Parcimonie en traitement du signal et des images

Partie 4: au delà de la parcimonie

Génie Mathématique - INSA
21/23/28 novembre 2016

N. Bertin, C. Herzet, A. Roumy, *Rémi Gribonval , A. Deleforge
remi.gribonval@inria.fr

Overall course content

Part 1 (N. Bertin): Fundamentals

- Definitions, first theoretical results, basic algorithmic principles

Part 2 (C. Herzet): Theoretical guarantees

- Finer conditions for feasibility and convergence

Part 3 (A. Roumy): Compressed sensing, probability results

- More conditions, information theory, number of measurements

Part 4 (R. Gribonval) & A. Deleforge): Beyond sparsity

- **Today**: From sparse vectors to low-rank matrices
- **Next**: Compressed matrix sensing; Well-posedness and algorithms for generic low-dimensional models; Dictionary learning

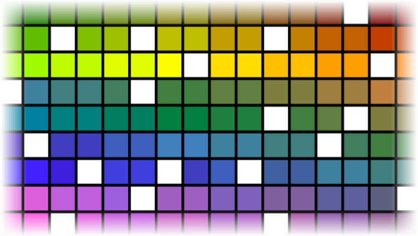
Detailed content of this part

1. Introduction: inverse problems **with matrices**
2. **Low-rank matrices**: definitions and reminders
3. Well-posedness of the **low-rank recovery problem**: a key result
4. Low-rank **recovery algorithms**: principles and first algorithms
5. Theoretical **guarantees**
6. **Dimension reduction**
7. Summary

Examples of matrix inverse problems: Matrix completion

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

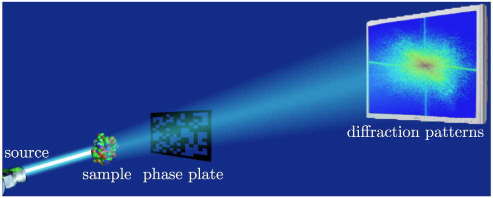
Examples of matrix inverse problems: Matrix completion



Goal = complete a large matrix

- Rows = movies (potentially tens of thousands)
- Columns = users (potentially several millions)
- Many missing entries (99.9% if each user rates ten items)

Examples of matrix inverse problems: Phase retrieval



- **Magnitude only Fourier measurements**: $y_i = |\langle \mathbf{a}_i, \mathbf{x} \rangle|$
  - **phase ambiguity**: no uniqueness $y_i = |\langle \mathbf{a}_i, \mathbf{x} \rangle| = |\langle \mathbf{a}_i, e^{j\phi}\mathbf{x} \rangle|$
  - **non-linear** inverse problem in vector $\mathbf{x}$ ...
  - ...yet convertible to *linear problem* in the matrix $\mathbf{X} \triangleq \mathbf{x}\mathbf{x}^H$:

$$z_i \triangleq y_i^2 = |\langle \mathbf{a}_i, \mathbf{x} \rangle|^2 = \mathbf{a}_i^H \mathbf{x}\mathbf{x}^H \mathbf{a}_i$$

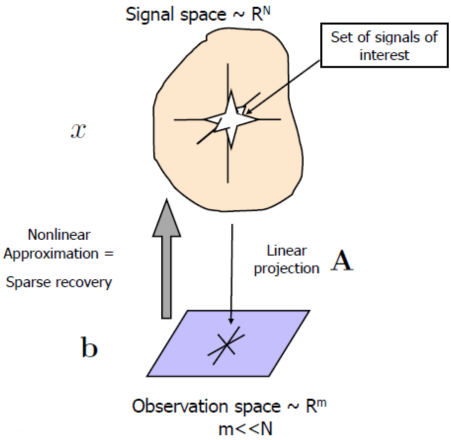
Inverse problems **with matrices**: mathematical expression

Matrix $\leftrightarrow$ vector conversions

- matrix to vector: $\mathbf{u} = \mathrm{vec}(\mathbf{U})$
- vector to matrix (of given size): $\mathbf{U} = \mathrm{mat}(\mathbf{u})$

- Linear observation operator $\mathcal{M}(\mathbf{X}) \triangleq (\langle \mathbf{a}_i, \mathrm{vec}(\mathbf{X}) \rangle)_{i=1}^m$
- Linear inverse problem:
  find $p \times q$ matrix $\mathbf{X}$ given $\mathbf{y} = \mathcal{M}(\mathbf{X})$, of dimension m
- Under-determined if $m < pq$ $\Rightarrow$ need a model

Reminder: iconic inverse problem with sparse regularization



Here: signals $\rightarrow$ matrices; what set of "matrices of interest" ?

