Non-Convex Methods for Low-Rank Matrix Reconstruction

Jian-Feng Cai

Department of Mathematics, Hong Kong University of Science and Technology

in Collaboration with
Ke Wei (UC Davis), Tianming Wang (U of Iowa),
Tony Chan, Shingyu Leung (HKUST)

May 29, 2017

Table of Contents

Low-rank matrix recovery

Assume $X \in \mathbb{R}^{m\times n}$ and $\mathrm{rank}(X) = r < \min(m, n)$. We want to reconstruct X from its linear measurement $y \in \mathbb{R}^p$

$$y = \mathcal{A}X,$$

where $\mathcal{A} : \mathbb{R}^{m\times n} \mapsto \mathbb{R}^p$ is a linear operator.

It is challenging to solve this problem because usually $p < mn$.

Example 1: Recommendation System

Netflix problem: Predict the rating of a viewer to a movie based on available ratings.

- X — rating matrix: x_{ij} is the rating of viewer i to movie j.
- Assume X is of low-rank — the rating is given by a few factors

$$x_{ij} = \sum_{k=1}^{r} \text{rating on factor } k = \sum_{k=1}^{r} p_{ik} q_{jk},$$

  where p_{ik} is the opinion of viewer i on factor k and q_{jk} is the quality of movie j on factor k.

- Only a small portion of entries of X is observed

$$\mathcal{A}X = \{x_{ij} \ : \ (i,j) \in \Omega\}, \quad \Omega \subset \{1,\dots,m\} \times \{1,\dots,n\}$$

- This problem is also called matrix completion.

Example 2: Phase Retrieval

Only intensities can be recorded by physical instruments. Can we recover the phase information?

- Let $x \in \mathbb{C}^n$ be an unknown vector.
- Intensities of its linear measurements are observed.

$$y_i = |\langle a_i, x \rangle|, \quad i = 1, \ldots, p.$$

- Instead of recovering x, we reconstruct the rank-1 matrix $X = xx^* \in \mathbb{C}^{n \times n}$.
- The observations are linear with respect to X

$$y_i^2 = \langle a_i a_i^*, X \rangle, \quad i = 1, \ldots, p$$

so that

$$\mathcal{A}X = \{\langle a_i a_i^*, X \rangle\}_{i=1}^p$$

More examples

- High-dimensional data.
  - Let $X = [\boldsymbol{x}_1, \ \boldsymbol{x}_2, \ \ldots, \ \boldsymbol{x}_n]$, where $\boldsymbol{x}_i \in \mathbb{R}^m$ for all i.
  - Assume all $\boldsymbol{x}_i$ lie on a low dimensional subspace in $\mathbb{R}^n$, which implies X is of low rank.
  - Linear inverse problems on high-dimensional data can be formulated by the problem of low-rank matrix recovery.

More examples

- High-dimensional data.
  - Let $X = [\boldsymbol{x}_1,\ \boldsymbol{x}_2,\ \ldots,\ \boldsymbol{x}_n]$, where $\boldsymbol{x}_i \in \mathbb{R}^m$ for all i.
  - Assume all $\boldsymbol{x}_i$ lie on a low dimensional subspace in $\mathbb{R}^n$, which implies X is of low rank.
  - Linear inverse problems on high-dimensional data can be formulated by the problem of low-rank matrix recovery.
- Homogeneous quadratic inverse problems.
  - Besides phase retrieval, some quadratic inverse problem may be converted to the recovery of a low-rank matrix.
  - Let $\boldsymbol{x} = [x_1,\ x_2,\ \ldots,\ x_m]^T$, and it is measured linear combinations of $x_i x_j^*$ for $1 \le i, j \le m$.
  - Let $X = \boldsymbol{x}\boldsymbol{x}^*$. Then the measurements are linear w.r.t. the rank-1 matrix X.
  - Examples: Phase Retrieval, Blind Deconvolution, Euclidean Embedding, Sensor Self-Calibration, ...

Table of Contents

Convex Optimization

To find a low-rank solution of $\mathcal{A}Z = y$

- Solve

$$\min_{Z} \ \operatorname{rank}(Z), \quad \text{s.t.} \quad \mathcal{A}Z = y.$$

- Non-convex, NP-hard.

Convex Optimization

To find a low-rank solution of $\mathcal{A}Z = y$

- Solve

$$\min_Z \ \mathrm{rank}(Z), \quad \text{s.t.} \quad \mathcal{A}Z = y.$$

- Non-convex, NP-hard.
- Convex relaxation:

$$\|Z\|_* = \sum_i \sigma_i(Z)$$

where $\|\boldsymbol{Z}\|_*$ is the nuclear norm of Z, the sum of all singular values of Z.

- Solve

$$\min_Z \ \|Z\|_*, \quad \text{s.t.} \quad \mathcal{A}Z = y.$$

Computation of Nuclear Norm Minimization

$$\min_{Z} \ \|Z\|_*, \quad \text{s.t.} \quad \mathcal{A}Z = y.$$

- $\|\cdot\|_*$ is non-smooth: the step size will be extremely small when a forward gradient descent (explicit) method is used.

Computation of Nuclear Norm Minimization

$$\min_Z \ \|Z\|_*, \quad \text{s.t.} \quad \mathcal{A}Z = y.$$

- $\|\cdot\|_*$ is non-smooth: the step size will be extremely small when a forward gradient descent (explicit) method is used.
- It will be faster to use a backward (implicit) method, where we need the proximity operator $(I + \lambda\partial\|\cdot\|_*)^{-1}$.

Theorem (Cai, Candes, Shen, 2010)

The proximal operator of $\|\cdot\|_*$ is the singular value thresholding (SVT). More precisely, let $Y = U\Sigma V^T \in \mathbb{R}^{m\times n}$ be a given matrix and its SVD. Then,

$$\mathcal{S}_\lambda(Y) = \arg\min_Z \frac{1}{2}\|Y - Z\|_F^2 + \lambda\|Z\|_*,$$

where

$$\mathcal{S}_\lambda(Y) = U\max(\Sigma - \lambda I, 0)_+ V^T.$$

Computation of Nuclear Norm Minimization

SVT is a fundamental element in many popular nuclear norm minimization algorithms.

