- The zero set: $\mathcal{Z} \subseteq \{1, ..., n\}$: $\widehat{x} \in G \Rightarrow \widehat{x}_{\mathcal{Z}} = 0$
- Zeros are found in a finite number of iterations (Hale et al., 2008): after a finite number of iterations $(x_k)_{\mathcal{Z}} = 0$.
- After that, if $B_{\mathcal{Z}}^T B_{\mathcal{Z}} \succeq \mu I$, with $\mu > 0$ (thus $\kappa(B_{\mathcal{Z}}^T B_{\mathcal{Z}}) = L/\mu < \infty$), we have linear convergence

$$\|x_{k+1} - \widehat{x}\|_2 \leq \frac{1-\kappa}{1+\kappa}\|x_k - \widehat{x}\|_2$$

for the optimal choice $\alpha = 2/(L + \mu)$ (see unconstrained theory).

Heavy Ball Acceleration: FISTA

- FISTA (*fast iterative shrinkage-thresholding algorithm*) is heavy-ball-type acceleration of IST (based on Nesterov (1983)) (Beck and Teboulle, 2009).

Initialize: Choose $\alpha \leq 1/L$, x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate: $x_k \leftarrow \text{prox}_{\tau\alpha\psi}\big(y_k - \alpha\nabla f(y_k)\big)$;

$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$

$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

Heavy Ball Acceleration: FISTA

- FISTA (*fast iterative shrinkage-thresholding algorithm*) is heavy-ball-type acceleration of IST (based on Nesterov (1983)) (Beck and Teboulle, 2009).

Initialize: Choose $\alpha \leq 1/L$, x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate: $x_k \leftarrow \text{prox}_{\tau\alpha\psi}\big(y_k - \alpha\nabla f(y_k)\big)$;

$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$

$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

- Acceleration:

$$\text{FISTA: } f(x_k) - f(\widehat{x}) \sim O\left(\frac{1}{k^2}\right) \qquad \text{IST: } f(x_k) - f(\widehat{x}) \sim O\left(\frac{1}{k}\right).$$

Heavy Ball Acceleration: FISTA

- FISTA (*fast iterative shrinkage-thresholding algorithm*) is heavy-ball-type acceleration of IST (based on Nesterov (1983)) (Beck and Teboulle, 2009).

Initialize: Choose $\alpha \leq 1/L$, x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate: $x_k \leftarrow \text{prox}_{\tau\alpha\psi}\big(y_k - \alpha\nabla f(y_k)\big)$;

$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$

$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

- Acceleration:

$$\text{FISTA: } f(x_k) - f(\widehat{x}) \sim O\left(\frac{1}{k^2}\right) \qquad \text{IST: } f(x_k) - f(\widehat{x}) \sim O\left(\frac{1}{k}\right).$$

- When L is not known, increase an estimate of L until it's big enough.

Heavy Ball Acceleration: TwIST

- TwIST (*two-step iterative shrinkage-thresholding* (Bioucas-Dias and Figueiredo, 2007)) is a heavy-ball-type acceleration of IST, for

$$\min_x \tfrac{1}{2}\|B\,x - b\|_2^2 + \tau\psi(x)$$

- Iterations (with $\alpha < 2/L$)

$$x_{k+1} = (\gamma - \beta)\,x_k + (1-\gamma)x_{k-1} + \beta\,\mathrm{prox}_{\alpha\tau\psi}\big(x_k - \alpha\,B^T(B\,x - b)\big)$$

Heavy Ball Acceleration: TwIST

- TwIST (*two-step iterative shrinkage-thresholding* (Bioucas-Dias and Figueiredo, 2007)) is a heavy-ball-type acceleration of IST, for

$$\min_x \tfrac{1}{2}\|B\,x - b\|_2^2 + \tau\psi(x)$$

- Iterations (with $\alpha < 2/L$)

$$x_{k+1} = (\gamma - \beta)\,x_k + (1-\gamma)x_{k-1} + \beta\,\mathrm{prox}_{\alpha\tau\psi}\big(x_k - \alpha\,B^T(B\,x - b)\big)$$

- Analysis in the strongly convex case: $\mu I \preceq B^T B \preceq LI$, with $\mu > 0$. Conditioning (as above) $\kappa = L/\mu < \infty$.

Heavy Ball Acceleration: TwIST

- TwIST (*two-step iterative shrinkage-thresholding* (Bioucas-Dias and Figueiredo, 2007)) is a heavy-ball-type acceleration of IST, for

$$\min_x \tfrac{1}{2}\|Bx - b\|_2^2 + \tau\psi(x)$$

- Iterations (with $\alpha < 2/L$)

$$x_{k+1} = (\gamma - \beta)\, x_k + (1-\gamma) x_{k-1} + \beta\, \mathrm{prox}_{\alpha\tau\psi}\big(x_k - \alpha\, B^T(Bx - b)\big)$$

- Analysis in the strongly convex case: $\mu I \preceq B^T B \preceq LI$, with $\mu > 0$. Conditioning (as above) $\kappa = L/\mu < \infty$.
- Optimal parameters: $\gamma = \rho^2 + 1$, $\beta = \frac{2\alpha}{\mu + L}$, where $\rho = \frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}}$, yield linear convergence

$$\|x_{k+1} - \widehat{x}\|_2 \leq \frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}} \|x_k - \widehat{x}\|_2$$

Heavy Ball Acceleration: TwIST

- TwIST (*two-step iterative shrinkage-thresholding* (Bioucas-Dias and Figueiredo, 2007)) is a heavy-ball-type acceleration of IST, for

$$\min_x \tfrac{1}{2}\|Bx - b\|_2^2 + \tau\psi(x)$$

- Iterations (with $\alpha < 2/L$)

$$x_{k+1} = (\gamma - \beta)\, x_k + (1-\gamma) x_{k-1} + \beta\, \mathrm{prox}_{\alpha\tau\psi}\big(x_k - \alpha\, B^T(Bx - b)\big)$$

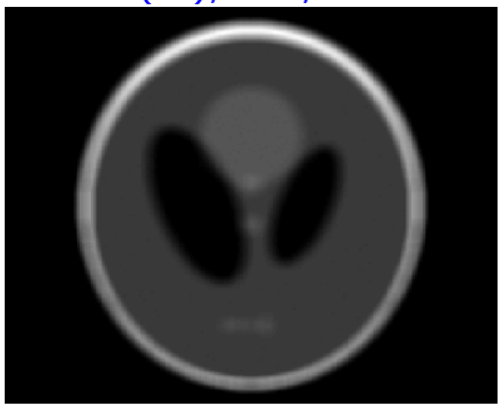
- Analysis in the strongly convex case: $\mu I \preceq B^T B \preceq LI$, with $\mu > 0$. Conditioning (as above) $\kappa = L/\mu < \infty$.
- Optimal parameters: $\gamma = \rho^2 + 1$, $\beta = \frac{2\alpha}{\mu + L}$, where $\rho = \frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}}$, yield linear convergence

$$\|x_{k+1} - \widehat{x}\|_2 \leq \frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}} \|x_k - \widehat{x}\|_2 \quad \left(\text{versus } \tfrac{1-\kappa}{1+\kappa} \text{ for IST}\right)$$

Illustration of the TwIST Acceleration

original

Blurred (B), 9x9, 40db noise

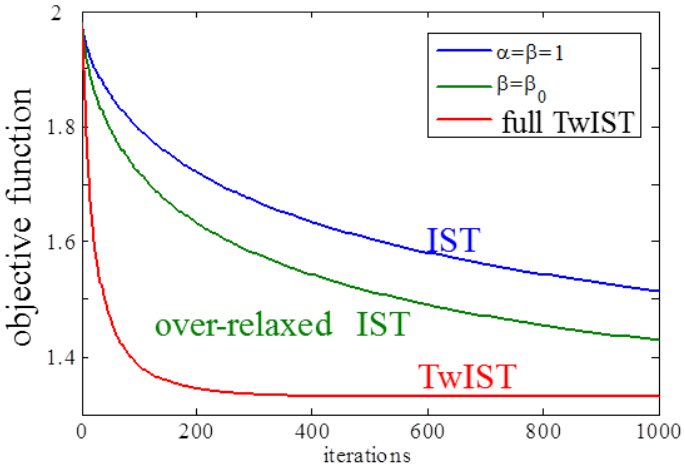


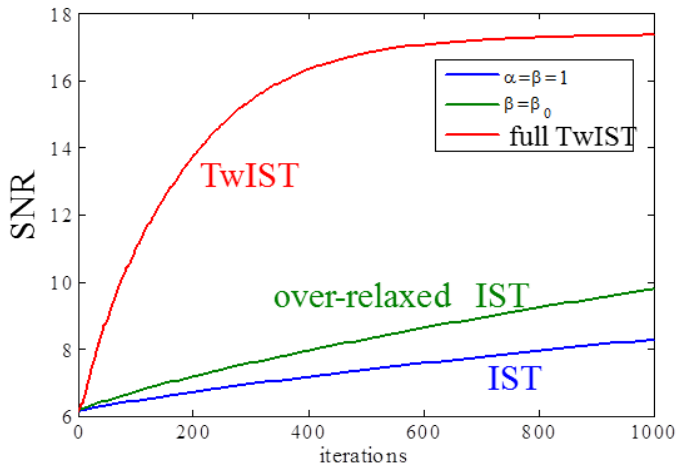
restored

$$\widehat{x} \quad \in \quad \arg\min_{x\in\mathbb{R}^n} \frac{1}{2}\|B\Psi x - u\|_2^2 + \tau\|x\|_1$$

representation coefficients

dictionary (e.g, wavelet basis, frame, ...)





Acceleration via Larger Steps: SpaRSA

- The standard step-size $\alpha_k \leq \frac{2}{L}$ in IST too timid

Acceleration via Larger Steps: SpaRSA

- The standard step-size $\alpha_k \leq \frac{2}{L}$ in IST too timid
- The SpARSA (sparse reconstruction by separable approximation) framework proposes bolder choices of α_k (Wright et al., 2009):
  - ✓ Barzilai-Borwein (see above), to mimic Newton steps — or at least get the scaling right.
  - ✓ keep increasing α_k until monotonicity is violated: backtrack.

Acceleration via Larger Steps: SpaRSA

- The standard step-size $\alpha_k \leq \frac{2}{L}$ in IST too timid
- The SpARSA (sparse reconstruction by separable approximation) framework proposes bolder choices of α_k (Wright et al., 2009):
  - ✓ Barzilai-Borwein (see above), to mimic Newton steps — or at least get the scaling right.
  - ✓ keep increasing α_k until monotonicity is violated: backtrack.
- Convergence to critical points (minima in the convex case) is guaranteed for a safeguarded version: ensure sufficient decrease w.r.t. the worst value in previous M iterations.

Another Approach: Gradient Projection

- $\min_x \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$ can be written as a standard QP:

$$\min_{u,v} \frac{1}{2}\|B(u-v) - b\|_2^2 + \tau u^T 1 + \tau u^T 1 \quad \text{s.t.} \quad u \geq 0,\ v \geq 0,$$

where $u_i = \max\{0, x_i\}$ and $v_i = \max\{0, -x_i\}$.

Another Approach: Gradient Projection

- $\min_x \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$ can be written as a standard QP:

$$\min_{u,v} \frac{1}{2}\|B(u-v) - b\|_2^2 + \tau u^T 1 + \tau u^T 1 \quad \text{s.t.} \quad u \geq 0,\ v \geq 0,$$

  where $u_i = \max\{0, x_i\}$ and $v_i = \max\{0, -x_i\}$.

- With $z = \begin{bmatrix} u \\ v \end{bmatrix}$, problem can be written in canonical form

$$\min_z \frac{1}{2} z^T Q z + c^T z \quad \text{s.t.} \quad z \geq 0$$

Another Approach: Gradient Projection

- $\min_x \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$ can be written as a standard QP:

$$\min_{u,v} \frac{1}{2}\|B(u-v) - b\|_2^2 + \tau u^T 1 + \tau u^T 1 \quad \text{s.t.} \quad u \geq 0,\ v \geq 0,$$

  where $u_i = \max\{0, x_i\}$ and $v_i = \max\{0, -x_i\}$.

- With $z = \begin{bmatrix} u \\ v \end{bmatrix}$, problem can be written in canonical form

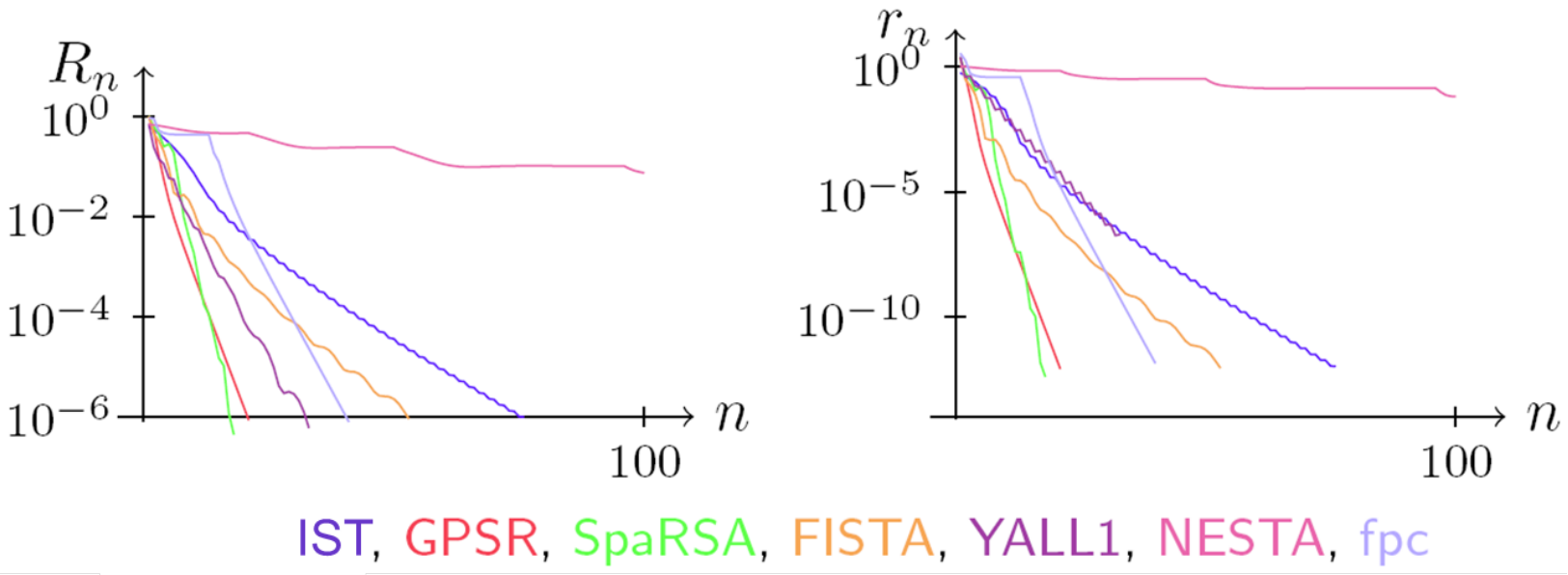
$$\min_z \frac{1}{2} z^T Q z + c^T z \quad \text{s.t.} \quad z \geq 0$$

- Solving this problem with projected gradient using Barzilai-Borwein steps: GPSR (gradient projection for sparse reconstruction).

Speed Comparisons

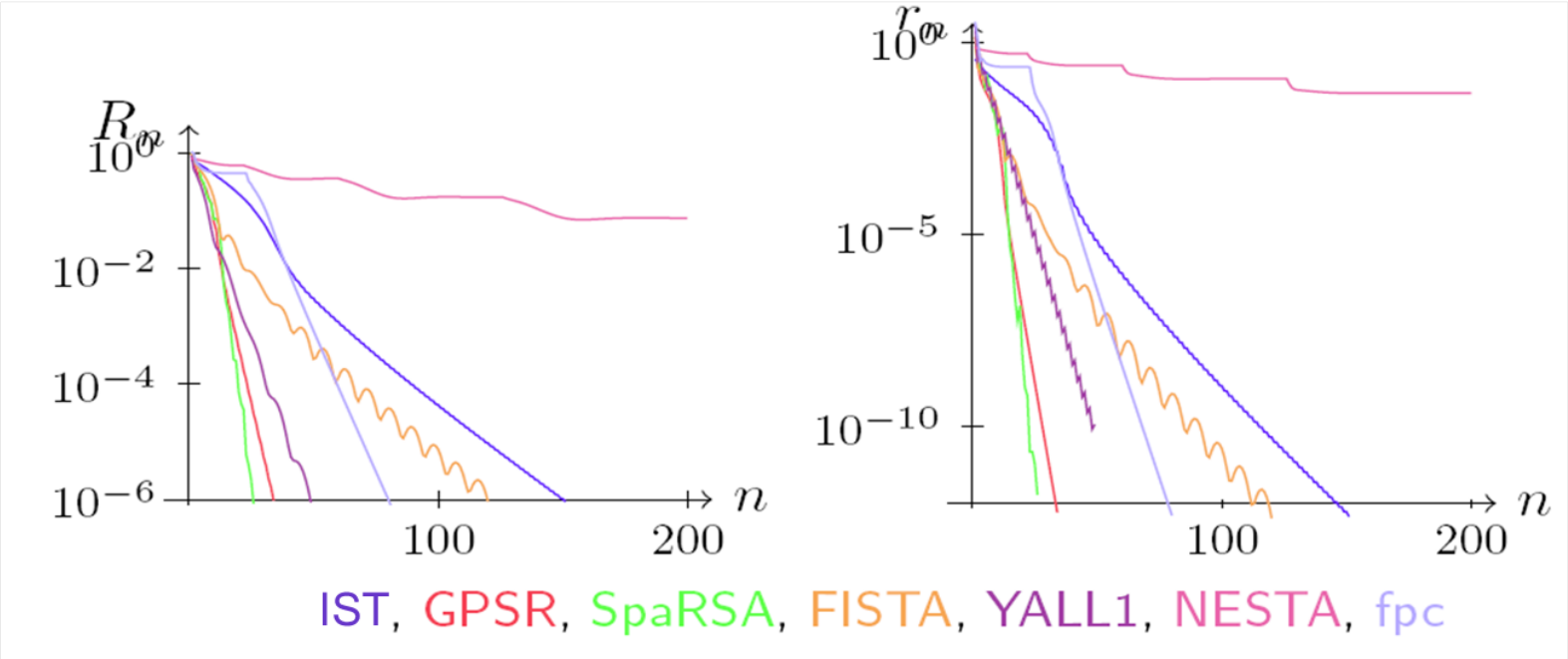
- Lorenz (2011) proposed a way of generating problem instances with known solution $\widehat{x}$: useful for speed comparison.
- Define: $R_k = \frac{\|x_k - \widehat{x}\|_2}{\|\widehat{x}\|_2}$ and $r_k = \frac{L(x_k) - L(\widehat{x})}{L(\widehat{x})}$ (where $L(x) = f(x) + \tau\psi(x)$).

Typical CS example: $\mathbf{A} = [\mathbf{I}\ \mathbf{U}]$ (512 x 1024), $\widehat{\mathbf{x}}$ has 80 non-zeros, $\tau = 0.1$



More Speed Comparisons

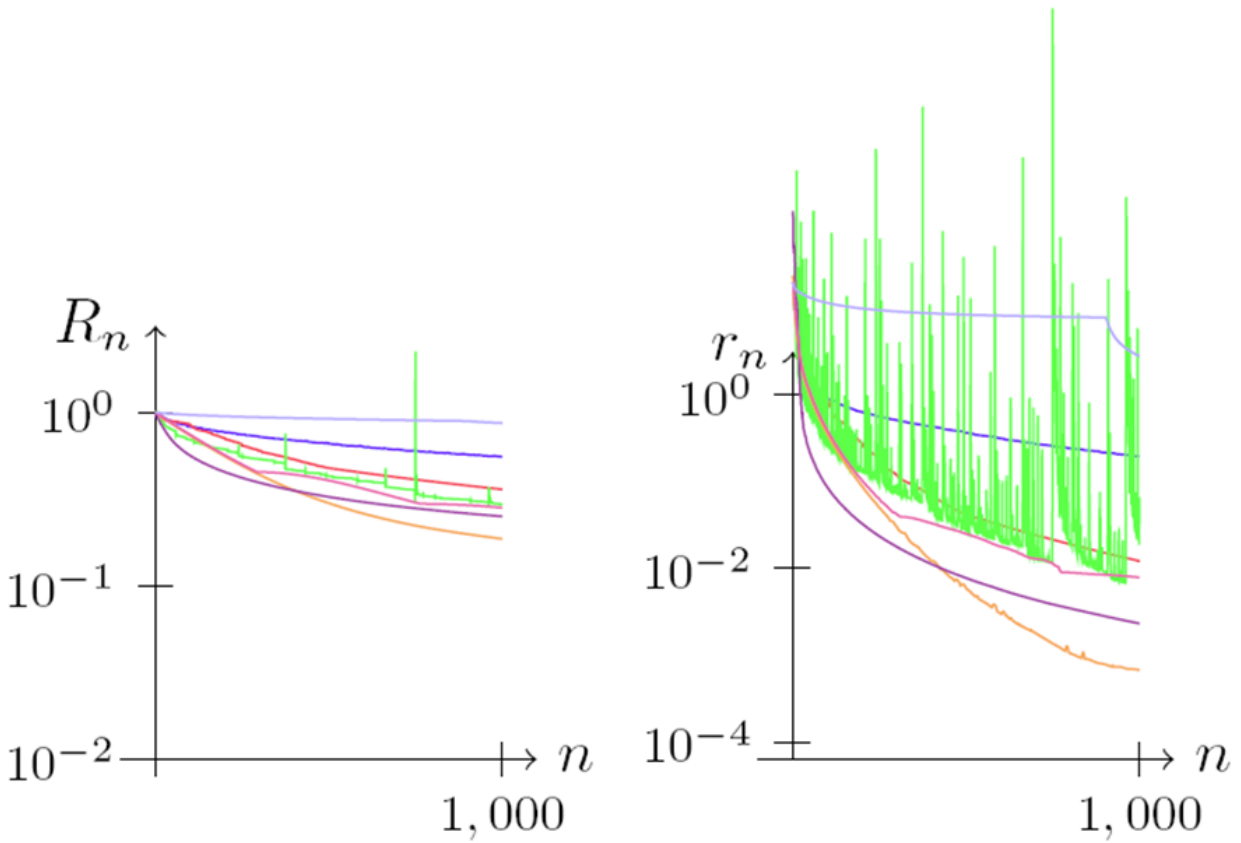
Typical CS example: $\mathbf{A} = [\mathbf{I}\ \mathbf{U}\ \mathbf{R}]$ (512 x 1536), $\widehat{\mathbf{x}}$ has 120 non-zeros, $\tau = 0.1$



Even More Speed Comparisons

A difficult problem: $\mathbf{A}$ is very coherent, τ is small $\tau = 10^{-3}$

All the solvers struggle…



IST, GPSR, SpaRSA, FISTA, YALL1, NESTA, fpc

Acceleration by Continuation

- IST/FBS/PGA can be very slow if τ is very small and/or f is poorly conditioned.

Acceleration by Continuation

- IST/FBS/PGA can be very slow if τ is very small and/or f is poorly conditioned.
- A very simple acceleration strategy: continuation/homotopy

Initialization: Set $\tau_0 \gg \tau$, starting point $\bar{x}$, factor $\sigma \in (0, 1)$, and $k = 0$.

