SI231 Matrix Analysis and Computations
Low-Rank Matrix Optimization

Ziping Zhao

Spring Term 2022–2023

School of Information Science and Technology
ShanghaiTech University, Shanghai, China

http://si231.sist.shanghaitech.edu.cn

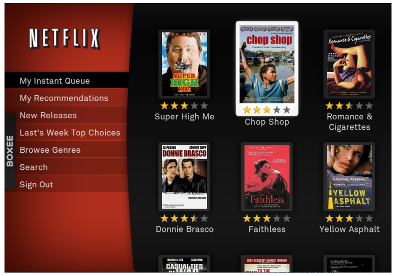
Low-Rank Matrix Optimization

- Motivation
- Problem Formulation
- Heuristics for Rank Minimization Problem
  - Nuclear Norm Heuristic
  - Log-det Heuristic
  - Matrix Factorization based Method
  - Rank Constraint via Convex Iteration
- Real Applications
  - Netflix Prize
  - Video Intrusion Detection
- Summary

Motivation – High Dimensional Data

- Data becomes increasingly massive, high dimensional...





- Images: compression, denoising, recognition. . .

- Videos: streaming, tracking, stabilization. . .

- User data: clustering, classification, recommendation. . .

- Web data: indexing, ranking, search. . .

Low Dimensional Structures in High Dimensional Data

- Low dimensional structures in visual data

- User Data: profiles of different users may share some common factors

- **How to extract low dimensional structures from such high dimensional data?**

Problem Formulation – Rank Minimization Problem

- In many scenarios, low dimensional structure is closely related to low rank.

- But in real applications, the true rank is usually unknown. A natural approach to solve this is to formulate it as a rank minimization problem, i.e., finding the matrix of lowest rank that satisfies some constraint

$$\begin{array}{ll} \underset{\mathbf{X}\in\mathbb{R}^{m\times n}}{\text{minimize}} & \text{rank}(\mathbf{X}) \\ \text{subject to} & \mathbf{X}\in\mathcal{C}, \end{array}$$

  where $\mathbf{X}$ is the optimization variable and $\mathcal{C}$ is a (possibly) convex set denoting the constraints.

- When $\mathbf{X}$ is restricted to be diagonal, $\text{rank}(\mathbf{X}) = \|\text{diag}(\mathbf{X})\|_0$ and the rank minimization problem reduces to the cardinality minimization problem (ℓ_0-norm minimization).

Matrix Rank

- The rank of a matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ is
  - the number of linearly independent rows of $\mathbf{X}$
  - the number of linearly independent columns of $\mathbf{X}$
  - the number of nonzero singular values of $\mathbf{X}$, i.e., $\|\boldsymbol{\sigma}(\mathbf{X})\|_0$.
  - the smallest number r such that there exists an $m \times r$ matrix $\mathbf{F}$ and an $r \times n$ matrix $\mathbf{G}$ with $\mathbf{X} = \mathbf{FG}$

Solving Rank Minimization Problem

- In general, the rank minimization problem is NP-hard, and there is little hope of finding the global minimum efficiently in all instances.

- What we are going to talk about, instead, are efficient *heuristics,* categorized into two groups:

  1. Approximate the rank function with some surrogate functions
     - Nuclear norm heuristic
     - Log-det heuristic
  2. Solving a sequence of rank-constrained feasibility problems
     - Matrix factorization based method
     - Rank constraint via convex iteration

Semidefinite Embedding Lemma

- It can be shown that any nonsquare matrix $\mathbf{X}$ can be associated with a positive semidefinite matrix whose rank is exactly twice the rank of $\mathbf{X}$.