- SVT algorithm [Cai, Candes, Shen, 2010]

$$\begin{cases} Y_{k+1} = Y_k - \delta \mathcal{A}^*(\mathcal{A}X_k - y) \\ X_{k+1} = \mathcal{S}_\delta(Y_{k+1}). \end{cases}$$

- Iterative soft-threhsolding [Ma et. al., 2011]

$$X_{k+1} = \mathcal{S}_{\lambda\delta}(X_k - \delta \mathcal{A}^*(\mathcal{A}X_k - y))$$

- ADMM [Chen et.al. 2012; Lin et.al. 2011]
- Proximity algorithms [Micchelli et.al. 2011; ...].

Computation of Nuclear Norm Minimization

The bottleneck of these algorithms is the computation of SVT $\mathcal{S}_\lambda(Y)$.

- All singular values exceeding λ and their associated singular vectors are computed.
- For large scale computation, a small rank of $\mathcal{S}_\lambda(Y)$ is needed at each iteration.

Computation of Nuclear Norm Minimization

The bottleneck of these algorithms is the computation of SVT $\mathcal{S}_\lambda(Y)$.

- All singular values exceeding λ and their associated singular vectors are computed.
- For large scale computation, a small rank of $\mathcal{S}_\lambda(Y)$ is needed at each iteration.

Disadvantage: The computation is expensive, and it consumes large memory.

Table of Contents

Non-Convex Optimization

Assume the rank r is known or estimated.

- Factorization based methods:

$$\min_{L\in\mathbb{R}^{n\times r},R\in\mathbb{R}^{m\times r}} \|\mathcal{A}(LR^T)-y\|_2^2.$$

Non-Convex Optimization

Assume the rank r is known or estimated.

- Factorization based methods:

$$\min_{L\in\mathbb{R}^{n\times r},R\in\mathbb{R}^{m\times r}} \|\mathcal{A}(LR^T)-y\|_2^2.$$

or

$$\min_{L\in\mathbb{R}^{n\times r},R\in\mathbb{R}^{m\times r}} \|\mathcal{A}(LR^T)-y\|_2^2+\lambda\|L^TL-R^TR\|_F^2.$$

  - The term $\|L^TL-R^TR\|_F^2$ to balance the magnitude of L and F.
  - Alternating minimization, Gradient descent, ...

Non-Convex Optimization

Assume the rank r is known or estimated.

- Factorization based methods:

$$\min_{L\in\mathbb{R}^{n\times r},R\in\mathbb{R}^{m\times r}} \|\mathcal{A}(LR^T)-y\|_2^2.$$

or

$$\min_{L\in\mathbb{R}^{n\times r},R\in\mathbb{R}^{m\times r}} \|\mathcal{A}(LR^T)-y\|_2^2+\lambda\|L^TL-R^TR\|_F^2.$$

  - The term $\|L^TL-R^TR\|_F^2$ to balance the magnitude of L and F.
  - Alternating minimization, Gradient descent, ...
  - Any local min is also the global min, and all other critical points are either strict saddle points or local max. [Ge et.al., 2017; Ge et.al., 2016; Sun etal., 2016]
  - Gradient descent algorithm with an arbitrary initial guess will not converge to a prescribed strict saddle point / local max almost surely. [Lee et.al., 2016]

Non-Convex Optimization

Assume the rank r is known or estimated.

- Factorization based methods:

$$\min_{L\in\mathbb{R}^{n\times r},R\in\mathbb{R}^{m\times r}} \|\mathcal{A}(LR^T)-y\|_2^2.$$

or

$$\min_{L\in\mathbb{R}^{n\times r},R\in\mathbb{R}^{m\times r}} \|\mathcal{A}(LR^T)-y\|_2^2+\lambda\|L^TL-R^TR\|_F^2.$$

  - The term $\|L^TL-R^TR\|_F^2$ to balance the magnitude of L and F.
  - Alternating minimization, Gradient descent, ...
  - Any local min is also the global min, and all other critical points are either strict saddle points or local max. [Ge et.al., 2017; Ge et.al., 2016; Sun etal., 2016]
  - Gradient descent algorithm with an arbitrary initial guess will not converge to a prescribed strict saddle point / local max almost surely. [Lee et.al., 2016]

- Rank constrained methods:

$$\min_{Z\in\mathbb{R}^{m\times n}} \|\mathcal{A}Z-y\|_2^2, \quad \text{s.t.} \quad \operatorname{rank}(Z)=r.$$

Iterative Hard-Thresholding.

We solve the rank constrained minimization

$$\min_{Z\in\mathbb{R}^{m\times n}} \|\mathcal{A}Z - y\|_2^2, \quad \text{s.t.} \quad \operatorname{rank}(Z) = r.$$

by projected gradient descent

$$X_{l+1} = \mathcal{H}_r(X_l - \alpha_l \mathcal{A}^*(\mathcal{A}X_l - y)),$$

where $\mathcal{H}_r(\cdot)$ is the r-truncated SVD.