Iterations: Find approx solution $x(\tau_k)$ of $\min_x f(x) + \tau_k \psi(x)$, starting from $\bar{x}$;

if $\tau_k = \tau_f$ STOP;

Set $\tau_{k+1} \leftarrow \max(\tau_f, \sigma\tau_k)$ and $\bar{x} \leftarrow x(\tau_k)$;

Acceleration by Continuation

- IST/FBS/PGA can be very slow if τ is very small and/or f is poorly conditioned.
- A very simple acceleration strategy: continuation/homotopy

Initialization: Set $\tau_0 \gg \tau$, starting point $\bar{x}$, factor $\sigma \in (0, 1)$, and $k = 0$.

Iterations: Find approx solution $x(\tau_k)$ of $\min_x f(x) + \tau_k \psi(x)$, starting from $\bar{x}$;

if $\tau_k = \tau_f$ STOP;

Set $\tau_{k+1} \leftarrow \max(\tau_f, \sigma\tau_k)$ and $\bar{x} \leftarrow x(\tau_k)$;

- Often the solution path $x(\tau)$, for a range of values of τ is desired, anyway (e.g., within an outer method to choose an optimal τ)

Acceleration by Continuation

- IST/FBS/PGA can be very slow if τ is very small and/or f is poorly conditioned.
- A very simple acceleration strategy: continuation/homotopy

Initialization: Set $\tau_0 \gg \tau$, starting point $\bar{x}$, factor $\sigma \in (0, 1)$, and $k = 0$.

Iterations: Find approx solution $x(\tau_k)$ of $\min_x f(x) + \tau_k \psi(x)$, starting from $\bar{x}$;

if $\tau_k = \tau_f$ STOP;

Set $\tau_{k+1} \leftarrow \max(\tau_f, \sigma\tau_k)$ and $\bar{x} \leftarrow x(\tau_k)$;

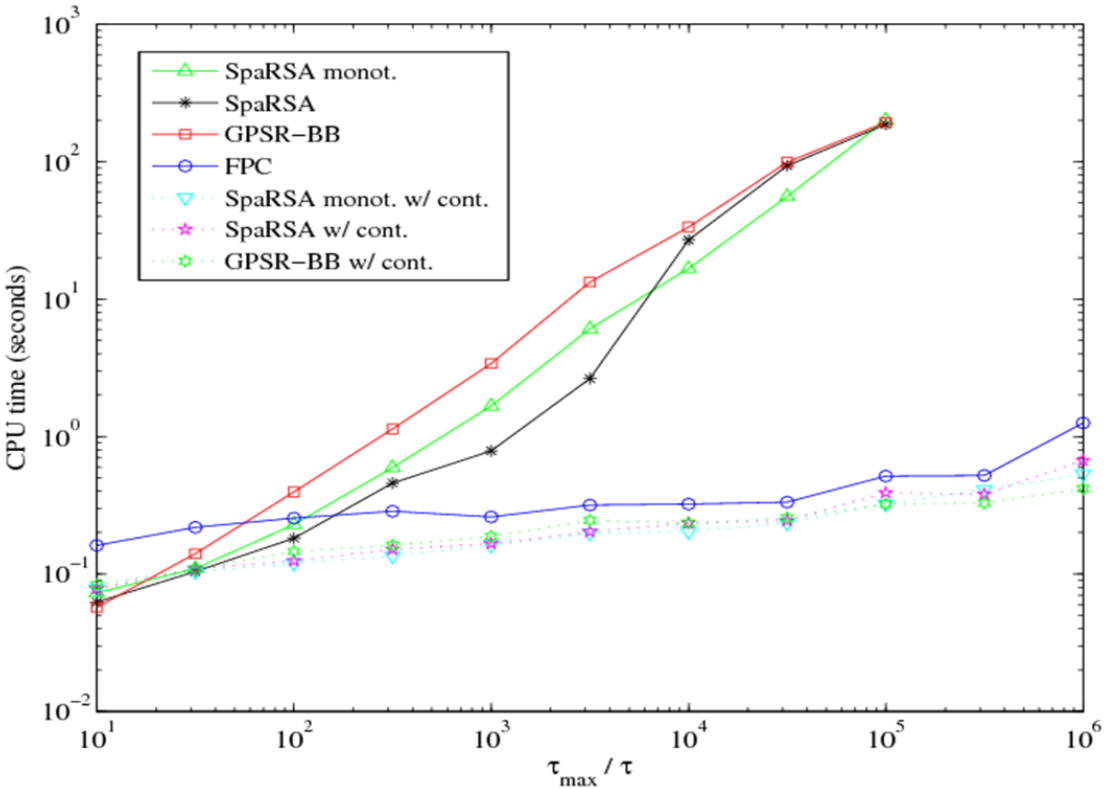
- Often the solution path $x(\tau)$, for a range of values of τ is desired, anyway (e.g., within an outer method to choose an optimal τ)
- Shown to be very effective in practice (Hale et al., 2008; Wright et al., 2009). Analyzed by Xiao and Zhang (2012).

Acceleration by Continuation: An Example

Classical sparse reconstruction problem (Wright et al., 2009)

$$\widehat{x} \in \arg\min_x \tfrac{1}{2}\|B\,x - b\|_2^2 + \tau\|x\|_1$$

with $B \in \mathbb{R}^{1024\times 4096}$ (thus $x \in \mathbb{R}^{4096}$ and $b \in \mathbb{R}^{1024}$).



A Final Touch: Debiasing

Consider problems of the form $\widehat{x} \in \arg\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$

Often, the original goal was to minimize the quadratic term, after the support of x had been found. But the ℓ_1 shrinks the non-zero values.

A Final Touch: Debiasing

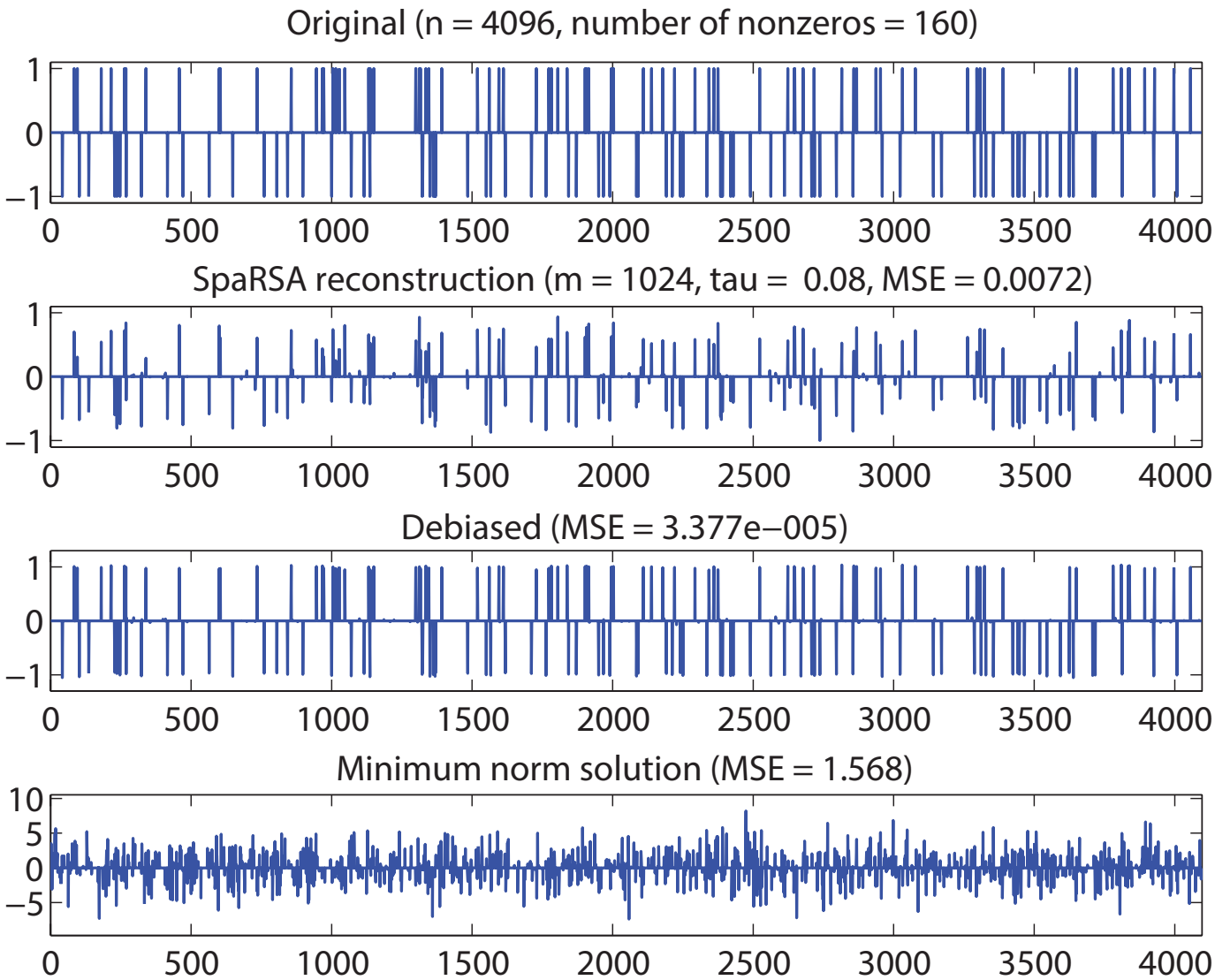
Consider problems of the form $\widehat{x} \in \arg\min_{x \in \mathbb{R}^n} \frac{1}{2}\|B\,x - b\|_2^2 + \tau\|x\|_1$

Often, the original goal was to minimize the quadratic term, after the support of x had been found. But the ℓ_1 shrinks the non-zero values.

Debiasing:

- ✓ find the zero set (complement of the support of $\widehat{x}$): $\mathcal{Z}(\widehat{x}) = \{1, ..., n\} \setminus \text{supp}(\widehat{x})$.
- ✓ solve $\min_x \|B\,x - b\|_2^2$ s.t. $x_{\mathcal{Z}(\widehat{x})} = 0$. (Fix the zeros and solve an unconstrained problem over the support.)
- ✓ Often, this has to be solved using an algorithm that only uses products by B and B^T, since this matrix cannot be partitioned.

Effect of Debiasing



Final Example: Matrix Recovery (Toh and Yun, 2010)

$$\widehat{M} \in \arg \min_{M \in \mathbb{R}^{n \times n}} \frac{1}{2} \|\Phi(M) - U\|_F^2 + \mu \|M\|_*$$

linear operator

...its adjoint

The proximal algorithm (IST) is as before:

$$X_{k+1} = \text{svt}_{\mu\, \beta_k} \left(X_k - \beta_k \, \Phi^*(\Phi(X_k) - U) \right)$$

Matrix completion: $\Phi(X) = X_\Omega$ (subset of entries) $|\Omega| = p$

Unknown M				IST			APG (FISTA)		
n/r	p	p/d_r	μ	iter	#sv	error	iter	#sv	error
100/10	5666	3	8.21e-03	7723	61	1.88e-01	655	13	1.06e-03
200/10	15665	4	1.05e-02	12180	96	2.45e-01	812	12	1.02e-03
500/10	49471	5	1.21e-02	10900	203	5.91e-01	1132	16	7.63e-04

Unknown M				continuation			APG + continuation		
n/r	p	p/d_r	μ	iter	#sv	error	iter	#sv	error
100/10	5666	3	8.21e-03	429	32	1.06e-03	74	10	1.46e-04
200/10	15665	4	1.05e-02	278	49	4.38e-04	73	10	1.02e-04
500/10	49471	5	1.21e-02	484	125	5.50e-04	72	10	8.06e-05

...the importance of acceleration!

References I

Barzilai, J. and Borwein, J. (1988). Two point step size gradient methods. *IMA Journal of Numerical Analysis*, 8:141–148.

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.

Bioucas-Dias, J. and Figueiredo, M. (2007). A new twist: two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image Processing*, 16:2992–3004.

Brucker, P. (1984). An O(n) algorithm for quadratic knapsack problems. *Operations Research Letters*, 3:163–166.

Combettes, P. and Pesquet, J.-C. (2011). Signal recovery by proximal forward-backward splitting. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer.

Combettes, P. and Wajs, V. (2006). Proximal splitting methods in signal processing. *Multiscale Modeling and Simulation*, 4:1168–1200.

Hale, E., Yin, W., and Zhang, Y. (2008). Fixed-point continuation for l1-minimization: Methodology and convergence. *SIAM Journal on Optimization*, 19:1107–1130.

Lorenz, D. (2011). Constructing test instances for basis pursuit denoising. arXiv.org/abs/1103.2897.

Maculan, N. and de Paula, G. G. (1989). A linear-time median-finding algorithm for projecting a vector on the simplex of $\mathbb{R}^n$. *Operations Research Letters*, 8:219–222.

References II

Moreau, J. (1962). Fonctions convexes duales et points proximaux dans un espace hilbertien. *CR Acad. Sci. Paris Sér. A Math*, 255:2897–2899.

Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Math. Doklady*, 27:372–376.

Toh, K.-C. and Yun, S. (2010). An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. *Pacific Journal of Optimization*, 6:615–640.

Wright, S., Nowak, R., and Figueiredo, M. (2009). Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57:2479–2493.

Xiao, L. and Zhang, T. (2012). A proximal-gradient homotopy method for the sparse least-squares problem. *SIAM Journal on Optimization*. (to appear; available at http://arxiv.org/abs/1203.3002).

Part 3: ADMM in Image Processing

Mário A. T. Figueiredo

[1]Instituto de Telecomunicações,
Instituto Superior Técnico,
Universidade de Lisboa, Portugal

CIMI-ANITI School on Optimisation, 2021

Outline

Alternating Direction Methof of Multipliers (ADMM)

- Canonical problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\, \mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{b} \end{aligned}$$

Alternating Direction Methof of Multipliers (ADMM)

- Canonical problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\, \mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{b} \end{aligned}$$

- Functions $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ and $g : \mathbb{R}^m \to \bar{\mathbb{R}}$ are closed, proper, and convex

Alternating Direction Methof of Multipliers (ADMM)

- Canonical problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\, \mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{b} \end{aligned}$$

- Functions $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ and $g : \mathbb{R}^m \to \bar{\mathbb{R}}$ are closed, proper, and convex

- Often used to re-write problems of the form

$$\min_{\mathbf{x}} \; f(\mathbf{x}) + g(\mathbf{H}\mathbf{x})$$

as

$$\min_{\mathbf{x},\mathbf{z}} \; f(\mathbf{x}) + g(\mathbf{z}) \quad \text{subject to} \quad \mathbf{H}\mathbf{x} = \mathbf{z}$$

Alternating Direction Method of Multipliers (ADMM)

- Canonical problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\, \mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{b} \end{aligned}$$

Alternating Direction Method of Multipliers (ADMM)

- Canonical problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\, \mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{b} \end{aligned}$$

- Canonical ADMM (in scaled form)

$$\begin{aligned} \mathbf{x}_{k+1} &= \arg\min_{\mathbf{x}} f(\mathbf{x}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_k - \mathbf{b} + \mathbf{u}_k\right\|_2^2 \\ \mathbf{z}_{k+1} &= \arg\min_{\mathbf{z}} g(\mathbf{z}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z} - \mathbf{b} + \mathbf{u}_k\right\|_2^2 \\ \mathbf{u}_{k+1} &= \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1} - \mathbf{b} \end{aligned}$$

Alternating Direction Method of Multipliers (ADMM)

- Canonical problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\, \mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{b} \end{aligned}$$

- Canonical ADMM (in scaled form)

$$\begin{aligned} \mathbf{x}_{k+1} &= \arg\min_{\mathbf{x}} f(\mathbf{x}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_k - \mathbf{b} + \mathbf{u}_k\right\|_2^2 \\ \mathbf{z}_{k+1} &= \arg\min_{\mathbf{z}} g(\mathbf{z}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z} - \mathbf{b} + \mathbf{u}_k\right\|_2^2 \\ \mathbf{u}_{k+1} &= \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1} - \mathbf{b} \end{aligned}$$

- Can be derived in several ways: method of multipliers (augmented Lagrangian); Douglas-Rachford for the dual; ...

Alternating Direction Method of Multipliers (ADMM)

- Canonical problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\, \mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{b} \end{aligned}$$

- Canonical ADMM (in scaled form)

$$\begin{aligned} \mathbf{x}_{k+1} &= \arg\min_{\mathbf{x}} f(\mathbf{x}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_k - \mathbf{b} + \mathbf{u}_k\right\|_2^2 \\ \mathbf{z}_{k+1} &= \arg\min_{\mathbf{z}} g(\mathbf{z}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z} - \mathbf{b} + \mathbf{u}_k\right\|_2^2 \\ \mathbf{u}_{k+1} &= \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1} - \mathbf{b} \end{aligned}$$

- Can be derived in several ways: method of multipliers (augmented Lagrangian); Douglas-Rachford for the dual; ...
- Introduced by French mathematicians in the 1970s
  [Gabay and Mercier, 1976], [Glowinski and Marrocco, 1975]

Alternating Direction Method of Multipliers (ADMM)

- Canonical problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\, \mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{b} \end{aligned}$$

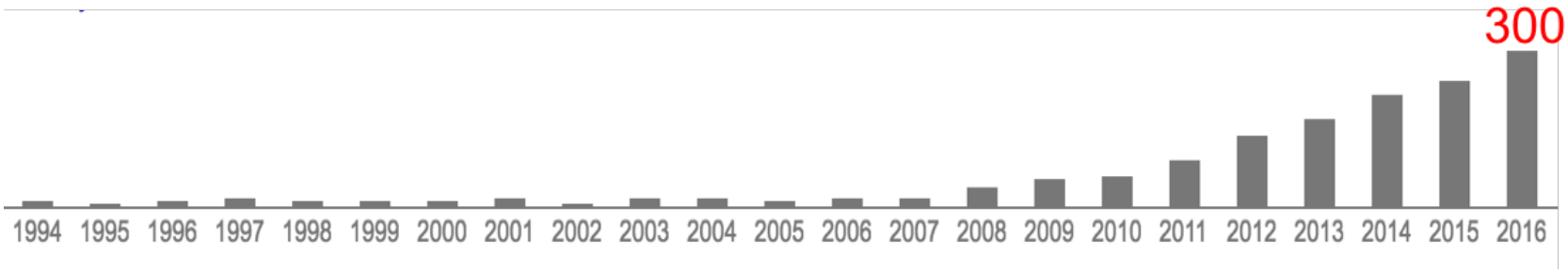
- Canonical ADMM (in scaled form)

$$\begin{aligned} \mathbf{x}_{k+1} &= \arg\min_{\mathbf{x}} f(\mathbf{x}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_k - \mathbf{b} + \mathbf{u}_k\right\|_2^2 \\ \mathbf{z}_{k+1} &= \arg\min_{\mathbf{z}} g(\mathbf{z}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z} - \mathbf{b} + \mathbf{u}_k\right\|_2^2 \\ \mathbf{u}_{k+1} &= \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1} - \mathbf{b} \end{aligned}$$

- Can be derived in several ways: method of multipliers (augmented Lagrangian); Douglas-Rachford for the dual; ...
- Introduced by French mathematicians in the 1970s
  [Gabay and Mercier, 1976], [Glowinski and Marrocco, 1975]
- Cornerstone work in the 1990s by Eckstein and Bertsekas [1992]

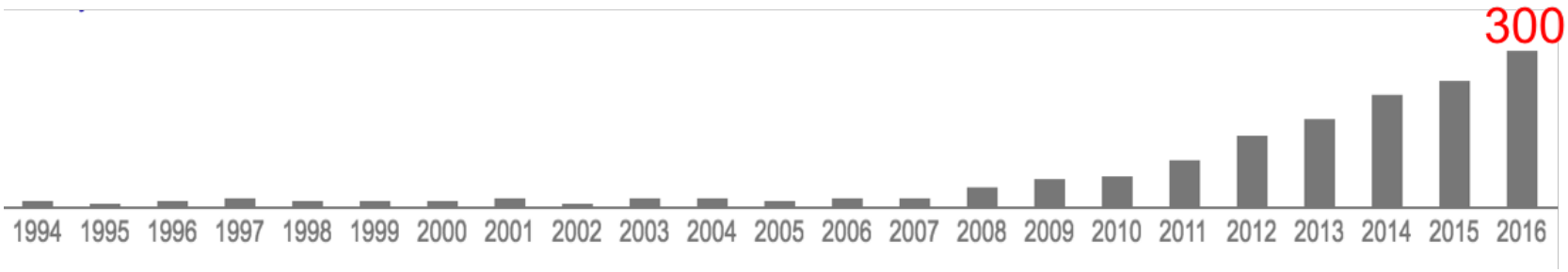
Explosion of Interest in ADMM

- Citations to paper by Eckstein and Bertsekas [1992]:

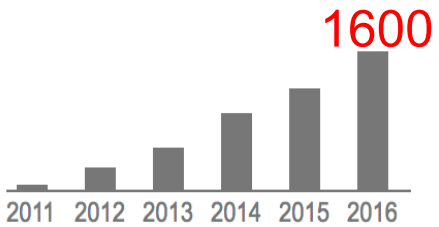


Explosion of Interest in ADMM

- Citations to paper by Eckstein and Bertsekas [1992]:

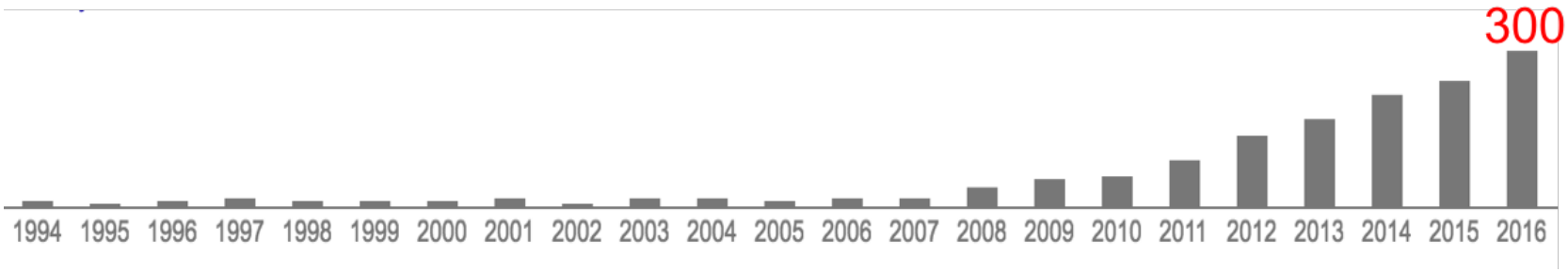


- Citations to review paper by Boyd et al. [2011]:

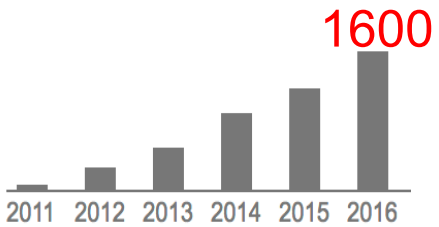


Explosion of Interest in ADMM

- Citations to paper by Eckstein and Bertsekas [1992]:



- Citations to review paper by Boyd et al. [2011]:



- Convergence properties: active research topic
  [Aspelmeier et al., 2016], [Bauschke et al., 2015], [Davis and Yin, 2014], [Deng and Yin, 2012], [Goldstein et al., 2014], [Nishihara et al., 2015], [Liang et al., 2015], [Patrinos et al., 2014], ...