Image courtesy of Mike Davies, Univ. Edinburgh.

Low-rank matrices: definitions and reminders

Definition: low-rank matrix

Rank of a matrix (real or complex)

- The rank of a $p \times q$ matrix $\mathbf{X}$, $\mathrm{rank}(\mathbf{X})$ is the dimension of the span of its columns (or equivalently of its rows).
- Given the SVD $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, with $\mathbf{\Sigma}$ "diagonal", we have $\mathrm{rank}(\mathbf{X}) = \|\mathrm{diag}(\mathbf{\Sigma})\|_0$. As a result $\mathrm{rank}(\mathbf{X}) \leq \min(p, q)$.

Informally, "low-rank" means $\mathrm{rank}(\mathbf{X}) \ll \min(p, q)$.

Definition: set of low-rank matrices

Low-rank matrices (real or complex)

- A $p \times q$ matrix $\mathbf{X}$ is *of rank at most* r iff it can be written as
  - $\mathbf{X} = \mathbf{U}\mathbf{V}^T$ where $\mathbf{U}$ is $p \times r$, $\mathbf{V}$ is $q \times r$; or equivalently
  - $\mathbf{X} = \sum_{i=1}^{r} \mathbf{u}_i \mathbf{v}_i^T$, where $\mathbf{u}_i$ is $p \times 1$, $\mathbf{v}_i$ is $q \times 1$.
- The set of all matrices of rank *at most* r is denoted:

$$\Sigma_r := \{\mathbf{X} \in \mathbb{R}^{p \times q}, \mathrm{rank}(\mathbf{X}) \leqslant r\}$$

Definition: set of low-rank matrices

Low-rank matrices (real or complex)

- A $p \times q$ matrix $\mathbf{X}$ is *of rank at most* r iff it can be written as
  - $\mathbf{X} = \mathbf{U}\mathbf{V}^T$ where $\mathbf{U}$ is $p \times r$, $\mathbf{V}$ is $q \times r$; or equivalently
  - $\mathbf{X} = \sum_{i=1}^{r} \mathbf{u}_i \mathbf{v}_i^T$, where $\mathbf{u}_i$ is $p \times 1$, $\mathbf{v}_i$ is $q \times 1$.
- The set of all matrices of rank *at most* r is denoted:

$$\Sigma_r := \{\mathbf{X} \in \mathbb{R}^{p \times q}, \mathrm{rank}(\mathbf{X}) \leqslant r\}$$

- Σ_r is **not a linear subspace** of the space of $p \times q$ matrices: If $\mathbf{X}, \mathbf{Y} \in \Sigma_r$, then $\mathbf{X} + \mathbf{Y} \in \Sigma_{2r}$ but in general $\mathbf{X} + \mathbf{Y} \notin \Sigma_r$

Definition: set of low-rank matrices

Low-rank matrices (real or complex)

- A $p \times q$ matrix $\mathbf{X}$ is *of rank at most* r iff it can be written as
  - $\mathbf{X} = \mathbf{U}\mathbf{V}^T$ where $\mathbf{U}$ is $p \times r$, $\mathbf{V}$ is $q \times r$; or equivalently
  - $\mathbf{X} = \sum_{i=1}^{r} \mathbf{u}_i \mathbf{v}_i^T$, where $\mathbf{u}_i$ is $p \times 1$, $\mathbf{v}_i$ is $q \times 1$.
- The set of all matrices of rank *at most* r is denoted:

$$\Sigma_r := \{\mathbf{X} \in \mathbb{R}^{p \times q}, \operatorname{rank}(\mathbf{X}) \leqslant r\}$$

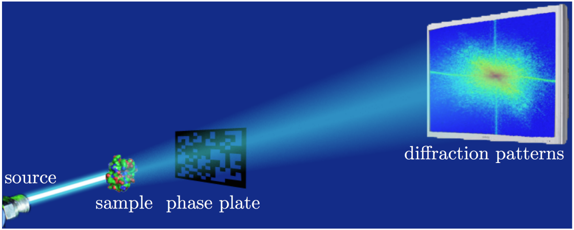
- Σ_r is **not a linear subspace** of the space of $p \times q$ matrices: If $\mathbf{X}, \mathbf{Y} \in \Sigma_r$, then $\mathbf{X} + \mathbf{Y} \in \Sigma_{2r}$ but in general $\mathbf{X} + \mathbf{Y} \notin \Sigma_r$
- In the sense of manifolds,

$$\dim(\Sigma_r) = r(p + q - r).$$

For small r, this is approximately $r(p+q)$.