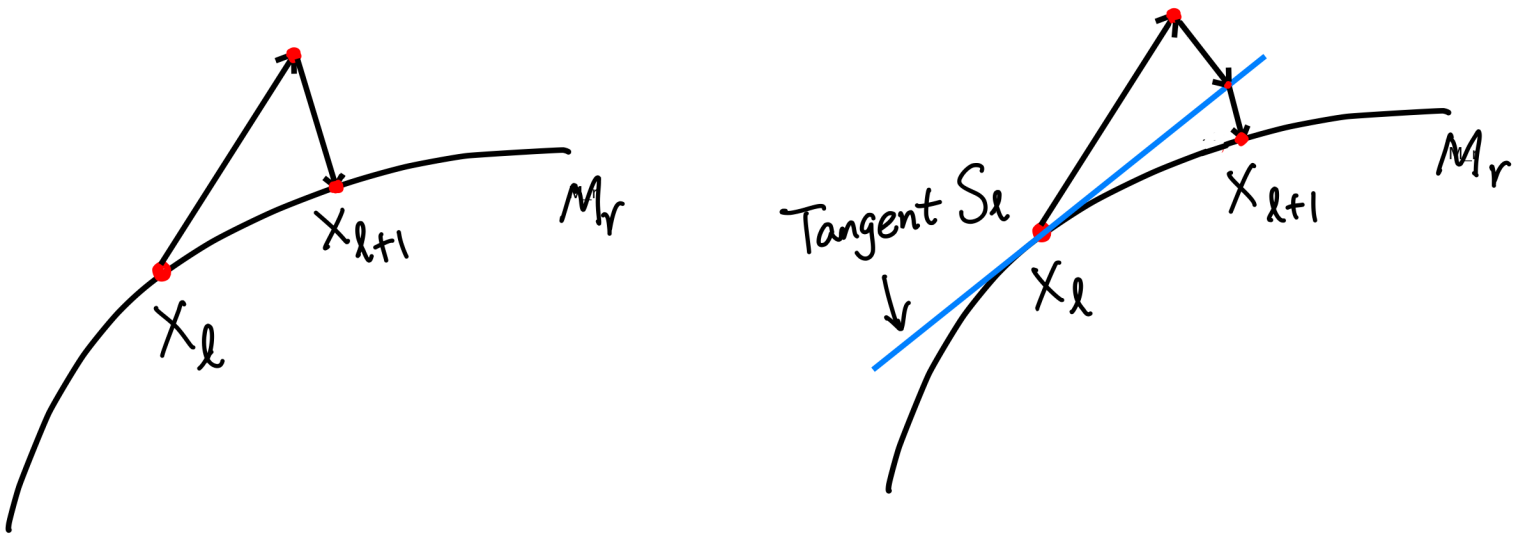
- When α_l is fixed, it is known as Singular Value Projection (SVP).
- When α_l is the steepest descent stepsize, it is called normalized IHT (NIHT).

IHT (cont.)

- In each iteration, SVD of $m \times n$ matrices is still needed in each step.
- How to avoid large size SVD?

IHT (cont.)

- In each iteration, SVD of $m \times n$ matrices is still needed in each step.
- How to avoid large size SVD?



Our algorithm [Wei, C., Chan, Leung, 2016]

$$X_{l+1} = \mathcal{H}_r \mathcal{P}_{S_l}(X_l - \alpha_l \mathcal{A}^*(\mathcal{A}X_l - y)).$$

No large scale SVD

- The subspace

$$\mathcal{S}_l = \{U_l P^T + Q V_l^T \ : \ P \in \mathbb{R}^{n\times r}, Q \in \mathbb{R}^{m\times r}\},$$

  is the tangent space of smooth manifold $\mathcal{M}_r$ at $X_l = U_l \Sigma_l V_l^T$, where $\mathcal{M}_r$ is the set of all rank-r matrices embedded in $\mathbb{R}^{m\times n}$.
- The projection $\mathcal{P}_{\mathcal{S}_l}$ is just matrix products.

No large scale SVD

- The subspace

$$\mathcal{S}_l = \{U_l P^T + Q V_l^T \ : \ P \in \mathbb{R}^{n\times r}, Q \in \mathbb{R}^{m\times r}\},$$

  is the tangent space of smooth manifold $\mathcal{M}_r$ at $X_l = U_l \Sigma_l V_l^T$, where $\mathcal{M}_r$ is the set of all rank-r matrices embedded in $\mathbb{R}^{m\times n}$.
- The projection $\mathcal{P}_{\mathcal{S}_l}$ is just matrix products.
- SVD of size only $2r \times 2r$ is needed.

$$W_l \in \mathcal{S}_l \Longrightarrow W_l = \overbrace{[U_l \ Q]}^{2r} \left.\begin{bmatrix} P^T \\ V_l^T \end{bmatrix}\right\} 2r,$$

  where W_l is the matrix before the application of $\mathcal{H}_r$.
  - First compute QR decompostion of $[U_l \ Q]$ and $\begin{bmatrix} P^T \\ V_l^T \end{bmatrix}$ respectively.
  - Then compute SVD of the product of R factors, which is of size $2r \times 2r$.

Riemannian optimization

The algorithm

$$X_{l+1} = \mathcal{H}_r(X_l - \alpha_l \mathcal{P}_{\mathcal{S}_l} \mathcal{A}^*(\mathcal{A}X_l - y)),$$

can be interpreted as a Gradient Descent Algorithm on the Riemannian manifold $\mathcal{M}_r$. [Vandereycken, 2013; Mishra, Apuroop, Sepulchre, 2013; Mishra, Meyer, Bonnabel, Sepulchre, 2013]

Riemannian Gradient Descent (RGad)

$$G_l = \mathcal{P}_{\mathcal{S}_l}(\mathcal{A}^*(\mathcal{A}X_l - y)) \qquad \text{(Gradient on the tangent space)}$$

$$\alpha_l = \frac{\|G_l\|_F^2}{\|\mathcal{A}G_l\|_2^2} \qquad \text{(Steepest Descent Step Size)}$$

$$W_l = X_l - \alpha_l G_l \qquad \text{(Update along the gradient)}$$

$$X_{l+1} = H_r(W_l) \qquad \text{(Retraction)}$$

Riemannian Conjugate Gradient

The algorithm can be further improved by conjugate gradient on Riemannian manifold for solving $\min_{Z \in \mathcal{M}_r} \|\mathcal{A}Z - y\|_2^2$

Riemannian Conjugate Gradient (RCG)

$$G_l = \mathcal{P}_{\mathcal{S}_l}\left(\mathcal{A}^*(\mathcal{A}X_l - y)\right) \quad \text{(Gradient on the tangent space)}$$

$$\beta_l = -\frac{\langle \mathcal{A}G_l, \mathcal{A}P_{l-1}\rangle}{\|\mathcal{A}P_{l-1}\|_2^2} \quad \text{(novel formula for } \beta \text{ [Wei, C., Chan, Leung, 2016])}$$

$$P_l = \mathcal{P}_{\mathcal{S}_l}(G_l + \beta_l P_{l-1}) \quad (P_l \text{ is conjugate to } P_{l-1})$$

$$\alpha_l = \frac{\langle G_l, P_l\rangle}{\|\mathcal{A}P_l\|_2^2}$$

$$W_l = X_l - \alpha_l P_l \quad \text{(Update along the search direction)}$$

$$X_{l+1} = H_r(W_l) \quad \text{(Retraction)}$$

Table of Contents

Theoretical Gurantee

Will RGrad and RCG find the true low-rank matrix X from $y = \mathcal{A}X$?
How many linear equations do we need?

