Sparse and Low-Rank Representation for Biometrics – Lecture III: Sparse Optimization and Numerical Implementation

Allen Y. Yang
Department of EECS, UC Berkeley
yang@eecs.berkeley.edu

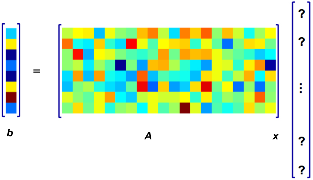
with John Wright and Yi Ma

ICB 2013 Tutorial

Two Previous Sparse Optimization Problems

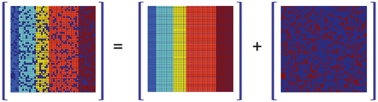
- ℓ_1**-min** seeks sparse solution in **underdetermined system** (A in general is full rank):

$$\min \|\mathbf{x}\|_1 \quad \text{subj. to} \quad \mathbf{b} = A\mathbf{x} \quad \text{where } A \in \mathbb{R}^{d\times n}, (d < n)$$



- **Robust PCA** seeks sparse and low-rank decomposition:

$$\min \|A\|_* + \lambda\|E\|_1 \quad \text{subj. to} \quad D = A + E \in \mathbb{R}^{m\times n}.$$



Efficient sparse optimization is challenging

- Generic second-order method toolboxes do exist: **CVX**.

Efficient sparse optimization is challenging

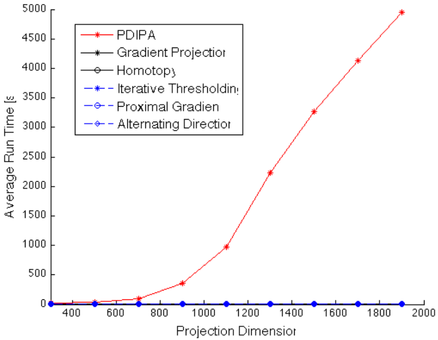
- Generic second-order method toolboxes do exist: **CVX**.
- However, standard interior-point methods are **very expensive** in HD space.

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{1}^T\mathbf{x} \\ \text{subj. to} \quad & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

The KKT condition:

$$\nabla f(\mathbf{x}^*) + \mu \nabla g(\mathbf{x}^*) + \lambda \nabla h(\mathbf{x}^*) = 0$$

Complexity: $O(n^3)$



- **Robust PCA:** CVX can solve smaller than 80×80 matrices on typical PC. Complexity bound: $O(n^6)$.

Sparse Optimization Literature: ℓ_1-Minimization

1. **Interior-point methods**
   - Log-Barrier [Frisch '55, Karmarkar '84, Megiddo '89, Monteiro-Adler '89, Kojima-Megiddo-Mizuno '93]
2. **Homotopy**
   - Homotopy [Osborne-Presnell-Turlach '00, Malioutov-Cetin-Willsky '05, Donoho-Tsaig '06]
   - Polytope Faces Pursuit (PFP) [Plumbley '06]
   - Least Angle Regression (LARS) [Efron-Hastie-Johnstone-Tibshirani '04]
3. **Gradient Projection**
   - Gradient Projection Sparse Representation (GPSR) [Figueiredo-Nowak-Wright '07]
   - Truncated Newton Interior-Point Method (TNIPM) [Kim-Koh-Lustig-Boyd-Gorinevsky '07]
4. **Iterative Soft-Thresholding**
   - Soft Thresholding [Donoho '95]
   - Sparse Reconstruction by Separable Approximation (SpaRSA) [Wright-Nowak-Figueiredo '08]
5. **Accelerated Proximal Gradient** [Nesterov '83, Nesterov '07]
   - FISTA [Beck-Teboulle '09]
   - Nesterov's Method (NESTA) [Becker-Bobin-Candés '09]
6. **Augmented Lagrangian Methods** [Bertsekas '82]
   - Bergman [Yin et al. '08]
   - SALSA [Figueiredo et al. '09]
   - Primal ALM, Dual ALM [AY et al '10]
7. **Alternating Direction Method of Multipliers**
   - YALL1 [Yang-Zhang '09]

Sparse Optimization Literature: Robust PCA

1. **Interior-point methods** [Candès-Li-Ma-Wright '09]
2. **Iterative Soft-Thresholding**
   - Singular Value Thresholding [Cai-Candès-Shen '09, Ma-Goldfarb-Chen '09]
3. **Accelerated Proximal Gradient** [Nesterov '83, Nesterov '07]
   - Accelerated Proximal Gradient [Toh-Yun '09, Ganesh-Lin-Ma-Wu-Wright '09]
4. **Augmented Lagrangian Methods** [Bertsekas '82]
   - ALM for Robust PCA [Lin-Chen-Ma '11]
5. **Alternating Direction Method of Multipliers** [Gabay-Mercier '76]
   - Principal Component Pursuit [Yuan-Yang '09, Candès-Li-Ma-Wright '09]

Outline

- First-Order Sparse Optimization Methods
  1. Iterative Soft-Thresholding (IST)
  2. Accelerated Proximal Gradient (APG)
  3. Augmented Lagrange Multipliers (ALM)
  4. Alternating Direction Methods of Multipliers (ADMM)
- Extensions

Problem Formulation

- Consider $F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$
  - **Approximate ℓ_1-min:**

$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{b} - A\mathbf{x}\|^2, \quad g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1.$$

  - **Robust PCA:**

$$f(\mathbf{x}) = \frac{\mu}{2}\|D - A - E\|_F^2, \quad g(\mathbf{x}) = \|A\|_* + \lambda\|E\|_1.$$

$f(\mathbf{x})$ is smooth, convex, and has Lipschitz continuous gradients:

$$\exists L \geq 0, \forall \mathbf{x}_1, \mathbf{x}_2 \in \Omega, \|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|$$

$g(\mathbf{x})$ can be a nonsmooth, nonconvex function.

