Matrix Decompositions, Approximation, and Completion

Emilia Magnani & Luca Pedrazzini

ETH Zürich

emagnani@student.ethz.ch & lucape@student.ethz.ch

19.11.2018

Overview

1. Main idea

Given data in the form of an $m \times n$ matrix $\mathbf{Z} = \{z_{ij}\}$, we look for a matrix $\hat{\mathbf{Z}}$ that approximates $\mathbf{Z}$.

- We want to gain understanding of the matrix $\mathbf{Z}$ through an approximation $\hat{\mathbf{Z}}$ that has a simple structure.
  The general approach is to consider estimators based on optimisation problems of the form

$$\hat{\mathbf{Z}} = \arg \min_{\mathbf{M} \in \mathbb{R}^{m \times n}} \| \mathbf{Z} - \mathbf{M} \|_F^2 \quad \text{subject to} \quad \Phi(\mathbf{M}) \leq c,$$

  where $\| \cdot \|_F^2$ is the (squared) Frobenius norm and $\Phi(\cdot)$ is a constraint function that encourages $\hat{\mathbf{Z}}$ to be sparse in some general sense.

- When $\mathbf{Z}$ has missing entries we would like to impute or fill the missing entries in $\mathbf{Z}$. This problem is known as *matrix completion*.

2. Missing values and matrix completion

In many applications measured data can be represented in a matrix $\mathbf{Z}_{m \times n}$, for which only a small number of entries are observed.

The problem is to "complete" the matrix based on the observed entries (*Matrix completion problem*, Candès and Recht, 2008; Candès and Tao, 2009; Rennie and Srebro, 2005).

The problem needs additional constraints on the unknown matrix $\mathbf{Z}$, and one common choice is a *rank constraint*.

Example: The NETFLIX Movie Challenge

Netflix launched a competition in 2006 to improve their system for recommending movies to their customers.[1]

	Dirty Dancing	Meet the Parents	Top Gun	The Sixth Sense	Catch Me If You Can	The Royal Tenenbaums	Con Air	Big Fish	The Matrix	A Few Good Men
Customer 1					4					
Customer 2			3				3			3
Customer 3		2		4					2	
Customer 4	3									
Customer 5	5	5			4					
Customer 6						2	4			
Customer 7			5					3		
Customer 8						2				3
Customer 9	3				5			5		
Customer 10										

$m = 480,189$ customers (rows),
$n = 17,770$ movies (columns).
8.6 billion potential entries

$z_{i,j} \in \{1, \dots 5\}$ rating
by viewer i for movie j

- The data matrix is very sparse: only 1% of the entries is observed.
- Contest goal: to predict the ratings for unrated movies and improve the RMSE by at least 10%.

[1] *Source:* Statistical Learning with Sparsity: The Lasso and Generalizations (p.172) [36]

Singular Value Decomposition (SVD)

Let $\mathbf{Z}$ be a $m \times n$ matrix with $m \geq n$.
Its *singular value decomposition takes the form* $\mathbf{Z} = \mathbf{U}\mathbf{D}\mathbf{V}^T$

$$\begin{pmatrix} \begin{bmatrix} & & \\ u_1 & \cdots & u_n \\ & & \end{bmatrix} \end{pmatrix} \begin{pmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{pmatrix} \begin{pmatrix} [& &] \\ & v_1 & \\ & \vdots & \\ [& &] \\ & v_n & \end{pmatrix}$$

- $\mathbf{U}$ is an $m \times n$ orthogonal matrix $(\mathbf{U}^T\mathbf{U} = \mathbf{I}_n)$
  whose columns $\mathbf{u}_j \in \mathbb{R}^m$ are called *left singular vectors*.
- $\mathbf{V}$ is an $n \times n$ orthogonal matrix $(\mathbf{V}^T\mathbf{V} = \mathbf{I}_n)$
  whose columns $\mathbf{v}_j \in \mathbb{R}^m$ are called *right singular vectors*.
- $\mathbf{D}$ is an $n \times n$ diagonal matrix with diagonal elements
  $d_1 \geq d_2 \geq \cdots \geq d_n \geq 0$ known as the *singular values*.

For a matrix $\mathbf{Z}_{m\times n}$ let $\Omega \subset \{1,\ldots,m\} \times \{1,\ldots,n\}$ denote the indices of observed entries.
A natural approach is to seek the lowest rank approximating matrix $\hat{\mathbf{Z}}$ and consider the following optimization problem:

$$\text{minimize rank}(\mathbf{M}) \quad \text{subject to} \sum_{(i,j)\in\Omega} (z_{ij} - m_{ij})^2 \leq \delta \tag{1}$$

This problem is nonconvex. So we consider a convenient convex relaxation of it :

$$\text{minimize} \parallel \mathbf{M} \parallel_\star \quad \textit{subject to} \sum_{(i,j)\in\Omega} (z_{ij} - m_{ij})^2 \leq \delta \tag{2}$$

where $\parallel \mathbf{M} \parallel_\star$ is the nuclear norm, or the sum of the singular values of $\mathbf{M}$.
We can reformulate (2) in Lagrange form:

$$\underset{\mathbf{M}}{\text{minimize}} \left\{ \frac{1}{2} \sum_{(i,j)\in\Omega} (z_{ij} - m_{ij})^2 + \lambda \parallel \mathbf{M} \parallel_\star \right\} \tag{3}$$

The parameter λ must be chosen from the data (typically by cross validation).