Classical Convergence Result

- Problem: $\min_{\mathbf{x}} \; f(\mathbf{x}) + g(\mathbf{H}\mathbf{x})$
- ADMM:

$$\begin{aligned}
\mathbf{x}^{(k+1)} &= \arg\min_{\mathbf{x}} f(\mathbf{x}) + \frac{\rho}{2}\left\|\mathbf{H}\mathbf{x} - \mathbf{v}^{(k)} - \mathbf{u}^{(k)}\right\|_2^2 \\
\mathbf{v}^{(k+1)} &= \arg\min_{\mathbf{v}} g(\mathbf{v}) + \frac{\rho}{2}\left\|\mathbf{H}\mathbf{x}^{(k+1)} - \mathbf{v} - \mathbf{u}^{(k)}\right\|_2^2, \\
\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} - \mathbf{H}\mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)},
\end{aligned}$$

Theorem (Eckstein and Bertsekas [1992])

Let $\mathbf{H}$ have full column rank, and $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ and $g : \mathbb{R}^m \to \bar{\mathbb{R}}$ be closed, proper, convex functions; let $\mathbf{v}_0, \mathbf{u}_0 \in \mathbb{R}^m$, and $\rho > 0$ be given. Then $(\mathbf{x}^{(k)})_{k=1,2,\ldots}$ converges to a solution, if one exists. If no solution exists, then at least one of the sequences $(\mathbf{v}^{(k)})_{k=1,2,\ldots}$ or $(\mathbf{u}^{(k)})_{k=1,2,\ldots}$ diverges.

Two or More Functions

- Problem template:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$

Two or More Functions

- Problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$
  - ✓ $g_j : \mathbb{R}^{m_j} \to \bar{\mathbb{R}}$ are closed, proper, and convex.

Two or More Functions

- Problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$
  - ✓ $g_j : \mathbb{R}^{m_j} \to \bar{\mathbb{R}}$ are closed, proper, and convex.
  - ✓ $\mathbf{H}_j \in \mathbb{R}^{m_j \times n}$

Two or More Functions

- Problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$
  - ✓ $g_j : \mathbb{R}^{m_j} \to \bar{\mathbb{R}}$ are closed, proper, and convex.
  - ✓ $\mathbf{H}_j \in \mathbb{R}^{m_j \times n}$
- Can be re-written in canonical form

$$\min_{\mathbf{x}} \; f(\mathbf{x}) + g(\mathbf{H}\mathbf{x}),$$

with

Two or More Functions

- Problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$
  - ✓ $g_j : \mathbb{R}^{m_j} \to \bar{\mathbb{R}}$ are closed, proper, and convex.
  - ✓ $\mathbf{H}_j \in \mathbb{R}^{m_j \times n}$
- Can be re-written in canonical form

$$\min_{\mathbf{x}} \; f(\mathbf{x}) + g(\mathbf{H}\mathbf{x}),$$

with $f = 0$,

Two or More Functions

- Problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$
  - ✓ $g_j : \mathbb{R}^{m_j} \to \bar{\mathbb{R}}$ are closed, proper, and convex.
  - ✓ $\mathbf{H}_j \in \mathbb{R}^{m_j \times n}$
- Can be re-written in canonical form

$$\min_{\mathbf{x}} \; f(\mathbf{x}) + g(\mathbf{H}\mathbf{x}),$$

with $$f = 0,\; \mathbf{z} = \begin{bmatrix} \mathbf{z}^{(1)} \\ \vdots \\ \mathbf{z}^{(J)} \end{bmatrix},$$

Two or More Functions

- Problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$
  - ✓ $g_j : \mathbb{R}^{m_j} \to \bar{\mathbb{R}}$ are closed, proper, and convex.
  - ✓ $\mathbf{H}_j \in \mathbb{R}^{m_j \times n}$
- Can be re-written in canonical form

$$\min_{\mathbf{x}} \; f(\mathbf{x}) + g(\mathbf{H}\mathbf{x}),$$

with $$f = 0,\; \mathbf{z} = \begin{bmatrix} \mathbf{z}^{(1)} \\ \vdots \\ \mathbf{z}^{(J)} \end{bmatrix},\; g(\mathbf{z}) = \sum_{j=1}^{J} g_j(\mathbf{z}^{(j)}),$$

Two or More Functions

- Problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$
  - ✓ $g_j : \mathbb{R}^{m_j} \to \bar{\mathbb{R}}$ are closed, proper, and convex.
  - ✓ $\mathbf{H}_j \in \mathbb{R}^{m_j \times n}$
- Can be re-written in canonical form

$$\min_{\mathbf{x}} \; f(\mathbf{x}) + g(\mathbf{H}\mathbf{x}),$$

with $$f = 0,\; \mathbf{z} = \begin{bmatrix} \mathbf{z}^{(1)} \\ \vdots \\ \mathbf{z}^{(J)} \end{bmatrix},\; g(\mathbf{z}) = \sum_{j=1}^{J} g_j(\mathbf{z}^{(j)}),\; \mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \vdots \\ \mathbf{H}_J \end{bmatrix}$$

ADMM for Two or More Functions

- General problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$

ADMM for Two or More Functions

- General problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$

- ADMM after re-writing in canonical form:

$$\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}} \sum_{j=1}^{J} \|\mathbf{H}_j\,\mathbf{x} - \mathbf{z}_k^{(j)} + \mathbf{u}_k^{(j)}\|_2^2$$

ADMM for Two or More Functions

- General problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$

- ADMM after re-writing in canonical form:

$$\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}} \sum_{j=1}^{J} \left\|\mathbf{H}_j\,\mathbf{x} - \mathbf{z}_k^{(j)} + \mathbf{u}_k^{(j)}\right\|_2^2$$

$$\mathbf{z}_{k+1}^{(1)} = \arg\min_{\mathbf{v}\in\mathbb{R}^{m_1}} g_1(\mathbf{v}) + \frac{\rho}{2}\left\|\mathbf{H}_1\,\mathbf{x}_{k+1} - \mathbf{v} + \mathbf{u}_k^{(1)}\right\|_2^2$$

ADMM for Two or More Functions

- General problem template: $\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$

- ADMM after re-writing in canonical form:

$$\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}} \sum_{j=1}^{J} \left\|\mathbf{H}_j\,\mathbf{x} - \mathbf{z}_k^{(j)} + \mathbf{u}_k^{(j)}\right\|_2^2$$

$$\mathbf{z}_{k+1}^{(1)} = \arg\min_{\mathbf{v}\in\mathbb{R}^{m_1}} g_1(\mathbf{v}) + \frac{\rho}{2}\left\|\mathbf{H}_1\,\mathbf{x}_{k+1} - \mathbf{v} + \mathbf{u}_k^{(1)}\right\|_2^2$$

$$\vdots \qquad\qquad \vdots$$

$$\mathbf{z}_{k+1}^{(J)} = \arg\min_{\mathbf{v}\in\mathbb{R}^{m_J}} g_J(\mathbf{v}) + \frac{\rho}{2}\left\|\mathbf{H}_J\,\mathbf{x}_{k+1} - \mathbf{v} + \mathbf{u}_k^{(J)}\right\|_2^2$$

ADMM for Two or More Functions

- General problem template: $$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$$

- ADMM after re-writing in canonical form:

$$\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}} \sum_{j=1}^{J} \left\|\mathbf{H}_j\,\mathbf{x} - \mathbf{z}_k^{(j)} + \mathbf{u}_k^{(j)}\right\|_2^2$$

$$\mathbf{z}_{k+1}^{(1)} = \arg\min_{\mathbf{v}\in\mathbb{R}^{m_1}} g_1(\mathbf{v}) + \frac{\rho}{2}\left\|\mathbf{H}_1\,\mathbf{x}_{k+1} - \mathbf{v} + \mathbf{u}_k^{(1)}\right\|_2^2$$

$$\vdots \qquad \vdots$$

$$\mathbf{z}_{k+1}^{(J)} = \arg\min_{\mathbf{v}\in\mathbb{R}^{m_J}} g_J(\mathbf{v}) + \frac{\rho}{2}\left\|\mathbf{H}_J\,\mathbf{x}_{k+1} - \mathbf{v} + \mathbf{u}_k^{(J)}\right\|_2^2$$

$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1}$$

ADMM for Two or More Functions

- General problem template: $\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$

- ADMM after re-writing in canonical form:

$$\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}} \sum_{j=1}^{J} \left\|\mathbf{H}_j\,\mathbf{x} - \mathbf{z}_k^{(j)} + \mathbf{u}_k^{(j)}\right\|_2^2$$

$$\mathbf{z}_{k+1}^{(1)} = \arg\min_{\mathbf{v}\in\mathbb{R}^{m_1}} g_1(\mathbf{v}) + \frac{\rho}{2}\left\|\mathbf{H}_1\,\mathbf{x}_{k+1} - \mathbf{v} + \mathbf{u}_k^{(1)}\right\|_2^2$$

$$\vdots \qquad \vdots$$

$$\mathbf{z}_{k+1}^{(J)} = \arg\min_{\mathbf{v}\in\mathbb{R}^{m_J}} g_J(\mathbf{v}) + \frac{\rho}{2}\left\|\mathbf{H}_J\,\mathbf{x}_{k+1} - \mathbf{v} + \mathbf{u}_k^{(J)}\right\|_2^2$$

$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1}$$

- SALSA, PIDAL, PIDSplit, SDMM

[F and Bioucas-Dias, 2010], [Setzer et al., 2010], [Combettes and Pesquet, 2011]

A Closer Look

- A closer look at the algorithm

$$\mathbf{x}_{k+1} = \left(\sum_{j=1}^{J} \mathbf{H}_j^T \mathbf{H}_j \right)^{-1} \sum_{j=1}^{J} \mathbf{H}_j \left(\mathbf{z}_k^{(j)} - \mathbf{u}_k^{(j)} \right)$$

A Closer Look

- A closer look at the algorithm

$$\mathbf{x}_{k+1} = \left(\sum_{j=1}^{J} \mathbf{H}_j^T \mathbf{H}_j \right)^{-1} \sum_{j=1}^{J} \mathbf{H}_j \left(\mathbf{z}_k^{(j)} - \mathbf{u}_k^{(j)} \right)$$

$$\mathbf{z}_{k+1}^{(1)} = \text{prox}_{g_1/\rho_k} \left(\mathbf{H}_1 \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(1)} \right)$$

$$\vdots \qquad \vdots$$

$$\mathbf{z}_{k+1}^{(J)} = \text{prox}_{g_J/\rho_k} \left(\mathbf{H}_J \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(J)} \right)$$

A Closer Look

- A closer look at the algorithm

$$\mathbf{x}_{k+1} = \left(\sum_{j=1}^{J} \mathbf{H}_j^T \mathbf{H}_j \right)^{-1} \sum_{j=1}^{J} \mathbf{H}_j \left(\mathbf{z}_k^{(j)} - \mathbf{u}_k^{(j)} \right)$$

$$\mathbf{z}_{k+1}^{(1)} = \text{prox}_{g_1/\rho_k} \left(\mathbf{H}_1 \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(1)} \right)$$

$$\vdots \qquad \vdots$$

$$\mathbf{z}_{k+1}^{(J)} = \text{prox}_{g_J/\rho_k} \left(\mathbf{H}_J \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(J)} \right)$$

$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1}$$

A Closer Look

- A closer look at the algorithm

$$\mathbf{x}_{k+1} = \left(\sum_{j=1}^{J} \mathbf{H}_j^T \mathbf{H}_j \right)^{-1} \sum_{j=1}^{J} \mathbf{H}_j \left(\mathbf{z}_k^{(j)} - \mathbf{u}_k^{(j)} \right)$$

$$\mathbf{z}_{k+1}^{(1)} = \text{prox}_{g_1/\rho_k} \left(\mathbf{H}_1 \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(1)} \right)$$

$$\vdots \qquad \vdots$$

$$\mathbf{z}_{k+1}^{(J)} = \text{prox}_{g_J/\rho_k} \left(\mathbf{H}_J \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(J)} \right)$$

$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1}$$

- Decoupled: a linear algebraic problem and a set of proximity operators

A Closer Look

- A closer look at the algorithm

$$\mathbf{x}_{k+1} = \left(\sum_{j=1}^{J} \mathbf{H}_j^T \mathbf{H}_j \right)^{-1} \sum_{j=1}^{J} \mathbf{H}_j \left(\mathbf{z}_k^{(j)} - \mathbf{u}_k^{(j)} \right)$$

$$\mathbf{z}_{k+1}^{(1)} = \text{prox}_{g_1/\rho_k} \left(\mathbf{H}_1 \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(1)} \right)$$

$$\vdots \qquad \vdots$$

$$\mathbf{z}_{k+1}^{(J)} = \text{prox}_{g_J/\rho_k} \left(\mathbf{H}_J \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(J)} \right)$$

$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1}$$

- Decoupled: a linear algebraic problem and a set of proximity operators
- Hinges on the efficient matrix inversion and simple proximity operators

A Closer Look

- A closer look at the algorithm

$$\mathbf{x}_{k+1} = \left(\sum_{j=1}^{J} \mathbf{H}_j^T \mathbf{H}_j \right)^{-1} \sum_{j=1}^{J} \mathbf{H}_j \left(\mathbf{z}_k^{(j)} - \mathbf{u}_k^{(j)} \right)$$

$$\mathbf{z}_{k+1}^{(1)} = \text{prox}_{g_1/\rho_k} \left(\mathbf{H}_1 \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(1)} \right)$$

$$\vdots \qquad \vdots$$

$$\mathbf{z}_{k+1}^{(J)} = \text{prox}_{g_J/\rho_k} \left(\mathbf{H}_J \, \mathbf{x}_{k+1} + \mathbf{u}_k^{(J)} \right)$$

$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{z}_{k+1}$$

- Decoupled: a linear algebraic problem and a set of proximity operators
- Hinges on the efficient matrix inversion and simple proximity operators
- Matrix inverse independent of ρ_k (good, if not kept constant)

Image Restoration/Reconstruction

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

  where $\mathbf{y}$ are observations and $\iota_C(\mathbf{x}) = \begin{cases} 0 & \Leftarrow \quad \mathbf{x} \in C \\ +\infty & \Leftarrow \quad \mathbf{x} \notin C \end{cases}$

Image Restoration/Reconstruction

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

  where $\mathbf{y}$ are observations and $\iota_C(\mathbf{x}) = \begin{cases} 0 & \Leftarrow \quad \mathbf{x} \in C \\ +\infty & \Leftarrow \quad \mathbf{x} \notin C \end{cases}$

- Ψ: the observation model (negative log-likelihood); namely,

Image Restoration/Reconstruction

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

  where $\mathbf{y}$ are observations and $\iota_C(\mathbf{x}) = \begin{cases} 0 & \Leftarrow \quad \mathbf{x} \in C \\ +\infty & \Leftarrow \quad \mathbf{x} \notin C \end{cases}$

- Ψ: the observation model (negative log-likelihood); namely,
  - ✓ Gaussian observations: $\Psi(\mathbf{u}, \mathbf{y}) = \frac{1}{2\sigma^2}\|\mathbf{u} - \mathbf{y}\|_2^2 = \frac{1}{2\sigma^2}\sum_i (u_i - y_i)^2$

Image Restoration/Reconstruction

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$

  where $\mathbf{y}$ are observations and $\iota_C(\mathbf{x}) = \begin{cases} 0 & \Leftarrow \quad \mathbf{x} \in C \\ +\infty & \Leftarrow \quad \mathbf{x} \notin C \end{cases}$

- Ψ: the observation model (negative log-likelihood); namely,
  - ✓ Gaussian observations: $\Psi(\mathbf{u}, \mathbf{y}) = \frac{1}{2\sigma^2}\|\mathbf{u} - \mathbf{y}\|_2^2 = \frac{1}{2\sigma^2}\sum_i (u_i - y_i)^2$
  - ✓ Poisson observations: $\Psi(\mathbf{u}, \mathbf{y}) = \sum_i \big(u_i + \iota_{\mathbb{R}_+}(u_i) - y_i \log(z_i)_+\big)$

Image Restoration/Reconstruction

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$

  where $\mathbf{y}$ are observations and $\iota_C(\mathbf{x}) = \begin{cases} 0 & \Leftarrow \quad \mathbf{x} \in C \\ +\infty & \Leftarrow \quad \mathbf{x} \notin C \end{cases}$

- Ψ: the observation model (negative log-likelihood); namely,
  - ✓ Gaussian observations: $\Psi(\mathbf{u}, \mathbf{y}) = \frac{1}{2\sigma^2}\|\mathbf{u} - \mathbf{y}\|_2^2 = \frac{1}{2\sigma^2}\sum_i (u_i - y_i)^2$
  - ✓ Poisson observations: $\Psi(\mathbf{u}, \mathbf{y}) = \sum_i \big(u_i + \iota_{\mathbb{R}_+}(u_i) - y_i \log(z_i)_+\big)$
  - ✓ Multiplicative noise: $\Psi(\mathbf{u}, \mathbf{y}) = M \sum_i \big(z_i + e^{y_i - z_i}\big)$

Image Restoration/Reconstruction

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$

  where $\mathbf{y}$ are observations and $\iota_C(\mathbf{x}) = \begin{cases} 0 & \Leftarrow \quad \mathbf{x} \in C \\ +\infty & \Leftarrow \quad \mathbf{x} \notin C \end{cases}$

- Ψ: the observation model (negative log-likelihood); namely,
  - ✓ Gaussian observations: $\Psi(\mathbf{u}, \mathbf{y}) = \frac{1}{2\sigma^2}\|\mathbf{u} - \mathbf{y}\|_2^2 = \frac{1}{2\sigma^2}\sum_i (u_i - y_i)^2$
  - ✓ Poisson observations: $\Psi(\mathbf{u}, \mathbf{y}) = \sum_i \big(u_i + \iota_{\mathbb{R}_+}(u_i) - y_i \log(z_i)_+\big)$
  - ✓ Multiplicative noise: $\Psi(\mathbf{u}, \mathbf{y}) = M \sum_i \big(z_i + e^{y_i - z_i}\big)$

- $\Phi \circ \mathbf{P}$ is a regularizer; *e.g.*, total variation (TV), or Φ is a norm

Image Restoration/Reconstruction

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$

  where $\mathbf{y}$ are observations and $\iota_C(\mathbf{x}) = \begin{cases} 0 & \Leftarrow \quad \mathbf{x} \in C \\ +\infty & \Leftarrow \quad \mathbf{x} \notin C \end{cases}$

- Ψ: the observation model (negative log-likelihood); namely,
  - ✓ Gaussian observations: $\Psi(\mathbf{u}, \mathbf{y}) = \frac{1}{2\sigma^2}\|\mathbf{u} - \mathbf{y}\|_2^2 = \frac{1}{2\sigma^2}\sum_i (u_i - y_i)^2$
  - ✓ Poisson observations: $\Psi(\mathbf{u}, \mathbf{y}) = \sum_i \big(u_i + \iota_{\mathbb{R}_+}(u_i) - y_i \log(z_i)_+\big)$
  - ✓ Multiplicative noise: $\Psi(\mathbf{u}, \mathbf{y}) = M \sum_i \big(z_i + e^{y_i - z_i}\big)$
- $\Phi \circ \mathbf{P}$ is a regularizer; *e.g.*, total variation (TV), or Φ is a norm
- $\mathbf{A}$: linear observation operator (blur, projections, ...)

Image Restoration: Observation Models Ψ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

Image Restoration: Observation Models Ψ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- All the above observation models have simple, component-wise proximity operators

Image Restoration: Observation Models Ψ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- All the above observation models have simple, component-wise proximity operators

  ✓ Gaussian observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{\sigma^2 u + \tau y}{\sigma^2 + \tau}$

Image Restoration: Observation Models Ψ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- All the above observation models have simple, component-wise proximity operators

  ✓ Gaussian observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{\sigma^2 u + \tau y}{\sigma^2 + \tau}$

  ✓ Poisson observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{1}{2}\left(y - \tau + \sqrt{(y-\tau)^2 + 4y\tau}\right)$

Image Restoration: Observation Models Ψ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- All the above observation models have simple, component-wise proximity operators

  ✓ Gaussian observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{\sigma^2 u + \tau y}{\sigma^2 + \tau}$

  ✓ Poisson observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{1}{2}\left(y - \tau + \sqrt{(y-\tau)^2 + 4y\tau}\right)$

  ✓ Multiplicative noise: $\text{prox}_{\tau\Psi}(u)$ uses Lambert's W-function

Image Restoration: Observation Models Ψ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- All the above observation models have simple, component-wise proximity operators

  - ✓ Gaussian observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{\sigma^2 u + \tau y}{\sigma^2 + \tau}$

  - ✓ Poisson observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{1}{2}\left(y - \tau + \sqrt{(y-\tau)^2 + 4y\tau}\right)$

  - ✓ Multiplicative noise: $\text{prox}_{\tau\Psi}(u)$ uses Lambert's W-function

- The proximity operator of ι_C is simply an Euclidean projection:

$$\text{prox}_{\iota_C}(\mathbf{u}) = \text{proj}_C(\mathbf{u});$$

Image Restoration: Observation Models Ψ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- All the above observation models have simple, component-wise proximity operators

  - ✓ Gaussian observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{\sigma^2 u + \tau y}{\sigma^2 + \tau}$

  - ✓ Poisson observations: $\text{prox}_{\tau\Psi}(u) = \dfrac{1}{2}\left(y - \tau + \sqrt{(y-\tau)^2 + 4y\tau}\right)$

  - ✓ Multiplicative noise: $\text{prox}_{\tau\Psi}(u)$ uses Lambert's W-function

- The proximity operator of ι_C is simply an Euclidean projection:

$$\text{prox}_{\iota_C}(\mathbf{u}) = \text{proj}_C(\mathbf{u});$$

e.g., if $C = \mathbb{R}^n_+$, then $\big(\text{proj}_C(\mathbf{u})\big)_i = \max\{0, u_i\}$

Image Restoration: Regularizers Φ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

Image Restoration: Regularizers Φ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$
- Classical regularizers with simple proximity operators

Image Restoration: Regularizers Φ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- Classical regularizers with simple proximity operators

  ✓ ℓ_1 norm: $\left(\text{prox}_{\tau\|\cdot\|_1}(\mathbf{u})\right)_i = \text{sign}(u_i)\max\{0, u_i - \tau\}$

Image Restoration: Regularizers Φ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- Classical regularizers with simple proximity operators

  ✓ ℓ_1 norm: $\left(\text{prox}_{\tau\|\cdot\|_1}(\mathbf{u})\right)_i = \text{sign}(u_i)\max\{0, u_i - \tau\} = \text{soft}(u_i, \tau)$

Image Restoration: Regularizers Φ

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$$

- Classical regularizers with simple proximity operators

  ✓ ℓ_1 norm: $\left(\text{prox}_{\tau\|\cdot\|_1}(\mathbf{u})\right)_i = \text{sign}(u_i)\max\{0, u_i - \tau\} = \text{soft}(u_i, \tau)$

  ✓ Squared ℓ_2 norm: $\text{prox}_{\tau\|\cdot\|_2^2}(\mathbf{u}) = \dfrac{\mathbf{u}}{1+\tau}$

Image Restoration: Regularizers Φ

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$
- Classical regularizers with simple proximity operators
  - ✓ ℓ_1 norm: $\left(\text{prox}_{\tau\|\cdot\|_1}(\mathbf{u})\right)_i = \text{sign}(u_i)\max\{0, u_i - \tau\} = \text{soft}(u_i, \tau)$
  - ✓ Squared ℓ_2 norm: $\text{prox}_{\tau\|\cdot\|_2^2}(\mathbf{u}) = \dfrac{\mathbf{u}}{1+\tau}$ (linear shrinkage)