Intuition: p degrees of freedom for each $\mathbf{u}_i$; q for each $\mathbf{v}_i$; $\Rightarrow p + q$ for each $\mathbf{u}_i \mathbf{v}_i^T$

Examples of low-rank matrices: Phase retrieval



- **Magnitude only measurements**: $y_i = |\langle \mathbf{a}_i, \mathbf{x} \rangle|$
  - **phase ambiguity**: no uniqueness $y_i = |\langle \mathbf{a}_i, \mathbf{x} \rangle| = |\langle \mathbf{a}_i, e^{j\phi}\mathbf{x} \rangle|$
  - **non-linear** inverse problem in vector $\mathbf{x}$ ...
  - ...yet convertible to *linear problem* in the matrix $\mathbf{X} \triangleq \mathbf{x}\mathbf{x}^H$:

$$z_i \triangleq y_i^2 = |\langle \mathbf{a}_i, \mathbf{x} \rangle|^2 = \mathbf{a}_i^H \mathbf{x}\mathbf{x}^H \mathbf{a}_i$$

Examples of low-rank matrices: Matrix completion

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

Simplistic user similarity model:

- r user categories;
- shared user preference profile in category i: $\mathbf{u}_i$;
- users in category i indicated by nonzero entry in $\mathbf{v}_i^T$;
- full matrix written $\mathbf{X} = \sum_{i=1}^{r} \mathbf{u}_i \mathbf{v}_i^T$.

Low-rank *approximation*

Real-world situations can deviate from the exact low-rank model.

- **Noise**:

$$\mathbf{y} = \mathcal{M}(\mathbf{X}) + \varepsilon$$

- **Approximate low-rank**:

$$\mathbf{X} \approx \mathbf{Z}$$

  where $\mathbf{Z}$ is a rank-r matrix. The matrix $\mathbf{X}$ is then often said to be **compressible** (rather than low-rank, which it is not).

Reminders on norms

$\|.\| : \mathbf{X} \to \mathbb{R}^+$ is:

- A norm iff for all $\mathbf{X}, \mathbf{Y}, \lambda$:
  (i) $\|\mathbf{X}\| = 0$ iff $\mathbf{X} = \mathbf{0}$ (definiteness)
  (ii) $\|\lambda \mathbf{X}\| = |\lambda|.\|\mathbf{X}\|$ (homogeneity)
  (iii) $\|\mathbf{X} + \mathbf{Y}\| \leqslant \|\mathbf{X}\| + \|\mathbf{Y}|$ (triangle inequality)
- A quasi-norm: (i), (ii) and for some constant C
  (iii) $\|\mathbf{X} + \mathbf{Y}\| \leqslant C(\|\mathbf{X}\| + \|\mathbf{Y}\|)$.

Shatten norms S_p

Shatten norm of a matrix

For $0 \leq p \leq \infty$: using the SVD, $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, define

$$\|\mathbf{M}\|_{S_p} \triangleq \|\mathrm{diag}(\mathbf{\Sigma})\|_p$$

This is a quasinorm for $0 < p < 1$; a norm for $1 \leq p \leq \infty$.

Special cases:

- $p = 0$: **rank** $\quad \mathrm{rank}(\mathbf{M}) = \|\mathbf{M}\|_{S_0}$
- $p = 1$: **trace/nuclear norm** $\quad \|\mathbf{M}\|_\star \triangleq \mathrm{trace}(\mathbf{\Sigma}) = \|\mathbf{M}\|_{S_1}$
- $p = 2$: **Frobenius norm** $\quad \|\mathbf{M}\|_F \triangleq \sqrt{\sum_{ij} \mathbf{M}_{ij}^2} = \|\mathbf{M}\|_{S_2}$
- $p = \infty$: **Spectral norm** $\quad \|\mathbf{M}\|_{op} \triangleq \sup_{\|\mathbf{x}\|_2 = 1} \|\mathbf{M}\mathbf{x}\|_2 = \|\mathbf{M}\|_{S_\infty}$