Problem Formulation

- Consider $F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$
  - **Approximate ℓ_1-min:**
$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{b} - A\mathbf{x}\|^2, \quad g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1.$$
  - **Robust PCA:**
$$f(\mathbf{x}) = \frac{\mu}{2}\|D - A - E\|_F^2, \quad g(\mathbf{x}) = \|A\|_* + \lambda\|E\|_1.$$

$f(\mathbf{x})$ is smooth, convex, and has Lipschitz continuous gradients:

$$\exists L \geq 0, \forall \mathbf{x}_1, \mathbf{x}_2 \in \Omega, \|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|$$

$g(\mathbf{x})$ can be a nonsmooth, nonconvex function.

Generic Composite Objective Function in Sparse Optimization:

$$F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$$

In general, solving a nonsmooth, nonconvex objective function is difficult with weak convergence guarantees.

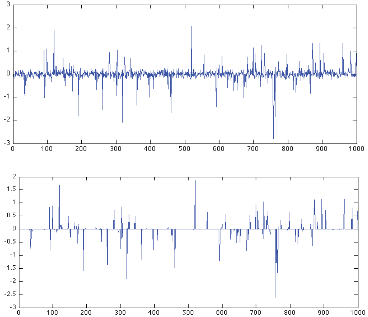
Soft-Thresholding: Special structure leads to efficient proximal operator

- Soft-thresholding function [Moreau '62, Donoho '95]:
  When $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$, $\lambda > 0$,

$$\arg\min_{\mathbf{x}} \lambda\|\mathbf{x}\|_1 + \frac{1}{2}\|\mathbf{u} - \mathbf{x}\|^2$$

  has a closed-form solution *element-wise*: (denoising)

$$x_i^* = \text{soft}(u_i, \lambda) \doteq \text{sign}(u_i) \cdot \max\{|u_i| - \lambda, 0\}.$$



Solving ℓ_1-Min via Iterative Soft-Thresholding

- Approximate ℓ_1-min objective function at $\mathbf{x}_k$:

$$\begin{aligned} F(\mathbf{x}) &= f(\mathbf{x}) + g(\mathbf{x}) = \frac{1}{2}\|\mathbf{b} - A\mathbf{x}\|^2 + \lambda\|\mathbf{x}\|_1 \\ &= f(\mathbf{x}_k) + (\mathbf{x} - \mathbf{x}_k)^T \nabla f(\mathbf{x}_k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_k)^T \nabla^2 f(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + g(\mathbf{x}) \\ &\approx f(\mathbf{x}_k) + (\mathbf{x} - \mathbf{x}_k)^T \nabla f(\mathbf{x}_k) + \frac{L}{2}\|\mathbf{x} - \mathbf{x}_k\|^2 + g(\mathbf{x}) \end{aligned}$$

where $\nabla^2 f(\mathbf{x}_k)$ is approximated by a diagonal matrix $L \cdot I$.

Solving ℓ_1-Min via Iterative Soft-Thresholding

- Approximate ℓ_1-min objective function at $\mathbf{x}_k$:

$$\begin{aligned} F(\mathbf{x}) &= f(\mathbf{x}) + g(\mathbf{x}) = \frac{1}{2}\|\mathbf{b} - A\mathbf{x}\|^2 + \lambda\|\mathbf{x}\|_1 \\ &= f(\mathbf{x}_k) + (\mathbf{x} - \mathbf{x}_k)^T \nabla f(\mathbf{x}_k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_k)^T \nabla^2 f(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + g(\mathbf{x}) \\ &\approx f(\mathbf{x}_k) + (\mathbf{x} - \mathbf{x}_k)^T \nabla f(\mathbf{x}_k) + \frac{L}{2}\|\mathbf{x} - \mathbf{x}_k\|^2 + g(\mathbf{x}) \end{aligned}$$

  where $\nabla^2 f(\mathbf{x}_k)$ is approximated by a diagonal matrix $L \cdot I$.

- Minimize $F(\mathbf{x})$ via iteration:

$$\begin{aligned} \mathbf{x}_{k+1} &= \min_{\mathbf{x}} F(\mathbf{x}) \\ &\approx \min_{\mathbf{x}} \left\{ (\mathbf{x} - \mathbf{x}_k)^T \nabla f(\mathbf{x}_k) + \frac{L}{2}\|\mathbf{x} - \mathbf{x}_k\|^2 + g(\mathbf{x}) \right\} \\ &= \min_{\mathbf{x}} \left\{ \|\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k) - \mathbf{x}\|^2 + \frac{\lambda}{L}\|\mathbf{x}\|_1 \right\} \end{aligned}$$

Solving ℓ_1-Min via Iterative Soft-Thresholding

- Approximate ℓ_1-min objective function at $\mathbf{x}_k$:

$$
\begin{aligned}
F(\mathbf{x}) &= f(\mathbf{x}) + g(\mathbf{x}) = \tfrac{1}{2}\|\mathbf{b} - A\mathbf{x}\|^2 + \lambda\|\mathbf{x}\|_1 \\
&= f(\mathbf{x}_k) + (\mathbf{x}-\mathbf{x}_k)^T\nabla f(\mathbf{x}_k) + \tfrac{1}{2}(\mathbf{x}-\mathbf{x}_k)^T\nabla^2 f(\mathbf{x}_k)(\mathbf{x}-\mathbf{x}_k) + g(\mathbf{x}) \\
&\approx f(\mathbf{x}_k) + (\mathbf{x}-\mathbf{x}_k)^T\nabla f(\mathbf{x}_k) + \tfrac{L}{2}\|\mathbf{x}-\mathbf{x}_k\|^2 + g(\mathbf{x})
\end{aligned}
$$

  where $\nabla^2 f(\mathbf{x}_k)$ is approximated by a diagonal matrix $L \cdot I$.