SOFT-IMPUTE Algorithm (SI)

We assume $\text{rank}(\mathbf{Z}) < \min(m, n)$.

Definition: Given an observed subset Ω of matrix entries, we define the *projection operator* $\mathcal{P}_\Omega(\mathbf{Z})_{i,j} = \begin{cases} z_{ij} & \text{if } (i,j) \in \Omega \\ 0 & \text{if } (i,j) \notin \Omega \end{cases}$

Remark: We can rewrite (3) as ($\|\cdot\|_F$ Frobenius norm)

$$\underset{\mathbf{M} \in \mathbb{R}^{m \times n}}{\text{minimize}} \left\{ \frac{1}{2} \| \mathcal{P}_\Omega(\mathbf{Z}) - \mathcal{P}_\Omega(\mathbf{M}) \|_F^2 + \lambda \| \mathbf{M} \|_\star \right\} \tag{4}$$

Definition: Given the singular value decomposition $\mathbf{Z} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ of a rank-r matrix $\mathbf{Z}$, we define its *soft-tresholded version* as

$$S_\lambda(\mathbf{Z}) \equiv \mathbf{U}\mathbf{D}_\lambda\mathbf{V}^T \quad \text{where} \quad \mathbf{D}_\lambda = \text{diag}[(d_1 - \lambda)_+, \ldots, (d_r - \lambda)_+]$$

Lemma: If $\mathbf{Z}$ is a a rank-r matrix, the solution to the optimization problem $\text{minimize}_\mathbf{M} \left\{ \frac{1}{2} \| \mathbf{Z} - \mathbf{M} \|_F^2 + \lambda \| \mathbf{M} \|_\star \right\}$ is given by $\hat{\mathbf{Z}} = S_\lambda(\mathbf{Z})$.

Algorithm: SOFT-IMPUTE *for matrix completion*

1. Initialize $\mathbf{Z}^{\text{old}} = \mathbf{0}$ and create a decreasing grid $\lambda_1 > ... > \lambda_K$.
2. For each $k = 1, ..., K$, set $\lambda = \lambda_k$ and iterate until convergence:
   Compute $\hat{\mathbf{Z}}_\lambda \leftarrow S_\lambda(\mathcal{P}_\Omega(\mathbf{Z}) + \mathcal{P}_\Omega^\perp(\mathbf{Z}^{\text{old}}))$.
   Update $\mathbf{Z}^{\text{old}} \leftarrow \hat{\mathbf{Z}}_\lambda$.
3. Output the sequence of solutions $\hat{\mathbf{Z}}_{\lambda_1}, ..., \hat{\mathbf{Z}}_{\lambda_K}$.

- ▷ Convergence to global minimum: The solution sequence generated converges asymptotically (as $k \to \infty$) to a minimizer of (4).
- ▷ Convergence speed: The algorithm converges at least sub-linearly, meaning that $\mathcal{O}(1/\delta)$ iterations are sufficient to compute a solution that is δ-close to the global optimum.
- ▷ Computational complexity: The computationally demanding part of the Algorithm is in $S_\lambda(\mathcal{P}_\Omega(\mathbf{Z}) + \mathcal{P}_\Omega^\perp(\mathbf{Z}^{\text{old}}))$. Note that:

$$\mathcal{P}_\Omega(\mathbf{Z}) + \mathcal{P}_\Omega^\perp(\mathbf{Z}^{\text{old}}) = \underbrace{\mathcal{P}_\Omega(\mathbf{Z}) - \mathcal{P}_\Omega(\mathbf{Z}^{\text{old}})}_{\text{sparse}} + \underbrace{\mathbf{Z}^{\text{old}}}_{\text{low rank}} \tag{5}$$

Application to the NETFLIX dataset

Netflix data matrix: 17,770 movies, 480,189 customers.
Extremely sparse, with 100,480,507 or 1% entries observed.

Netflix's algorithm in 2006 ("Cinematch") had an RMSE of 0.9525.
Contest goal: to improve this by 10% (or of 0.8572 or better).
Competition identified: *training set* + *qualifying set* (most recent ratings)

λ	Rank	Time (hrs)	RMSE
$\lambda_0/250$	42	1.36	0.9622
$\lambda_0/300$	66	2.21	0.9572
$\lambda_0/500$	81	2.83	0.9543
$\lambda_0/600$	95	3.27	0.9497

Results of applying SOFT-IMPUTE to the Netflix data. $\lambda_0 = \| \mathcal{P}_\Omega(Z) \|$

Computations were done on a Intel Xeon Linux 3GHz processor; timings are reported based on MATLAB implementations of PROPACK and SI algorithm. RMSE is root- mean squared error.[2]

[2] *Source:* Mazumder, R., Hastie, T. and Tibshirani, R. (2010), Spectral regularization algorithms for learning large incomplete matrices, Journal of Machine Learning Research 11, 2287–2322

Maximum Margin Matrix Factorization (MMMF)

Another class of techniques used in *collaborative filtering problems* are Maximum Margin Matrix Factorization (MMMF) methods.
They use a factor model for approximating the matrix $\mathbf{Z}$.