- The analysis depends on applications.

$$\text{Good Initialization} + \text{Local Convergence}$$
$$\Longrightarrow \text{Convergence to the true low-rank solution}$$

Initialization

- We choose
$$X_0 = H_r(\mathcal{A}^* y)$$
- X_0 is one step of IHT with initial guess 0.

Initialization

- We choose
$$X_0 = H_r(\mathcal{A}^* y)$$
- X_0 is one step of IHT with initial guess 0.
- A probabilistic explanation: Assume A_i, $i = 1, \ldots, p$ have i.i.d. entries with expectation 0 and variance $1/p$. Then

$$\mathbf{E}([\mathcal{A}^* y]_{jk}) = \mathbf{E}\left(\sum_i \langle A_i, X\rangle [A_i]_{jk}\right) = \mathbf{E}\left(\sum_{i,a,b} [A_i]_{jk}[A_i]_{ab} X_{ab}\right)$$
$$= \mathbf{E}\left(\sum_{i=1}^{p} [A_i]_{jk}^2\right) \cdot X_{jk} = X_{jk}$$

Case I: Guarantee for $\mathcal{A}$ satisfying RIP

Restricted Isometric Property (s-RIP)

There exists a constant $R_s \in (0,1)$ such that

$$(1-R_s)\|Z\|_F^2 \le \|\mathcal{A}Z\|_F^2 \le (1+R_s)\|Z\|_F^2, \quad \forall Z \in \mathcal{M}_s.$$

Theorem (Wei, Cai, Chan, Leung, SIMAX, 2016)

Assume $\mathcal{A}$ satisfies RIP with

$$R_{3r} \le \frac{1}{\mathrm{Cond}^2(X)} \frac{1}{25\sqrt{r}}.$$

Then the RGrad algorithm with initial guess $X_0 = H_r(\mathcal{A}^* y)$ converges linearly to X, provided the rank of X is r and $y = \mathcal{A}X$.

When $\mathcal{A}$ satisfies RIP

Theorem (Wei, Cai, Chan, Leung, SIMAX, 2016)

Assume $\mathcal{A}$ satisfies RIP with

$$R_{3r} \leq \frac{1}{\mathrm{Cond}^2(X)} \frac{1}{40\sqrt{r}}.$$

Then the Riemannian conjugate gradient algorithm with:

- *Initial guess $X_0 = H_r(\mathcal{A}^* y)$*
- *Restarting when either $\frac{\langle G_l, P_{l-1} \rangle}{\|G_l\|_F \|P_{l-1}\|_F} \leq 0.1$ or $\|G_l\|_F \leq \|P_{l-1}\|_F$ violated*

converges linearly to X, provided the rank of X is r and $y = \mathcal{A}X$.

Numerical Experiments

$\mathcal{A}$ is the random Gaussian.

Table: Average computational time (seconds) and average number of iterations of RGrad, RCG, RCG restarted, and ASD over ten random rank r matrices per (m, n, p, r) tuple for $m = n \in \{80, 160\}$, $r \in \{5, 10\}$ and $p/(m + n - r)r \in \{2, 3\}$; Gaussian sensing.

r	5						10					
$1/\rho$	2			3			2			3		
	rel.err	iter	time	rel.err	iter	time	rel.err	iter	time	rel.err	iter	time
	$m = n = 80$											
RGrad	3.3e-05	137	8.52	2.2e-05	58	5.61	3.2e-05	130	24.9	2.1e-05	57	15.5
RCG	2.2e-05	34	2.38	1.4e-05	22	2.71	2.1e-05	34	8.47	1.4e-05	22	7.72
RCG res.	2.2e-05	35	2.81	1.5e-05	22	2.79	2.2e-05	36	8.95	1.3e-05	23	8.12
ASD	2.5e-05	143	10.3	1.7e-05	73	9.37	2.4e-05	210	53.8	1.7e-05	224	82.2
	$m = n = 160$											
RGrad	3.3e-05	142	103	2e-05	61	66.2	3.2e-05	135	194	2.1e-05	58	123
RCG	2.3e-05	35	33.0	1.5e-05	22	31.2	2.2e-05	35	65.7	1.4e-05	23	62.9
RCG res.	2.4e-05	36	33.9	1.5e-05	23	32.1	2.2e-05	36	67.7	1.4e-05	24	66.1
ASD	2.5e-05	147	140	1.8e-05	81	117	2.4e-05	213	407	1.6e-05	149	426

Table: Phase transition table for Gaussian sensing with $m = n = 80$. For each (m, n, p) with $p = \delta \cdot mn$, the algorithm can recover all of the ten random test matrices when $r \leq r_{\min}$, but fails to recover each of the randomly drawn matrices when $r \geq r_{\max}$.

	RGrad				RCG				RCG restarted			
δ	$r_{\min}$	$r_{\max}$	$\rho_{\min}$	$\rho_{\max}$	$r_{\min}$	$r_{\max}$	$\rho_{\min}$	$\rho_{\max}$	$r_{\min}$	$r_{\max}$	$\rho_{\min}$	$\rho_{\max}$
0.1	3	4	0.74	0.97	3	4	0.74	0.97	3	4	0.74	0.97
0.15	4	6	0.65	0.96	4	6	0.65	0.96	4	6	0.65	0.96
0.2	6	8	0.72	0.95	6	8	0.72	0.95	6	8	0.72	0.95
0.25	8	10	0.76	0.94	8	10	0.76	0.94	8	10	0.76	0.94
0.3	11	12	0.85	0.93	11	13	0.85	1	11	13	0.85	1
0.35	12	15	0.79	0.97	12	15	0.79	0.97	11	15	0.73	0.97
0.4	14	17	0.8	0.95	14	17	0.8	0.95	14	17	0.8	0.95
0.45	17	19	0.84	0.93	17	19	0.84	0.93	17	19	0.84	0.93
0.5	20	22	0.88	0.95	20	22	0.88	0.95	20	22	0.88	0.95
0.55	22	24	0.86	0.93	22	24	0.86	0.93	22	24	0.86	0.93
0.6	25	27	0.88	0.94	26	28	0.91	0.96	26	28	0.91	0.96
0.65	28	30	0.89	0.94	28	32	0.89	0.98	28	32	0.89	0.98
0.7	31	33	0.89	0.94	31	35	0.89	0.98	31	35	0.89	0.98
0.75	34	36	0.89	0.93	35	38	0.91	0.97	35	38	0.91	0.97
0.8	38	40	0.91	0.94	40	42	0.94	0.97	40	42	0.94	0.97
0.85	42	44	0.91	0.94	44	47	0.94	0.98	44	47	0.94	0.98
0.9	47	50	0.92	0.95	50	53	0.95	0.98	50	53	0.95	0.98
0.95	52	54	0.92	0.94	57	61	0.97	0.99	57	61	0.97	0.99