- Minimize $F(\mathbf{x})$ via iteration:

$$
\begin{aligned}
\mathbf{x}_{k+1} &= \min_{\mathbf{x}} F(\mathbf{x}) \\
&\approx \min_{\mathbf{x}} \left\{(\mathbf{x}-\mathbf{x}_k)^T\nabla f(\mathbf{x}_k) + \tfrac{L}{2}\|\mathbf{x}-\mathbf{x}_k\|^2 + g(\mathbf{x})\right\} \\
&= \min_{\mathbf{x}} \left\{\|\mathbf{x}_k - \tfrac{1}{L}\nabla f(\mathbf{x}_k) - \mathbf{x}\|^2 + \tfrac{\lambda}{L}\|\mathbf{x}\|_1\right\}
\end{aligned}
$$

ℓ_1-Min via IST

- **Update rule:** $\mathbf{x}_{k+1} = \text{soft}\left(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k), \frac{\lambda}{L}\right)$.
- **Complexity:** Most expensive operation is $\nabla f(\mathbf{x}) = A^T A\mathbf{x} - A^T b = O(dn)$.
- **Rate of convergence:** $F(\mathbf{x}_k) - F(\mathbf{x}^*) \leq \frac{L_f\|\mathbf{x}_0 - \mathbf{x}^*\|^2}{2k} = O(\frac{1}{k})$ (Linear).

IST: Pros and Cons

- **Strong points:**
  1. Avoid expensive matrix factorization and inverse in interior-point methods.
  2. Efficient inter-loop proximal operator: $\text{soft}(\mathbf{u}, \lambda)$.
  3. Scalable, easy to parallelize, i.e., matrix-vector product.

IST: Pros and Cons

- **Strong points:**
  1. Avoid expensive matrix factorization and inverse in interior-point methods.
  2. Efficient inter-loop proximal operator: soft($\mathbf{u}, \lambda$).
  3. Scalable, easy to parallelize, i.e., matrix-vector product.
- **Weak points:**

Linear rate of convergence translates to many iterations to converge.

Question: Can we improve the rate of convergence while still using first-order methods?

Accelerated Proximal Gradient Method

Theorem [Nesterov '83]

If f is differentiable with Lipschitz continuous gradient

$$\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|,$$

there exists a first-order algorithm with quadratic convergence rate $O(\frac{1}{k^2})$ in function values.

(even earlier than [Karmarkar '84] for solving linear programs)

Accelerated Proximal Gradient Method

Theorem [Nesterov '83]

If f is differentiable with Lipschitz continuous gradient

$$\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|,$$

there exists a first-order algorithm with quadratic convergence rate $O(\frac{1}{k^2})$ in function values.

(even earlier than [Karmarkar '84] for solving linear programs)
Procedure:

- Construct a quadratic upper bound $Q_L(\mathbf{x}, \mathbf{w}) \geq f(\mathbf{x})$:

$$Q_{L_f}(\mathbf{x}, \mathbf{w}) = f(\mathbf{w}) + (\mathbf{x} - \mathbf{w})^T \nabla f(\mathbf{w}) + \frac{L}{2}\|\mathbf{x} - \mathbf{w}\|^2.$$

- $\mathbf{w}_k$ is called *proximal point*.
  If $\mathbf{w}_k = \mathbf{x}_k$, and $\mathbf{x}_{k+1} = \arg\min Q_{L_f}(\mathbf{x}, \mathbf{x}_k) + g(\mathbf{x})$
  $\Rightarrow$ Same IST update rule leads to linear convergence rate.

Accelerated Proximal Gradient Method

Theorem [Nesterov '83]

If f is differentiable with Lipschitz continuous gradient

$$\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|,$$

there exists a first-order algorithm with quadratic convergence rate $O(\frac{1}{k^2})$ in function values.

(even earlier than [Karmarkar '84] for solving linear programs)
Procedure:

- Construct a quadratic upper bound $Q_L(\mathbf{x}, \mathbf{w}) \geq f(\mathbf{x})$:

$$Q_{L_f}(\mathbf{x}, \mathbf{w}) = f(\mathbf{w}) + (\mathbf{x} - \mathbf{w})^T \nabla f(\mathbf{w}) + \frac{L}{2}\|\mathbf{x} - \mathbf{w}\|^2.$$

- $\mathbf{w}_k$ is called *proximal point*.
  If $\mathbf{w}_k = \mathbf{x}_k$, and $\mathbf{x}_{k+1} = \arg\min Q_{L_f}(\mathbf{x}, \mathbf{x}_k) + g(\mathbf{x})$
  $\Rightarrow$ Same IST update rule leads to linear convergence rate.
- A nonconventional update rule leads to **quadratic rate of convergence**:

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}, \quad \mathbf{w}_{k+1} = \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}_k - \mathbf{x}_{k-1}).$$

Accelerated Proximal Gradient Methods in Sparse Optimization

- Fast IST Algorithm (FISTA): [Beck-Teboulle '09]
  APG applies to **approximate ℓ_1-min function** with a fixed λ:

$$F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) \le Q(\mathbf{x}, \mathbf{w}) + \lambda \|\mathbf{x}\|_1.$$

FISTA

1. $\mathbf{x}_{k+1} = \text{soft}\left(\mathbf{w}_k - \frac{1}{L}\nabla f(\mathbf{w}_k), \frac{\lambda}{L}\right)$
2. $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
3. $\mathbf{w}_{k+1} = \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}_k - \mathbf{x}_{k-1})$

How does APG improve IST?