Let $\mathbf{M} = \mathbf{A}\mathbf{B}^T$ where $\mathbf{A}$ and $\mathbf{B}$ are $m \times r$ and $n \times r$ respectively.
Consider the optimization problem

$$\underset{\substack{\mathbf{A}\in\mathbb{R}^{m\times r}\\ \mathbf{B}\in\mathbb{R}^{n\times r}}}{\text{minimize}} \left\{ \| \mathcal{P}_\Omega(\mathbf{Z}) - \mathcal{P}_\Omega(\mathbf{A}\mathbf{B}^T) \|_F^2 + \lambda(\| \mathbf{A} \|_F^2 + \| \mathbf{B} \|_F^2) \right\}. \tag{6}$$

This problem turns out to be equivalent to the nuclear norm regularized problem (3)

$$\underset{\mathbf{M}\in\mathbb{R}^{m\times n}}{\text{minimize}} \frac{1}{2} \| \mathcal{P}_\Omega(\mathbf{Z}) - \mathcal{P}_\Omega(\mathbf{M}) \|_F^2 + \lambda \| \mathbf{M} \|_\star$$

since

$$\| \mathbf{M} \|_\star = \min_{\substack{\mathbf{A}\in\mathbb{R}^{m\times r},\ \mathbf{B}\in\mathbb{R}^{n\times r}\\ \mathbf{M}=\mathbf{A}\mathbf{B}^T}} \frac{1}{2}(\| \mathbf{A} \|_F^2 + \| \mathbf{B} \|_F^2) \tag{7}$$

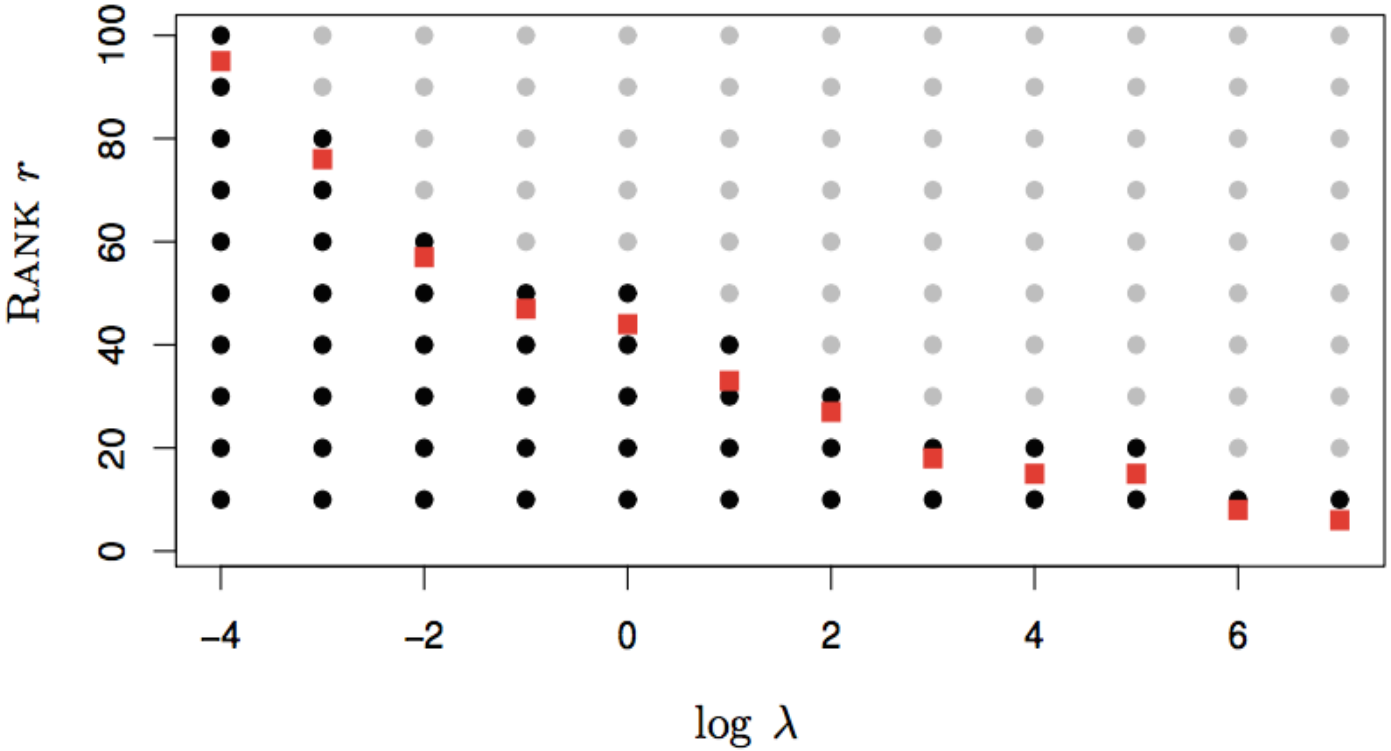
In particular, the one-dimensional SOFT- IMPUTE family lies in the two-dimensional (r, λ) MMMF family:

Theorem

Let $\mathbf{Z}$ be an $m \times n$ matrix with observed entries indexed by Ω.

(a) *Let $r = \min(m, n)$. Then the solutions to MMMF (6) and SI (3) coincide for all $\lambda \geq 0$.*

(b) *For some fixed $\lambda^* > 0$ suppose that (3) has an optimal solution with rank r^*. Then for any optimal soution $(\hat{\mathbf{A}}, \hat{\mathbf{B}})$, to the problem (6) with $r \geq r^*$ and $\lambda = \lambda^*$, the matrix $\hat{\mathbf{M}} = \hat{\mathbf{A}}\hat{\mathbf{B}}^T$ is an optimal solution for (3). This implies that the solution space of (3) is contained in that of (6).*



3

- While (3) is a convex optimization problem in $\mathbf{M}$, (6) is a non-convex problem in the variables $\mathbf{A}$, $\mathbf{B}$ and has possibly several local minima.
- The MMMF criterion (6) defines a two-dimensional family of models indexed by the pair (r, λ), while the SI criterion (3) defines a one-dimensional family.
  In light of the theorem, this family is a special path in the two-dimensionl grid of solutions $(\hat{\mathbf{A}}_{(r,\lambda)}, \hat{\mathbf{B}}_{(r,\lambda)})$.