Image Restoration: Regularizers Φ

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$
- Classical regularizers with simple proximity operators
  - ✓ ℓ_1 norm: $\left(\text{prox}_{\tau\|\cdot\|_1}(\mathbf{u})\right)_i = \text{sign}(u_i)\max\{0, u_i - \tau\} = \text{soft}(u_i, \tau)$
  - ✓ Squared ℓ_2 norm: $\text{prox}_{\tau\|\cdot\|_2^2}(\mathbf{u}) = \dfrac{\mathbf{u}}{1+\tau}$ (linear shrinkage)
  - ✓ ℓ_2 norm: $\text{prox}_{\tau\|\cdot\|_2}(\mathbf{u}) = \dfrac{\mathbf{u}\ \max\{0, \|\mathbf{u}\|_2 - \tau\}}{\max\{0, \|\mathbf{u}\|_2 - \tau\} + \tau}$

Image Restoration: Regularizers Φ

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x}) + \iota_C(\mathbf{x})$
- Classical regularizers with simple proximity operators
  - ✓ ℓ_1 norm: $\left(\text{prox}_{\tau\|\cdot\|_1}(\mathbf{u})\right)_i = \text{sign}(u_i)\max\{0, u_i - \tau\} = \text{soft}(u_i, \tau)$
  - ✓ Squared ℓ_2 norm: $\text{prox}_{\tau\|\cdot\|_2^2}(\mathbf{u}) = \dfrac{\mathbf{u}}{1+\tau}$ (linear shrinkage)
  - ✓ ℓ_2 norm: $\text{prox}_{\tau\|\cdot\|_2}(\mathbf{u}) = \dfrac{\mathbf{u}\ \max\{0, \|\mathbf{u}\|_2 - \tau\}}{\max\{0, \|\mathbf{u}\|_2 - \tau\} + \tau} = \text{vect-soft}(\mathbf{u}, \tau)$
- Total variation can be written as $\Phi \circ \mathbf{P}$, where

$$\mathbf{P} : \mathbb{R}^n \to (\mathbb{R}^2)^n, \text{ with } (\mathbf{P}\mathbf{x})_i = \begin{bmatrix} x_i - x_{h(i)} \\ x_i - x_{v(i)} \end{bmatrix}, \text{ and } \Phi(\mathbf{v}) = \sum_i \|\mathbf{v}_i\|_2$$

with $h(i)$ and $v(i)$ the horizontal and vertical neighbours of pixel i

Image Restoration: Synthesis Formulation

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$$

Image Restoration: Synthesis Formulation

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$$
- Synthesis formulation: $\mathbf{A} = \mathbf{B}\mathbf{W}$ and $\mathbf{P} = \mathbf{I}$

Image Restoration: Synthesis Formulation

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$$
- Synthesis formulation: $\mathbf{A} = \mathbf{B}\mathbf{W}$ and $\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{W}$ the synthesis operator of a Parseval frame: $\mathbf{W}\,\mathbf{W}^T = \mathbf{I}$

Image Restoration: Synthesis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Synthesis formulation: $\mathbf{A} = \mathbf{B}\mathbf{W}$ and $\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{W}$ the synthesis operator of a Parseval frame: $\mathbf{W}\,\mathbf{W}^T = \mathbf{I}$
  - ✓ $\mathbf{B}$ is the observation operator

Image Restoration: Synthesis Formulation

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$$
- Synthesis formulation: $\mathbf{A} = \mathbf{B}\mathbf{W}$ and $\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{W}$ the synthesis operator of a Parseval frame: $\mathbf{W}\,\mathbf{W}^T = \mathbf{I}$
  - ✓ $\mathbf{B}$ is the observation operator
  - ✓ $\mathbf{x}$ contains the representation coefficients, not the image itself

Image Restoration: Synthesis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Synthesis formulation: $\mathbf{A} = \mathbf{B}\mathbf{W}$ and $\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{W}$ the synthesis operator of a Parseval frame: $\mathbf{W}\,\mathbf{W}^T = \mathbf{I}$
  - ✓ $\mathbf{B}$ is the observation operator
  - ✓ $\mathbf{x}$ contains the representation coefficients, not the image itself
- Using the Sherman-Morrison-Woodbury matrix inversion formula

$$\left(\mathbf{W}^T\mathbf{B}^T\mathbf{B}\mathbf{W} + \mathbf{I}\right)^{-1} = \mathbf{I} - \mathbf{W}^T\mathbf{B}^T\left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}\mathbf{B}\mathbf{W}$$

Image Restoration: Synthesis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Synthesis formulation: $\mathbf{A} = \mathbf{B}\mathbf{W}$ and $\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{W}$ the synthesis operator of a Parseval frame: $\mathbf{W}\,\mathbf{W}^T = \mathbf{I}$
  - ✓ $\mathbf{B}$ is the observation operator
  - ✓ $\mathbf{x}$ contains the representation coefficients, not the image itself
- Using the Sherman-Morrison-Woodbury matrix inversion formula

$$\left(\mathbf{W}^T\mathbf{B}^T\mathbf{B}\mathbf{W} + \mathbf{I}\right)^{-1} = \mathbf{I} - \mathbf{W}^T\mathbf{B}^T\left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}\mathbf{B}\mathbf{W}$$

- Can $\mathbf{B}^T\mathbf{B} + \mathbf{I}$ be inverted efficiently?

Image Restoration: Analysis Formulation

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$$

Image Restoration: Analysis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Analysis formulation: $\mathbf{A} = \mathbf{B}$

Image Restoration: Analysis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Analysis formulation: $\mathbf{A} = \mathbf{B}$
  - ✓ $\mathbf{P}$ the analysis operator of a Parseval frame: $\mathbf{P}^T\mathbf{P} = \mathbf{I}$

Image Restoration: Analysis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Analysis formulation: $\mathbf{A} = \mathbf{B}$
  - ✓ $\mathbf{P}$ the analysis operator of a Parseval frame: $\mathbf{P}^T\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{B}$ is the observation operator

Image Restoration: Analysis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Analysis formulation: $\mathbf{A} = \mathbf{B}$
  - ✓ $\mathbf{P}$ the analysis operator of a Parseval frame: $\mathbf{P}^T\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{B}$ is the observation operator
  - ✓ $\mathbf{x}$ contains the image itself

Image Restoration: Analysis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Analysis formulation: $\mathbf{A} = \mathbf{B}$
  - ✓ $\mathbf{P}$ the analysis operator of a Parseval frame: $\mathbf{P}^T\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{B}$ is the observation operator
  - ✓ $\mathbf{x}$ contains the image itself
- Matrix inversion:

$$\left(\mathbf{B}^T\mathbf{B} + \mathbf{P}^T\mathbf{P}\right)^{-1} = \left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}$$

Image Restoration: Analysis Formulation

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Analysis formulation: $\mathbf{A} = \mathbf{B}$
  - ✓ $\mathbf{P}$ the analysis operator of a Parseval frame: $\mathbf{P}^T\mathbf{P} = \mathbf{I}$
  - ✓ $\mathbf{B}$ is the observation operator
  - ✓ $\mathbf{x}$ contains the image itself
- Matrix inversion:

$$\left(\mathbf{B}^T\mathbf{B} + \mathbf{P}^T\mathbf{P}\right)^{-1} = \left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}$$

- Can $\mathbf{B}^T\mathbf{B} + \mathbf{I}$ be inverted efficiently?

Image Restoration: Constrained (Morozov) Formulations

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$$

Image Restoration: Constrained (Morozov) Formulations

- General formulation: $$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$$

- Constrained (or Morozov) formulation:

$$\hat{\mathbf{x}} \in \min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{P}\mathbf{x}) \text{ subject to } \Lambda(\mathbf{A}\mathbf{x}, \mathbf{y}) \leq 1$$

Image Restoration: Constrained (Morozov) Formulations

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Constrained (or Morozov) formulation:

$$\hat{\mathbf{x}} \in \min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{P}\mathbf{x}) \text{ subject to } \Lambda(\mathbf{A}\mathbf{x}, \mathbf{y}) \leq 1$$

- Can be written in the general formulation, with

$$\Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) = \iota_{D(\mathbf{y})}(\mathbf{A}\mathbf{x}), \text{ with } D(\mathbf{y}) = \{\mathbf{x} : \Lambda(\mathbf{x}, \mathbf{y}) \leq 1\}$$

Image Restoration: Constrained (Morozov) Formulations

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Constrained (or Morozov) formulation:

$$\hat{\mathbf{x}} \in \min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{P}\mathbf{x}) \;\text{ subject to }\; \Lambda(\mathbf{A}\mathbf{x}, \mathbf{y}) \leq 1$$

- Can be written in the general formulation, with

$$\Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) = \iota_{D(\mathbf{y})}(\mathbf{A}\mathbf{x}), \;\text{ with }\; D(\mathbf{y}) = \{\mathbf{x} : \Lambda(\mathbf{x}, \mathbf{y}) \leq 1\}$$

- Classical example: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{P}\mathbf{x})$ s.t. $\xi\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2 \leq 1$

  Thus, $D(\mathbf{y})$ is a unit Euclidean ball around $\mathbf{y}$; projection is trivial.

Image Restoration: Constrained (Morozov) Formulations

- General formulation: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) + \Phi(\mathbf{P}\mathbf{x})$
- Constrained (or Morozov) formulation:

$$\hat{\mathbf{x}} \in \min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{P}\mathbf{x}) \text{ subject to } \Lambda(\mathbf{A}\mathbf{x}, \mathbf{y}) \leq 1$$

- Can be written in the general formulation, with

$$\Psi(\mathbf{A}\mathbf{x}, \mathbf{y}) = \iota_{D(\mathbf{y})}(\mathbf{A}\mathbf{x}), \text{ with } D(\mathbf{y}) = \{\mathbf{x} : \Lambda(\mathbf{x}, \mathbf{y}) \leq 1\}$$

- Classical example: $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{P}\mathbf{x})$ s.t. $\xi\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2 \leq 1$

  Thus, $D(\mathbf{y})$ is a unit Euclidean ball around $\mathbf{y}$; projection is trivial.

- Applies both to synthesis and analysis formulations

Image Restoration: Matrix Inversions

- The required inversion $\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}$ is simple in many relevant cases: [Afonso et al., 2011], [F and Bioucas-Dias, 2010]

Image Restoration: Matrix Inversions

- The required inversion $\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}$ is simple in many relevant cases: [Afonso et al., 2011], [F and Bioucas-Dias, 2010]

  ✓ Periodic deconvolution: $\mathbf{B}=\mathbf{U}^H\mathbf{F}\mathbf{U}$,

  $\mathbf{F}$ is diagonal; $\mathbf{U}$ is the DFT matrix ($\mathbf{U}^H\mathbf{U}=\mathbf{U}\mathbf{U}^H=\mathbf{I}$)

$$\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}=\mathbf{U}^H\underbrace{\left(|\mathbf{F}|^2+\mathbf{I}\right)^{-1}}_{\text{diagonal}}\mathbf{U}$$

Image Restoration: Matrix Inversions

- The required inversion $\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}$ is simple in many relevant cases: [Afonso et al., 2011], [F and Bioucas-Dias, 2010]

  ✓ Periodic deconvolution: $\mathbf{B}=\mathbf{U}^H\mathbf{F}\mathbf{U}$,

  $\mathbf{F}$ is diagonal; $\mathbf{U}$ is the DFT matrix ($\mathbf{U}^H\mathbf{U}=\mathbf{U}\mathbf{U}^H=\mathbf{I}$)

$$\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}=\mathbf{U}^H\underbrace{\left(|\mathbf{F}|^2+\mathbf{I}\right)^{-1}}_{\text{diagonal}}\mathbf{U}$$

  ✓ Inpainting: $\mathbf{B}\in\{0,1\}^{m\times n}$, with m rows of $\mathbf{I}$; thus, $\mathbf{B}^T\mathbf{B}$ is diagonal

Image Restoration: Matrix Inversions

- The required inversion $\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}$ is simple in many relevant cases: [Afonso et al., 2011], [F and Bioucas-Dias, 2010]

  ✓ Periodic deconvolution: $\mathbf{B}=\mathbf{U}^H\mathbf{F}\mathbf{U}$,

  $\mathbf{F}$ is diagonal; $\mathbf{U}$ is the DFT matrix ($\mathbf{U}^H\mathbf{U}=\mathbf{U}\mathbf{U}^H=\mathbf{I}$)

$$\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}=\mathbf{U}^H\underbrace{\left(|\mathbf{F}|^2+\mathbf{I}\right)^{-1}}_{\text{diagonal}}\mathbf{U}$$

  ✓ Inpainting: $\mathbf{B}\in\{0,1\}^{m\times n}$, with m rows of $\mathbf{I}$; thus, $\mathbf{B}^T\mathbf{B}$ is diagonal

  ✓ Compressive Fourier imaging (MRI, multi-coil MRI): $\mathbf{B}=\mathbf{M}\mathbf{U}$, where $\mathbf{M}\in\{0,1\}^{m\times n}$, with m rows of $\mathbf{I}$; thus, $\mathbf{M}\mathbf{M}^T=\mathbf{I}$

$$\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}=\mathbf{I}-\frac{1}{2}\,\mathbf{U}^H\underbrace{\mathbf{M}^T\mathbf{M}}_{\text{diagonal}}\,\mathbf{U}$$

Image Restoration: Matrix Inversions

- The required inversion $\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}$ is simple in many relevant cases: [Afonso et al., 2011], [F and Bioucas-Dias, 2010]

  ✓ Periodic deconvolution: $\mathbf{B}=\mathbf{U}^H\mathbf{F}\mathbf{U}$,

  $\mathbf{F}$ is diagonal; $\mathbf{U}$ is the DFT matrix ($\mathbf{U}^H\mathbf{U}=\mathbf{U}\mathbf{U}^H=\mathbf{I}$)

$$\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}=\mathbf{U}^H\underbrace{\left(|\mathbf{F}|^2+\mathbf{I}\right)^{-1}}_{\text{diagonal}}\mathbf{U}$$

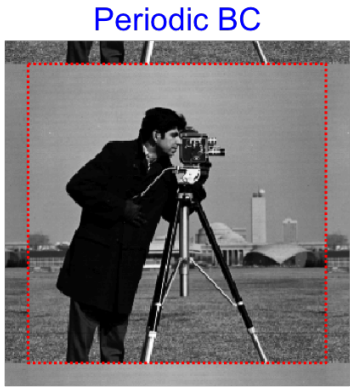
  ✓ Inpainting: $\mathbf{B}\in\{0,1\}^{m\times n}$, with m rows of $\mathbf{I}$; thus, $\mathbf{B}^T\mathbf{B}$ is diagonal

  ✓ Compressive Fourier imaging (MRI, multi-coil MRI): $\mathbf{B}=\mathbf{M}\mathbf{U}$, where $\mathbf{M}\in\{0,1\}^{m\times n}$, with m rows of $\mathbf{I}$; thus, $\mathbf{M}\mathbf{M}^T=\mathbf{I}$

$$\left(\mathbf{B}^T\mathbf{B}+\mathbf{I}\right)^{-1}=\mathbf{I}-\frac{1}{2}\,\mathbf{U}^H\underbrace{\mathbf{M}^T\mathbf{M}}_{\text{diagonal}}\,\mathbf{U}$$

- Cost is at most $O(n\log n)$

Non-periodic Deconvolution



- Periodic boundary conditions are usually unnatural

Non-periodic Deconvolution

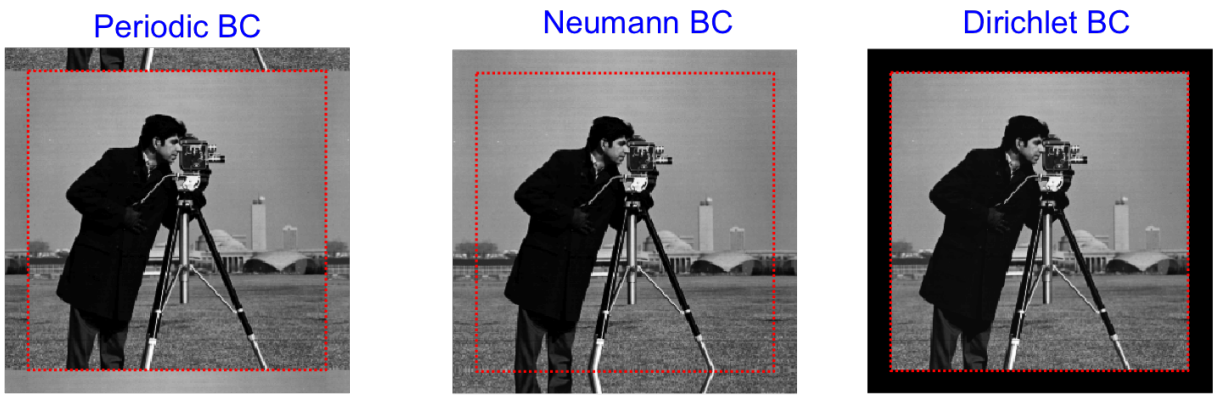
Periodic BC

Neumann BC

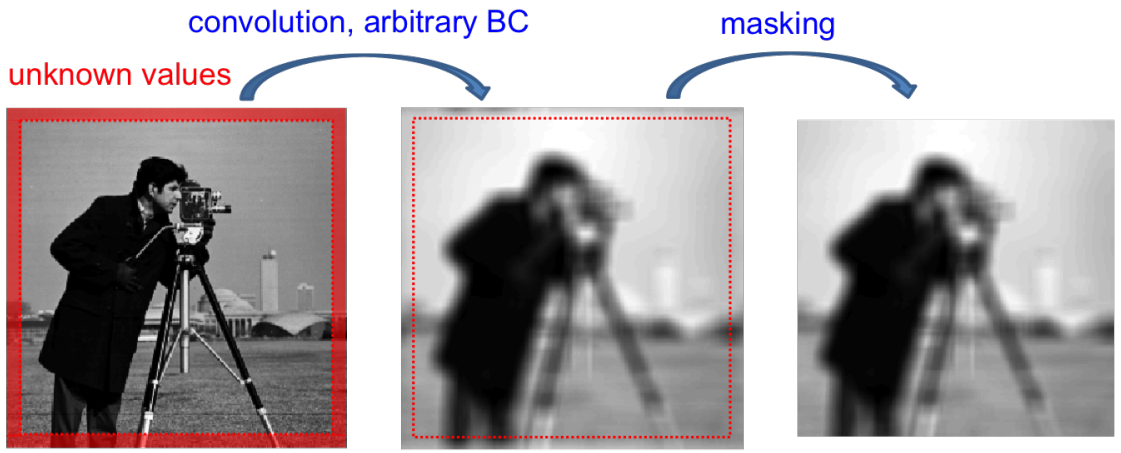
Dirichlet BC

- Periodic boundary conditions are usually unnatural
- ...as are other standard BC: Neumann, Dirichlet.

Non-periodic Deconvolution



- Periodic boundary conditions are usually unnatural
- ...as are other standard BC: Neumann, Dirichlet.
- A more natural choice: unknown boundaries [Reeves, 2005], [Chan et al., 2005], [Almeida and F, 2013a], [Ramani and Fessler, 2013]



Non-periodic Deconvolution (2)

- Gaussian noise model: $\Psi(\mathbf{Bx}, \mathbf{y}) = \frac{1}{2\sigma^2} \| \overbrace{\mathbf{M}}^{\text{mask}} \underbrace{\mathbf{U}^H \mathbf{F} \mathbf{U}}_{\text{period. conv.}} \mathbf{x} - \mathbf{y} \|_2^2$

Non-periodic Deconvolution (2)

- Gaussian noise model: $\Psi(\mathbf{Bx}, \mathbf{y}) = \frac{1}{2\sigma^2} \| \overbrace{\mathbf{M}}^{\text{mask}} \underbrace{\mathbf{U}^H \mathbf{F} \mathbf{U}}_{\text{period. conv.}} \mathbf{x} - \mathbf{y} \|_2^2$

- Choosing $\mathbf{B} = \mathbf{M}\mathbf{U}^H\mathbf{F}\mathbf{U}$, makes $\left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}$ hard to compute

Non-periodic Deconvolution (2)

- Gaussian noise model: $\Psi(\mathbf{Bx}, \mathbf{y}) = \frac{1}{2\sigma^2} \| \overbrace{\mathbf{M}}^{\text{mask}} \underbrace{\mathbf{U}^H \mathbf{F} \mathbf{U}}_{\text{period. conv.}} \mathbf{x} - \mathbf{y} \|_2^2$

- Choosing $\mathbf{B} = \mathbf{M}\mathbf{U}^H\mathbf{F}\mathbf{U}$, makes $\left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}$ hard to compute

- Better option: $\mathbf{B} = \mathbf{U}^H\mathbf{F}\mathbf{U}$ (as in periodic deconvolution), and

$$\Psi(\mathbf{u}, \mathbf{y}) = \frac{1}{2\sigma^2} \|\mathbf{M}\mathbf{u} - \mathbf{y}\|_2^2$$

Non-periodic Deconvolution (2)

- Gaussian noise model: $\Psi(\mathbf{Bx}, \mathbf{y}) = \frac{1}{2\sigma^2} \| \overbrace{\mathbf{M}}^{\text{mask}} \underbrace{\mathbf{U}^H \mathbf{F} \mathbf{U}}_{\text{period. conv.}} \mathbf{x} - \mathbf{y} \|_2^2$
- Choosing $\mathbf{B} = \mathbf{M}\mathbf{U}^H\mathbf{F}\mathbf{U}$, makes $\left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}$ hard to compute
- Better option: $\mathbf{B} = \mathbf{U}^H\mathbf{F}\mathbf{U}$ (as in periodic deconvolution), and

$$\Psi(\mathbf{u}, \mathbf{y}) = \frac{1}{2\sigma^2} \|\mathbf{M}\mathbf{u} - \mathbf{y}\|_2^2$$

- The proximity operator is still simple:

$$\text{prox}_{\tau\Psi}(\mathbf{u}) = \underbrace{\left(\tau\mathbf{M}^T\mathbf{M} + \sigma^2\mathbf{I}\right)^{-1}}_{\text{diagonal}} \left(\tau\mathbf{M}^T\mathbf{y} + \sigma^2\mathbf{u}\right)$$

Non-periodic Deconvolution (2)

- Gaussian noise model: $\Psi(\mathbf{Bx}, \mathbf{y}) = \frac{1}{2\sigma^2} \| \overbrace{\mathbf{M}}^{\text{mask}} \underbrace{\mathbf{U}^H \mathbf{F} \mathbf{U}}_{\text{period. conv.}} \mathbf{x} - \mathbf{y} \|_2^2$
- Choosing $\mathbf{B} = \mathbf{M}\mathbf{U}^H\mathbf{F}\mathbf{U}$, makes $\left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}$ hard to compute
- Better option: $\mathbf{B} = \mathbf{U}^H\mathbf{F}\mathbf{U}$ (as in periodic deconvolution), and

$$\Psi(\mathbf{u}, \mathbf{y}) = \frac{1}{2\sigma^2} \|\mathbf{M}\mathbf{u} - \mathbf{y}\|_2^2$$

- The proximity operator is still simple:

$$\text{prox}_{\tau\Psi}(\mathbf{u}) = \underbrace{\left(\tau\mathbf{M}^T\mathbf{M} + \sigma^2\mathbf{I}\right)^{-1}}_{\text{diagonal}} \left(\tau\mathbf{M}^T\mathbf{y} + \sigma^2\mathbf{u}\right)$$

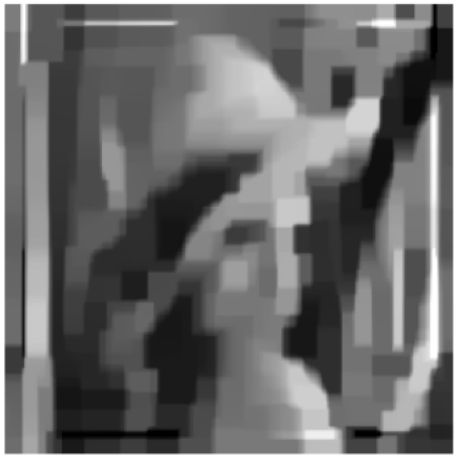
- Similar formulations:
  - ✓ deconvolution + inpainting ($\mathbf{M}$ masks the boundary and missing pixels)
  - ✓ super-resolution (filtering + downsampling mask)