Strong points:

1. Retain efficient inter-loop complexity: Overhead of calculating proximal points is neglectable.
2. Dramatically reduces the number of iteration needed than IST as converges quadratically.

How does APG improve IST?

Strong points:

1. Retain efficient inter-loop complexity: Overhead of calculating proximal points is neglectable.
2. Dramatically reduces the number of iteration needed than IST as converges quadratically.

Weak points:

Continuation strategy in approximate solutions is inefficient

In $F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$, the equality constraint $h(\mathbf{x}) = 0$ is approximated by a quadratic penalty

$$\frac{1}{\lambda} f(\mathbf{x}) \doteq \frac{1}{2\lambda} \|h(\mathbf{x})\|^2 = \frac{1}{2\lambda} \|\mathbf{b} - A\mathbf{x}\|^2.$$

Equality can only be achieved when $\lambda \searrow 0$. This is called **the continuation technique**.

Question: Is there a better framework than continuation to approximate solutions?

Augmented Lagrangian Method (ALM)

- ℓ_1-Min:

$$\mathbf{x}^* = \arg\min \|\mathbf{x}\|_1 \quad \text{subj. to} \quad \mathbf{b} = A\mathbf{x}$$

(adding a **quadratic penalty term** for the equality constraint)

$$F_\mu(\mathbf{x}) = \|\mathbf{x}\|_1 + \frac{\mu}{2}\|\mathbf{b} - A\mathbf{x}\|_2^2 \quad \text{subj. to} \quad \mathbf{b} = A\mathbf{x}.$$

Augmented Lagrangian Method (ALM)

- ℓ_1-Min:

$$\mathbf{x}^* = \arg\min \|\mathbf{x}\|_1 \quad \text{subj. to} \quad \mathbf{b} = A\mathbf{x}$$

(adding a **quadratic penalty term** for the equality constraint)

$$F_\mu(\mathbf{x}) = \|\mathbf{x}\|_1 + \frac{\mu}{2}\|\mathbf{b} - A\mathbf{x}\|_2^2 \quad \text{subj. to} \quad \mathbf{b} = A\mathbf{x}.$$

- **Augmented Lagrange Function:** [Hestenes 69, Powell 69]

$$F_\mu(\mathbf{x}, \mathbf{y}) = \|\mathbf{x}\|_1 + \langle \mathbf{y}, \mathbf{b} - A\mathbf{x}\rangle + \frac{\mu}{2}\|\mathbf{b} - A\mathbf{x}\|_2^2,$$

where $\mathbf{y}$ is the Lagrange multipliers for the constraint $\mathbf{b} = A\mathbf{x}$.

Augmented Lagrangian Method (ALM)

- ℓ_1-Min:

$$\mathbf{x}^* = \arg\min \|\mathbf{x}\|_1 \quad \text{subj. to} \quad \mathbf{b} = A\mathbf{x}$$

(adding a **quadratic penalty term** for the equality constraint)

$$F_\mu(\mathbf{x}) = \|\mathbf{x}\|_1 + \frac{\mu}{2}\|\mathbf{b} - A\mathbf{x}\|_2^2 \quad \text{subj. to} \quad \mathbf{b} = A\mathbf{x}.$$

- **Augmented Lagrange Function:** [Hestenes 69, Powell 69]

$$F_\mu(\mathbf{x}, \mathbf{y}) = \|\mathbf{x}\|_1 + \langle \mathbf{y}, \mathbf{b} - A\mathbf{x}\rangle + \frac{\mu}{2}\|\mathbf{b} - A\mathbf{x}\|_2^2,$$

where $\mathbf{y}$ is the Lagrange multipliers for the constraint $\mathbf{b} = A\mathbf{x}$.

- **Method of Multipliers:** Given a fixed $\rho > 1$

$$\begin{aligned} \mathbf{x}_{k+1} &= \arg\min F_{\mu_k}(\mathbf{x}, \mathbf{y}_k) \\ \mathbf{y}_{k+1} &= \mathbf{y}_k + \mu_k h(\mathbf{x}_{k+1}) \\ \mu_{k+1} &= \rho \mu_k \end{aligned} \quad (1)$$

Convergence of ALM

Theorem [Bertsekas '03]

When optimize $F_\mu(\mathbf{x}, \mathbf{y})$ w.r.t. a sequence $\mu^k \to \infty$, and $\{\mathbf{y}^k\}$ is bounded, then the limit point of $\{\mathbf{x}^k\}$ is the global minimum with a **quadratic rate of convergence**: $F(\mathbf{x}_k) - F(\mathbf{x}^*) \leq O(1/k^2)$.

Convergence of ALM

Theorem [Bertsekas '03]

When optimize $F_\mu(\mathbf{x}, \mathbf{y})$ w.r.t. a sequence $\mu^k \to \infty$, and $\{\mathbf{y}^k\}$ is bounded, then the limit point of $\{\mathbf{x}^k\}$ is the global minimum with a **quadratic rate of convergence**: $F(\mathbf{x}_k) - F(\mathbf{x}^*) \leq O(1/k^2)$.