Lemma 1. *Let $\mathbf{X} \in \mathbb{R}^{m \times n}$ be a given matrix. Then rank$(\mathbf{X}) \leq r$ if and only if there exist matrices $\mathbf{Y} \in \mathbb{S}^m$ and $\mathbf{Z} \in \mathbb{S}^n$ such that*

$$\begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} \succeq \mathbf{0}, \quad \mathrm{rank}(\mathbf{Y}) + \mathrm{rank}(\mathbf{Z}) \leq 2r.$$

- Based on the semidefinite embedding lemma, minimizing the rank of a general nonsquare matrix $\mathbf{X}$ is equivalent to minimizing the rank of the positive semidefinite, block diagonal matrix blkdiag$(\mathbf{Y}, \mathbf{Z})$:

$$\begin{array}{ll} \underset{\mathbf{X},\mathbf{Y},\mathbf{Z}}{\text{minimize}} & \frac{1}{2}\mathrm{rank}(\mathsf{blkdiag}(\mathbf{Y}, \mathbf{Z})) \\ \text{subject to} & \begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} \succeq \mathbf{0} \\ & \mathbf{X} \in \mathcal{C}. \end{array}$$

Proof of Semidefinite Embedding Lemma

- the "$\Longrightarrow$" part: Suppose that $\operatorname{rank}(\mathbf{X}) = r_0 \leq r$. Then $\operatorname{rank}(\mathbf{X})$ can be factored as $\operatorname{rank}(\mathbf{X}) = \mathbf{F}\mathbf{G}$ where $\mathbf{F} \in \mathbb{R}^{m \times r_0}$ and $\mathbf{G} \in \mathbb{R}^{r_0 \times m}$, and $\operatorname{rank}(\mathbf{F}) = \operatorname{rank}(\mathbf{G}) = r_0$. Set $\mathbf{Y}$ and $\mathbf{Z}$ to be the rank r_0 matrices $\mathbf{F}\mathbf{F}^T$ and $\mathbf{G}^T\mathbf{G}$, respectively. Then we have

$$\begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ \mathbf{G}^T \end{bmatrix} \begin{bmatrix} \mathbf{F}^T & \mathbf{G} \end{bmatrix} \succeq \mathbf{0}$$

- the "$\Longleftarrow$" part: we begin with a result, which is a generalization of the well known Schur complement condition for positive semidefinite matrices.

Pseudo-Schur Complement

- let

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix},$$

  where $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}$, $\mathbf{B} \in \mathbb{R}^{m_1 \times n_2}$, $\mathbf{C} \in \mathbb{R}^{m_2 \times n_1}$, and $\mathbf{D} \in^{m_2 \times n_2}$.

  Note $\mathbf{D}$ can be rectangular or square and singular. We define the pseudo-Schur complement of $\mathbf{A}$ in $\mathbf{M}$ by

$$\mathbf{S}_{D^\dagger} = \mathbf{A} - \mathbf{B}\mathbf{D}^\dagger\mathbf{C},$$

  where $\mathbf{D}^\dagger$ is the pseudo-inverse (or Moore-Penrose inverse) of $\mathbf{D}$.

- Similarly, we can also define $\mathbf{S}_{A^\dagger} = \mathbf{D} - \mathbf{C}\mathbf{A}^\dagger\mathbf{B}$, $\mathbf{S}_{B^\dagger} = \mathbf{C} - \mathbf{D}\mathbf{B}^\dagger\mathbf{A}$, and $\mathbf{S}_{C^\dagger} = \mathbf{B} - \mathbf{A}\mathbf{C}^\dagger\mathbf{D}$.

Pseudo-Schur Complement Condition for PSD Matrices

- the Schur complement: let

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{D} \end{bmatrix},$$

where $\mathbf{A} \in \mathbb{S}^m$, $\mathbf{B} \in \mathbb{R}^{m \times n}$, and $\mathbf{D} \in \mathbb{S}^n$.