[3] *Source:* Statistical Learning with Sparsity: The Lasso and Generalizations (p.183) [36]

Introduction

Part 2

In the first part of the presentation we have seen the matrix completion.

Now our task will be the sparse matrix decomposition, where we seek for sparse singular vectors. Matrix completion won't be be anymore our primary goal.

3. Rank-r SVD

The rank-r SVD is a decomposition based on the optimization problem

$$\underset{\text{rank}(\mathbf{M})=r}{\text{minimize}} \|\mathbf{Z} - \mathbf{M}\|_F,$$

where we have $\mathbf{Z} = \mathbf{UDV}^T$ and assume that $r \leq \text{rank}(\mathbf{Z})$.

It has a closed form solution $\hat{\mathbf{Z}}_r = \mathbf{UD}_r\mathbf{V}^T$, where $\mathbf{D}_r$ is a diagonal matrix $\mathbf{D}$ with all but the first r diagonal entries set to zero. The solution is sparse in the sense that has all but r singular values that are zero. This is the decomposition known as the rank-r SVD and is used, for example, in:

- machine learning;
- natural language processing, i.e. interactions between computers and human languages;
- recommender system, i.e. system that tries to predict the preference that an user would give to an item (where the matrix has missing data).

4. Penalized Matrix Decomposition

Maximum-margin matrix factorization methods lead to other forms of regularization. Consider the ℓ_1-penalized version

$$\underset{\mathbf{U}\in\mathbb{R}^{m\times r},\ \mathbf{V}\in\mathbb{R}^{m\times r},\ \mathbf{D}\in\mathbb{R}^{r\times r}}{\text{minimize}} \left\{ \|\mathbf{Z}-\mathbf{U}\mathbf{D}\mathbf{V}^T\|_F^2 + \lambda_1\|\mathbf{U}\|_1 + \lambda_2\|\mathbf{V}\|_1 \right\}$$

with $\mathbf{D}$ diagonal and non-negative and $\mathbf{U}\mathbf{D}\mathbf{V}^T$ the SVD.
We assume that all values of $\mathbf{Z}$ are observed.

$\Rightarrow$ obtain sparse left and right singular vectors

4. Penalized Matrix Decomposition
Optimization

We want to optimize the previous formula $\left(\underset{\mathbf{U},\mathbf{V},\mathbf{D}}{\text{minimize}} \left\{ \|\mathbf{Z} - \mathbf{U}\mathbf{D}\mathbf{V}^T\|_F^2 + \lambda_1 \|\mathbf{U}\|_1 + \lambda_2 \|\mathbf{V}\|_1 \right\}\right)$

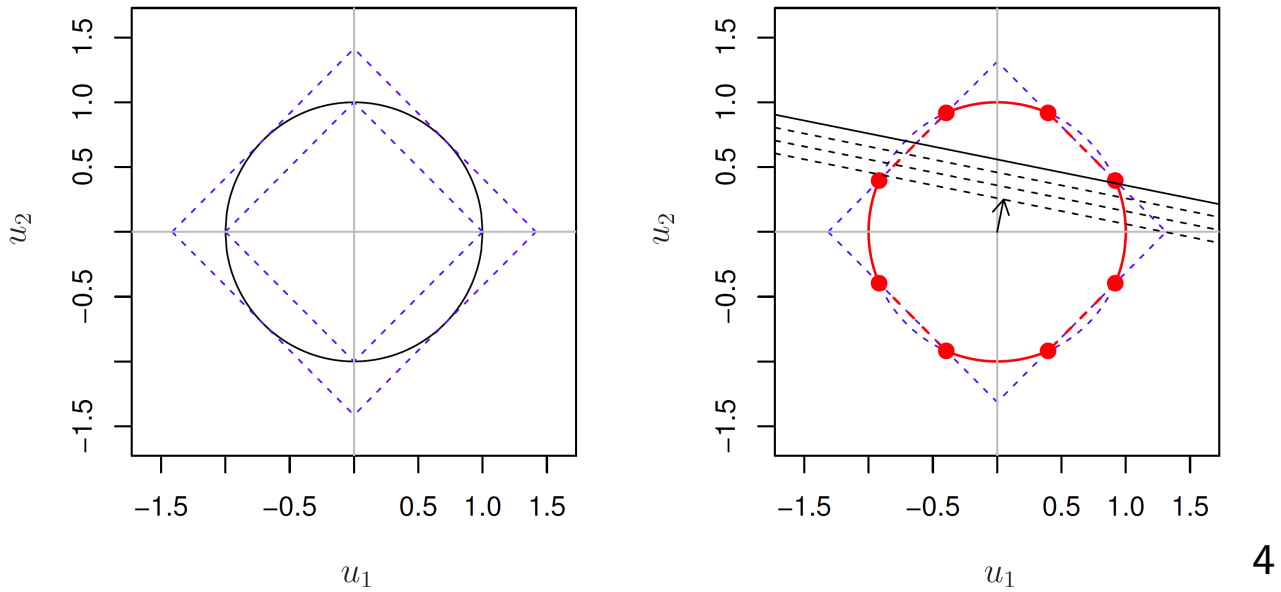
We write it in the constrained form and we consider the one-dimensional case, i.e. r=1.