Deconvolution with Unknown Boundaries: Example

original (256×256)

observed (238×238)

Assuming periodic BC



FA-BC (ISNR = -2.52dB)

Edge tapering

FA-ET (ISNR = 3.06dB)

Unknown BC by ADMM

FA-MD (ISRN = 10.63dB)

Deconvolution + Inpainting with Unknown BC: Example

original (256×256)

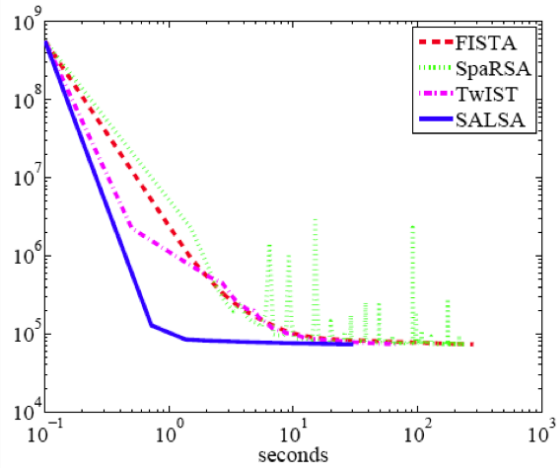
observed (238×238)

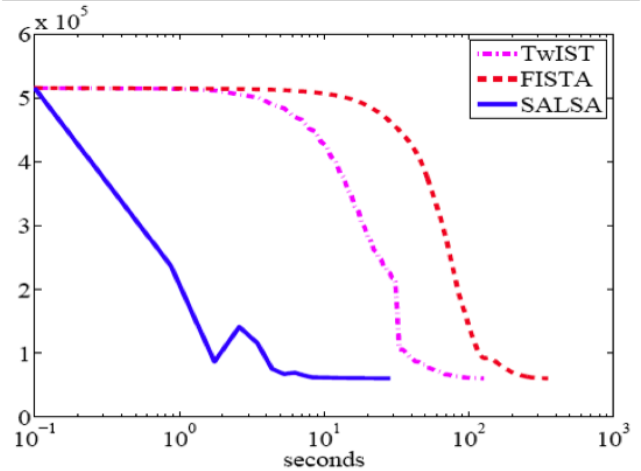
FA-CG (SNR = 20.58dB)

FA-MD (SNR = 20.57dB)

Speed

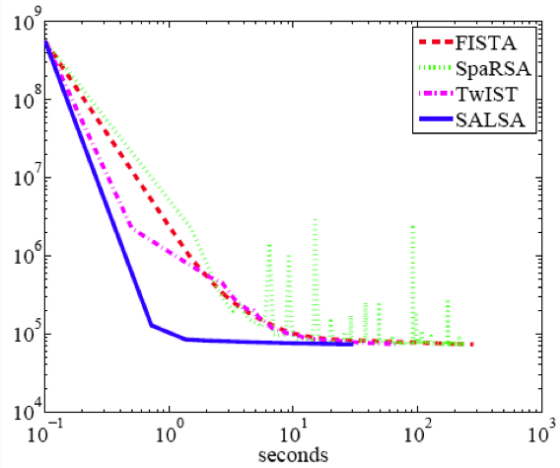
- Benchmark deblurring problem (9×9 blur, 40dB SNR, Haar frame, ℓ_1) and inpainting problem (50% missing data) [Afonso et al., 2011]

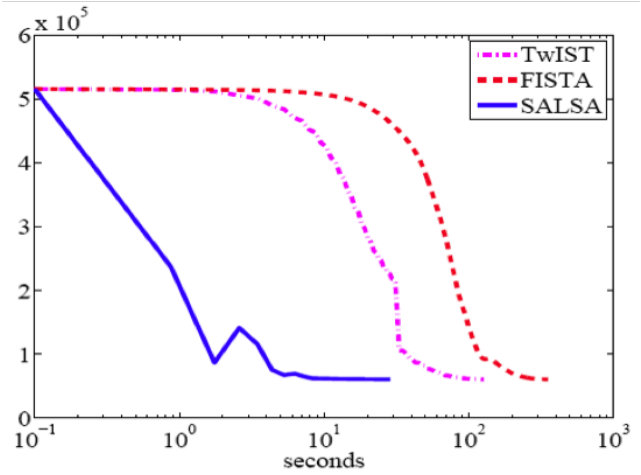




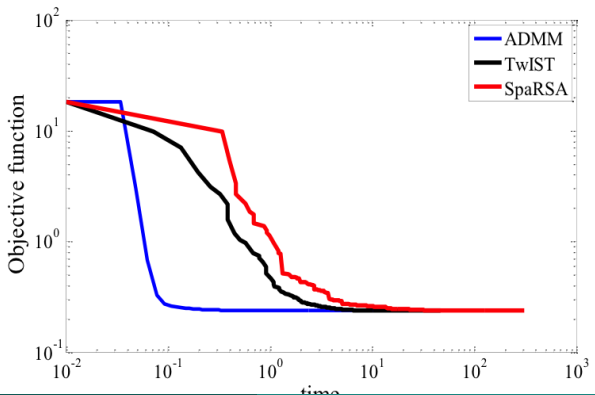
Speed

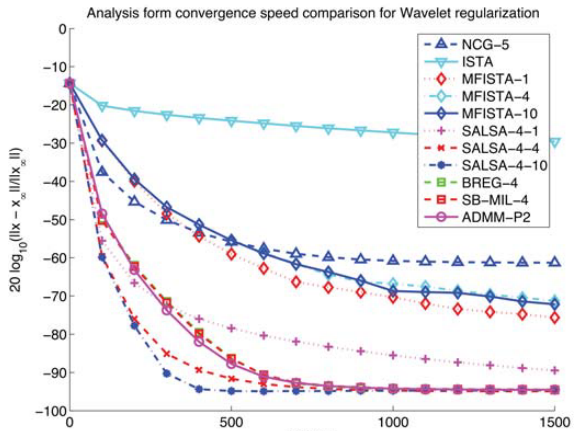
- Benchmark deblurring problem (9×9 blur, 40dB SNR, Haar frame, ℓ_1) and inpainting problem (50% missing data) [Afonso et al., 2011]





- Deconvolution with unknown BC [Almeida and F, 2013a], [Ramani and Fessler, 2013]





Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:

Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization

Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization
  - ✓ Total variation regularization

Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization
  - ✓ Total variation regularization
  - ✓ ...or combinations thereof

Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization
  - ✓ Total variation regularization
  - ✓ ...or combinations thereof
  - ✓ Tikhonov, Morozov, Ivanov formulations

Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization
  - ✓ Total variation regularization
  - ✓ ...or combinations thereof
  - ✓ Tikhonov, Morozov, Ivanov formulations
  - ✓ Gaussian, Poissonian, multiplicative noise, ...

Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization
  - ✓ Total variation regularization
  - ✓ ...or combinations thereof
  - ✓ Tikhonov, Morozov, Ivanov formulations
  - ✓ Gaussian, Poissonian, multiplicative noise, ...
  - ✓ Deconvolution, inpainting, compressive Fourier sensing (MRI), super-resolution, ...

Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization
  - ✓ Total variation regularization
  - ✓ ...or combinations thereof
  - ✓ Tikhonov, Morozov, Ivanov formulations
  - ✓ Gaussian, Poissonian, multiplicative noise, ...
  - ✓ Deconvolution, inpainting, compressive Fourier sensing (MRI), super-resolution, ...
  - ✓ Periodic or unknown boundaries

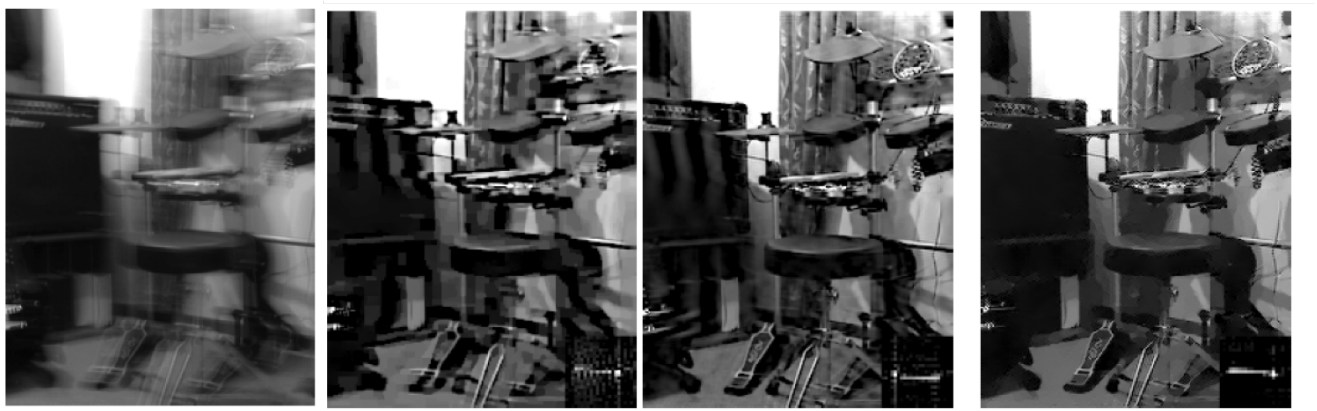
Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization
  - ✓ Total variation regularization
  - ✓ ...or combinations thereof
  - ✓ Tikhonov, Morozov, Ivanov formulations
  - ✓ Gaussian, Poissonian, multiplicative noise, ...
  - ✓ Deconvolution, inpainting, compressive Fourier sensing (MRI), super-resolution, ...
  - ✓ Periodic or unknown boundaries
  - ✓ Blind deconvolution
- Convergence guaranteed by classical results [Eckstein and Bertsekas, 1992] ...functions are closed, proper, convex; matrices have full column rank (except blind deconvolution)

Summaryzing

- ...a flexible toolbox for ADMM-based image restoration:
  - ✓ Frame-based analysis or synthesis regularization
  - ✓ Total variation regularization
  - ✓ ...or combinations thereof
  - ✓ Tikhonov, Morozov, Ivanov formulations
  - ✓ Gaussian, Poissonian, multiplicative noise, ...
  - ✓ Deconvolution, inpainting, compressive Fourier sensing (MRI), super-resolution, ...
  - ✓ Periodic or unknown boundaries
  - ✓ Blind deconvolution
- Convergence guaranteed by classical results [Eckstein and Bertsekas, 1992] ...functions are closed, proper, convex; matrices have full column rank (except blind deconvolution)
- Current research: choice of parameter ρ_k [Xu et al., 2016]

Blind Deconvolution: Real Examples

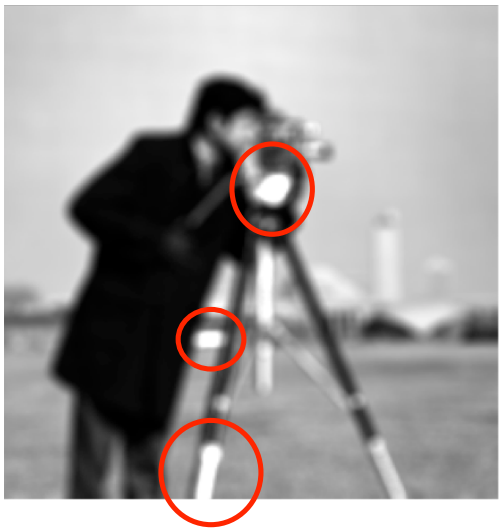


Blurred photo | [14], 70 seconds | [16], 100 seconds | Proposed method, 55 seconds

Observed photo. | [Almeida et al, 2010] | proposed

Results from [Almeida and F, 2013b]

Blind Deconvolution: The Importance of Inpainting



blur and saturations

blind deblurring
ignoring saturations

blind deblurring
accounting for saturations

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$
- ADMM directly applied to this problem has the form

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{prox}_{\Phi/\rho}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$
- ADMM directly applied to this problem has the form

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{prox}_{\Phi/\rho}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

- The prox of the regularizer Φ is a denoising operation

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$
- ADMM directly applied to this problem has the form

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{prox}_{\Phi/\rho}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

- The prox of the regularizer Φ is a denoising operation
- Prox of convex regularizer (frames, TV): not state-of-the-art denoising

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$
- ADMM directly applied to this problem has the form

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{prox}_{\Phi/\rho}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

- The prox of the regularizer Φ is a denoising operation
- Prox of convex regularizer (frames, TV): not state-of-the-art denoising
- State-of-the-art denoising methods are patch-based, non-local:

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$
- ADMM directly applied to this problem has the form

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{prox}_{\Phi/\rho}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

- The prox of the regularizer Φ is a denoising operation
- Prox of convex regularizer (frames, TV): not state-of-the-art denoising
- State-of-the-art denoising methods are patch-based, non-local:
  - ✓ Collaborative filtering (BM3D) [Dabov et al., 2007]

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$
- ADMM directly applied to this problem has the form

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{prox}_{\Phi/\rho}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

- The prox of the regularizer Φ is a denoising operation
- Prox of convex regularizer (frames, TV): not state-of-the-art denoising
- State-of-the-art denoising methods are patch-based, non-local:
  - ✓ Collaborative filtering (BM3D) [Dabov et al., 2007]
  - ✓ Non-local Bayes [Lebrun et al., 2013]

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$
- ADMM directly applied to this problem has the form

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{prox}_{\Phi/\rho}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

- The prox of the regularizer Φ is a denoising operation
- Prox of convex regularizer (frames, TV): not state-of-the-art denoising
- State-of-the-art denoising methods are patch-based, non-local:
  - ✓ Collaborative filtering (BM3D) [Dabov et al., 2007]
  - ✓ Non-local Bayes [Lebrun et al., 2013]
  - ✓ Gaussian mixture models [Zoran and Weiss, 2011], [Teodoro et al., 2015]

Denoising Step in ADMM

- Restoration (w/ Gauss noise): $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \Phi(\mathbf{x})$
- ADMM directly applied to this problem has the form

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{prox}_{\Phi/\rho}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

- The prox of the regularizer Φ is a denoising operation
- Prox of convex regularizer (frames, TV): not state-of-the-art denoising
- State-of-the-art denoising methods are patch-based, non-local:
  - ✓ Collaborative filtering (BM3D) [Dabov et al., 2007]
  - ✓ Non-local Bayes [Lebrun et al., 2013]
  - ✓ Gaussian mixture models [Zoran and Weiss, 2011], [Teodoro et al., 2015]
- Can we use one of these denoisers instead of some proximity operator?

Plug-and-Play ADMM

- Plug a black-box denoiser into ADMM [Venkatakrishnan et al., 2013]

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{denoise}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}$$

Plug-and-Play ADMM

- Plug a black-box denoiser into ADMM [Venkatakrishnan et al., 2013]

$$\begin{aligned}
\mathbf{x}_{k+1} &= \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right) \\
\mathbf{z}_{k+1} &= \text{denoise}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right) \\
\mathbf{u}_{k+1} &= \mathbf{u}_k + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}
\end{aligned}$$

- If denoiser $=$ prox$_\phi$, for convex ϕ, convergence is well-known [Eckstein and Bertsekas, 1992, Boyd et al., 2011].

Plug-and-Play ADMM

- Plug a black-box denoiser into ADMM [Venkatakrishnan et al., 2013]

$$\begin{aligned}\mathbf{x}_{k+1} &= \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)\\ \mathbf{z}_{k+1} &= \text{denoise}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)\\ \mathbf{u}_{k+1} &= \mathbf{u}_k + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}\end{aligned}$$

- If denoiser $=$ prox$_\phi$, for convex ϕ, convergence is well-known [Eckstein and Bertsekas, 1992, Boyd et al., 2011].

- ...what about convergence of PnP-ADMM? [Sreehari et al., 2016, Teodoro et al., 2017a, Chan et al., 2017]

Plug-and-Play ADMM

- Plug a black-box denoiser into ADMM [Venkatakrishnan et al., 2013]

$$\begin{aligned}\mathbf{x}_{k+1} &= \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k - \mathbf{u}_k)\right)\\ \mathbf{z}_{k+1} &= \text{denoise}\left(\mathbf{x}_{k+1} + \mathbf{u}_k\right)\\ \mathbf{u}_{k+1} &= \mathbf{u}_k + \mathbf{x}_{k+1} - \mathbf{z}_{k+1}\end{aligned}$$

- If denoiser $=$ prox$_\phi$, for convex ϕ, convergence is well-known [Eckstein and Bertsekas, 1992, Boyd et al., 2011].

- ...what about convergence of PnP-ADMM? [Sreehari et al., 2016, Teodoro et al., 2017a, Chan et al., 2017]

- Empirical results: competitive!

GMM-Based Denoising

- Observation model: $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$

GMM-Based Denoising

- Observation model: $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Decompose noisy image into overlapping patches $\mathbf{y}_i$

GMM-Based Denoising

- Observation model: $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Decompose noisy image into overlapping patches $\mathbf{y}_i$
- Denoise each patch independently under GMM prior:

$$p(\mathbf{x}_i) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_j, \mathbf{C}_j)$$

GMM-Based Denoising

- Observation model: $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Decompose noisy image into overlapping patches $\mathbf{y}_i$
- Denoise each patch independently under GMM prior:

$$p(\mathbf{x}_i) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_j, \mathbf{C}_j)$$

- The minimum mean squared error (MMSE) estimate (not the MAP) has closed-form:

$$\hat{\mathbf{x}}_i = \mathbb{E}[\mathbf{X}_i|\mathbf{y}_i]$$

GMM-Based Denoising

- Observation model: $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Decompose noisy image into overlapping patches $\mathbf{y}_i$
- Denoise each patch independently under GMM prior:

$$p(\mathbf{x}_i) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_j, \mathbf{C}_j)$$

- The minimum mean squared error (MMSE) estimate (not the MAP) has closed-form:

$$\hat{\mathbf{x}}_i = \mathbb{E}[\mathbf{X}_i|\mathbf{y}_i]$$

- Assemble the denoised image by putting the estimated patches at their locations, averaging overlapping pixel estimates

GMM-Based Denoising

- Observation model: $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Decompose noisy image into overlapping patches $\mathbf{y}_i$
- Denoise each patch independently under GMM prior:

$$p(\mathbf{x}_i) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_j, \mathbf{C}_j)$$

- The minimum mean squared error (MMSE) estimate (not the MAP) has closed-form:

$$\hat{\mathbf{x}}_i = \mathbb{E}[\mathbf{X}_i|\mathbf{y}_i]$$

- Assemble the denoised image by putting the estimated patches at their locations, averaging overlapping pixel estimates
- Estimating the mixture:

GMM-Based Denoising

- Observation model: $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Decompose noisy image into overlapping patches $\mathbf{y}_i$
- Denoise each patch independently under GMM prior:

$$p(\mathbf{x}_i) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_j, \mathbf{C}_j)$$

- The minimum mean squared error (MMSE) estimate (not the MAP) has closed-form:

$$\hat{\mathbf{x}}_i = \mathbb{E}[\mathbf{X}_i|\mathbf{y}_i]$$

- Assemble the denoised image by putting the estimated patches at their locations, averaging overlapping pixel estimates
- Estimating the mixture:
  - ✓ From a collection of clean image patches [Zoran and Weiss, 2011]

GMM-Based Denoising

- Observation model: $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Decompose noisy image into overlapping patches $\mathbf{y}_i$
- Denoise each patch independently under GMM prior:

$$p(\mathbf{x}_i) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_j, \mathbf{C}_j)$$

- The minimum mean squared error (MMSE) estimate (not the MAP) has closed-form:

$$\hat{\mathbf{x}}_i = \mathbb{E}[\mathbf{X}_i|\mathbf{y}_i]$$

- Assemble the denoised image by putting the estimated patches at their locations, averaging overlapping pixel estimates
- Estimating the mixture:
  - ✓ From a collection of clean image patches [Zoran and Weiss, 2011]
  - ✓ From the noisy image itself using EM [Teodoro et al., 2015]

MMSE Estimate with GMM Prior

- Gaussian noisy observations: $f_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$

MMSE Estimate with GMM Prior

- Gaussian noisy observations: $f_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Gaussian prior: $f_{\mathbf{X}}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C})$

MMSE Estimate with GMM Prior

- Gaussian noisy observations: $f_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Gaussian prior: $f_{\mathbf{X}}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C})$
- MMSE estimate:

$$\arg\min_{\hat{\mathbf{x}}} \mathbb{E}[\|\hat{\mathbf{x}} - \mathbf{X}\|_2^2|\mathbf{y}] = \mathbb{E}[\mathbf{X}|\mathbf{y}] = \left(\sigma^2\mathbf{C} + \mathbf{I}\right)^{-1}\left(\sigma^2\mathbf{C}^{-1}\boldsymbol{\mu} + \mathbf{y}\right)$$

MMSE Estimate with GMM Prior

- Gaussian noisy observations: $f_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Gaussian prior: $f_{\mathbf{X}}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C})$
- MMSE estimate:

$$\arg\min_{\hat{\mathbf{x}}} \mathbb{E}[\|\hat{\mathbf{x}} - \mathbf{X}\|_2^2|\mathbf{y}] = \mathbb{E}[\mathbf{X}|\mathbf{y}] = \left(\sigma^2\mathbf{C} + \mathbf{I}\right)^{-1}\left(\sigma^2\mathbf{C}^{-1}\boldsymbol{\mu} + \mathbf{y}\right)$$

- Gaussian mixture prior: $f_{\mathbf{X}}(\mathbf{x}) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \mathbf{C}_j)$

MMSE Estimate with GMM Prior

- Gaussian noisy observations: $f_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Gaussian prior: $f_{\mathbf{X}}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C})$
- MMSE estimate:

$$\arg\min_{\hat{\mathbf{x}}} \mathbb{E}[\|\hat{\mathbf{x}} - \mathbf{X}\|_2^2|\mathbf{y}] = \mathbb{E}[\mathbf{X}|\mathbf{y}] = \left(\sigma^2\mathbf{C} + \mathbf{I}\right)^{-1}\left(\sigma^2\mathbf{C}^{-1}\boldsymbol{\mu} + \mathbf{y}\right)$$

- Gaussian mixture prior: $f_{\mathbf{X}}(\mathbf{x}) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \mathbf{C}_j)$
- MMSE estimate

$$\mathbb{E}[\mathbf{X}|\mathbf{y}] = \sum_{j=1}^{K} \beta_j(\mathbf{y})\left(\sigma^2\mathbf{C}_j + \mathbf{I}\right)^{-1}\left(\sigma^2\mathbf{C}_j^{-1}\boldsymbol{\mu}_j + \mathbf{y}\right)$$

where $\beta_j(\mathbf{y}) \propto \alpha_j \mathcal{N}(\mathbf{y}|\boldsymbol{\mu}_j, \mathbf{C}_j + \sigma^2\mathbf{I})$, with $\sum_{j=1}^{K} \beta_j(\mathbf{y}) = 1$

MMSE Estimate with GMM Prior

- Gaussian noisy observations: $f_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma^2\mathbf{I})$
- Gaussian prior: $f_{\mathbf{X}}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C})$
- MMSE estimate:

$$\arg\min_{\hat{\mathbf{x}}} \mathbb{E}[\|\hat{\mathbf{x}} - \mathbf{X}\|_2^2|\mathbf{y}] = \mathbb{E}[\mathbf{X}|\mathbf{y}] = \left(\sigma^2\mathbf{C} + \mathbf{I}\right)^{-1}\left(\sigma^2\mathbf{C}^{-1}\boldsymbol{\mu} + \mathbf{y}\right)$$