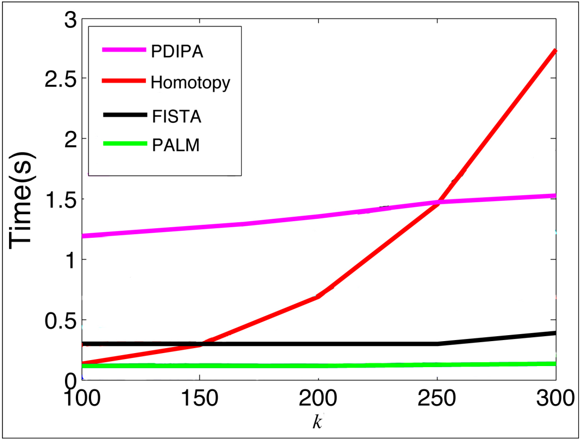
- ALM guarantees quadratic convergence of the outer-loop.
- ALM is efficient only if the inter-loop solving each $F_\mu(\mathbf{x}, \mathbf{y})$ is also efficient!

$$\mathbf{x}_{k+1} = \arg\min F_{\mu_k}(\mathbf{x}, \mathbf{y}_k)$$

- ℓ_1-**Min:** use FISTA.

Simulation Benchmark on ℓ_1-Min

Figure : $n = 1000$ and varying sparsity



References: (Matlab/C code available on our website)

AY, Zhou, Ganesh, Ma. *Fast L1-minimization algorithms for robust face recognition*, IEEE TIP, 2013.

ADMM: Alternating Direction Method of Multipliers

Question: When ALM is too slow, is there yet another solution to the rescue?

ADMM: Alternating Direction Method of Multipliers

Question: When ALM is too slow, is there yet another solution to the rescue?

- Consider the same objective function in ALM:

$$F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) \quad \text{subj. to} \quad h(\mathbf{x}) = 0.$$

In case $f(\cdot)$, $g(\cdot)$, and/or their composite function are too complex:

$$\begin{array}{rcl} F(\mathbf{x}, \mathbf{z}) & = & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subj. to} & & h(\mathbf{x}) = 0, \mathbf{x} - \mathbf{z} = 0. \end{array}$$

Berkeley University of California

ADMM: Alternating Direction Method of Multipliers

Question: When ALM is too slow, is there yet another solution to the rescue?

- Consider the same objective function in ALM:

$$F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) \quad \text{subj. to} \quad h(\mathbf{x}) = 0.$$

In case $f(\cdot)$, $g(\cdot)$, and/or their composite function are too complex:

$$\begin{array}{rcl} F(\mathbf{x}, \mathbf{z}) & = & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subj. to} & & h(\mathbf{x}) = 0, \mathbf{x} - \mathbf{z} = 0. \end{array}$$

- Apply ALM

$$F(\mathbf{x}, \mathbf{z}, \mathbf{y}_1, \mathbf{y}_2, \mu) \doteq f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}_1^T h(\mathbf{x}) + \mathbf{y}_2^T (\mathbf{x} - \mathbf{z}) + \frac{\mu}{2}\|h(\mathbf{x})\|^2 + \frac{\mu}{2}\|\mathbf{x} - \mathbf{z}\|^2$$

ADMM: Alternating Direction Method of Multipliers

Question: When ALM is too slow, is there yet another solution to the rescue?

- Consider the same objective function in ALM:

$$F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) \quad \text{subj. to} \quad h(\mathbf{x}) = 0.$$

In case $f(\cdot)$, $g(\cdot)$, and/or their composite function are too complex:

$$\begin{array}{rcl} F(\mathbf{x}, \mathbf{z}) & = & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subj. to} & & h(\mathbf{x}) = 0, \mathbf{x} - \mathbf{z} = 0. \end{array}$$

- Apply ALM

$$F(\mathbf{x}, \mathbf{z}, \mathbf{y}_1, \mathbf{y}_2, \mu) \doteq f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}_1^T h(\mathbf{x}) + \mathbf{y}_2^T(\mathbf{x} - \mathbf{z}) + \frac{\mu}{2}\|h(\mathbf{x})\|^2 + \frac{\mu}{2}\|\mathbf{x} - \mathbf{z}\|^2$$

- Alternating direction technique:

$$\begin{array}{rcl} \mathbf{x}_{k+1} & = & \arg\min_{\mathbf{x}} F(\mathbf{x}, \mathbf{z}_k, \mathbf{y}_{1k}, \mathbf{y}_{2k}, \mu_k) \\ \mathbf{z}_{k+1} & = & \arg\min_{\mathbf{x}} F(\mathbf{x}_{k+1}, \mathbf{z}_k, \mathbf{y}_{1k}, \mathbf{y}_{2k}, \mu_k) \\ \mathbf{y}_{1k+1} & = & \mathbf{y}_{1k} + \mu_k h(\mathbf{x}) \\ \mathbf{y}_{2k+1} & = & \mathbf{y}_{2k} + \mu_k(\mathbf{x} - \mathbf{z}) \\ \mu_{k+1} & \nearrow & \infty \end{array}$$

Solving ℓ_1-Min via ADMM

- ℓ_1-min update rule:

$$\begin{aligned} \mathbf{x}_{k+1} &= \text{soft}\left(\mathbf{w}_k - \frac{1}{L}\nabla f(\mathbf{w}_k), \frac{\lambda}{L}\right) \\ \mathbf{w}_{k+1} &= \mathbf{x}_{k+1} + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}_{k+1} - \mathbf{x}_k) \\ \mathbf{y}_{k+1} &= \mathbf{y}_k + \mu_k h(\mathbf{x}_{k+1}) \end{aligned} \quad (2)$$