Case II: Guarantee for matrix completion

- The operator $\mathcal{A} = \mathcal{P}_\Omega$ doesn't satisfies RIP.
- Matrix completion may fail for any algorithms.
  Example: if the $(1,1)$-entry is not sampled, then any algorithm cannot distinct the following matrices

$$e_1 e_1^T, \quad 2e_1 e_1^T, \quad \dots$$

- The singular vectors cannot be too sparse.

Assumption 1 [Candes, Recht, 2009; Candes, Tao, 2010]

Let X be an $n \times n$, rank-r matrix with compact SVD $X = U\Sigma V^T$.
Assume there exist two positive constants μ_0 and μ_1 such that

$$\frac{n}{r} \max_{1\le i\le n} \max\left\{\|\mathcal{P}_U(e_i)\|_2^2, \|\mathcal{P}_V(e_i)\|_2^2\right\} \le \mu_0, \quad \|X\|_\infty \le \mu_1 \sqrt{\frac{r}{n^2}} \|X\|_2.$$

Matrix completion

- Initialization: $X_0 = H_r(\mathcal{A}^* y)$
- The first $O(\log N)$ steps uses resampling and trimming.

Theorem (Wei, Cai, Chan, Leung, *preprint*, 2016)

Let X be fixed and satisfying Assumption 1. Suppose Ω is sampled uniformly at random with $|\Omega| = m$. Then both RGrad and restarted RCG converges linearly to X with probability at least $1 - n^{-2}$ provided

$$m \geq Cnr^2 \log^2 n$$

for some constant $C > 0$.

Matrix completion

Key inequalities in the proof

- RIP in the tangent space of $\mathcal{M}_r$ at X

$$\left\| \mathcal{P}_T \left(\mathcal{I} - \frac{mn}{p} \mathcal{P}_\Omega \right) \mathcal{P}_T \right\| \le \epsilon$$

[Candes, Rechet, 2009; Candes, Tao, 2010]

- "asymmetric" isometric property

$$\left\| \mathcal{P}_{\widehat{T}_\ell} \left(\mathcal{I} - \frac{mn}{p} \mathcal{P}_{\widehat{\Omega}_{\ell+1}} \right) \left(\mathcal{P}_U - \mathcal{P}_{\widehat{U}_\ell} \right) \right\| \le \epsilon$$

[Wei, C., Chan, Leung, 2016]

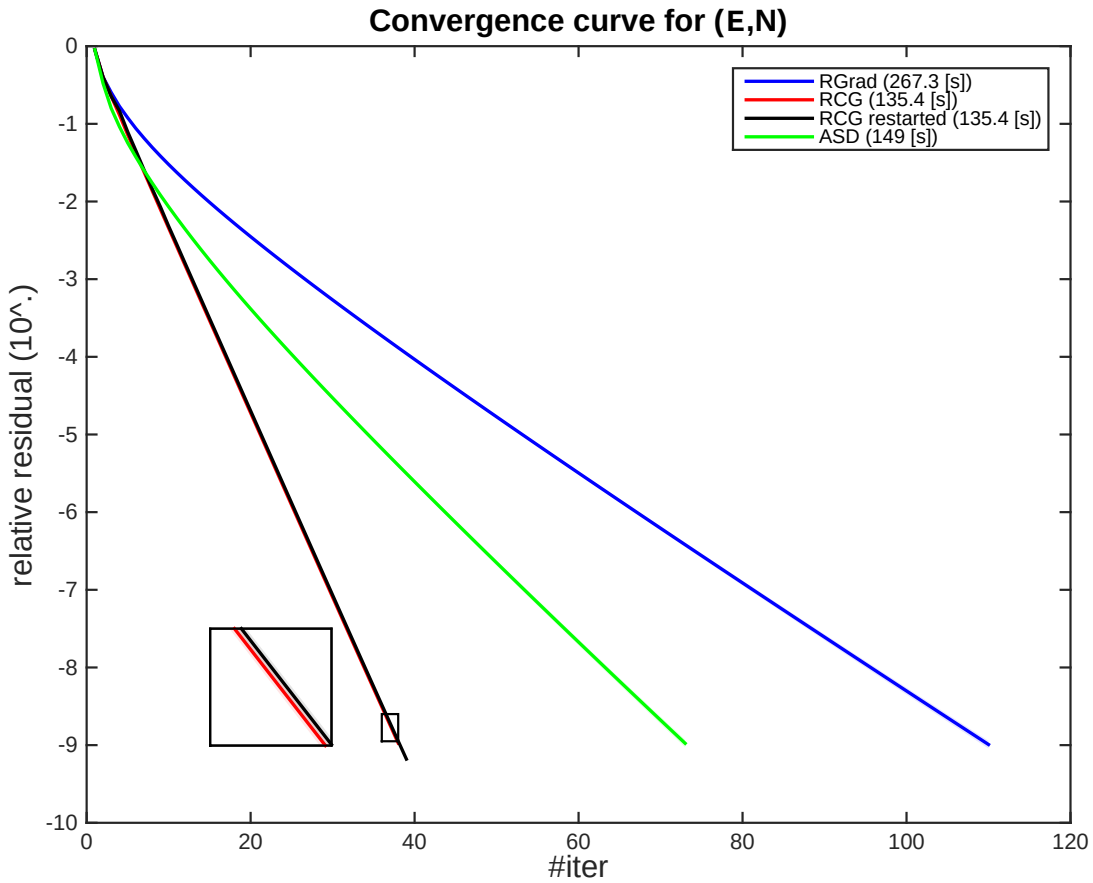
Numerical Experiments: Matrix Completion