- Gaussian mixture prior: $f_{\mathbf{X}}(\mathbf{x}) = \sum_{j=1}^{K} \alpha_j \, \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \mathbf{C}_j)$
- MMSE estimate (the MAP solution has no closed form)

$$\mathbb{E}[\mathbf{X}|\mathbf{y}] = \sum_{j=1}^{K} \beta_j(\mathbf{y})\left(\sigma^2\mathbf{C}_j + \mathbf{I}\right)^{-1}\left(\sigma^2\mathbf{C}_j^{-1}\boldsymbol{\mu}_j + \mathbf{y}\right)$$

where $\beta_j(\mathbf{y}) \propto \alpha_j \mathcal{N}(\mathbf{y}|\boldsymbol{\mu}_j, \mathbf{C}_j + \sigma^2\mathbf{I})$, with $\sum_{j=1}^{K} \beta_j(\mathbf{y}) = 1$

Plug-and-Play ADMM: Deblurring of Generic Images

- Generic GMM prior

Image:	Cameraman						House					
Experiment:	1	2	3	4	5	6	1	2	3	4	5	6
IDD-BM3D [Danielyan et al., 2012]	**8.85**	**7.12**	**10.45**	**3.98**	**4.31**	**4.89**	**9.95**	**8.55**	**12.89**	**5.79**	**5.74**	**7.13**
ADMM-GMM [Teodoro et al., 2016b]	8.39	6.36	9.80	3.47	4.16	4.88	9.66	8.22	12.43	5.50	5.42	6.82

(a) Original

(b) Blurred

(c) IDD-BM3D

(d) ADMM-GMM

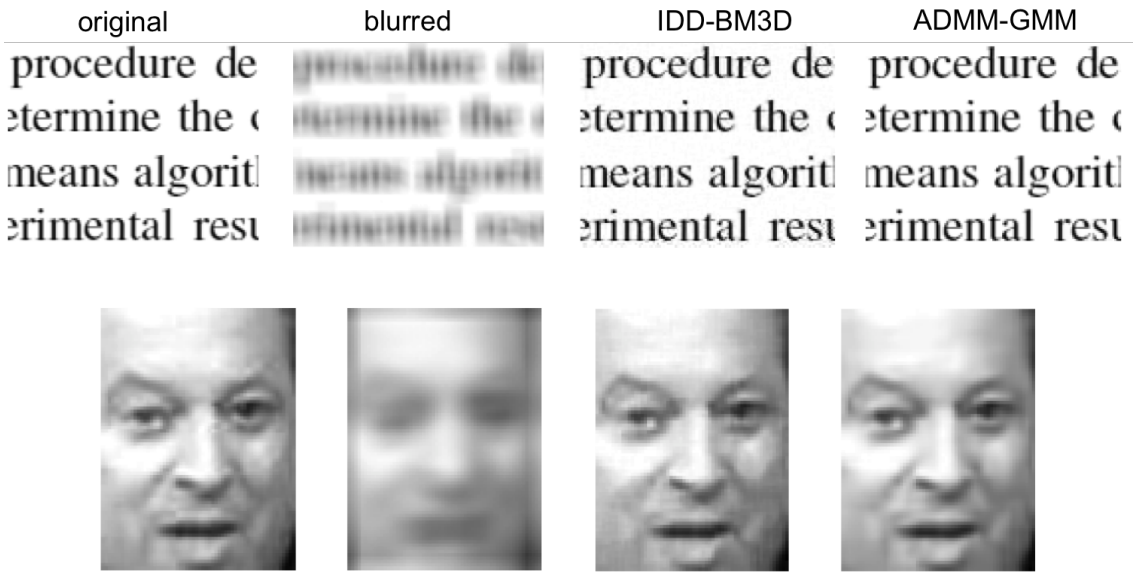
- For generic natural images: competitive, but does not beat state-of-the-art

Class-Adapted GMM-Based Restoration

- Learn a GMM for class of images, plug the corresponding denoiser into ADMM [Teodoro et al., 2016b]

Class-Adapted GMM-Based Restoration

- Learn a GMM for class of images, plug the corresponding denoiser into ADMM [Teodoro et al., 2016b]



Class-Adapted GMM-Based Restoration

- Learn a GMM for class of images, plug the corresponding denoiser into ADMM [Teodoro et al., 2016b]

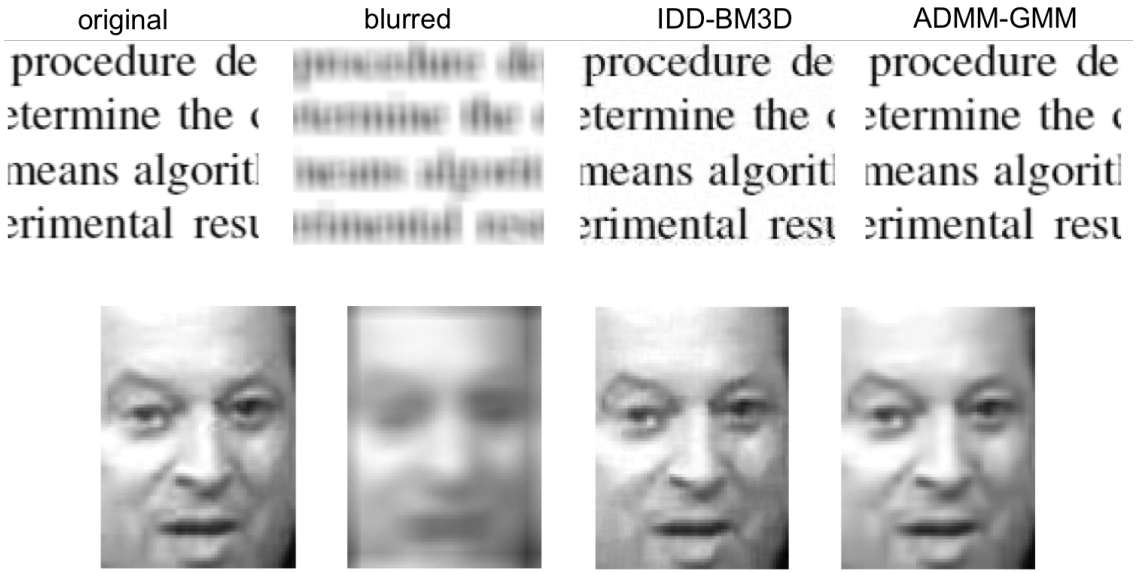


Image class:	Text						Face					
Experiment:	1	2	3	4	5	6	1	2	3	4	5	6
BSNR	26.07	20.05	40.00	15.95	24.78	18.11	28.28	22.26	40.00	15.89	26.22	15.37
Input PSNR	14.14	14.13	12.13	16.83	14.48	28.73	25.61	22.54	20.71	26.49	24.79	30.03
IDD-BM3D	11.97	8.91	16.29	5.88	6.81	4.87	13.66	11.16	14.96	7.31	10.33	6.18
ADMM-GMM	**16.24**	**11.55**	**23.11**	**8.88**	**10.77**	**8.34**	**15.05**	**12.59**	**17.28**	**8.84**	**11.69**	**7.32**

Blind Deblurring

- Blind image deblurring/deconvolution

$$\mathbf{y} = \mathbf{h} * \mathbf{x} + \mathbf{n}$$

where both $\mathbf{x}$ and $\mathbf{h}$ are unknown

Blind Deblurring

- Blind image deblurring/deconvolution

$$\mathbf{y} = \mathbf{h} * \mathbf{x} + \mathbf{n} = \mathbf{H}(\mathbf{h})\, \mathbf{x} + \mathbf{n}$$

where both $\mathbf{x}$ and $\mathbf{h}$ are unknown

Blind Deblurring

- Blind image deblurring/deconvolution

$$\mathbf{y} = \mathbf{h} * \mathbf{x} + \mathbf{n} = \mathbf{H}(\mathbf{h})\,\mathbf{x} + \mathbf{n} = \mathbf{X}(\mathbf{x})\,\mathbf{h} + \mathbf{n}$$

where both $\mathbf{x}$ and $\mathbf{h}$ are unknown

Blind Deblurring

- Blind image deblurring/deconvolution

$$\mathbf{y} = \mathbf{h} * \mathbf{x} + \mathbf{n} = \mathbf{H}(\mathbf{h})\,\mathbf{x} + \mathbf{n} = \mathbf{X}(\mathbf{x})\,\mathbf{h} + \mathbf{n}$$

  where both $\mathbf{x}$ and $\mathbf{h}$ are unknown

- Joint criterion (under Gaussian noise) [Almeida and F, 2013b]

$$(\hat{\mathbf{x}}, \hat{\mathbf{h}}) \in \arg\min_{\mathbf{x},\mathbf{h}} \underbrace{\frac{1}{2}\|\mathbf{h} * \mathbf{x} - \mathbf{y}\|_2^2 + \Phi(\mathbf{x}) + \Psi(\mathbf{h})}_{O(\mathbf{x},\mathbf{h})}$$

  where Φ and Ψ are regularizers

Blind Deblurring

- Blind image deblurring/deconvolution

$$\mathbf{y} = \mathbf{h} * \mathbf{x} + \mathbf{n} = \mathbf{H}(\mathbf{h})\,\mathbf{x} + \mathbf{n} = \mathbf{X}(\mathbf{x})\,\mathbf{h} + \mathbf{n}$$

  where both $\mathbf{x}$ and $\mathbf{h}$ are unknown

- Joint criterion (under Gaussian noise) [Almeida and F, 2013b]

$$(\hat{\mathbf{x}}, \hat{\mathbf{h}}) \in \arg\min_{\mathbf{x},\mathbf{h}} \underbrace{\frac{1}{2}\|\mathbf{h} * \mathbf{x} - \mathbf{y}\|_2^2 + \Phi(\mathbf{x}) + \Psi(\mathbf{h})}_{O(\mathbf{x},\mathbf{h})}$$

  where Φ and Ψ are regularizers

- Even if Φ and Ψ are convex, this is a non-convex problem

Algorithm

- Proximal alternating minimization [Attouch et al., 2007]

Algorithm

- Proximal alternating minimization [Attouch et al., 2007]
- Solver for each minimization: ADMM/SALSA

Algorithm

- Proximal alternating minimization [Attouch et al., 2007]
- Solver for each minimization: ADMM/SALSA

Initialization: $\hat{\mathbf{x}} = \mathbf{y}$, $\hat{\mathbf{h}}$ - identity filter

while stopping criterion is not satisfied **do**

$$\hat{\mathbf{x}} \leftarrow \underset{\mathbf{x}}{\text{argmin}} \; O(\mathbf{x}, \hat{\mathbf{h}}) + \frac{\rho_x}{2} \|\mathbf{x} - \hat{\mathbf{x}}^{\text{previous}}\|^2$$

$$\hat{\mathbf{h}} \leftarrow \underset{\mathbf{h}}{\text{argmin}} \; O(\hat{\mathbf{x}}, \mathbf{h}) + \frac{\rho_h}{2} \|\mathbf{h} - \hat{\mathbf{h}}^{\text{previous}}\|^2$$

end while

Algorithm

- Proximal alternating minimization [Attouch et al., 2007]
- Solver for each minimization: ADMM/SALSA

Initialization: $\hat{\mathbf{x}} = \mathbf{y}$, $\hat{\mathbf{h}}$ - identity filter

while stopping criterion is not satisfied **do**

$$\hat{\mathbf{x}} \leftarrow \underset{\mathbf{x}}{\text{argmin}} \; O(\mathbf{x}, \hat{\mathbf{h}}) + \frac{\rho_x}{2} \|\mathbf{x} - \hat{\mathbf{x}}^{\text{previous}}\|^2$$

$$\hat{\mathbf{h}} \leftarrow \underset{\mathbf{h}}{\text{argmin}} \; O(\hat{\mathbf{x}}, \mathbf{h}) + \frac{\rho_h}{2} \|\mathbf{h} - \hat{\mathbf{h}}^{\text{previous}}\|^2$$

end while

- Image regularizer: class-adapted plug-and-play priors

Algorithm

- Proximal alternating minimization [Attouch et al., 2007]
- Solver for each minimization: ADMM/SALSA

Initialization: $\hat{\mathbf{x}} = \mathbf{y}$, $\hat{\mathbf{h}}$ - identity filter

while stopping criterion is not satisfied **do**

$$\hat{\mathbf{x}} \leftarrow \underset{\mathbf{x}}{\text{argmin}} \; O(\mathbf{x}, \hat{\mathbf{h}}) + \frac{\rho_x}{2} \|\mathbf{x} - \hat{\mathbf{x}}^{\text{previous}}\|^2$$

$$\hat{\mathbf{h}} \leftarrow \underset{\mathbf{h}}{\text{argmin}} \; O(\hat{\mathbf{x}}, \mathbf{h}) + \frac{\rho_h}{2} \|\mathbf{h} - \hat{\mathbf{h}}^{\text{previous}}\|^2$$

end while

- Image regularizer: class-adapted plug-and-play priors
- Filter regularizer: positivity and support, or sparsity

Priors

- Plug-and-play image priors:

Priors

- Plug-and-play image priors:
  - ✓ **GMM**-based patch denoiser, trained on a dataset of clean images (text, faces, fingerprint)

Priors

- Plug-and-play image priors:
  - ✓ **GMM**-based patch denoiser, trained on a dataset of clean images (text, faces, fingerprint)
  - ✓ **Dictionary**-based patch denoiser, learned from clean images (same classes)

Priors

- Plug-and-play image priors:
  - ✓ **GMM**-based patch denoiser, trained on a dataset of clean images (text, faces, fingerprint)
  - ✓ **Dictionary**-based patch denoiser, learned from clean images (same classes)
  - ✓ General-purpose **BM3D** denoiser.

Priors

- Plug-and-play image priors:
  - ✓ **GMM**-based patch denoiser, trained on a dataset of clean images (text, faces, fingerprint)
  - ✓ **Dictionary**-based patch denoiser, learned from clean images (same classes)
  - ✓ General-purpose **BM3D** denoiser.
- Blur filter priors

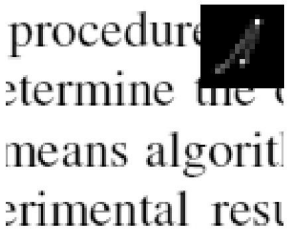
Priors

- Plug-and-play image priors:
  - ✓ **GMM**-based patch denoiser, trained on a dataset of clean images (text, faces, fingerprint)
  - ✓ **Dictionary**-based patch denoiser, learned from clean images (same classes)
  - ✓ General-purpose **BM3D** denoiser.
- Blur filter priors
  - ✓ Constraint: positivity and maximum support

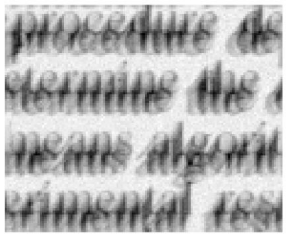
Priors

- Plug-and-play image priors:
  - ✓ **GMM**-based patch denoiser, trained on a dataset of clean images (text, faces, fingerprint)
  - ✓ **Dictionary**-based patch denoiser, learned from clean images (same classes)
  - ✓ General-purpose **BM3D** denoiser.
- Blur filter priors
  - ✓ Constraint: positivity and maximum support
  - ✓ Sparsity (adequate for motion blur)

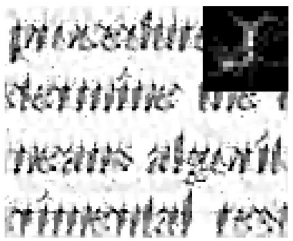
Results: GMM-based prior for text images



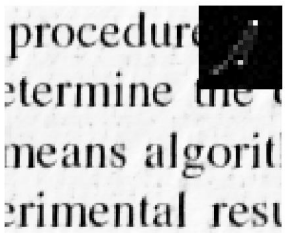
original



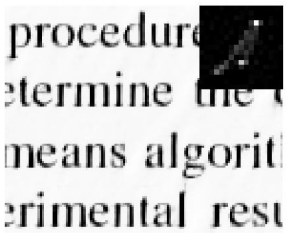
blurred



[Pan et al., 2014]



BM3D: 9.97 dB



GMM: 11.16 dB

Experiments

original

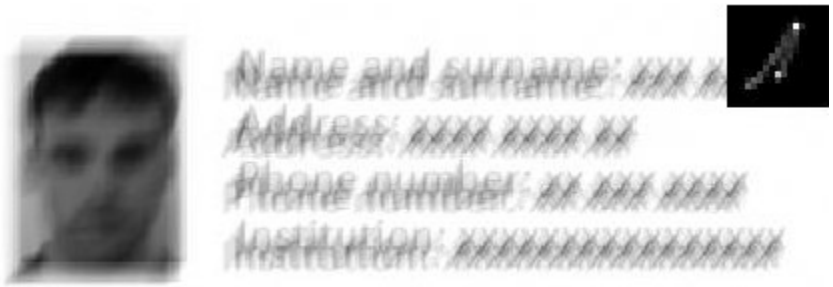
blurred

[Almeida and F, 2013b]
0.36 dB

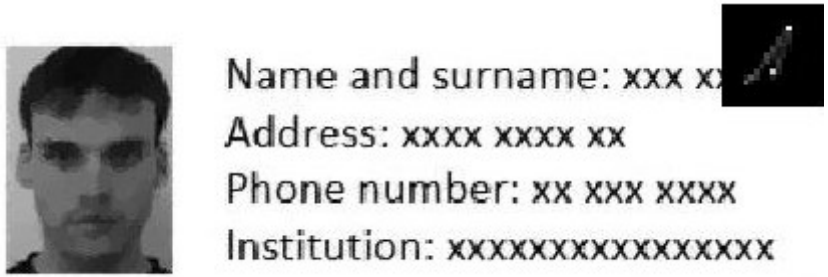
BM3D: 0.66 dB

GMM: 1.19 dB

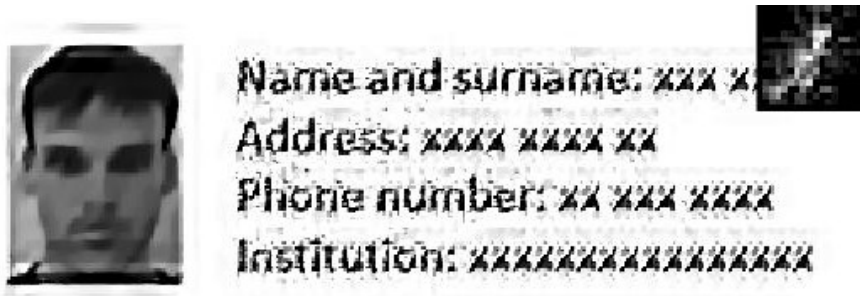
Experiments



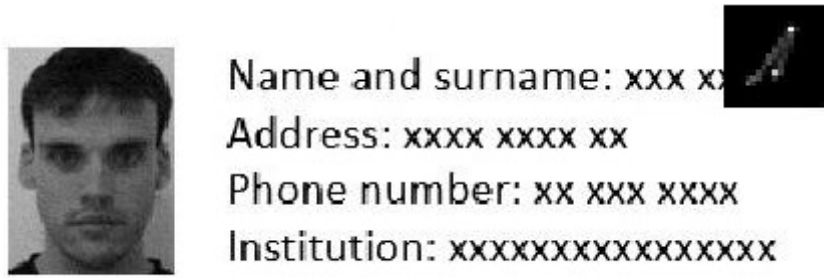
(a) Blurred image



(b) [Almeida and F, 2013b]

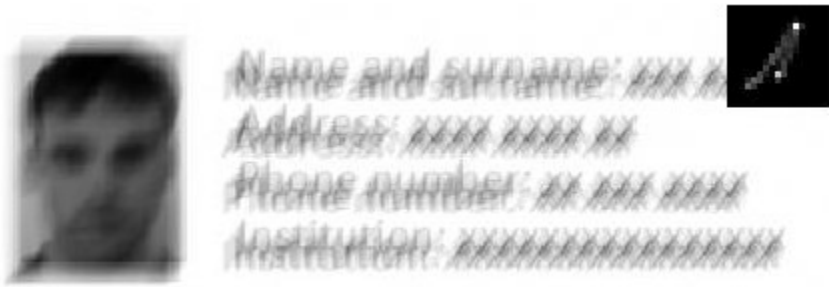


(c) [Pan et al., 2014]

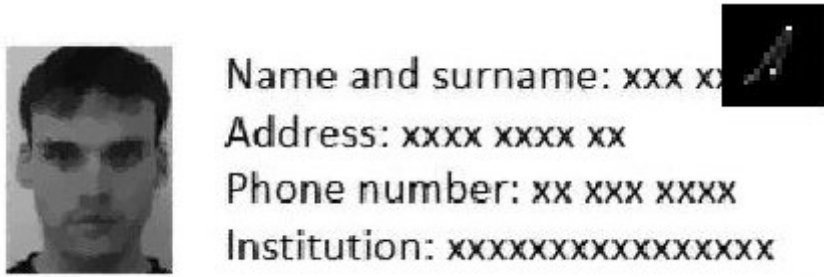


(d) Proposed

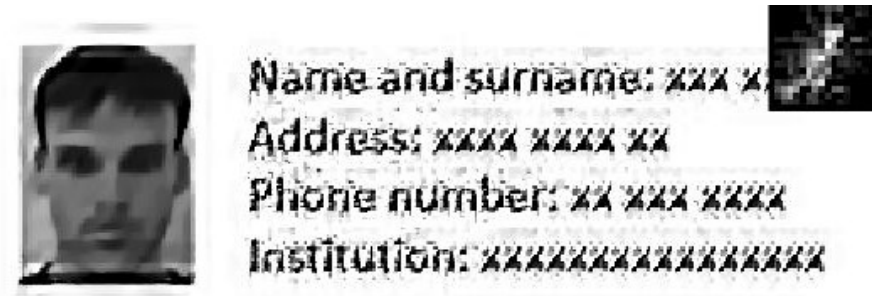
Experiments



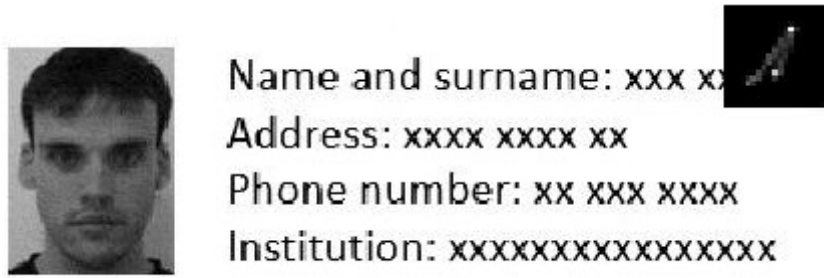
(a) Blurred image



(b) [Almeida and F, 2013b]



(c) [Pan et al., 2014]



(d) Proposed

- Uses a concatenation of two dictionaries: face and text

Experiments

original

blurred

[Krishnan et al, 2011]

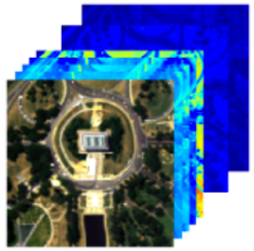
[Xu and Jia, 2011]

[Pan et al, 2014]

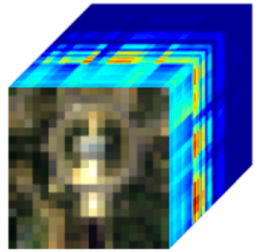
proposed

An Extreme Case of Adaptation: Hyperspectral Fusion

- Spectral-spatial resolution trade-off:



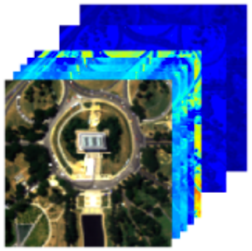
Multi-spectral:
high spatial resolution
low spectral resolution



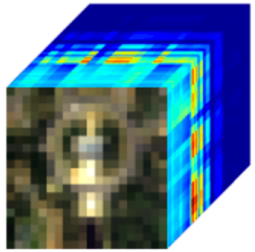
Hyper-spectral:
low spatial resolution
high spectral resolution

An Extreme Case of Adaptation: Hyperspectral Fusion

- Spectral-spatial resolution trade-off:



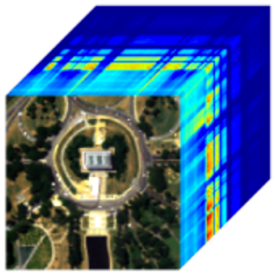
Multi-spectral:
high spatial resolution
low spectral resolution



Hyper-spectral:
low spatial resolution
high spectral resolution

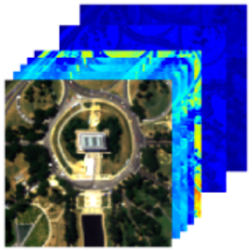
- Fuse MS and HS data:

  high spatial & spectral resolutions

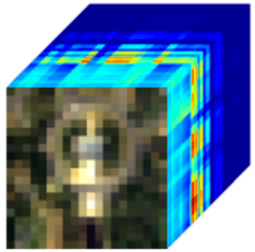


An Extreme Case of Adaptation: Hyperspectral Fusion

- Spectral-spatial resolution trade-off:



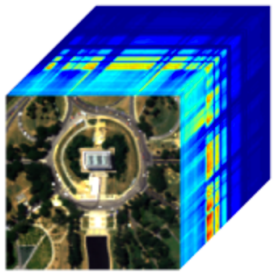
Multi-spectral:
high spatial resolution
low spectral resolution



Hyper-spectral:
low spatial resolution
high spectral resolution

- Fuse MS and HS data:

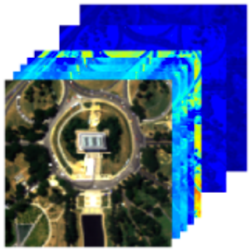
  high spatial & spectral resolutions



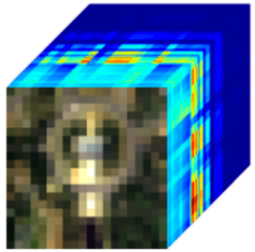
- Extreme case: pansharpening (panchromatic rather than MS image).