- We have

$$\begin{aligned} \mathbf{M} \succeq \mathbf{0} \quad & \Longleftrightarrow \quad \mathbf{A} \succeq \mathbf{0},\ (\mathbf{I} - \mathbf{A}\mathbf{A}^\dagger)\mathbf{B} = \mathbf{0},\ \mathbf{S}_{A^\dagger} \succeq \mathbf{0} \\ & \Longleftrightarrow \quad \mathbf{D} \succeq \mathbf{0},\ (\mathbf{I} - \mathbf{D}\mathbf{D}^\dagger)\mathbf{B}^T = \mathbf{0},\ \mathbf{S}_{D^\dagger} \succeq \mathbf{0} \end{aligned}$$

- proof:

$$\begin{aligned} \mathbf{M} & = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{B}^T\mathbf{A}^\dagger & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{D} - \mathbf{B}^T\mathbf{A}^\dagger\mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^\dagger\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \\ & = \begin{bmatrix} \mathbf{I} & \mathbf{B}\mathbf{D}^\dagger \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} - \mathbf{B}\mathbf{D}^\dagger\mathbf{B}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{D}^\dagger\mathbf{B}^T & \mathbf{I} \end{bmatrix} \end{aligned}$$

Proof of Semidefinite Embedding Lemma

- the "$\Longleftarrow$" part: based on the pseudo-Schur complement condition for positive semidefinite matrices (the result in the first line), our goal is to show that they imply $\text{rank}(\mathbf{Y}) \geq \text{rank}(\mathbf{X})$ and $\text{rank}(\mathbf{Z}) \geq \text{rank}(\mathbf{X})$.

  Assume, without loss of generality, that $\text{rank}(\mathbf{Y}) \leq \text{rank}(\mathbf{Z})$ (if this were not the case, we could use the result in the second line). From second condition, we have

$$\mathbf{X}^T(\mathbf{I} - \mathbf{Y}\mathbf{Y}^\dagger) = \mathbf{0}.$$

  Since $(\mathbf{I} - \mathbf{Y}\mathbf{Y}^\dagger)$ is a projection operator for $\mathcal{N}(\mathbf{Y})$, it follows that

$$\mathcal{N}(\mathbf{X}^T) \supseteq \mathcal{N}(\mathbf{Y}) \Rightarrow \dim \mathcal{N}(\mathbf{X}^T) \geq \dim \mathcal{N}(\mathbf{Y}).$$

  Recall that $\text{rank}(\mathbf{X}) = n - \dim \mathcal{N}(\mathbf{X}) = m - \dim \mathcal{N}(\mathbf{X}^T)$. Then, we can conclude that $\text{rank}(\mathbf{Y}) \geq \text{rank}(\mathbf{X}^T) = \text{rank}(\mathbf{X})$. Hence,

$$\text{rank}(\mathbf{X}) \leq \text{rank}(\mathbf{Y}) \leq \frac{1}{2}(\text{rank}(\mathbf{Y}) + \text{rank}(\mathbf{Z}))$$

Nuclear Norm Heuristic

A well known heuristic for rank minimization problem is replacing the rank function in the objective with the nuclear norm

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & \|\mathbf{X}\|_* \\ \text{subject to} & \mathbf{X} \in \mathcal{C} \end{array}$$

- Proposed by Fazel (2002) [Fazel, 2002].
- The nuclear norm $\|\mathbf{X}\|_*$ is defined as the sum of singular values, i.e., $\|\mathbf{X}\|_* = \sum_{i=1}^{r} \sigma_i$.
- If $\mathbf{X} \succeq \mathbf{0}$, $\|\mathbf{X}\|_*$ is just $\mathrm{tr}(\mathbf{X})$ and the "nuclear norm heuristic" reduces to the "trace heuristic".

Why Nuclear Norm?

- Nuclear norm can be viewed as the ℓ_1-norm of the vector of singular values.

- Just as ℓ_1-norm $\Rightarrow$ sparsity, nuclear norm $\Rightarrow$ sparse singular value vector, i.e., low rank.

- When $\mathbf{X}$ is restricted to be diagonal, $\|\mathbf{X}\|_* = \|\mathrm{diag}(\mathbf{X})\|_1$ and the nuclear norm heuristic for rank minimization problem reduces to the ℓ_1-norm heuristic for cardinality minimization problem.