Table: Average computational time (seconds) and average number of iterations of RGrad, RCG, RCG restarted, and ASD over ten random rank r matrices per (m, n, p, r) tuple for $m = n \in \{8000, 16000\}$, $r \in \{50, 100\}$ and $p/(m + n - r)r \in \{2, 3\}$; Entry sensing.

r	50						100					
$1/\rho$	2			3			2			3		
	rel.err	iter	time	rel.err	iter	time	rel.err	iter	time	rel.err	iter	time
	$m = n = 8000$											
RGrad	3.2e-05	116	58.9	2.1e-05	52	37.9	3.1e-05	107	184	1.9e-05	54	129
RCG	2.2e-05	36	27.59	1.5e-05	23	24.7	2.1e-05	33	82.0	1.1e-05	22	75.2
RCG res.	2.3e-05	36	27.6	1.6e-05	23	24.6	1.9e-05	34	83.0	1.2e-05	22	75.3
ASD	3.2e-05	89	71.4	2.1e-05	40	38.0	3e-05	74	119	1.9e-05	35	70.0
	$m = n = 16000$											
RGrad	3.2e-05	116	151	2e-05	48	89.3	3.1e-05	97	453	2.1e-05	55	353
RCG	2.3e-05	36	66.9	1.3e-05	24	61.7	2.2e-05	34	209	1.5e-05	22	187
RCG res.	2.1e-05	37	67.4	1.2e-05	24	62.3	2.2e-05	34	211	1.6e-05	22	187
ASD	3.3e-05	92	262	2.1e-05	41	132	3.1e-05	76	351	1.9e-05	36	204

Table: Phase transition table for entry sensing with $m = n = 800$. For each (m, n, p) with $p = \delta \cdot mn$, the algorithm can recover all of the ten random test matrices when $r \leq r_{\min}$, but failes to recover each of the randomly drawn matrices when $r \geq r_{\max}$.

	RGrad				RCG				RCG restarted			
δ	$r_{\min}$	$r_{\max}$	$\rho_{\min}$	$\rho_{\max}$	$r_{\min}$	$r_{\max}$	$\rho_{\min}$	$\rho_{\max}$	$r_{\min}$	$r_{\max}$	$\rho_{\min}$	$\rho_{\max}$
0.1	36	38	0.88	0.93	35	37	0.86	0.9	36	37	0.88	0.9
0.15	55	59	0.89	0.95	55	57	0.89	0.92	55	57	0.89	0.92
0.2	76	78	0.9	0.93	74	77	0.88	0.92	74	77	0.88	0.92
0.25	97	99	0.91	0.93	96	98	0.9	0.92	96	98	0.9	0.92
0.3	119	121	0.92	0.93	117	119	0.9	0.92	117	119	0.9	0.92
0.35	142	143	0.92	0.93	140	142	0.91	0.92	140	142	0.91	0.92
0.4	166	167	0.93	0.93	163	166	0.91	0.93	163	166	0.91	0.93
0.45	190	192	0.93	0.94	188	191	0.92	0.93	188	191	0.92	0.93
0.5	217	219	0.94	0.95	214	217	0.93	0.94	214	217	0.93	0.94
0.55	244	248	0.94	0.95	242	246	0.93	0.95	242	245	0.93	0.94
0.6	274	276	0.95	0.95	272	274	0.94	0.95	272	274	0.94	0.95
0.65	306	308	0.95	0.96	302	306	0.94	0.95	304	306	0.95	0.95
0.7	340	343	0.96	0.96	338	340	0.95	0.96	338	340	0.95	0.96
0.75	378	380	0.96	0.97	374	378	0.96	0.96	374	378	0.96	0.96
0.8	418	422	0.96	0.97	416	420	0.96	0.97	416	420	0.96	0.97
0.85	466	470	0.97	0.98	464	468	0.97	0.97	464	468	0.97	0.97
0.9	524	527	0.98	0.98	522	526	0.98	0.98	522	526	0.98	0.98
0.95	600	604	0.99	0.99	600	604	0.99	0.99	600	604	0.99	0.99


Convergence curve for (E,N)
RGrad (267.3 [s])
RCG (135.4 [s])
RCG restarted (135.4 [s])
ASD (149 [s])
relative residual (10^.)
#iter
0
-1
-2
-3
-4
-5
-6
-7
-8
-9
-10
0
20
40
60
80
100
120

Case III: Guarantee for Phase Retrieval

- Phase Retrieval: Solve $\boldsymbol{x} \in \mathbb{C}^n$ from $|\boldsymbol{A}\boldsymbol{x}| = \boldsymbol{y}$ with known $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ and $\boldsymbol{y} \in \mathbb{R}^m_+$.
- The problem can be reformulated as

$$\mathcal{A}\boldsymbol{X} = \boldsymbol{b},$$

  where $X = \boldsymbol{x}\boldsymbol{x}^*$, $[\mathcal{A}\boldsymbol{X}]_i = \boldsymbol{a}_i^* X \boldsymbol{a}_i$, and $\boldsymbol{b} = \boldsymbol{y}^2$.
- $\mathcal{A}$ doesn't satisfies RIP.

Phase Retrieval

- Initialization: $X_0 = H_1(\mathcal{A}^* y)$
- Use only "good" measurements at each iteration.

Theorem (Cai, Wei, *working paper*, 2017)

Assume entries of $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ *are i.i.d. complex Gaussian. Then the RGrad algorithm converges linearly to* X *with probability at least* $1 - c_0 e^{-c_1 n}$, *provided* $m \geq Cn$.

- Experimental results show RCG is much faster than popular non-convex methods, e.g., Wirtinger flow, truncated Wirtinger flow.