Solving ℓ_1-Min via ADMM

- ℓ_1-min update rule:

$$\begin{aligned} \mathbf{x}_{k+1} &= \text{soft}\left(\mathbf{w}_k - \frac{1}{L}\nabla f(\mathbf{w}_k), \frac{\lambda}{L}\right) \\ \mathbf{w}_{k+1} &= \mathbf{x}_{k+1} + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}_{k+1} - \mathbf{x}_k) \\ \mathbf{y}_{k+1} &= \mathbf{y}_k + \mu_k h(\mathbf{x}_{k+1}) \end{aligned} \tag{2}$$

- Compared to ALM:

$$\begin{aligned} (\mathbf{x}_{k+1}, \mathbf{w}_{k+1}) &= \arg\min F_{\mu_k}(\mathbf{x}, \mathbf{w}, \mathbf{y}_k) \\ \mathbf{y}_{k+1} &= \mathbf{y}_k + \mu_k h(\mathbf{x}_{k+1}) \end{aligned} \tag{3}$$

Solving ℓ_1-Min via ADMM

- ℓ_1-min update rule:

$$\begin{aligned} \mathbf{x}_{k+1} &= \text{soft}\left(\mathbf{w}_k - \tfrac{1}{L}\nabla f(\mathbf{w}_k), \tfrac{\lambda}{L}\right) \\ \mathbf{w}_{k+1} &= \mathbf{x}_{k+1} + \tfrac{t_k - 1}{t_{k+1}}(\mathbf{x}_{k+1} - \mathbf{x}_k) \\ \mathbf{y}_{k+1} &= \mathbf{y}_k + \mu_k h(\mathbf{x}_{k+1}) \end{aligned} \qquad (2)$$

- Compared to ALM:

$$\begin{aligned} (\mathbf{x}_{k+1}, \mathbf{w}_{k+1}) &= \arg\min F_{\mu_k}(\mathbf{x}, \mathbf{w}, \mathbf{y}_k) \\ \mathbf{y}_{k+1} &= \mathbf{y}_k + \mu_k h(\mathbf{x}_{k+1}) \end{aligned} \qquad (3)$$

In ADMM, update of **x**, **w**, and **y** alternates only once per loop regardless of convergence.

References:
Yang, Zhang. *Alternating direction algorithms for ℓ_1-problems in compressive sensing*, 2009.
Boyd et al. *Distributed optimization and statistical learning via the alternating direction method of multipliers*, 2010.

RPCA via ADMM

- RPCA: $\min_{A,E} \|A\|_* + \lambda\|E\|_1 \quad \text{subj. to} \quad D = A + E$

ALM Objective

$$\begin{aligned} F(A,E,Y,\mu) &= \|A\|_* + \lambda\|E\|_1 + \langle Y, D-A-E\rangle + \frac{\mu}{2}\|D-A-E\|_F^2 \\ &= g_1(A) + g_2(E) + f(A,E,Y,\mu) \end{aligned}$$

RPCA via ADMM

- RPCA: $\min_{A,E} \|A\|_* + \lambda\|E\|_1 \quad$ subj. to $\quad D = A + E$

ALM Objective

$$
\begin{aligned}
F(A, E, Y, \mu) &= \|A\|_* + \lambda\|E\|_1 + \langle Y, D - A - E\rangle + \tfrac{\mu}{2}\|D - A - E\|_F^2 \\
&= g_1(A) + g_2(E) + f(A, E, Y, \mu)
\end{aligned}
$$

- **For matrix ℓ_1-norm**:
  When $g(\mathbf{x}) = \lambda\|X\|_1$, $\lambda > 0$,

$$
\arg\min_X \lambda\|X\|_1 + \frac{1}{2}\|Q - X\|_F^2
$$

has a closed-form proximal function *element-wise*: soft (soft-thresholding)

$$
x_{ij}^* = \text{soft}(q_{ij}, \lambda) = \text{sign}(q_{ij}) \cdot \max\{|q_{ij}| - \lambda, 0\}.
$$

RPCA via ADMM

- RPCA: $\min_{A,E} \|A\|_* + \lambda\|E\|_1$ subj. to $D = A + E$

ALM Objective

$$\begin{aligned} F(A,E,Y,\mu) &= \|A\|_* + \lambda\|E\|_1 + \langle Y, D-A-E\rangle + \frac{\mu}{2}\|D-A-E\|_F^2 \\ &= g_1(A) + g_2(E) + f(A,E,Y,\mu) \end{aligned}$$

- **For matrix ℓ_1-norm**:
  When $g(\mathbf{x}) = \lambda\|X\|_1$, $\lambda > 0$,

$$\arg\min_X \lambda\|X\|_1 + \frac{1}{2}\|Q - X\|_F^2$$

  has a closed-form proximal function *element-wise*: soft (soft-thresholding)

$$x_{ij}^* = \mathrm{soft}(q_{ij}, \lambda) = \mathrm{sign}(q_{ij}) \cdot \max\{|q_{ij}| - \lambda, 0\}.$$

- IST update rule for E: $Q_E = D - A - \frac{1}{\mu}Y$

$$E_{k+1} = \mathrm{soft}(D - A_k - \frac{1}{\mu_k}Y_k, \frac{\lambda}{\mu_k})$$

Singular-Value Thresholding

- **For matrix nuclear norm**: [Cai-Candès-Shen '08]
  When $g(\mathbf{x}) = \lambda\|X\|_*$, $\lambda > 0$,

$$\arg\min_{\mathbf{x}} \lambda\|X\|_* + \frac{1}{2}\|Q - X\|_F^2$$

has a closed-form proximal function: $S(\cdot,\cdot)$ (singular-value thresholding)

$$X^* = U\,\mathrm{soft}(\Sigma, \lambda)V^T \doteq S(Q, \lambda),$$

where $Q = U\Sigma V^T$.