- $\|\mathbf{x}\|_1$ is the convex envelope of $\mathrm{card}(\mathbf{x})$ over $\{\mathbf{x} | \, \|\mathbf{x}\|_\infty \leq 1\}$. Similarly, $\|\mathbf{X}\|_*$ is the convex envelope of $\mathrm{rank}(\mathbf{X})$ on the convex set $\{\mathbf{X} | \, \|\mathbf{X}\|_2 \leq 1\}$.

Equivalent SDP Formulation

Lemma 2. *For* $\mathbf{X} \in \mathbb{R}^{m\times n}$ *and* $t \in \mathbb{R}$, *we have* $\|\mathbf{X}\|_* \leq t$ *if and only if there exist matrices* $\mathbf{Y} \in \mathbb{R}^{m\times m}$ *and* $\mathbf{Z} \in \mathbb{R}^{n\times n}$ *such that*

$$\begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} \succeq \mathbf{0}, \quad \mathrm{tr}(\mathbf{Y}) + \mathrm{tr}(\mathbf{Z}) \leq 2t.$$

- Based on the above lemma, the nuclear norm minimization problem is equivalent to

$$\begin{array}{ll} \underset{\mathbf{X},\mathbf{Y},\mathbf{Z}}{\text{minimize}} & \frac{1}{2}\mathrm{tr}(\mathbf{Y}+\mathbf{Z}) \\ \text{subject to} & \begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} \succeq \mathbf{0} \\ & \mathbf{X} \in \mathcal{C}. \end{array}$$

- This SDP formulation can also be obtained by applying the "trace heuristic" to the PSD form of the rank minimization problem.

Log-det Heuristic

- In the log-det heuristic, log-det function is used as a smooth surrogate for rank function.

- Symmetric positive semidefinite case:

$$\begin{array}{ll} \underset{\mathbf{X} \succeq \mathbf{0}}{\text{minimize}} & \log\det(\mathbf{X} + \delta \mathbf{I}) \\ \text{subject to} & \mathbf{X} \in \mathcal{C}, \end{array}$$

  where $\delta > 0$ is a small regularization constant.

- Note that $\log\det(\mathbf{X} + \delta\mathbf{I}) = \sum_i \log(\sigma_i(\mathbf{X} + \delta\mathbf{I}))$, $\mathrm{rank}(\mathbf{X}) = \|\boldsymbol{\sigma}(\mathbf{X})\|_0$, and $\log(s + \delta)$ can be seen as a surrogate function of $\mathrm{card}(s)$.

- However, the surrogate function $\log\det(\mathbf{X} + \delta\mathbf{I})$ is not convex (in fact, it is concave).

Log-det Heuristic

- An iterative linearization and minimization scheme (called majorization-minimization) is used to find a local minimum.

- Let $\mathbf{X}^{(k)}$ denote the kth iterate of the optimization variable $\mathbf{X}$. The first-order Taylor series expansion of $\log\det\left(\mathbf{X}+\delta\mathbf{I}\right)$ about $\mathbf{X}^{(k)}$ is given by

$$\log\det\left(\mathbf{X}+\delta\mathbf{I}\right) \leq \log\det\left(\mathbf{X}^{(k)}+\delta\mathbf{I}\right) + \mathrm{tr}\left(\left(\mathbf{X}^{(k)}+\delta\mathbf{I}\right)^{-1}\left(\mathbf{X}-\mathbf{X}^{(k)}\right)\right).$$

  Then, one could minimize $\log\det\left(\mathbf{X}+\delta\mathbf{I}\right)$ by iteratively minimizing the local linearization, which leads to

$$\mathbf{X}^{(k+1)} = \arg\min_{\mathbf{X}\in\mathcal{C}} \mathrm{tr}\left(\left(\mathbf{X}^{(k)}+\delta\mathbf{I}\right)^{-1}\mathbf{X}\right).$$

Interpretation of Log-det Heuristic

- If we choose $\mathbf{X}^{(0)} = \mathbf{I}$, the first iteration is equivalent to minimizing the trace of $\mathbf{X}$, which is just the trace heuristic. The iterations that follow try to reduce the rank further. In this sense, we can view this heuristic as a refinement of the trace heuristic.