Best low-rank *approximation*

- The **error of best rank-r approximation of $\mathbf{X}$ is**:

$$\sigma_r(\mathbf{X}) \triangleq \inf\{\|\mathbf{X} - \mathbf{Z}\|, \mathrm{rank}(\mathbf{Z}) \leq r\}$$

- Consider $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ the SVD of $\mathbf{X}$. The matrix $\mathbf{Z} \triangleq \mathbf{U}\hat{\mathbf{\Sigma}}\mathbf{V}^T$, where $\hat{\mathbf{\Sigma}}$ matches $\mathbf{\Sigma}$ on the r largest diagonal entries and is zero everywhere else, realizes this infimum for Shatten norms $\|\mathbf{X} - \mathbf{Z}\|_{S_p}$, no matter the value of p (>0).

Problem formulation: Ideal low-rank regularization

Given the observation $\mathbf{y}$, with known measurement operator $\mathcal{M}$, we wish to solve:

$$\min_{\mathbf{X}} \operatorname{rank}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{y} = \mathcal{M}(\mathbf{X})$$

Contents

Well-posedness of the low-rank recovery problem

Given a low-rank matrix $\mathbf{X}_0$ and $\mathbf{y} \triangleq \mathcal{M}(\mathbf{X}_0)$, consider the **low-rank matrix recovery problem**:

$$\min_{\mathbf{X}} \operatorname{rank}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{y} = \mathcal{M}(\mathbf{X})$$

Theorem. Well-posedness of the low-rank recovery problem

The following properties are equivalent.

(i) Uniqueness of solutions of rank at most r: for any pair of matrices $(\mathbf{X}_0,\mathbf{X}_1)$ of rank at most r, if $\mathcal{M}(\mathbf{X_0}) = \mathcal{M}(\mathbf{X_1})$ then $\mathbf{X}_0 = \mathbf{X}_1$.

(ii) The null space $\operatorname{Ker}(\mathcal{M})$ does not contain any matrix of rank at most $2r$ other than the zero matrix.

Well-posedness of the low-rank recovery problem

A shorter, easy to memorize formulation of the previous theorem is:

$$\boxed{\begin{array}{c} \forall \mathbf{X}_0, \mathbf{X}_1 \in \Sigma_r,\ \mathcal{M}(\mathbf{X}_0) = \mathcal{M}(\mathbf{X}_1) \Rightarrow \mathbf{X}_0 = \mathbf{X}_1 \\ \Leftrightarrow \\ \mathrm{Ker}(\mathcal{M}) \cap \Sigma_{2r} = \{\mathbf{0}\} \end{array}}$$

Problem (Homework): consequence for Matrix Completion

- what is the measurement operator $\mathcal{M}$?
- what is the rank of an s-sparse matrix $\mathbf{X}$?
- for what rank r does the above property hold ?

Demonstration

Homework

Prove the theorem.

Number of measurements and sparsity

Comments on the previous theorem:

- This is a **worst case analysis**
  - does not provide guarantees for matrix completion.
  - more advanced analysis with *random sampling* (random missing entries) and *incoherence* of $\mathbf{X}$ are available.
- Necessary number of measurements: .

$$m \geq \dim(\Sigma_{2r}) = 2r(p+q-2r)$$

Number of measurements and sparsity

Comments on the previous theorem:

- This is a **worst case analysis**
  - does not provide guarantees for matrix completion.
  - more advanced analysis with *random sampling* (random missing entries) and *incoherence* of $\mathbf{X}$ are available.
- Necessary number of measurements: .

$$m \geq \dim(\Sigma_{2r}) = 2r(p+q-2r) \ll pq \qquad \text{for small } r$$

Contents

Rank minimization is NP hard

We want to solve the problem:

$$\min_{\mathbf{X}} \operatorname{rank}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{y} = \mathcal{M}(\mathbf{X})$$

Bad luck : this is NP-hard, just as the ℓ^0 minimization problem!
(not really a surprise perhaps?)