Singular-Value Thresholding

- **For matrix nuclear norm**: [Cai-Candès-Shen '08]
  When $g(\mathbf{x}) = \lambda\|X\|_*$, $\lambda > 0$,

$$\arg\min_{\mathbf{x}} \lambda\|X\|_* + \frac{1}{2}\|Q - X\|_F^2$$

  has a closed-form proximal function: $S(\cdot,\cdot)$ (singular-value thresholding)

$$X^* = U\,\mathrm{soft}(\Sigma,\lambda)V^T \doteq S(Q,\lambda),$$

  where $Q = U\Sigma V^T$.
- IST update rule for A: $Q_A = D - E - \frac{1}{\mu}Y$

ALM Objective

$$\begin{aligned} F(A,E,Y,\mu) &= \|A\|_* + \lambda\|E\|_1 + \langle Y, D - A - E\rangle + \tfrac{\mu}{2}\|D - A - E\|_F^2 \\ &= g_1(A) + g_2(E) + f(A,E,Y,\mu) \end{aligned}$$

$$A_{k+1} = S(D - E_k - \frac{1}{\mu_k}Y, \frac{1}{\mu_k})$$

Solving RPCA via ALM and ADMM

- RPCA-ALM

RPCA-ALM

1. $(A_{k+1}, E_{k+1}) = \arg\min_{A,E}\{A, E, Y_k, \mu_k\}$
   Repeat soft$(Q_E, \cdot)$ and $S(Q_A, \cdot)$ until converges.
2. $Y_{k+1} = Y_k + \mu_k(D - A_{k+1} - E_{k+1})$
3. $\mu_{k+1} \nearrow \infty$

Drawback: When μ_k grows, the inner-loop slows down $\Rightarrow$ The total number of SVDs grows!

Solving RPCA via ALM and ADMM

- RPCA-ALM

RPCA-ALM

1. $(A_{k+1}, E_{k+1}) = \arg\min_{A,E}\{A, E, Y_k, \mu_k\}$
   Repeat soft$(Q_E, \cdot)$ and $S(Q_A, \cdot)$ until converges.
2. $Y_{k+1} = Y_k + \mu_k(D - A_{k+1} - E_{k+1})$
3. $\mu_{k+1} \nearrow \infty$

Drawback: When μ_k grows, the inner-loop slows down $\Rightarrow$ The total number of SVDs grows!

- RPCA-ADMM: Minimizing RPCA w.r.t. A or E *separately* is easy

RPCA-ADMM

1. $A_{k+1} = S(D - E_k - \frac{1}{\mu_k}Y, \frac{1}{\mu_k})$
2. $E_{k+1} = \text{soft}(D - A_{k+1} - \frac{1}{\mu_k}Y_k, \frac{\lambda}{\mu_k})$
3. $Y_{k+1} = Y_k + \mu_k(D - A_{k+1} - E_{k+1})$
4. $\mu_{k+1} \nearrow \infty$

A, E, and Y are updated only once in each loop regardless of convergence.

Summary

- Convergence:

Theorem [Lin-Chen-Wu-Ma '11]

In ADMM, if μ_k is nondecreasing, then (A_k, E_k) converge globally to an optimal solution to the RPCA problem iff

$$\sum_{k=1}^{+\infty} \frac{1}{\mu_k} = +\infty.$$

Summary

- Convergence:

Theorem [Lin-Chen-Wu-Ma '11]

In ADMM, if μ_k is nondecreasing, then (A_k, E_k) converge globally to an optimal solution to the RPCA problem iff

$$\sum_{k=1}^{+\infty} \frac{1}{\mu_k} = +\infty.$$

- Pros:
  1. Separately optimizing $g_1(A)$ and $g_2(E)$ simplifies the problem.
  2. In each iteration, most expensive operation is SVD.
  3. Each subproblem can be solved in a *distributed fashion*, maximizing the usage of CPUs and memory.

Berkeley
University of California

Summary

- Convergence:

Theorem [Lin-Chen-Wu-Ma '11]

In ADMM, if μ_k is nondecreasing, then (A_k, E_k) converge globally to an optimal solution to the RPCA problem iff

$$\sum_{k=1}^{+\infty} \frac{1}{\mu_k} = +\infty.$$

- Pros:
  1. Separately optimizing $g_1(A)$ and $g_2(E)$ simplifies the problem.
  2. In each iteration, most expensive operation is SVD.
  3. Each subproblem can be solved in a *distributed fashion*, maximizing the usage of CPUs and memory.
- Cons:
  1. Analysis of convergence become more involved. [Boyd et al. '10, Lin-Chen-Wu-Ma '11]
  2. Inexact optimization may lead to more iterations to converge.

References:
Boyd et al. *Distributed optimization and statistical learning via the alternating direction method of multipliers*, 2010.
Lin, Chen, Wu, Ma. *The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrix*, 2011.

RPCA Benchmark

Table : Solving a $1,000 \times 1,000$ RPCA problem.