- At each iteration we solve a weighted trace minimization problem, with weights $\mathbf{W}^{(k)} = \left(\mathbf{X}^{(k)} + \delta\mathbf{I}\right)^{-1}$. Thus, the log-det heuristic can be considered as an extension of the iterative reweighted ℓ_1-norm heuristic for cardinality minimization problem to the matrix case.

Log-det Heuristic for General Matrix

- For general nonsquare matrix $\mathbf{X}$, we can apply the log-det heuristic to the equivalent PSD form and obtain

$$\begin{array}{ll} \underset{\mathbf{X},\mathbf{Y},\mathbf{Z}}{\text{minimize}} & \log\det(\mathsf{blkdiag}(\mathbf{Y},\mathbf{Z}) + \delta\mathbf{I}) \\ \text{subject to} & \begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} \succeq \mathbf{0} \\ & \mathbf{X} \in \mathcal{C}. \end{array}$$

- Linearizing as before, at iteration k we solve the following problem to get $\mathbf{X}^{(k+1)}$, $\mathbf{Y}^{(k+1)}$ and $\mathbf{Z}^{(k+1)}$

$$\begin{array}{ll} \underset{\mathbf{X},\mathbf{Y},\mathbf{Z}}{\text{minimize}} & \operatorname{tr}\left(\left(\mathsf{blkdiag}(\mathbf{Y}^{(k)},\mathbf{Z}^{(k)}) + \delta\mathbf{I}\right)^{-1}\mathsf{blkdiag}(\mathbf{Y},\mathbf{Z})\right) \\ \text{subject to} & \begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} \succeq \mathbf{0} \\ & \mathbf{X} \in \mathcal{C}. \end{array}$$

Matrix Factorization based Method

- The idea behind factorization based methods is that $\mathrm{rank}(\mathbf{X}) \le r$ if and only if $\mathbf{X}$ can be factorized as $\mathbf{X} = \mathbf{F}\mathbf{G}$, where $\mathbf{F} \in \mathbb{R}^{m \times r}$ and $\mathbf{G} \in \mathbb{R}^{r \times n}$.

- For each given r, we check if there exists a feasible $\mathbf{X}$ of rank less than or equal to r by checking if any $\mathbf{X} \in \mathcal{C}$ can be factored as above.

- The expression $\mathbf{X} = \mathbf{F}\mathbf{G}$ is not convex in $\mathbf{X}$, $\mathbf{F}$ and $\mathbf{G}$ simultaneously, but it is convex in $(\mathbf{X}, \mathbf{F})$ when $\mathbf{G}$ is fixed and convex in $(\mathbf{X}, \mathbf{G})$ when $\mathbf{F}$ is fixed.

- Various heuristics can be applied to handle this non-convex equality constraint, but it is not guaranteed to find an $\mathbf{X}$ with rank r even if one exists.

Matrix Factorization based Method

- Coordinate descent method: Fix $\mathbf{F}$ and $\mathbf{G}$ one at a time and iteratively solve a convex problem at each iteration.
  - Choose $\mathbf{F}^{(0)} \in \mathbb{R}^{m \times r}$. Set $k = 1$.
  - **repeat**

$$
\begin{aligned}
(\tilde{\mathbf{X}}^{(k)}, \mathbf{G}^{(k)}) &= \underset{\mathbf{X} \in \mathcal{C}, \mathbf{G} \in \mathbb{R}^{r \times n}}{\operatorname{argmin}} \left\| \mathbf{X} - \mathbf{F}^{(k-1)} \mathbf{G} \right\|_F \\
(\mathbf{X}^{(k)}, \mathbf{F}^{(k)}) &= \underset{\mathbf{X} \in \mathcal{C}, \mathbf{F} \in \mathbb{R}^{m \times r}}{\operatorname{argmin}} \left\| \mathbf{X} - \mathbf{F} \mathbf{G}^{(k)} \right\|_F \\
e^{(k)} &= \left\| \mathbf{X}^{(k)} - \mathbf{F}^{(k)} \mathbf{G}^{(k)} \right\|_F,
\end{aligned}
$$