$$\underset{\mathbf{u}\in\mathbb{R}^m,\ \mathbf{v}\in\mathbb{R}^n,\ d\geq 0}{\text{minimize}} \|\mathbf{Z} - d\mathbf{u}\mathbf{v}^T\|_F^2 \text{ subject to } \|\mathbf{u}\|_1 \leq c_1 \text{ and } \|\mathbf{v}\|_1 \leq c_2$$

We see that the estimator in the constrained form tends to produce solutions that are too sparse. For avoiding it we add a ℓ_2-norm constraints.

$$\underset{\mathbf{u}\in\mathbb{R}^m,\ \mathbf{v}\in\mathbb{R}^n,\ d\geq 0}{\text{minimize}} \|\mathbf{Z} - d\mathbf{u}\mathbf{v}^T\|_F^2 \text{ subject to } \|\mathbf{u}\|_1 \leq c_1,\ \|\mathbf{v}\|_1 \leq c_2,$$
$$\|\mathbf{u}\|_2 \leq 1,\ \|\mathbf{v}\|_2 \leq 1$$

4. Penalized Matrix Decomposition
Graphical representation

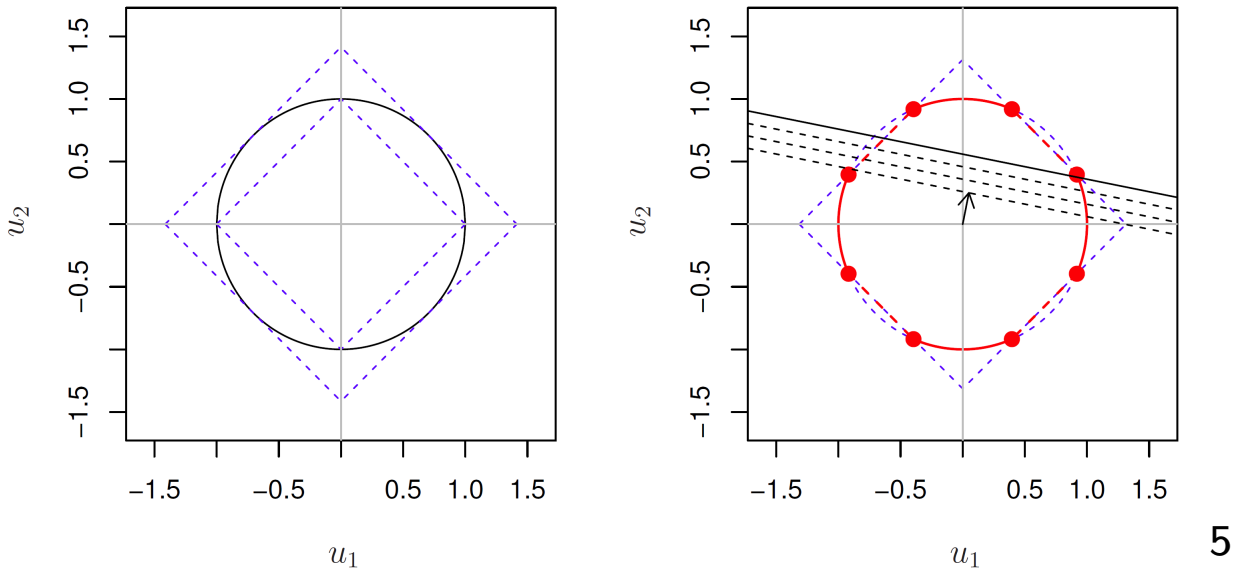


[4]

Figure: A graphical representation of the ℓ_1 and the ℓ_2 constraints on $\mathbf{u} \in \mathbb{R}^2$ in the penalized matrix decomposition criterion. The constraints are $\|\mathbf{u}\|_2^2 \leq 1$ and $\|\mathbf{u}\|_1 \leq c$. *Left panel:* The circle ℓ_2 constraint is the solide circle. For both the ℓ_1 and ℓ_2 constraints to be active, the constraint radius c must be between 1 and $\sqrt{2}$. The constraint on $\|\mathbf{u}\|_1 = 1$ and $\|\mathbf{u}\|_1 = \sqrt{2}$ are shown using the dashed lines.

[4] *Source:* Statistical Learning with Sparsity: The Lasso and Generalizations (p.188) [36]

4. Penalized Matrix Decomposition
Graphical representation



[5]

Figure: *Right panel:* The ℓ_2 and ℓ_1 constraints on $\mathbf{u}$ are shown for some c between 1 and $\sqrt{2}$. Red dots indicate the points where both the ℓ_1 and the ℓ_2 constraints are active. The red contour shows the boundary of the constraint region. The black lines are the linear contours of the criterion as a function of $\mathbf{u}$, which increase as we move to the upper right in this example. The solid red arcs indicate the solutions that occur when $\lambda_1 = 0$ in the next algorithm (ℓ_2 active, ℓ_1 not). The figure shows that in the two dimensions, the points where both the ℓ_1 and the ℓ_2 constraints are active have neither u_1 nor u_2 equal to zero. We also see that without the ℓ_2 constraints, we would always end up at a corner; this would lead to trivial solutions.