Three practical philosophies

$$\min_{\mathbf{X}} \operatorname{rank}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{y} = \mathcal{M}(\mathbf{X})$$

Idea 1 | **Idea 2** | **Idea 3**

Focus on $\operatorname{rank}(\mathbf{X})$

1. Add rank-one component
2. Check if $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$
3. Do it again until happy

Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$
2. Force it to be low-rank
3. Do it again until happy

Solve a nicer problem

1. Replace $\operatorname{rank}(\cdot)$ by nicer norm
2. Write a convex optim. problem
3. Use your favorite solver

Three practical philosophies

$$\min_{\mathbf{X}} \operatorname{rank}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{y} = \mathcal{M}(\mathbf{X})$$

Idea 1

Focus on $\operatorname{rank}(\mathbf{X})$

1. Add rank-one component
2. Check if $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$
3. Do it again until happy

Greedy algorithms

Idea 2

Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$
2. Force it to be low-rank
3. Do it again until happy

Idea 3

Solve a nicer problem

1. Replace $\operatorname{rank}(\cdot)$ by nicer norm
2. Write a convex optim. problem
3. Use your favorite solver

Three practical philosophies

$$\min_{\mathbf{X}} \operatorname{rank}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{y} = \mathcal{M}(\mathbf{X})$$

Idea 1

Focus on $\operatorname{rank}(\mathbf{X})$

1. Add rank-one component
2. Check if $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$
3. Do it again until happy

Greedy algorithms

Idea 2

Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$
2. Force it to be low-rank
3. Do it again until happy

Hard thresholding algorithms

Idea 3

Solve a nicer problem

1. Replace $\operatorname{rank}(\cdot)$ by nicer norm
2. Write a convex optim. problem
3. Use your favorite solver

Three practical philosophies

$$\min_{\mathbf{X}} \operatorname{rank}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{y} = \mathcal{M}(\mathbf{X})$$

Idea 1 | **Idea 2** | **Idea 3**

Focus on $\operatorname{rank}(\mathbf{X})$

1. Add rank-one component
2. Check if $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$
3. Do it again until happy

Greedy algorithms

Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$
2. Force it to be low-rank
3. Do it again until happy

Hard thresholding algorithms

Solve a nicer problem

1. Replace $\operatorname{rank}(\cdot)$ by nicer norm
2. Write a convex optim. problem
3. Use your favorite solver

Convex relaxation algorithms

Greedy algorithms

Kiryung Lee and Yoram Bresler. Admira: Atomic decomposition for minimum rank approximation, 2009. arXiv:0905.0044

Hard thresholding algorithms

Hard thresholding algorithms

- Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$
- Idea:
  1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$

Hard thresholding algorithms

Hard thresholding algorithms

- Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$
- Idea:
  1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined

Hard thresholding algorithms

Hard thresholding algorithms

- Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$
- Idea:
  1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
     - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
     - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

Hard thresholding algorithms

Hard thresholding algorithms

- Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$
- Idea:

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
   - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
   - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

2. Force it to be low-rank

Hard thresholding algorithms

Hard thresholding algorithms

- Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$
- Idea:
  1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
     - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
     - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

  2. Force it to be low-rank $\Rightarrow$ best rank-r approximation

Hard thresholding algorithms

Hard thresholding algorithms

- Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$
- Idea:
  1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
     - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
     - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

  2. Force it to be low-rank $\Rightarrow$ best rank-r approximation
     - keep r largest singular values of $\mathbf{X}_{n+1/2}$, set the other to zero.
     - that is to say **hard thresholding** of singular values with $H_r(\cdot)$.

Hard thresholding algorithms

Hard thresholding algorithms

- Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$
- Idea:

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
   - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
   - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

2. Force it to be low-rank $\Rightarrow$ best rank-r approximation
   - keep r largest singular values of $\mathbf{X}_{n+1/2}$, set the other to zero.
   - that is to say **hard thresholding** of singular values with $H_r(\cdot)$.
3. Do it again until happy

Hard thresholding algorithms

Hard thresholding algorithms

- Focus on $\mathbf{y} \approx \mathcal{M}(\mathbf{X})$
- Idea:

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
   - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
   - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

2. Force it to be low-rank $\Rightarrow$ best rank-r approximation
   - keep r largest singular values of $\mathbf{X}_{n+1/2}$, set the other to zero.
   - that is to say **hard thresholding** of singular values with $H_r(\cdot)$.
3. Do it again until happy $\Rightarrow$ stopping criterion

Summary: Singular Value Projection

Singular Value Projection (SVP)