  - **until** $e^{(k)} \leq \epsilon$, or $e^{(k-1)}$ and $e^{(k)}$ are approximately equal.

Rank Constraint via Convex Iteration

- Consider a semidefinite rank-constrained feasibility problem

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{find}} & \mathbf{X} \\ \text{subject to} & \mathbf{X} \in \mathcal{C} \\ & \mathbf{X} \succeq \mathbf{0} \\ & \operatorname{rank}(\mathbf{X}) \leq r, \end{array}$$

- It is proposed in [Dattorro, 2005] to solve this problem via iteratively solving the following two convex problems:

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & \operatorname{tr}(\mathbf{W}^{\star}\mathbf{X}) \\ \text{subject to} & \mathbf{X} \in \mathcal{C} \\ & \mathbf{X} \succeq \mathbf{0} \end{array} \qquad \begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \operatorname{tr}(\mathbf{W}\mathbf{X}^{\star}) \\ \text{subject to} & \mathbf{0} \preceq \mathbf{W} \preceq \mathbf{I} \\ & \operatorname{tr}(\mathbf{W}) = n - r, \end{array}$$

where $\mathbf{W}^{\star}$ is the optimal solution of the second problem and $\mathbf{X}^{\star}$ is the optimal solution of the first problem.

Rank Constraint via Convex Iteration

- An optimal solution to the second problem is known in closed form. Given non-increasingly ordered diagonalization $\mathbf{X}^\star = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$, then matrix $\mathbf{W}^\star = \mathbf{U}^\star\mathbf{U}^{\star T}$ is optimal where $\mathbf{U}^\star = \mathbf{V}(:, r+1 : n) \in \mathbb{R}^{n\times n-r}$, and

$$\mathrm{tr}(\mathbf{W}^\star\mathbf{X}^\star) = \sum_{i=r+1}^{n} \lambda_i(\mathbf{X}^\star).$$

- We start from $\mathbf{W}^\star = \mathbf{I}$ and iteratively solving the two convex problems. Note that in the first iteration the first problem is just the "trace heuristic".

- Suppose at convergence, $\mathrm{tr}(\mathbf{W}^\star\mathbf{X}^\star) = \tau$, if $\tau = 0$, then $\mathrm{rank}(\mathbf{X}^\star) \leq r$ and $\mathbf{X}^\star$ is a feasible point. But this is not guaranteed, only local convergence can be established, i.e., converging to some $\tau \geq 0$.

Rank Constraint via Convex Iteration

- For general nonsquare matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$, we have an equivalent PSD form

$$\begin{array}{ll} \underset{\mathbf{X}\in\mathbb{R}^{m\times n}}{\text{find}} & \mathbf{X} \\ \text{subject to} & \mathbf{X} \in \mathcal{C} \\ & \text{rank}(\mathbf{X}) \leq r \end{array} \quad \Leftrightarrow \quad \begin{array}{ll} \underset{\mathbf{X},\mathbf{Y},\mathbf{Z}}{\text{find}} & \mathbf{X} \\ \text{subject to} & \mathbf{X} \in \mathcal{C} \\ & \mathbf{G} = \begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} \succeq \mathbf{0} \\ & \text{rank}(\mathbf{G}) \leq r. \end{array}$$

- The same convex iterations can be applied now. Note that if we start from $\mathbf{W}^\star = \mathbf{I}$, now the first problem is just the "nuclear norm heuristic" for the first iteration.