[5] *Source:* Statistical Learning with Sparsity: The Lasso and Generalizations (p.188) [36]

4. Penalized Matrix Decomposition
Soft-thresholding operation

Out of the previous graphical representation, the optimization problem is well defined as long as $1 \leq c_1 \leq \sqrt{m}$ and $1 \leq c_2 \leq \sqrt{n}$.

The previous formula is biconvex and it can be minimized in an alternating fashion $\rightarrow$ the solution is a soft-thresholding operation, where the soft-thresholding operator is applied element-wise on the vector.

We fix $\mathbf{v} \in \mathbb{R}^n$ such that it satisfy the constraints.
The update is $\mathbf{u} \leftarrow \frac{S_{\lambda_1}(\mathbf{Zv})}{\|S_{\lambda_1}(\mathbf{Zv})\|_2}$ / The update is $\mathbf{v} \leftarrow \frac{S_{\lambda_2}(\mathbf{Z}^T\mathbf{u})}{\|S_{\lambda_2}(\mathbf{Z}^T\mathbf{u})\|_2}$.

The threshold λ_1 (resp. λ_2) must be chosen so that it satisfies the constraints. This means that $\lambda_1 = 0$ if $\|\mathbf{u}\|_1 \leq c_1$ and $\lambda_1 > 0$ if $\|\mathbf{u}\|_1 = c_1$.

If the ℓ_1 constraints have no effect, then the algorithm reduces to the power method that finds the largest singular vector.

4. Penalized Matrix Decomposition
Algorithm (rank-one)

Algorithm: *Alternating soft-thresholding for rank-one penalized matrix decomposition*

1. Set $\mathbf{v}$ to the top left singular vector from the SVD of $\mathbf{Z}$.
2. Perform the update $\mathbf{v} \in \mathbb{R}^n$ is $\mathbf{u} \leftarrow \frac{S_{\lambda_1}(\mathbf{Zv})}{\|S_{\lambda_1}(\mathbf{Zv})\|_2}$, with λ_1 being the smallest value such that $\|\mathbf{u}\|_1 \leq c_1$.
3. Perform the update $\mathbf{u} \in \mathbb{R}^m$ is $\mathbf{v} \leftarrow \frac{S_{\lambda_2}(\mathbf{Z}^T\mathbf{u})}{\|S_{\lambda_2}(\mathbf{Z}^T\mathbf{u})\|_2}$, with λ_2 being the smallest value such that $\|\mathbf{v}\|_1 \leq c_2$.
4. Iterate steps 2 and 3 until convergence.
5. Return $\mathbf{u}$, $\mathbf{v}$ and $d = \mathbf{u}^T\mathbf{Zv}$.

The algorithm has the same solution of the previous optimization problem

$$\underset{\mathbf{u}\in\mathbb{R}^m,\ \mathbf{v}\in\mathbb{R}^n,\ d\geq 0}{\text{minimize}} \|\mathbf{Z} - d\mathbf{u}\mathbf{v}^T\|_F^2 \text{ subject to } \|\mathbf{u}\|_1 \leq c_1,\ \|\mathbf{v}\|_1 \leq c_2,\ \|\mathbf{u}\|_2 \leq 1,\ \|\mathbf{v}\|_2 \leq 1.$$

We can modify the algorithm so that we can handle missing matrix entries. For example, by omitting missing values when computing $\mathbf{Zv}$ and $\mathbf{Z}^T\mathbf{u}$.

4. Penalized Matrix Decomposition
Algorithm (multifactor)

Algorithm: *Multifactor penalized matrix decomposition*

1. Let $\mathbf{R} \leftarrow \mathbf{Z}$.
2. For $k = 1, \ldots K$:
   a. Find $\mathbf{u}_k$, $\mathbf{v}_k$, and d_k by applying the single-factor algorithm to data $\mathbf{R}$.
   b. Update $\mathbf{R} \leftarrow \mathbf{R} - d_k \mathbf{u}_k \mathbf{v}_k^T$.

The point is to apply the rank-one Algorithm successively to the matrix $\mathbf{Z}$.

If we assume that the ℓ_1 penalties don't exist, then the multifactor PMD algorithm leads to the rank-K SVD of $\mathbf{Z}$.
Recall: the rank-K SVD $\hat{\mathbf{Z}}_K = \mathbf{U}\mathbf{D}_r\mathbf{V}^T$.

In general the optimization

$$\underset{\mathbf{u}\in\mathbb{R}^m,\ \mathbf{v}\in\mathbb{R}^n,\ d\geq 0}{\text{minimize}} \|\mathbf{Z} - d\mathbf{u}\mathbf{v}^T\|_F^2 \text{ subject to } \|\mathbf{u}\|_1 \leq c_1,\ \|\mathbf{v}\|_1 \leq c_2,$$
$$\|\mathbf{u}\|_2 \leq 1,\ \|\mathbf{v}\|_2 \leq 1$$

can be used for also other type of penalties. For example for the fused lasso penalty

$$\Phi(\mathbf{u}) = \sum_{j=2}^{m} |u_j - u_{j-1}|,$$

where $\mathbf{u} = (u_1, u_2, \ldots u_m)$.

4. Penalized Matrix Decomposition
Example: application of the algorithm

The previous algorithms are used to find out the protein complexes in the human body, when we have a large-scale protein interaction data. The other algorithms developed for this purpose, didn't work for sparse protein-protein interaction (PPI).
We model the PPI network as a weight graph $G = (V, E, \omega)$, where V is the set of nodes (proteins), E the set of edges (protein pairs) and ω is the set of similarity value between each protein pairs.

