Matrix Completion

Jiayu Zhou

[1] Department of Computer Science and Engineering
Michigan State University
East Lansing, MI USA

April 19, 2016

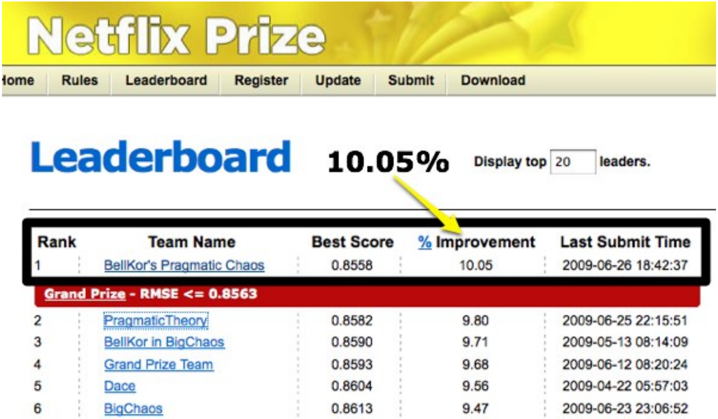
Some slides are from "Matrix Completion and Large-scale SVD Computations." by Trevor Hastie.

Table of contents

1. Introduction
2. Convex Approaches
    - Problem Formulation
3. Non-Convex Approaches
4. Applications
    - Image Inpainting
    - Recommender Systems
    - Robust PCA

Introduction

Motivation: The Netflix Prize


Netflix Prize

Home | Rules | Leaderboard | Register | Update | Submit | Download

Leaderboard **10.05%** Display top 20 leaders.

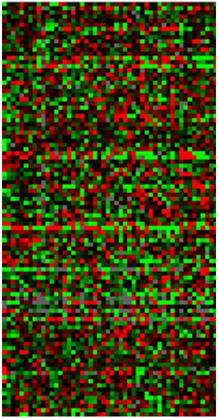
Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	BellKor's Pragmatic Chaos	0.8558	10.05	2009-06-26 18:42:37
Grand Prize - RMSE <= 0.8563				
2	PragmaticTheory	0.8582	9.80	2009-06-25 22:15:51
3	BellKor in BigChaos	0.8590	9.71	2009-05-13 08:14:09
4	Grand Prize Team	0.8593	9.68	2009-06-12 08:20:24
5	Dace	0.8604	9.56	2009-04-22 05:57:03
6	BigChaos	0.8613	9.47	2009-06-23 23:06:52

The Netflix Data Set

	Movie I	Movie II	Movie III	Movie IV	
User A	1	?	5	4	...
User B	?	2	3	?	...
User C	4	1	2	?	...
User D	?	5	1	3	...
User E	1	2	?	?	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

- **Training Data**: 480K users, 18K movies, 100M ratings (1-5), (99% ratings missing)
- **Goal**: \$ 1M prize for 10% reduction in RMSE over Cinematch
- **BellKor's Pragmatic Chaos** declared winners on 9/21/2009 used ensemble of models, an important ingradient being **low-rank factorization**

Expression Arrays



- The rows are genes (variables)
- The columns are observations (samples, DNA arrays).
- Typical numbers are 6-10K genes, 50-150 samples.
- Often 10-15% N/As

Matrix Completion / Collaborative Filtering

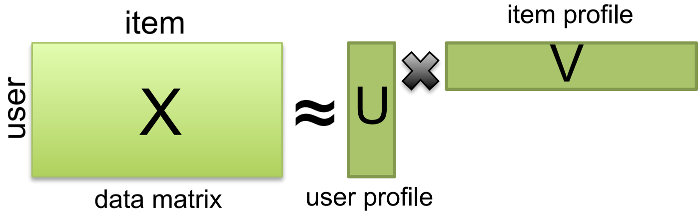
Problem Definition

- **Large** matrices
  rows/columsn $\approx 10^5, 10^6$ and even higher.
- Very **sparse**:
  often only $1-2\%$ observed
- Exploit matrix **structure**
  row/column interactions
- Task: **"fill-in"** missing entries
- Application: recommender systems, image-processing, imputation of NAs for genomic data, rank estimation for SVD.