Recommender Systems

amazon.com

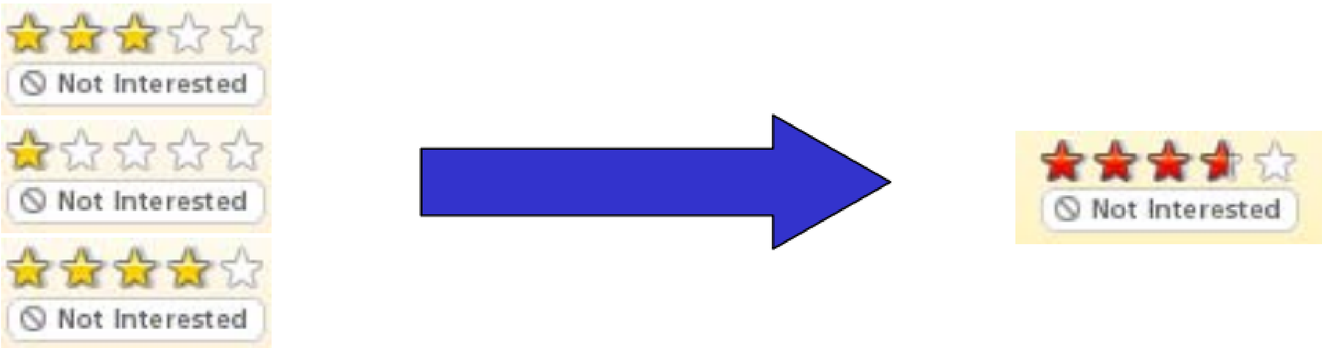
NETFLIX

match.com chemistry eHarmony

- How does Amazon recommend commodities?
- How does Netflix recommend movies?

Netflix Prize

- Given 100 million ratings on a scale of 1 to 5, predict 3 million ratings to highest accuracy



- 17,770 total movies, 480,189 total users
- How to fill in the blanks?
- Can you improve the recommendation accuracy by 10% over what Netflix was using? $\Longrightarrow$ **One million dollars!**

Abstract Setup: Matrix Completion

- Consider a rating matrix $\mathbf{R} \in \mathbb{R}^{m \times n}$ with r_{ij} representing the rating user i gives to movie j.

- But some r_{ij} are unknown since no one watches all movies

$$\mathbf{R} = \begin{array}{cc} & \text{Movies} \\ & \begin{bmatrix} 2 & 3 & ? & ? & 5 & ? \\ 1 & ? & ? & 4 & ? & 3 \\ ? & ? & 3 & 2 & ? & 5 \\ 4 & ? & 3 & ? & 2 & 4 \end{bmatrix} \text{Users} \end{array}$$

- We would like to predict how users will like unwatched movies.

Structure of the Rating Matrix

- The rating matrix is very big, $480,189$ (number of users) times $17,770$ (number of movies) in the Netflix case.

- But there are much fewer types of people and movies than there are people and movies.

- So it is reasonable to assume that for each user i, there is a k-dimensional vector $\mathbf{p}_i$ explaining the user's movie taste and for each movie j, there is also a k-dimensional vector $\mathbf{q}_j$ explaining the movie's appeal. And the inner product between these two vectors, $\mathbf{p}_i^T\mathbf{q}_j$, is the rating user i gives to movie j, i.e., $r_{ij} = \mathbf{p}_i^T\mathbf{q}_j$. Or equivalently in matrix form, $\mathbf{R}$ is factorized as $\mathbf{R} = \mathbf{P}^T\mathbf{Q}$, where $\mathbf{P} \in \mathbb{R}^{k\times m}$, $\mathbf{Q} \in \mathbb{R}^{k\times n}$, $k \ll \min(m,n)$.

- It is the same as assuming the matrix $\mathbf{R}$ is of low rank.