Table of Contents

Low-rank Hankel matrix reconstruction

Find a rank-r Hankel matrix

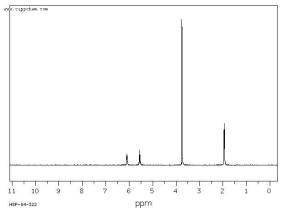
from its partially known anti-diagonals.

$$\begin{bmatrix} x_0 & x_1 & ? & x_3 & ? \\ x_1 & ? & x_3 & ? & x_5 \\ ? & x_3 & ? & x_5 & x_6 \\ x_3 & ? & x_5 & x_6 & ? \\ ? & x_5 & x_6 & ? & x_7 \end{bmatrix}$$

- Standard matrix completion may needs an unnecessarily large number of samples.

Motivating Example: NMR spectroscopy





- The signal can be modelled well by a weighted sum of a few of (damped) multidimensional sinusoids. In the one-dimensional case,

$$x(t) = \sum_{k=1}^{r} d_k e^{2\pi \imath f_k t} e^{-\tau_k t},$$

where $f_k \in [0, 1)$ after normalization and $\tau_k \in \mathbb{R}_+$.

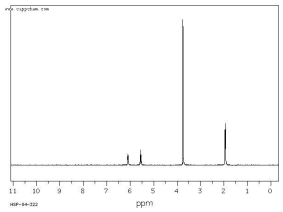
- Unfortunately, the full sampling of

$$\boldsymbol{x} = [x(0),\ x(1),\ x(2), \ldots,\ x(n-1)]^T \in \mathbb{C}^n.$$

for a specimen may take a few weeks.

Motivating Example: NMR spectroscopy





- The signal can be modelled well by a weighted sum of a few of (damped) multidimensional sinusoids. In the one-dimensional case,

$$x(t) = \sum_{k=1}^{r} d_k e^{2\pi \imath f_k t} e^{-\tau_k t},$$

where $f_k \in [0, 1)$ after normalization and $\tau_k \in \mathbb{R}_+$.

- Unfortunately, the full sampling of

$$\boldsymbol{x} = [x(0),\ x(1),\ x(2), \ldots,\ x(n-1)]^T \in \mathbb{C}^n.$$

for a specimen may take a few weeks.

- To save time and cost, non-uniform sampling (NUS) is popular in NMR spectroscopy.

Finite vs Infinite dictionaries

- Assume only $\boldsymbol{x}_\Omega$ with $m := |\Omega| < n$ is observed, and we want to recover $\boldsymbol{x}$.

Finite vs Infinite dictionaries

- Assume only $\boldsymbol{x}_\Omega$ with $m := |\Omega| < n$ is observed, and we want to recover $\boldsymbol{x}$.
- Conventional Compressed Sensing suffers basis mismatch caused by the discretization of the frequency domain $[0, 1)$, and the resolution in the spectral domain is finite.

Finite vs Infinite dictionaries

- Assume only $\boldsymbol{x}_\Omega$ with $m := |\Omega| < n$ is observed, and we want to recover $\boldsymbol{x}$.
- Conventional Compressed Sensing suffers basis mismatch caused by the discretization of the frequency domain $[0, 1)$, and the resolution in the spectral domain is finite.
- To achieve super-resolution, structured matrix completion based methods are proposed. [Tang et.al., 2013; Chen et.al., 2014; Candes et.al., 2012; Cho et.al., 2016; Cai et.al., 2017]

Finite vs Infinite dictionaries

- Assume only $\boldsymbol{x}_\Omega$ with $m := |\Omega| < n$ is observed, and we want to recover $\boldsymbol{x}$.
- Conventional Compressed Sensing suffers basis mismatch caused by the discretization of the frequency domain $[0, 1)$, and the resolution in the spectral domain is finite.
- To achieve super-resolution, structured matrix completion based methods are proposed. [Tang et.al., 2013; Chen et.al., 2014; Candes et.al., 2012; Cho et.al., 2016; Cai et.al., 2017]
- We use the low-rank Hankel formulation.

From Spectrally Sparsity to Low-Rank Hankel Matrix

- Define the Hankel matrix formed by $\boldsymbol{x}$ as

$$\mathcal{H}\boldsymbol{x} = \left[x_{j+k}\right]_{j,k} \in \mathbb{C}^{n_1 \times n_2},$$

where n_1 and n_2 are prescribed integers satisfying $n_1 + n_2 = n - 1$.

- Then $\text{rank}(\mathcal{H}\boldsymbol{x}) = r$ because of the following Vandermonde decomposition:

$$\mathcal{H}\boldsymbol{x} = \underbrace{\boldsymbol{V}_L}_{n_1 \times r} \underbrace{\boldsymbol{D}}_{r \times r} \underbrace{\boldsymbol{V}_R^T}_{r \times n_2},$$

where $\boldsymbol{V}_L = [e^{2\pi\imath f_k j} e^{-\tau_k j}]_{j,k}$, $\boldsymbol{V}_R = [e^{2\pi\imath f_k j} e^{-\tau_k j}]_{j,k}$ are Vandermonde matrices, and $\boldsymbol{D} = \text{diag}(d_1, d_2, \ldots, d_r)$

- The spectrally sparse signal reconstruction can be converted to Low-rank Hankel Matrix Completion:

Find the rank-r Hankel matrix $\mathcal{H}\boldsymbol{x}$

from its partially known anti-diagonals $\boldsymbol{x}_\Omega$.