Require: $\mathbf{y}$, $\mathcal{M}$, r

1: Initialize estimate: $\mathbf{X}_0 = \mathbf{0}$
2: **while** (some stopping criterion is met) **do**
3: $\quad \mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$
4: $\quad [\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = SVD(\mathbf{X}_{n+1/2})$
5: $\quad \mathbf{X}_{n+1} = \mathbf{U}\mathrm{diag}(H_r(\mathrm{diag}(\mathbf{\Sigma})))\mathbf{V}^T$
6: **end while**
7: **return** $\mathbf{X}_n$

- Inspired by Iterative Hard Thresholding (IHT) for sparse recovery, see e.g. [Blumensath & Davies, 2008] (first paper with theoretical results: convergence to a stationary point)
- Described with recovery guarantees (see later) in: R. Meka, P. Jain and I. S. Dhillon, Guaranteed rank minimization via singular value projection, Advances in Neural Information Processing Systems, (2010), pp. 937–945

Comments on SVP

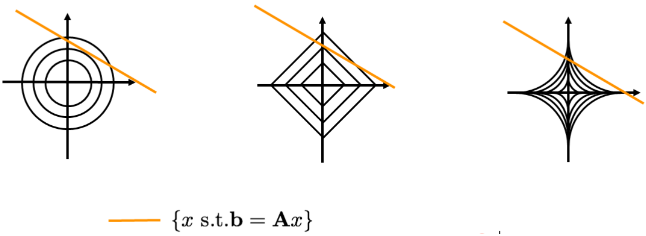
- Requires to know the expected rank r
- Requires that $||\mathcal{M}||_{op} < 1$ for convergence
- Recent works refines this to avoid cost of full SVD (with "lazy" SVD to get part associated to $2r$ largest singular values only)

To go further

- Many other algorithms exist of this type exist, for instance the introduction of a step size in the gradient descent step: $\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mu_n \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n)))$
- Some variants focus on dealing with the case where r is unknown:
  - Iterative *soft* thresholding or *shrinkage* (see next: convex relaxations)
  - Varying r along the algorithm

Convex relaxation: reminder on ℓ_p norms for sparse recovery

We can get a visual intuition of the interest of ℓ_p norms for that:

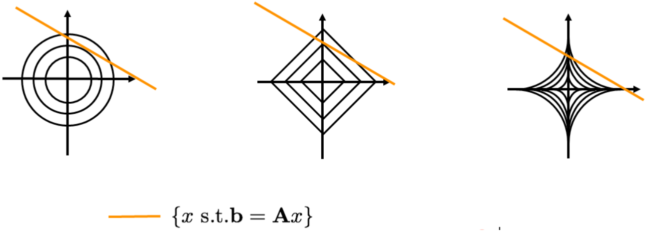


Both **convex** and **promoting sparsity**

- **For sparse recovery**: ℓ_1 norm

Convex relaxation: reminder on ℓ_p norms for sparse recovery

We can get a visual intuition of the interest of ℓ_p norms for that:



Both **convex** and **promoting sparsity** / **promoting low-rank**

- **For sparse recovery**: ℓ_1 norm
- **For low-rank recovery**: **trace/nuclear/Shatten-1 norm**

$$\|\mathbf{X}\|_\star = \|\mathbf{X}\|_{S_1} = \mathrm{Trace}(\mathbf{\Sigma}(\mathbf{X})) = \|\mathrm{diag}(\mathbf{\Sigma}(\mathbf{X}))\|_1$$

Low-rank recovery as optimization problems

- **Approximation**

$$\min_{\mathbf{X}} \|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 \quad \text{such that} \quad \|\mathbf{X}\|_\star \leqslant \eta$$

- **Rank reduction**

$$\min_{\mathbf{x}} \|\mathbf{X}\|_\star \quad \text{such that} \quad \|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 \leqslant \varepsilon$$

- **Regularization**

$$\frac{1}{2} \min_{\mathbf{x}} \|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda \|\mathbf{X}\|_\star$$

All can be cast as Second Order Cone Programs (SOCP) and addressed with standard tools. This however does not take into account their specific structure.

Black boxes

Cvx in Matlab:

```
p=10;q=11;
m=10;
M = randn(m,p*q);
y = randn(m,1);
cvx_begin
  variable X(p,q)
  minimize ( norm_nuc(X) )
    subject to
    M*X(:)  = y
cvx_end
```

To go further

- Usage of generic optimization algorithms to solve those problems may not take benefit of their particularities.
- Advanced specific algorithms ? No surprise: yes, just as for ℓ_1 !