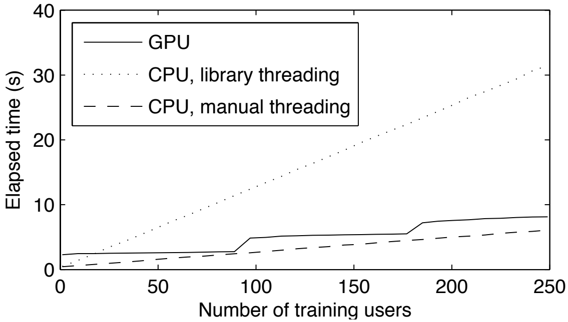
Algorithm	Accuracy	Rank	$\|E\|_0$ (%)	Iterations	Time (sec)
IST	5.99e-6	50	10.12	8,550	119,370.3
APG	5.91e-6	50	10.03	134	82.7
ALM	2.07e-7	50	10.00	34	37.5
ADMM	3.83e-7	50	10.00	23	11.8

Take-Home Message

1. More than 10,000 times speed acceleration in the above simulation.
2. Complexity of RPCA is a small constant times that of SVD.

Further Numerical Acceleration Achievable via Parallelization

Figure : Face Alignment Time vs Number of Subjects



Reference (Parallel C/CUDA-C implementation available upon request):
Shia, AY, Sastry, Ma, *Fast ℓ_1-minimization and parallelization for face recognition*, **Asilomar Conf**, 2011.

Nonlinear Sparse Optimization

- ℓ_1-Min assumes an underdetermined linear system: $\mathbf{b} = A\mathbf{x}$

$$(P_0): \quad \min \|\mathbf{b} - A\mathbf{x}\|^2 \quad \text{subj. to} \quad \|\mathbf{x}\|_1 \leq k.$$

Nonlinear Sparse Optimization

- ℓ_1-Min assumes an underdetermined linear system: $\mathbf{b} = A\mathbf{x}$

$$(P_0): \quad \min \|\mathbf{b} - A\mathbf{x}\|^2 \quad \text{subj. to} \quad \|\mathbf{x}\|_1 \leq k.$$

- Generalize to any **nonlinear differentiable function** $f(\cdot)$:

$$\min f(\mathbf{x}) \quad \text{subj. to} \quad \|\mathbf{x}\|_1 \leq k.$$

Nonlinear Sparse Optimization

- ℓ_1-Min assumes an underdetermined linear system: $\mathbf{b} = A\mathbf{x}$

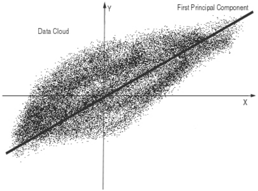
$$(P_0): \quad \min \|\mathbf{b} - A\mathbf{x}\|^2 \quad \text{subj. to} \quad \|\mathbf{x}\|_1 \leq k.$$

- Generalize to any **nonlinear differentiable function** $f(\cdot)$:

$$\min f(\mathbf{x}) \quad \text{subj. to} \quad \|\mathbf{x}\|_1 \leq k.$$

- **Example:** Sparse PCA

$$\mathbf{x}^* = \arg \max_{\|\mathbf{x}\|_2 = 1} \left(\mathbf{x}^T \Sigma \mathbf{x} - \rho \|\mathbf{x}\|_1 \right)$$



A General Framework in Nonlinear Sparse Optimization

Greedy Simplex Method (Matching Pursuit/Coordinate Descent)

1. If $\|\mathbf{x}^l\|_0 < k$
   1. Find $t_p \in \arg\min_i f(\mathbf{x}^l + t_i \mathbf{e}_i)$.
   2. $\mathbf{x}^{l+1} = \mathbf{x}^l + t_p \mathbf{e}_p$.
2. if $\|\mathbf{x}^l\|_0 = k$
   1. Find $(t_p, x_q) \in \arg\min_{i,j} f(\mathbf{x}^l - x_j \mathbf{e}_j + t_i \mathbf{e}_i)$.
   2. $\mathbf{x}^{l+1} = \mathbf{x}^l - x_q \mathbf{e}_q + t_p \mathbf{e}_p$.
3. Continue $l \leftarrow l + 1$, until some stopping criterion is satisfied.

- **Convergence:**
  GSM converges to a local minimum: $f(\mathbf{x}^{l+1}) \leq f(\mathbf{x}^l)$ for every $l \geq 0$.

Reference:
Beck & Eldar, *Sparsity constrained nonlinear optimization*, 2012.

References

Website:

- ℓ_1-Min: http://www.eecs.berkeley.edu/~yang/
- RPCA: http://perception.csl.illinois.edu/matrix-rank/

More Publications:

- AY, Ganesh, Zhou, Sastry, Ma. "Fast ℓ_1-minimization algorithms for robust face recognition." arXiv, 2010.
- Shia, AY, Sastry, Ma. "Fast ℓ_1-minimization and parallelization for face recognition." Asilomar, 2011.
- Singaraju, Tron, Elhamifar, AY, Sastry. "On the Lagrangian biduality of sparsity minimization problems." ICASSP, 2012.
- Naikal, AY, Sastry. "Informative feature selection for objection recognition via Sparse PCA." ICCV, 2011.
- Slaughter, AY, Bagwell, Checkles, Sentis, Vishwanath. "Sparse online low-rank projection and outlier rejection (SOLO) for 3-D rigid-body motion registration." ICRA, 2012.
- Ohlsson, AY, Dong, Sastry. "CPRL – An extension of compressive sensing to the phase retrieval problem." NIPS, 2012.