The method that uses the PMD is divided into 3 steps:

1. the adjacent matrix $n \times n$ of the protein interaction network is transformed with the PCA into a $n \times p$ matrix, where each row represents a protein in each n samples (protein complexes) and each column represents the expression level of a sample in each p proteins;
2. then we decompose it into 3 factor matrices ($\mathbf{U}$,$\mathbf{V}$,$\mathbf{D}$);
3. we define appropriate constraints on the 3 factor matrices so that using the PMD we can detect the protein complexes when we have sparse PPI networks.

For more information see the article
Detection of Protein Complexes Based on Penalized Matrix Decomposition in a Sparse Protein–Protein Interaction Network, written by *Buwen Cao, Shuguang Deng, Hua Qin, Pingjian Ding, Shaopeng Chen and Guanghui Li (2018)*.

5. Additive Matrix Decomposition

We want to decompose a matrix into the sum of two or more matrices. The components in this addition must have complementary structure, for example we can decompose a matrix into the sum of a low rank matrix and a sparse matrix.

There are a lot of variety of applications for the additive matrix decomposition, for example:

- robust form of PCA (principal component analysis);
- factor analysis;
- robust matrix completion;
- multivariate regression.

5. Additive Matrix Decomposition
Noisy linear observation model

We can describe most of these applications in terms of the noisy linear observation model, that is $\mathbf{Z} = \mathbf{L}^* + \mathbf{S}^* + \mathbf{W}$, where $\mathbf{W}$ is a noise matrix. Here we have the pair $(\mathbf{L}^*, \mathbf{S}^*)$ that specify the additive matrix decomposition into a low rank matrix $\mathbf{L}^*$ and a sparse components matrix $\mathbf{S}^*$.

We want to find estimators of the pair $(\mathbf{L}^*, \mathbf{S}^*)$ based on

$$\underset{\mathbf{L},\mathbf{S}\in\mathbb{R}^{m\times n}}{\text{minimize}} \left\{ \frac{1}{2}\|\mathbf{Z} - (\mathbf{L} + \mathbf{S})\|_{\text{F}}^2 + \lambda_1 \Phi_1(\mathbf{L}) + \lambda_2 \Phi_2(\mathbf{S}) \right\},$$

where we have that Φ_1 and Φ_2 are penalty functions.

Here we have a look to the situation with a low rank and sparse matrices with $\Phi_1(\mathbf{L}) = \|\mathbf{L}\|_\star$ and $\Phi_2(\mathbf{S}) = \|\mathbf{S}\|_1$.

The standard principal component analysis is based on performing an SVD of a data matrix $\mathbf{Z} \in \mathbb{R}^{N \times p}$. The row i is the i^{th} sample of a p-dimensional data vector.

We have seen that the rank-r SVD can be obtained by minimizing $\|\mathbf{Z} - \mathbf{L}\|_F^2$ subject to a rank constraint on $\mathbf{L}$.

PROBLEM: What if some entries of $\mathbf{Z}$ are corrupted?
$\rightarrow$ PCA solution is sensible to corrupted data.

5. Additive Matrix Decomposition
Robust PCA

We can't simply approximate the matrix $\mathbf{Z}$ with a low rank matrix $\mathbf{L}$, but we also have to add a sparse matrix $\mathbf{S}$ that model the corrupted variables.

Given sparsity k and a target rank r $\rightarrow$ solve the optimization problem

$$\underset{\text{rank}(\mathbf{L}) \leq r,\ \text{card}(\mathbf{S}) \leq k}{\text{minimize}} \frac{1}{2} \|\mathbf{Z} - (\mathbf{L} + \mathbf{S})\|_F^2$$

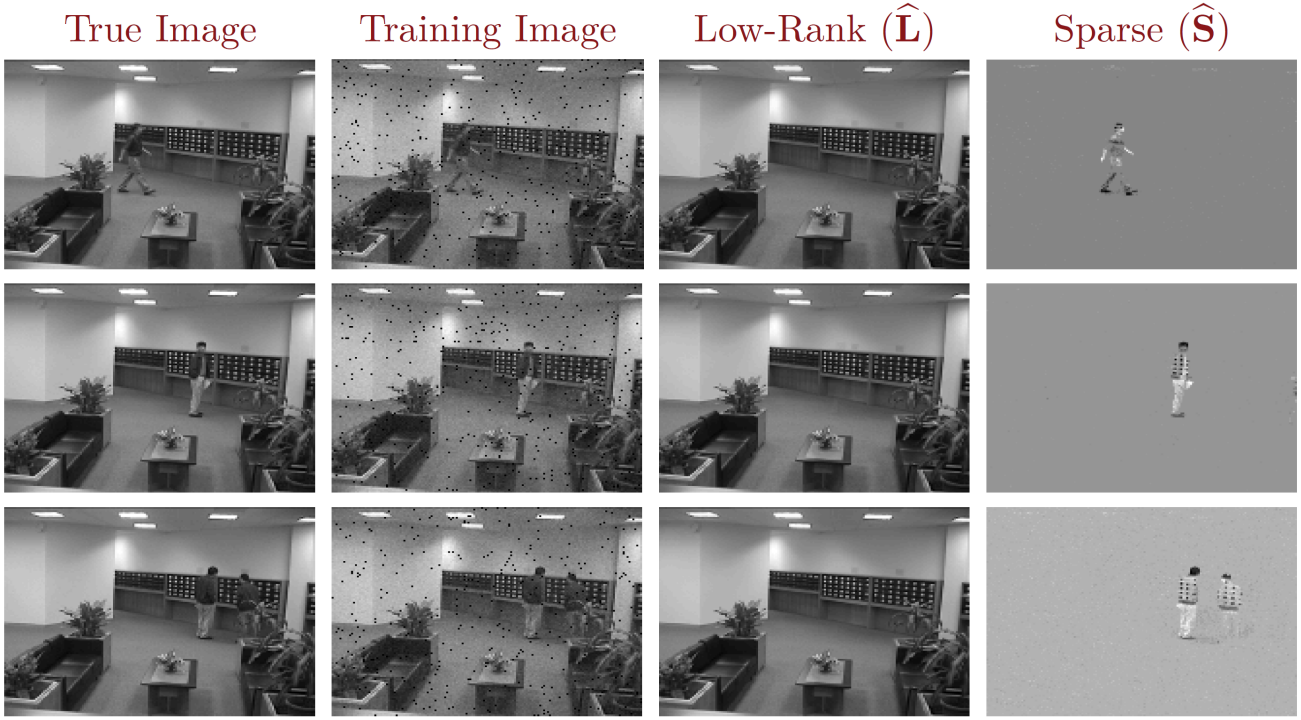
This problem with the rank and cardinality constraints is non-convex.