An Extreme Case of Adaptation: Hyperspectral Fusion

- Spectral-spatial resolution trade-off:



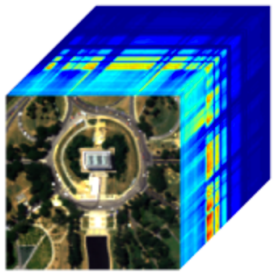
Multi-spectral:
high spatial resolution
low spectral resolution



Hyper-spectral:
low spatial resolution
high spectral resolution

- Fuse MS and HS data:

  high spatial & spectral resolutions



- Extreme case: pansharpening (panchromatic rather than MS image).

Hyperspectral Fusion: Formulation

- Observation model [Simões et al., 2015]

$$\begin{aligned} \mathbf{Y}_h &= \overbrace{\mathbf{EX}}^{\mathbf{Z}}\mathbf{BM} + \mathbf{N}_h \\ \mathbf{Y}_m &= \mathbf{R}\underbrace{\mathbf{EX}}_{\mathbf{Z}} + \mathbf{N}_m \end{aligned}$$

hyperspectral data $\in \mathbb{R}^{L_h \times n_h}$

multispectral data $\in \mathbb{R}^{L_m \times n_m}$

$L_h > L_m$ and $n_h < n_m$

Hyperspectral Fusion: Formulation

- Observation model [Simões et al., 2015]

$$\mathbf{Y}_h = \overbrace{\mathbf{EX}}^{\mathbf{Z}}\mathbf{BM} + \mathbf{N}_h \qquad \text{hyperspectral data} \in \mathbb{R}^{L_h \times n_h}$$

$$\mathbf{Y}_m = \mathbf{R}\underbrace{\mathbf{EX}}_{\mathbf{Z}} + \mathbf{N}_m \qquad \text{multispectral data} \in \mathbb{R}^{L_m \times n_m}$$

$$L_h > L_m \quad \text{and} \quad n_h < n_m$$

✓ $\mathbf{E} \in \mathbb{R}^{L_h \times p}$: the p-dimensional subspace containing the fused image $\mathbf{Z}$

Hyperspectral Fusion: Formulation

- Observation model [Simões et al., 2015]

$$\mathbf{Y}_h = \overbrace{\mathbf{EX}}^{\mathbf{Z}}\mathbf{BM} + \mathbf{N}_h \qquad \text{hyperspectral data} \in \mathbb{R}^{L_h \times n_h}$$

$$\mathbf{Y}_m = \mathbf{R}\underbrace{\mathbf{EX}}_{\mathbf{Z}} + \mathbf{N}_m \qquad \text{multispectral data} \in \mathbb{R}^{L_m \times n_m}$$

$$L_h > L_m \text{ and } n_h < n_m$$

- ✓ $\mathbf{E} \in \mathbb{R}^{L_h \times p}$: the p-dimensional subspace containing the fused image $\mathbf{Z}$
- ✓ $\mathbf{X} \in \mathbb{R}^{p \times n_h}$: the corresponding coefficients ($p \ll L_h$)

Hyperspectral Fusion: Formulation

- Observation model [Simões et al., 2015]

$$\mathbf{Y}_h = \overbrace{\mathbf{EX}}^{\mathbf{Z}}\mathbf{BM} + \mathbf{N}_h \qquad \text{hyperspectral data} \in \mathbb{R}^{L_h \times n_h}$$

$$\mathbf{Y}_m = \mathbf{R}\underbrace{\mathbf{EX}}_{\mathbf{Z}} + \mathbf{N}_m \qquad \text{multispectral data} \in \mathbb{R}^{L_m \times n_m}$$

$$L_h > L_m \quad \text{and} \quad n_h < n_m$$

- ✓ $\mathbf{E} \in \mathbb{R}^{L_h \times p}$: the p-dimensional subspace containing the fused image $\mathbf{Z}$
- ✓ $\mathbf{X} \in \mathbb{R}^{p \times n_h}$: the corresponding coefficients ($p \ll L_h$)
- ✓ $(\mathbf{B}\,\mathbf{M}) \in \mathbb{R}^{n_m \times n_h}$ models spatial convolution & subsampling

Hyperspectral Fusion: Formulation

- Observation model [Simões et al., 2015]

$$\mathbf{Y}_h = \overbrace{\mathbf{E}\mathbf{X}}^{\mathbf{Z}}\mathbf{B}\mathbf{M} + \mathbf{N}_h \qquad \text{hyperspectral data} \in \mathbb{R}^{L_h \times n_h}$$

$$\mathbf{Y}_m = \mathbf{R}\underbrace{\mathbf{E}\mathbf{X}}_{\mathbf{Z}} + \mathbf{N}_m \qquad \text{multispectral data} \in \mathbb{R}^{L_m \times n_m}$$

$$L_h > L_m \text{ and } n_h < n_m$$

- ✓ $\mathbf{E} \in \mathbb{R}^{L_h \times p}$: the p-dimensional subspace containing the fused image $\mathbf{Z}$
- ✓ $\mathbf{X} \in \mathbb{R}^{p \times n_h}$: the corresponding coefficients ($p \ll L_h$)
- ✓ $(\mathbf{B}\,\mathbf{M}) \in \mathbb{R}^{n_m \times n_h}$ models spatial convolution & subsampling
- ✓ $\mathbf{R} \in \mathbb{R}^{L_m \times L_h}$ models the spectral responses of the MS sensors

Hyperspectral Fusion: Formulation

- Observation model [Simões et al., 2015]

$$\mathbf{Y}_h = \overbrace{\mathbf{E}\mathbf{X}}^{\mathbf{Z}}\mathbf{B}\mathbf{M} + \mathbf{N}_h \qquad \text{hyperspectral data} \in \mathbb{R}^{L_h \times n_h}$$

$$\mathbf{Y}_m = \mathbf{R}\underbrace{\mathbf{E}\mathbf{X}}_{\mathbf{Z}} + \mathbf{N}_m \qquad \text{multispectral data} \in \mathbb{R}^{L_m \times n_m}$$

$$L_h > L_m \text{ and } n_h < n_m$$

- ✓ $\mathbf{E} \in \mathbb{R}^{L_h \times p}$: the p-dimensional subspace containing the fused image $\mathbf{Z}$
- ✓ $\mathbf{X} \in \mathbb{R}^{p \times n_h}$: the corresponding coefficients ($p \ll L_h$)
- ✓ $(\mathbf{B}\,\mathbf{M}) \in \mathbb{R}^{n_m \times n_h}$ models spatial convolution & subsampling
- ✓ $\mathbf{R} \in \mathbb{R}^{L_m \times L_h}$ models the spectral responses of the MS sensors
- ✓ $\mathbf{N}_h$ and $\mathbf{N}_m$ model noise

Hyperspectral Fusion via PnP-ADMM

- Assuming Gaussian noise:

$$\widehat{\mathbf{X}} \in \arg \min_{\mathbf{X} \in \mathbb{R}^{p \times n_h}} \frac{1}{2}\|\mathbf{EXBM} - \mathbf{Y}_h\|_F^2 + \frac{\lambda_m}{2}\|\mathbf{REX} - \mathbf{Y}_m\|_F^2 + \text{“}\phi(\mathbf{X})\text{”}$$

Hyperspectral Fusion via PnP-ADMM

- Assuming Gaussian noise:

$$\widehat{\mathbf{X}} \in \arg \min_{\mathbf{X} \in \mathbb{R}^{p \times n_h}} \frac{1}{2}\|\mathbf{EXBM} - \mathbf{Y}_h\|_F^2 + \frac{\lambda_m}{2}\|\mathbf{REX} - \mathbf{Y}_m\|_F^2 + \text{“}\phi(\mathbf{X})\text{”}$$

- ...which fits nicely the SALSA template ($J = 3$): $\min_{\mathbf{x}} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\, \mathbf{x}\right)$

Hyperspectral Fusion via PnP-ADMM

- Assuming Gaussian noise:

$$\widehat{\mathbf{X}} \in \arg \min_{\mathbf{X} \in \mathbb{R}^{p \times n_h}} \frac{1}{2}\|\mathbf{EXBM} - \mathbf{Y}_h\|_F^2 + \frac{\lambda_m}{2}\|\mathbf{REX} - \mathbf{Y}_m\|_F^2 + \text{“}\phi(\mathbf{X})\text{”}$$

- ...which fits nicely the SALSA template ($J = 3$): $\min_{\mathbf{x}} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\, \mathbf{x}\right)$

- Matrix inversion computable via FFT (with periodic or unknown BC)

Hyperspectral Fusion via PnP-ADMM

- Assuming Gaussian noise:

$$\widehat{\mathbf{X}} \in \arg \min_{\mathbf{X} \in \mathbb{R}^{p \times n_h}} \frac{1}{2}\|\mathbf{EXBM} - \mathbf{Y}_h\|_F^2 + \frac{\lambda_m}{2}\|\mathbf{REX} - \mathbf{Y}_m\|_F^2 + \text{“}\phi(\mathbf{X})\text{”}$$

- ...which fits nicely the SALSA template ($J = 3$): $\min_{\mathbf{x}} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$

- Matrix inversion computable via FFT (with periodic or unknown BC)

- Proximity operators:
  - ✓ The one involving $\mathbf{RE}$: a single $p \times p$ inversion; decoupled across pixels

Hyperspectral Fusion via PnP-ADMM

- Assuming Gaussian noise:

$$\widehat{\mathbf{X}} \in \arg \min_{\mathbf{X} \in \mathbb{R}^{p \times n_h}} \frac{1}{2}\|\mathbf{EXBM} - \mathbf{Y}_h\|_F^2 + \frac{\lambda_m}{2}\|\mathbf{REX} - \mathbf{Y}_m\|_F^2 + \text{“}\phi(\mathbf{X})\text{”}$$

- ...which fits nicely the SALSA template ($J = 3$): $\min_{\mathbf{x}} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$

- Matrix inversion computable via FFT (with periodic or unknown BC)

- Proximity operators:
  - ✓ The one involving $\mathbf{RE}$: a single $p \times p$ inversion; decoupled across pixels
  - ✓ The one involving $\mathbf{BM}$: solved by FFT, decoupled across bands

Hyperspectral Fusion via PnP-ADMM

- Assuming Gaussian noise:

$$\widehat{\mathbf{X}} \in \arg \min_{\mathbf{X} \in \mathbb{R}^{p \times n_h}} \frac{1}{2}\|\mathbf{EXBM} - \mathbf{Y}_h\|_F^2 + \frac{\lambda_m}{2}\|\mathbf{REX} - \mathbf{Y}_m\|_F^2 + \text{“}\phi(\mathbf{X})\text{”}$$

- ...which fits nicely the SALSA template ($J = 3$): $\min_{\mathbf{x}} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$

- Matrix inversion computable via FFT (with periodic or unknown BC)

- Proximity operators:
  - ✓ The one involving $\mathbf{RE}$: a single $p \times p$ inversion; decoupled across pixels
  - ✓ The one involving $\mathbf{BM}$: solved by FFT, decoupled across bands
  - ✓ The prox of ϕ is replaced by an adapted GMM-based denoiser

Hyperspectral Fusion via PnP-ADMM

- Assuming Gaussian noise:

$$\widehat{\mathbf{X}} \in \arg \min_{\mathbf{X} \in \mathbb{R}^{p \times n_h}} \frac{1}{2}\|\mathbf{EXBM} - \mathbf{Y}_h\|_F^2 + \frac{\lambda_m}{2}\|\mathbf{REX} - \mathbf{Y}_m\|_F^2 + \text{“}\phi(\mathbf{X})\text{”}$$

- ...which fits nicely the SALSA template ($J = 3$): $\min_{\mathbf{x}} \sum_{j=1}^{J} g_j\left(\mathbf{H}_j\,\mathbf{x}\right)$

- Matrix inversion computable via FFT (with periodic or unknown BC)

- Proximity operators:
  - ✓ The one involving $\mathbf{RE}$: a single $p \times p$ inversion; decoupled across pixels
  - ✓ The one involving $\mathbf{BM}$: solved by FFT, decoupled across bands
  - ✓ The prox of ϕ is replaced by an adapted GMM-based denoiser

- The GMM is learned from patches of $\mathbf{Y}_m$ (high spatial resolution) [Teodoro et al., 2016a]

Convergence

- PnP-ADMM with a patch-based GMM-MMSE denoiser

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k + \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{denoiser}\left(\mathbf{x}_{k+1} - \mathbf{u}_k, 1/\rho\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} - \mathbf{x}_{k+1} + \mathbf{z}_{k+1}$$

Convergence

- PnP-ADMM with a patch-based GMM-MMSE denoiser

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k + \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{denoiser}\left(\mathbf{x}_{k+1} - \mathbf{u}_k, 1/\rho\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} - \mathbf{x}_{k+1} + \mathbf{z}_{k+1}$$

- denoiser is the prox of a convex function $\Rightarrow$ convergence.

Convergence

- PnP-ADMM with a patch-based GMM-MMSE denoiser

$$\begin{aligned}\mathbf{x}_{k+1} &= \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k + \mathbf{u}_k)\right)\\ \mathbf{z}_{k+1} &= \text{denoiser}\left(\mathbf{x}_{k+1} - \mathbf{u}_k, 1/\rho\right)\\ \mathbf{u}_{k+1} &= \mathbf{u}_{k+1} - \mathbf{x}_{k+1} + \mathbf{z}_{k+1}\end{aligned}$$

- denoiser is the prox of a convex function $\Rightarrow$ convergence.

- From Moreau [1965]: some map $p : \mathbb{R}^n \to \mathbb{R}^n$ is the prox of a convex function if and only if:

  a) p is non-expansive, *i.e.*,
  $\forall\, \mathbf{x}, \mathbf{x}',\ \|p(\mathbf{x}) - p(\mathbf{x}')\| \leq \|\mathbf{x} - \mathbf{x}'\|$

  b) and p is subgradient of a convex function, *i.e.*,
  $\exists\, \phi : \mathbb{R}^n \to \mathbb{R} :\ p(\mathbf{x}) \in \partial\phi(\mathbf{x}),\ \forall\, \mathbf{x}$

Convergence

- PnP-ADMM with a patch-based GMM-MMSE denoiser

$$\mathbf{x}_{k+1} = \left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{A}^T\mathbf{y} + \rho(\mathbf{z}_k + \mathbf{u}_k)\right)$$
$$\mathbf{z}_{k+1} = \text{denoiser}\left(\mathbf{x}_{k+1} - \mathbf{u}_k, 1/\rho\right)$$
$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1} - \mathbf{x}_{k+1} + \mathbf{z}_{k+1}$$

- denoiser is the prox of a convex function $\Rightarrow$ convergence.

- From Moreau [1965]: some map $p : \mathbb{R}^n \to \mathbb{R}^n$ is the prox of a convex function if and only if:

  a) p is non-expansive, *i.e.*,
  $\forall\, \mathbf{x}, \mathbf{x}',\ \|p(\mathbf{x}) - p(\mathbf{x}')\| \leq \|\mathbf{x} - \mathbf{x}'\|$

  b) and p is subgradient of a convex function, *i.e.*,
  $\exists\, \phi : \mathbb{R}^n \to \mathbb{R} :\ p(\mathbf{x}) \in \partial\phi(\mathbf{x}),\ \forall\, \mathbf{x}$

- Does the patch-based GMM-MMSE denoiser satisfy these conditions?

Convergence (2)

- Is the patch-based GMM-MMSE denoiser non-expansive?

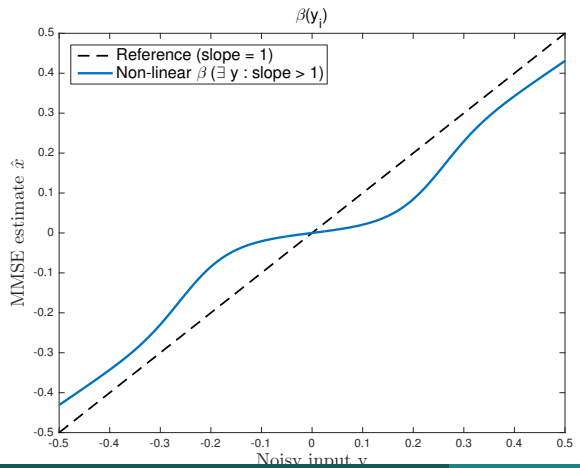
Convergence (2)

- Is the patch-based GMM-MMSE denoiser non-expansive?
- No! A simple univariate counter-example:
  - ✓ Spike-and-slab-type prior:
    $p(x) = \frac{1}{2}\mathcal{N}(x; 0, \tau_1) + \frac{1}{2}\mathcal{N}(x; 0, \tau_2), \quad \tau_2 \gg \tau_1$

Convergence (2)

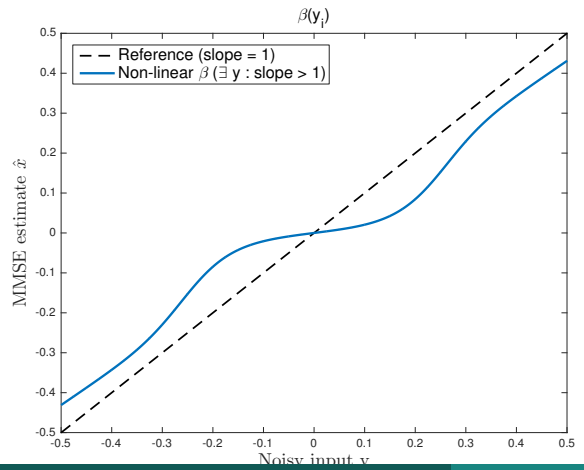
- Is the patch-based GMM-MMSE denoiser non-expansive?
- No! A simple univariate counter-example:
  - ✓ Spike-and-slab-type prior:
    $p(x) = \frac{1}{2}\mathcal{N}(x; 0, \tau_1) + \frac{1}{2}\mathcal{N}(x; 0, \tau_2), \quad \tau_2 \gg \tau_1$
  - ✓ MMSE estimate under Gaussian noise of unit variance:

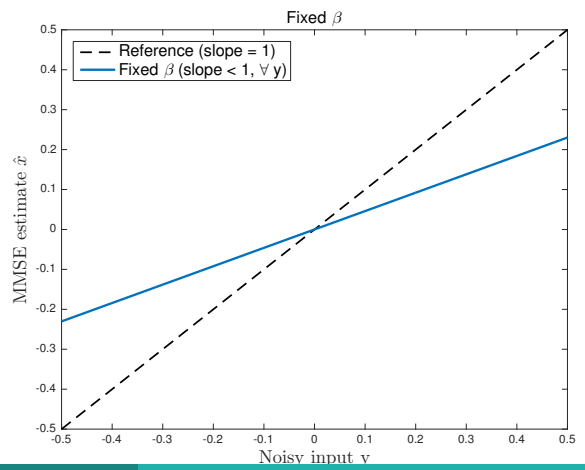
$$\hat{x} = \mathbb{E}[X|y] = \frac{\frac{\tau_1\, y}{\tau_1+1}\, \beta_1(y) + \frac{\tau_2\, y}{\tau_2+1}\, \beta_2(y)}{\beta_1(y) + \beta_2(y)}, \qquad \text{where } \beta_i(y) = \mathcal{N}(y; 0, \tau_i+1)$$



Convergence (2)

- Is the patch-based GMM-MMSE denoiser non-expansive?
- No! A simple univariate counter-example:
  - ✓ Spike-and-slab-type prior:
    $p(x) = \frac{1}{2}\mathcal{N}(x; 0, \tau_1) + \frac{1}{2}\mathcal{N}(x; 0, \tau_2), \quad \tau_2 \gg \tau_1$
  - ✓ MMSE estimate under Gaussian noise of unit variance:
    $$\hat{x} = \mathbb{E}[X|y] = \frac{\frac{\tau_1\, y}{\tau_1+1}\, \beta_1(y) + \frac{\tau_2\, y}{\tau_2+1}\, \beta_2(y)}{\beta_1(y) + \beta_2(y)}, \qquad \text{where } \beta_i(y) = \mathcal{N}(y; 0, \tau_i+1)$$
- With β_i fixed: $\hat{x} = y\big(\beta_1 \frac{\tau_1}{\tau_1+1} + \beta_2 \frac{\tau_2}{\tau_2+1}\big)/(\beta_1 + \beta_2)$





Convergence (2)

- Freeze the weights (β_m) after a certain number of iterations.