Soft thresholding algorithms

Soft thresholding algorithms

- Idea to adress regularized problem

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_\star$$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$

Soft thresholding algorithms

Soft thresholding algorithms

- Idea to adress regularized problem

$$\min_{\mathbf{X}} \tfrac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_\star$$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined

Soft thresholding algorithms

Soft thresholding algorithms

- Idea to adress regularized problem

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_\star$$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
   - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
   - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

Soft thresholding algorithms

Soft thresholding algorithms

- Idea to adress regularized problem

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_\star$$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
    - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
    - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

2. Force small nuclear norm

Soft thresholding algorithms

Soft thresholding algorithms

- Idea to adress regularized problem

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_\star$$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
    - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
    - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

2. Force small nuclear norm $\Rightarrow$ singular value thresholding

Soft thresholding algorithms

Soft thresholding algorithms

- Idea to adress regularized problem

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_\star$$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
   - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
   - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

2. Force small nuclear norm $\Rightarrow$ singular value thresholding
   - **soft thresholding** of singular values $\mathbf{\Sigma}(\mathbf{X}_{n+1/2})$ with $S_\lambda(\cdot)$.

Soft thresholding algorithms

Soft thresholding algorithms

- Idea to adress regularized problem

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_\star$$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
    - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
    - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

2. Force small nuclear norm $\Rightarrow$ singular value thresholding
    - **soft thresholding** of singular values $\boldsymbol{\Sigma}(\mathbf{X}_{n+1/2})$ with $S_\lambda(\cdot)$.
3. Do it again until happy

Soft thresholding algorithms

Soft thresholding algorithms

- Idea to adress regularized problem

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_\star$$

1. Find some $\mathbf{y} \approx \mathcal{M}(\mathbf{X}_n)$ $\Rightarrow$ underdetermined
   - replace by: decrease the error $\mathbf{y} - \mathcal{M}(\mathbf{X}_n)$ at each iteration n
   - in practice: use of a **gradient descent** (Landweber iterations):

$$\mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$$

2. Force small nuclear norm $\Rightarrow$ singular value thresholding
   - **soft thresholding** of singular values $\mathbf{\Sigma}(\mathbf{X}_{n+1/2})$ with $S_\lambda(\cdot)$.
3. Do it again until happy $\Rightarrow$ stopping criterion

Summary: Singular Value Thresholding

Singular Value Thresholding (SVT)

Require: $\mathbf{y}$, $\mathcal{M}$, r

1: Initialize estimate: $\mathbf{X}_0 = \mathbf{0}$
2: **while** (some stopping criterion is met) **do**
3: $\quad \mathbf{X}_{n+1/2} = \mathbf{X}_n + \mathcal{M}^*(\mathbf{y} - \mathcal{M}(\mathbf{X}_n))$
4: $\quad [\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = SVD(\mathbf{X}_{n+1/2})$
5: $\quad \mathbf{X}_{n+1} = \mathbf{U}\mathrm{diag}(S_\lambda(\mathrm{diag}(\mathbf{\Sigma})))\mathbf{V}^T$
6: **end while**
7: **return** $\mathbf{X}_n$

- Described e.g. in [Cai, Candès, Chen 2010]
- Inspired by Iterative Shrinkage Thresholding Algorithm (ISTA) for sparse recovery, see e.g. [Daubechies, De Frise, De Mol, 2004] for global convergence guarantees

Comments on SVT

- Does not require to know the expected rank r ...
- ... but requires to choose the regularization parameter λ (serves as a threshold)
- Requires that $||\mathcal{M}||_{op} < 1$ for convergence
- Recent works refines this to avoid cost of full SVD

A tentative big picture

	Iterative/Greedy	Optimization
Principle	$\mathbf{r}_n = \mathbf{y} - \mathcal{M}(\mathbf{X}_n)$	$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{M}(\mathbf{X}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{X}\|_{S_p}^p$
Tuning	Stopping criterion Target rank r	Regularization parameter λ
Variants	Selection criterion Update strategy	Choice of sparsity measure p Optimization algorithm

Homework 1

Repeat the steps of tutorial session 1 with a `cvx` implementation of nuclear norm minimization; with SVP instead of IHT; with SVT instead of ISTA.