Our algorithms

We solve the non-convex optimization

$$\min_{\boldsymbol{z}} \sum_{j\in\Omega} |z_j - x_j|^2 \quad \text{s.t.} \quad \operatorname{rank}(\mathcal{H}\boldsymbol{z}) = r.$$

Our algorithms

We solve the non-convex optimization

$$\min_{\boldsymbol{z}} \sum_{j\in\Omega} |z_j - x_j|^2 \quad \text{s.t.} \quad \operatorname{rank}(\mathcal{H}\boldsymbol{z}) = r.$$

- Iterative Hard Thresholding (IHT)

$$\boldsymbol{x}_{\ell+1} = \mathcal{H}^{\dagger}\mathcal{T}_r\mathcal{H}(\boldsymbol{x}_\ell - p^{-1}\mathcal{P}_\Omega(\boldsymbol{x}_\ell - \boldsymbol{x})),$$

Our algorithms

We solve the non-convex optimization

$$\min_{\boldsymbol{z}} \sum_{j\in\Omega} |z_j - x_j|^2 \quad \text{s.t.} \quad \operatorname{rank}(\mathcal{H}\boldsymbol{z}) = r.$$

- Iterative Hard Thresholding (IHT)

$$\boldsymbol{x}_{\ell+1} = \mathcal{H}^\dagger \mathcal{T}_r \mathcal{H}(\boldsymbol{x}_\ell - p^{-1}\mathcal{P}_\Omega(\boldsymbol{x}_\ell - \boldsymbol{x})),$$

- To avoid large scale SVD, we apply our new framework for low-rank matrix reconstruction to get Fast IHT (FIHT)

$$\boldsymbol{x}_{\ell+1} = \mathcal{H}^\dagger \mathcal{T}_r \mathcal{P}_{\mathcal{S}_\ell} \mathcal{H}(\boldsymbol{x}_\ell - p^{-1}\mathcal{P}_\Omega(\boldsymbol{x}_\ell - \boldsymbol{x})),$$

Our algorithms

We solve the non-convex optimization

$$\min_{\boldsymbol{z}} \sum_{j\in\Omega} |z_j - x_j|^2 \quad \text{s.t.} \quad \operatorname{rank}(\mathcal{H}\boldsymbol{z}) = r.$$

- Iterative Hard Thresholding (IHT)

$$\boldsymbol{x}_{\ell+1} = \mathcal{H}^\dagger \mathcal{T}_r \mathcal{H}(\boldsymbol{x}_\ell - p^{-1}\mathcal{P}_\Omega(\boldsymbol{x}_\ell - \boldsymbol{x})),$$

- To avoid large scale SVD, we apply our new framework for low-rank matrix reconstruction to get Fast IHT (FIHT)

$$\boldsymbol{x}_{\ell+1} = \mathcal{H}^\dagger \mathcal{T}_r \mathcal{P}_{\mathcal{S}_\ell} \mathcal{H}(\boldsymbol{x}_\ell - p^{-1}\mathcal{P}_\Omega(\boldsymbol{x}_\ell - \boldsymbol{x})),$$

  - Every step can be implemented by FFTs.
  - Can use Takagi decompostion to save half computational cost when the matrix is square.

Theoretical Guarantee of FIHT

We will show that FIHT converges to $\boldsymbol{x}$ linearly provided $m \sim O(r^2 \log^2 n)$

- Assumptions:
  - The elements of Ω is sampled independently and uniformly from $\{0, 1, \ldots, n-1\}$ with replacement.
  - $\mathcal{H}\boldsymbol{x}$ is μ_0-incoherent, which may be viewed as a condition on the separation of frequencies.

Definition

The Hankel matrix $\mathcal{H}\boldsymbol{x}$ with the Vandermonde decomposition $\mathcal{H}\boldsymbol{x} = \boldsymbol{V}_L \boldsymbol{D} \boldsymbol{V}_R^T$ is said of μ_0-incoherent if

$$\sigma_{\min}(\boldsymbol{V}_L^* \boldsymbol{V}_L) \geq \frac{n_1}{\mu_0}, \quad \sigma_{\min}(\boldsymbol{V}_R^* \boldsymbol{V}_R) \geq \frac{n_2}{\mu_0}$$

Guarantee of FIHT

FIHT converges linearly to the correct solution when it is initialized by $L = O(\log n)$ resampling and trimming, provided $m \sim O(r^2 \log^2 n)$.

Theorem (Theoretical Guarantee of FIHT,[Cai, Wang, Wei, 2017])

Assume $\mathcal{H}\boldsymbol{x}$ is μ_0-incoherent. Let $0 < \varepsilon_0 < \frac{1}{10}$ and $L = \left\lceil 6 \log\left(\frac{\sqrt{n}\log(n)}{16\varepsilon_0}\right)\right\rceil$. Define $\nu = 10\varepsilon_0 < 1$. Then with probability at least $1 - (2L+3)\, n^{-2}$, the iterates generated by FIHT with our initialization satisfies

$$\|\boldsymbol{x}_\ell - \boldsymbol{x}\| \le \nu^\ell \|\boldsymbol{L}_0 - \mathcal{H}\boldsymbol{x}\|_F,$$

provided

$$m \ge C\mu_0 c_s \kappa^6 r^2 \log(n) \log\left(\frac{\sqrt{n}\log(n)}{16\varepsilon_0}\right)$$

for some universal constant $C > 0$.

Table of Contents

Conclusion

- The new framework of applying $\mathcal{H}_r\mathcal{P}_{\mathcal{S}_\ell}$ is better than $\mathcal{H}_r$ solely for low-rank matrix recovery problems.

Conclusion

- The new framework of applying $\mathcal{H}_r\mathcal{P}_{\mathcal{S}_\ell}$ is better than $\mathcal{H}_r$ solely for low-rank matrix recovery problems.
- The projection $\mathcal{P}_{\mathcal{S}_\ell}$ onto the tangent space helps
  - Computationally: reduce SVD of size $n \times n$ to $O(r) \times O(r)$.
  - Theoretically: help to prove the theoretical guarantee, because the isometric property holds true only in the tangent space.

References:

[1] K. Wei, **J.-F. Cai**, T.F. Chan and S. Leung, Guarantees of Riemannian Optimization for Low Rank Matrix Recovery, *SIAM J. Matrix Anal. & Appl.*, 37(3):1198–1222, 2016.

[2] K. Wei, **J.-F. Cai**, T.F. Chan and S. Leung, Guarantees of Riemannian Optimization for Low Rank Matrix Completion, *preprint*, 2016.

[3] **J.-F. Cai**, T. Wang, and K. Wei, Fast and Provable Algorithms for Spectrally Sparse Signal Reconstruction via Low-Rank Hankel Matrix Completion, *Applied and Computational Harmonic Analysis*, to appear.

[4] **J.-F. Cai**, and K. Wei, Phase Retrival via Riemannian Optimization: Theory and Algorithms, *in preparation*, 2017.

Thanks for your attention!

Questions?