Convergence (2)

- Freeze the weights (β_m) after a certain number of iterations.
- Patch estimate:

$$\hat{\mathbf{x}}_i = \sum_{m=1}^{K} \beta_m^i \, \mathbf{C}_m \Big(\mathbf{C}_m + \sigma^2 \, \mathbf{I} \Big)^{-1} \mathbf{y}_i$$

Convergence (2)

- Freeze the weights (β_m) after a certain number of iterations.
- Patch estimate:

$$\hat{\mathbf{x}}_i = \sum_{m=1}^{K} \beta_m^i \, \mathbf{C}_m \Big(\mathbf{C}_m + \sigma^2 \, \mathbf{I} \Big)^{-1} \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{y}_i$$

Convergence (2)

- Freeze the weights (β_m) after a certain number of iterations.
- Patch estimate:

$$\hat{\mathbf{x}}_i = \sum_{m=1}^{K} \beta_m^i \, \mathbf{C}_m \Big(\mathbf{C}_m + \sigma^2 \, \mathbf{I} \Big)^{-1} \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{P}_i \, \mathbf{y}$$

$\mathbf{P}_i$ is the operator (binary matrix) that extracts the i-th patch

Convergence (2)

- Freeze the weights (β_m) after a certain number of iterations.
- Patch estimate:

$$\hat{\mathbf{x}}_i = \sum_{m=1}^{K} \beta_m^i \, \mathbf{C}_m \Big(\mathbf{C}_m + \sigma^2 \, \mathbf{I} \Big)^{-1} \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{P}_i \, \mathbf{y}$$

$\mathbf{P}_i$ is the operator (binary matrix) that extracts the i-th patch

(weights are normalized, to simplify the notation: $\beta_m^i \leftarrow \beta_m^i / \sum_j \beta_j^i$)

Convergence (2)

- Freeze the weights (β_m) after a certain number of iterations.
- Patch estimate:

$$\hat{\mathbf{x}}_i = \sum_{m=1}^{K} \beta_m^i \, \mathbf{C}_m \Big(\mathbf{C}_m + \sigma^2 \, \mathbf{I} \Big)^{-1} \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{P}_i \, \mathbf{y}$$

$\mathbf{P}_i$ is the operator (binary matrix) that extracts the i-th patch

(weights are normalized, to simplify the notation: $\beta_m^i \leftarrow \beta_m^i / \sum_j \beta_j^i$)

- Global image estimate: aggregate the patch estimates:

$$\hat{\mathbf{x}} = \frac{1}{n_p} \sum_{i=1}^{N} \mathbf{P}_i^T \mathbf{F}_i(\sigma^2) \mathbf{P}_i \, \mathbf{y} = \mathbf{W}(\sigma^2) \, \mathbf{y}$$

Convergence (2)

- Freeze the weights (β_m) after a certain number of iterations.
- Patch estimate:

$$\hat{\mathbf{x}}_i = \sum_{m=1}^{K} \beta_m^i \, \mathbf{C}_m \Big(\mathbf{C}_m + \sigma^2 \, \mathbf{I} \Big)^{-1} \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{y}_i = \mathbf{F}_i(\sigma^2) \, \mathbf{P}_i \, \mathbf{y}$$

  $\mathbf{P}_i$ is the operator (binary matrix) that extracts the i-th patch

  (weights are normalized, to simplify the notation: $\beta_m^i \leftarrow \beta_m^i / \sum_j \beta_j^i$)

- Global image estimate: aggregate the patch estimates:

$$\hat{\mathbf{x}} = \frac{1}{n_p} \sum_{i=1}^{N} \mathbf{P}_i^T \mathbf{F}_i(\sigma^2) \mathbf{P}_i \, \mathbf{y} = \mathbf{W}(\sigma^2) \, \mathbf{y}$$

- Key properties of $\mathbf{W}$ [Teodoro et al., 2017b]: for any $\sigma^2 > 0$,

$$\mathbf{W}(\sigma^2) = \mathbf{W}(\sigma^2)^T, \qquad \mathbf{W}(\sigma^2) \succeq 0, \qquad \lambda_{\max}\big(\mathbf{W}(\sigma^2)\big) < 1$$

Convergence (2)

- Freezing the weights (β_m) after a certain number of iterations,

$$\text{denoiser}(\mathbf{y}, \sigma^2) = \mathbf{W}(\sigma^2)\mathbf{y}$$

Convergence (2)

- Freezing the weights (β_m) after a certain number of iterations,

$$\text{denoiser}(\mathbf{y}, \sigma^2) = \mathbf{W}(\sigma^2)\mathbf{y}$$

- Recalling Moreau's corollary, this is a proximity operator:
  - It is non-expansive: $\mathbf{W}(\sigma^2)$ is symmetric with $\lambda_{\text{max}}\big(\mathbf{W}(\sigma^2)\big) < 1$
  - It is the gradient of a convex function: $\mathbf{W}(\sigma^2)\mathbf{y} = \nabla_{\mathbf{y}}\big(\frac{1}{2}\mathbf{y}^T\mathbf{W}(\sigma^2)\mathbf{y}\big)$

Convergence (2)

- Freezing the weights (β_m) after a certain number of iterations,

$$\text{denoiser}(\mathbf{y}, \sigma^2) = \mathbf{W}(\sigma^2)\mathbf{y}$$

- Recalling Moreau's corollary, this is a proximity operator:
  - It is non-expansive: $\mathbf{W}(\sigma^2)$ is symmetric with $\lambda_{\text{max}}\big(\mathbf{W}(\sigma^2)\big) < 1$
  - It is the gradient of a convex function: $\mathbf{W}(\sigma^2)\mathbf{y} = \nabla_{\mathbf{y}}\big(\frac{1}{2}\mathbf{y}^T\mathbf{W}(\sigma^2)\mathbf{y}\big)$
- Can we identify the function of which this denoiser is the prox?

$$\phi(\mathbf{x}) = \iota_{S(\mathbf{W})}(\mathbf{x}) + \frac{1}{2}\mathbf{x}^T\bar{\mathbf{Q}}(\bar{\Lambda}^{-1} - \mathbf{I})\bar{\mathbf{Q}}^T\mathbf{x}$$

where $S(\mathbf{W})$ is the column span of $\mathbf{W}$, $\bar{\Lambda}$ has the positive eigenvalues of $\mathbf{W}$, and $\bar{\mathbf{Q}}$ the corresponding eigenvectors.

Convergence (2)

- Freezing the weights (β_m) after a certain number of iterations,

$$\text{denoiser}(\mathbf{y}, \sigma^2) = \mathbf{W}(\sigma^2)\mathbf{y}$$

- Recalling Moreau's corollary, this is a proximity operator:
  - It is non-expansive: $\mathbf{W}(\sigma^2)$ is symmetric with $\lambda_{\text{max}}\big(\mathbf{W}(\sigma^2)\big) < 1$
  - It is the gradient of a convex function: $\mathbf{W}(\sigma^2)\mathbf{y} = \nabla_{\mathbf{y}}\big(\frac{1}{2}\mathbf{y}^T\mathbf{W}(\sigma^2)\mathbf{y}\big)$
- Can we identify the function of which this denoiser is the prox?

$$\phi(\mathbf{x}) = \iota_{S(\mathbf{W})}(\mathbf{x}) + \frac{1}{2}\mathbf{x}^T\bar{\mathbf{Q}}(\bar{\Lambda}^{-1} - \mathbf{I})\bar{\mathbf{Q}}^T\mathbf{x}$$

  where $S(\mathbf{W})$ is the column span of $\mathbf{W}$, $\bar{\Lambda}$ has the positive eigenvalues of $\mathbf{W}$, and $\bar{\mathbf{Q}}$ the corresponding eigenvectors.

- Conclusion: the problem has a solution and PnP-ADMM converges

Hyperspectral Fusion: Synthetic Example

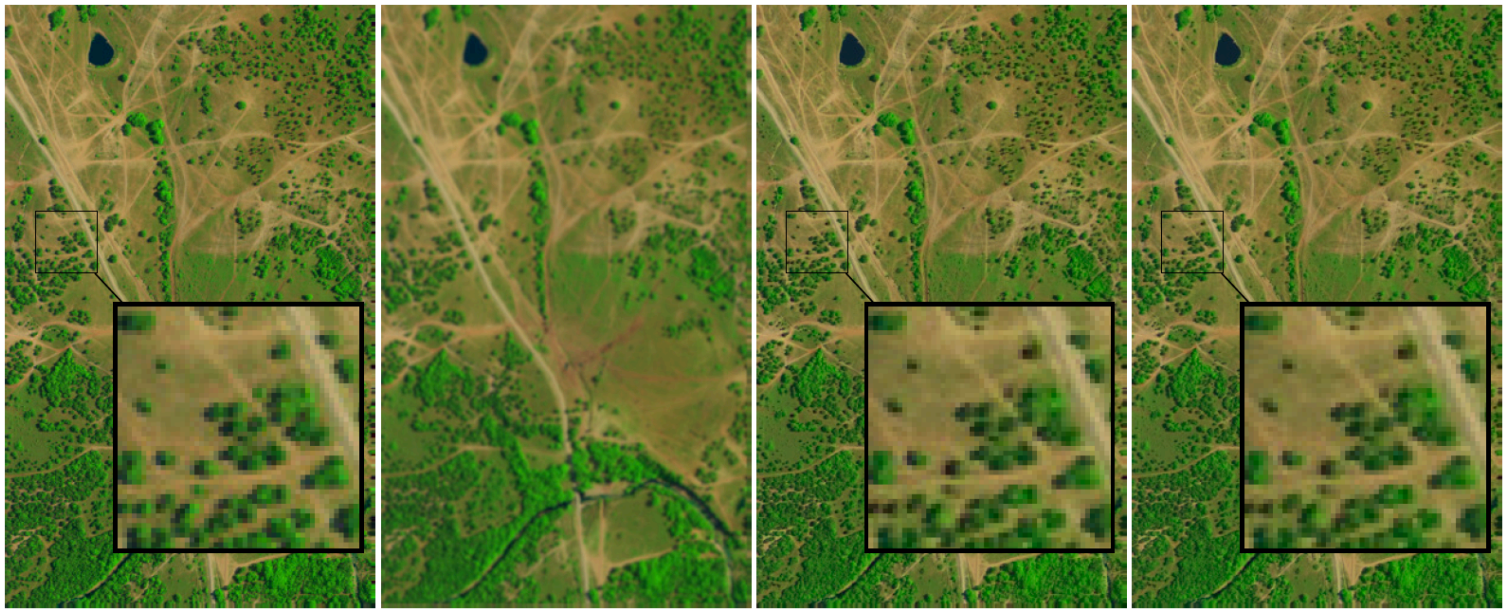
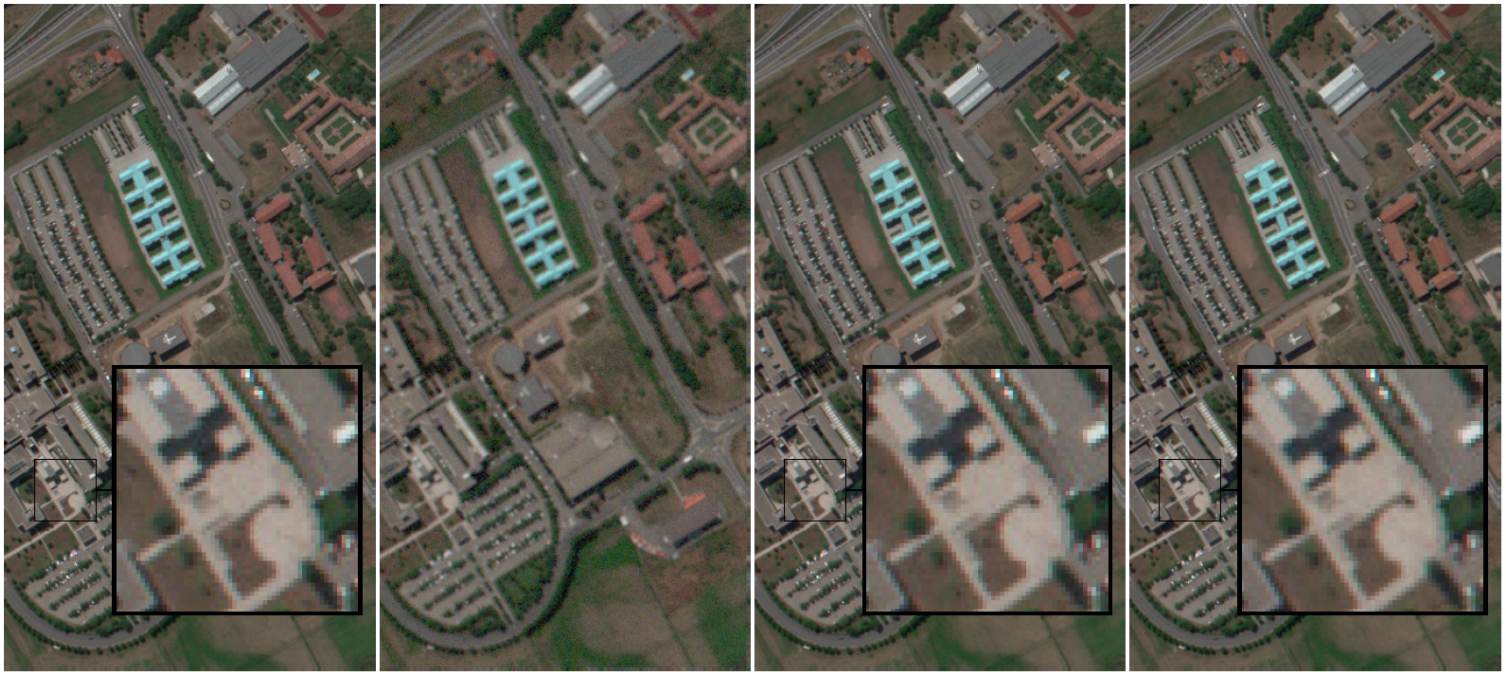


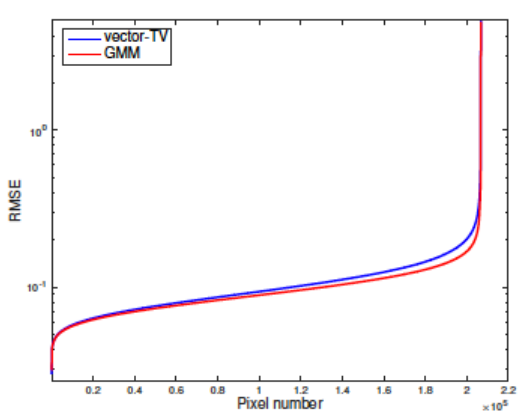
Table 3: HS and MS fusion on RTerrain dataset.

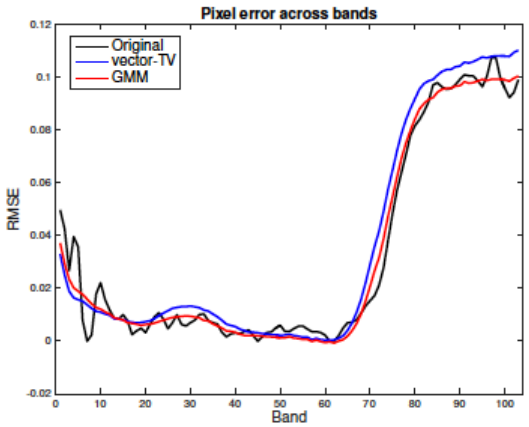
	Exp. 1 (PAN)			Exp. 2 (PAN)			Exp. 3 (R,G,B,N-IR)			Exp. 4 (R,G,B,N-IR)		
SNR ($\mathbf{Y}_m$)	50dB			30dB			50dB			30dB		
SNR ($\mathbf{Y}_h$)	50dB			20dB			50dB			20dB		
Metric	ERGAS	SAM	SRE	ERGAS	SAM	SRE	ERGAS	SAM	SRE	ERGAS	SAM	SRE
HySure	2.62	5.34	21.46	2.77	5.35	20.86	1.08	2.68	28.71	1.53	3.42	26.07
Proposed	2.58	**5.15**	**21.69**	**2.75**	**5.33**	**21.12**	**0.91**	**2.20**	**30.86**	**1.29**	**2.85**	**27.85**
ADMM-BM3D	**2.57**	5.17	21.65	2.76	5.36	21.08	0.93	2.22	30.80	1.31	2.91	27.72

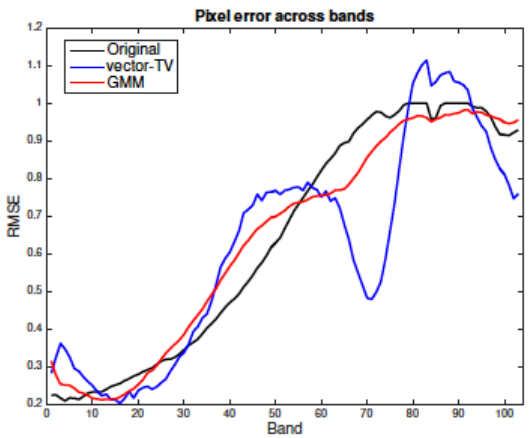
[Teodoro et al., 2016a]

Hyperspectral Fusion: Synthetic Example









Final Remarks

- ADMM/SALSA: a flexible toolbox for a variety inverse problems

Final Remarks

- ADMM/SALSA: a flexible toolbox for a variety inverse problems
- Its speed hinges on the inversion of $(\mathbf{B}^T\mathbf{B} + \mathbf{I})$ (à la quasi-Newton)

Final Remarks

- ADMM/SALSA: a flexible toolbox for a variety inverse problems
- Its speed hinges on the inversion of $(\mathbf{B}^T\mathbf{B} + \mathbf{I})$ (à la quasi-Newton)
- Plug-and-play (PnP) denoisers "can" be used with ADMM

Final Remarks

- ADMM/SALSA: a flexible toolbox for a variety inverse problems
- Its speed hinges on the inversion of $(\mathbf{B}^T\mathbf{B} + \mathbf{I})$ (à la quasi-Newton)
- Plug-and-play (PnP) denoisers "can" be used with ADMM
- Convergence properties of PnP-ADMM with fixed linear denoiser

Thank you.

References I

M.V. Afonso, J.M. Bioucas-Dias, and M.A.T. F. An augmented lagrangian approach to the constrained optimization formulation of imaging inverse problems. *IEEE Transactions on Image Processing*, 20:681–695, 2011.

M. Almeida and M. F. Deconvolving images with unknown boundaries using the alternating direction method of multipliers. *IEEE Transactions on Image Processing*, 22:3074–3086, 2013a.

M. Almeida and M. F. Blind image deblurring with unknown boundary conditions using the alternating direction method of multipliers. In *IEEE International Conf. on Image Processing*, 2013b.

T. Aspelmeier, C. Charitha, and D. R. Luke. Local linear convergence of the ADMM/Douglas-Rachford algorithms without strong convexity and application to statistical imaging. *SIAM J. Imaging Science*, 2016. to appear.

H. Attouch, P. Redont, and A. Soubeyran. A new class of alternating proximal minimization algorithms with costs to move. *SIAM Journal on Optimization*, 18:1061–1081, 2007.

H. H. Bauschke, M. N. Dao, D. Noll, and H. M. Phan. On Slater's condition and finite convergence of the Douglas-Rachford algorithm for solving convex feasibility problems in Euclidean spaces. *Journal of Global Optimization*, pages 1–21, 2015.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3:1–122, 2011.

References II

S. Chan, X. Wang, and O. Elgendy. Plug-and-play ADMM for image restoration: Fixed point convergence and applications. *IEEE Transactions on Computational Imaging*, 3:in press, 2017.

T. Chan, A. Yip, and F. Park. Simultaneous total variation image inpainting and blind deconvolution. *International Journal of Imaging Systems Technology*, 15:92–102, 2005.

P.L. Combettes and J.C. Pesquet. Proximal splitting methods in signal processing. *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212, 2011.

K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16: 2080–2095, 2007.

A. Danielyan, V. Katkovnik, and K. Egiazarian. BM3D frames and variational image deblurring. *IEEE Transactions on Image Processing*, 21:1715–1728, 2012.

D. Davis and W. Yin. Convergence rates of relaxed Peaceman-Rachford and ADMM under regularity assumptions. *arXiv:1407.5210*, 2014.

W. Deng and W. Yin. On the global and linear convergence of generalized alternating direction method of multipliers. *DTIC Technical Report*, 2012.

J. Eckstein and D. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 5:293–318, 1992.

M. F and J. Bioucas-Dias. Restoration of Poissonian images using alternating direction optimization. *IEEE Transactions on Image Processing*, 19:3133–3145, 2010.

References III

D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximations. *Computers and Mathematics with Applications*, 2:17–40, 1976.

R. Glowinski and A. Marrocco. Sur l'approximation, par elements finis d'ordre un, et la resolution, par penalisation-dualité, d'une classe de problemes de Dirichlet non lineares. *Revue Française d'Automatique, Informatique et Recherche Opérationelle*, 9:41–76, 1975.

T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7:1588–1623, 2014.

M. Lebrun, A. Buades, and J. M. Morel. A nonlocal Bayesian image denoising algorithm. *SIAM Journal on Imaging Science*, 6:1665–1688, 2013.

J. Liang, M. J. Fadili, G. PeyrÃ©, and R. Luke. Activity identification and local linear convergence of Douglas-Rachford/ADMM under partial smoothness. In *Scale Space and Variational Methods in Computer Vision*, pages 642–653. Springer, 2015.

J. J. Moreau. ProximitÃ© et dualtitÃ© dans un espace hilbertien. *Bulletin de la SociÃ©tÃ© MathÃ©matique de France*, 93:273–299, 1965.

R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. Jordan. A general analysis of the convergence of ADMM. In *International Conference on Machine Learning*, pages 343–352, 2015.

References IV

J. Pan, Z. Hu, Z. Su, and M. Yang. Deblurring text images via ℓ_0-regularized intensity and gradient prior. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

P. Patrinos, L. Stella, and A. Bemporad. Douglas-Rachford splitting: Complexity estimates and accelerated variants. In *IEEE Conference on Decision and Control*, pages 4234–4239, 2014.

A. Matakos S. Ramani and J. Fessler. Accelerated edge-preserving image restoration without boundary artifacts. *IEEE Transactions on Image Processing*, 22:2019–2029, 2013.

S. Reeves. Fast image restoration without boundary artifacts. *IEEE Transactions on Image Processing*, 14:1448–1453, 2005.

S. Setzer, G. Steidl, and T. Teuber. Deblurring Poissonian images by split Bregman techniques. *Journal of Visual Communication and Image Representation*, 21:193–199, 2010.

M. Simões, J. Bioucas-Dias, L. Almeida, and J. Chanussot. A convex formulation for hyperspectral image superresolution via subspace-based regularization. *IEEE Trans. Geoscience and Remote Sensing*, 55:3373–3388, 2015.

S. Sreehari, S. Venkatakrishnan, B. Wohlberg, G. Buzzard, L. Drummy, J. Simmons, and A. Bouman. Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, 2:408–423, 2016.

A. Teodoro, M. Almeida, and M. Figueiredo. Single-frame image denoising and inpainting using Gaussian mixtures. In *4th International Conference on Pattern Recognition Applications and Methods*, 2015.

References V

A. Teodoro, J. Bioucas-Dias, and M. F. Sharpening hyperspectral images using plug-and-play priors. submitted, 2016a.

A. Teodoro, J. Bioucas-Dias, and M. Figueiredo. Image restoration and reconstruction using variable splitting and class-adapted image priors. In *IEEE International Conference on Image Processing*, 2016b.

A. Teodoro, J. Bioucas-Dias, and M. Figueiredo. Hyperspectral sharpening using class-adapted Gaussian mixture priors, 2017a. submitted.

A. Teodoro, J. Bioucas-Dias, and M. Figueiredo. Scene-adapted plug-and-play with convergence guarantees. In *IEEE International Workshop on Machine Learning for Signal Processing*, 2017b.

S. Venkatakrishnan, C. Bouman, E. Chu, and B. Wohlberg. Plug-and-play priors for model based reconstruction. In *IEEE Global Conference on Signal and Information Processing*, pages 945–948, 2013.

Z. Xu, M. Figueiredo, and T. Goldstein. Adaptive ADMM with spectral penalty parameter selection. *arXiv:1605.07246*, 2016.

D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *International Conference on Computer Vision*, pages 479–486, 2011.