Matrix Completion

- The true rank is unknown, a natural approach is to find the minimum rank solution

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & \text{rank}(\mathbf{X}) \\ \text{subject to} & x_{ij} = r_{ij}, \quad \forall (i,j) \in \Omega, \end{array}$$

  where Ω is the set of observed entries.

- In practice, instead of requiring strict equality for the observed entries, one may allow some error and the formulation becomes

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & \text{rank}(\mathbf{X}) \\ \text{subject to} & \sum_{(i,j)\in\Omega} (x_{ij} - r_{ij})^2 \leq \epsilon. \end{array}$$

- Then, all the heuristics can be applied, e.g., log-det heuristic, matrix factorization.

What did the Winners Use?

- What algorithm did the final winner of the Netflix Prize use?

- You can find the report from the Netflix Prize website. The winning solution is really a cocktail of many methods combined and thousands of model parameters fine-tuned specially to the training set provided by Netflix.

- But one key idea they used is just the factorization of the rating matrix as the product of two low rank matrices [Koren et al., 2009], [Koren and Bell, 2011].

Video Intrusion Detection – Background Extraction from Video

- Given video sequence $\mathbf{F}_i$, $i = 1, \ldots, n$.

- The objective is to extract the background in the video sequence, i.e., separating the background from the human activities.

Low-Rank Matrix + Sparse Matrix

- Stacking the images and grouping the video sequence $\mathbf{Y} = [\mathrm{vec}(\mathbf{F}_1), \ldots, \mathrm{vec}(\mathbf{F}_n)]$
- The background component is of low rank, since the background is static within a short period (ideally it is rank one as the image would be the same).
- The foreground component is sparse, as activities only occupy a small fraction of space.
- The problem fits into the following signal model

$$\mathbf{Y} = \mathbf{X} + \mathbf{E},$$

where $\mathbf{Y}$ is the observation, $\mathbf{X}$ is a low rank matrix (the low rank background) and $\mathbf{E}$ is a sparse matrix (the sparse foreground).

Low-Rank and Sparse Matrix Recovery

- Low-rank and sparse matrix recovery

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & \text{rank}(\mathbf{X}) + \gamma \left\|\text{vec}(\mathbf{E})\right\|_0 \\ \text{subject to} & \mathbf{Y} = \mathbf{X} + \mathbf{E}. \end{array}$$

- Applying the nuclear norm heuristic and ℓ_1-norm heuristic simultaneously

$$\begin{array}{ll} \underset{\mathbf{X}}{\text{minimize}} & \left\|\mathbf{X}\right\|_* + \gamma \left\|\text{vec}(\mathbf{E})\right\|_1 \\ \text{subject to} & \mathbf{Y} = \mathbf{X} + \mathbf{E}. \end{array}$$

- Recently, some theoretical results indicate that when $\mathbf{X}$ is of low rank and $\mathbf{E}$ is sparse enough, exact recovery happens with high probability [Wright et al., 2009].

Background Extraction Result

- row 1: the original video sequences.
- row 2: the extracted low-rank background.
- row 3: the extracted sparse foreground.

Summary

- We have introduced the rank minimization problem.
- We have developed different heuristics to solve the rank minimization problem:
  - Nuclear norm heuristic
  - Log-det heuristic
  - Matrix factorization based method
  - Rank constraint via convex iteration
- Real applications are provided.

References

[Candès and Recht, 2009] Candès, E. J. and Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772.

[Dattorro, 2005] Dattorro, J. (2005). *Convex Optimization & Euclidean Distance Geometry*. Meboo Publishing USA.

[Fazel, 2002] Fazel, M. (2002). *Matrix rank minimization with applications*. PhD thesis, Stanford University.

[Koren and Bell, 2011] Koren, Y. and Bell, R. (2011). Advances in collaborative filtering. In *Recommender Systems Handbook*, pages 145–186. Springer.

[Koren et al., 2009] Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

[Wright et al., 2009] Wright, J., Ganesh, A., Rao, S., Peng, Y., and Ma, Y. (2009). Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in neural information processing systems*, pages 2080–2088.