We have a convex relaxation with the previous general optimization

$$\underset{\mathbf{L}, \mathbf{S} \in \mathbb{R}^{m \times n}}{\text{minimize}} \left\{ \frac{1}{2} \|\mathbf{Z} - (\mathbf{L} + \mathbf{S})\|_F^2 + \lambda_1 \Phi_1(\mathbf{L}) + \lambda_2 \Phi_2(\mathbf{S}) \right\},$$

with the constraints $\phi_1(\mathbf{L}) = \|\mathbf{L}\|_\star$ and $\phi_2(\mathbf{S}) = \sum_{i,j} |s_{ij}|$ for element-wise sparsity.

5. Additive Matrix Decomposition
Robust PCA (Example)



[6]

Figure: Video surveillance. The columns of the data matrix $\mathbf{Z}$ are frames from a video surveillance camera (they are noisy and have missing pixel values).

$\Rightarrow$ possibility to separate a sparse matrix or a low rank matrix from the main estimator (here we could just analyse the behaviour of the people).

[6] *Source:* Statistical Learning with Sparsity: The Lasso and Generalizations (p.193) [36]

Robust matrix completion is a method for matrix-completion.
They are used in:

- collaborative filtering, i.e. a method that recommends specific things to the consumer depending on the preferences of many consumers (music, Netflix,...);
- recommender systems, i.e. a software that seeks to predict the rate that a user would give to something.

Sometimes also ratings may be corrupted.
For example, it happened that the Amazon system had been corrupted by some users, so that it would recommend a sex manual to those users who were interested in Christian literature.

As in the robust PCA, we add a sparse component $\mathbf{S}$ to our low rank matrix $\mathbf{L}$.
The nature of sparsity depends on how we model the adversarial behaviour:

- small fraction of entries corrupted $\longrightarrow$ element-wise sparsity via the ℓ_1-norm;
- rows (users) are corrupted $\longrightarrow$ row-wise sparsity penalty via the group lasso norm $\|\mathbf{S}\|_{1,2} = \sum_{i=1}^{m} \|\mathbf{S}_i\|_2$, where $\mathbf{S}_i \in \mathbb{R}^n$ is the i^{th} row of the matrix.

We now have a look to the optimization for a row-wise sparsity penalty. This is

$$\underset{\mathbf{L},\mathbf{S}\in\mathbb{R}^{m\times n}}{\text{minimize}} \left\{ \frac{1}{2} \sum_{(i,j)\in\Omega} (z_{ij} - (\mathbf{L}_{ij} + \mathbf{S}_{ij})^2 + \lambda_1 \|\mathbf{L}\|_\star + \lambda_2 \sum_{i=1}^{m} \|\mathbf{S}_i\|_2 \right\} \quad (8)$$

We go back to the previous simple formula (in the chapter of Matrix completion using nuclear norm)

$$\underset{\mathbf{M}}{\text{minimize}} \left\{ \frac{1}{2} \sum_{(i,j)\in\Omega} (z_{ij} - m_{ij})^2 + \lambda \|\mathbf{M}\|_\star \right\} \quad (9)$$

$\rightarrow$ (8) is simply a modification of (9).

Conclusions

Today we have tried to find a matrix $\hat{\mathbf{Z}}$ that approximates $\mathbf{Z}$.
We did it for 2 main reasons:

1) filling in missing data of $\mathbf{Z} \rightarrow$ *completion*;
2) approximating $\mathbf{Z}$ with a matrix that has a simpler structure $\rightarrow$ *decomposition*.

We have seen different methods of optimization, like: singular value decomposition, penalized matrix decomposition and some matrix completion methods.

These optimization problems are really useful in machine learning and in recommender systems.

Questions

References

Trevor Hastie, Robert Tibshirani, Martin Wainwright (2015)

Statistical Learning with Sparsity: The Lasso and Generalizations, CRC Press

Mazumder, R., Hastie, T. and Tibshirani, R. (2010)

Spectral regularization algorithms for learning large incomplete matrices

Journal of Machine Learning Research 228–2322

Buwen Cao, Shuguang Deng, Hua Qin, Pingjian Ding, Shaopeng Chen and Guanghui Li (2018)

Detection of Protein Complexes Based on Penalized Matrix Decomposition in a Sparse Protein–Protein Interaction Network