Convex Approaches

Model Assumption: Low Rank + Noise

- The low-rank assumption $X = UV$



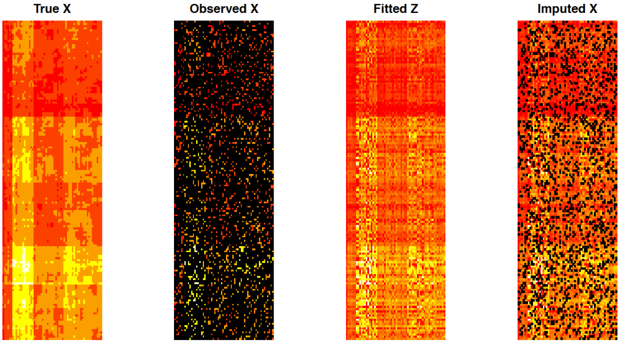
- Meaningful?
  - **Interpretation** - User and Item factors induce collaboration
  - **Empirical** - Netflix success.
  - **Theoretical** - "reconstruction" possible under low-rank and regularity conditions.

Problem Formulation

Find $Z_{n\times m}$ of (small) rank r such that training error is mall.

$$\min_{Z} \sum_{(i,j)\in\Omega} (X_{ij} - Z_{ij})^2 \quad \text{s.t.} \operatorname{rank}(Z) = r$$

where Ω is the set of indices of observed elements. Impute missing X_{ij} with Z_{ij}.



Nuclear Norm Relaxation

- The rank(Z) constraint makes the problem non-convex and combinatorially very hard (although good heurist algorithms exist)

Nuclear Norm Relaxation

- The rank(Z) constraint makes the problem non-convex and combinatorially very hard (although good heurist algorithms exist)
- $\|Z\|_* = \sum_j \sigma_j(Z)$ – sum of singular values of Z – is convex in Z. Called the **nuclear norm/trace norm** of Z.

Nuclear Norm Relaxation

- The rank(Z) constraint makes the problem non-convex and combinatorially very hard (although good heurist algorithms exist)
- $\|Z\|_* = \sum_j \sigma_j(Z)$ – sum of singular values of Z – is convex in Z. Called the **nuclear norm/trace norm** of Z.
- $\|Z\|_*$ is **the tightest convex relaxation** of rank(Z) (Fazel, Boyd, 2002)

Nuclear Norm Relaxation

- The $\text{rank}(Z)$ constraint makes the problem non-convex and combinatorially very hard (although good heurist algorithms exist)
- $\|Z\|_* = \sum_j \sigma_j(Z)$ – sum of singular values of Z – is convex in Z. Called the **nuclear norm/trace norm** of Z.
- $\|Z\|_*$ is **the tightest convex relaxation** of $\text{rank}(Z)$ (Fazel, Boyd, 2002)

We solve instead

$$\min_Z \sum_{(i,j)\in\Omega} (X_{ij} - Z_{ij})^2, \quad \text{s.t. } \|Z\|_* \le \tau$$

which is convex in Z.

Notation

Following *Cai et al* (2010) define $P_\Omega(X)_{n\times m}$: projection onto the observed entries

$$P_\Omega(X)_{i,j} = \begin{cases} X_{i,j} & \text{if } (i,j) \text{ is observed} \\ 0 & \text{if } (i,j) \text{ is missing} \end{cases}$$

Notation

Following *Cai et al* (2010) define $P_\Omega(X)_{n\times m}$: projection onto the observed entries

$$P_\Omega(X)_{i,j} = \begin{cases} X_{i,j} & \text{if } (i,j) \text{ is observed} \\ 0 & \text{if } (i,j) \text{ is missing} \end{cases}$$

Criterion rewritten as:

$$\sum_{(i,j)\in\Omega} (X_{ij} - Z_{ij})^2 = \|P_\Omega(X) - P_\Omega(Z)\|_F^2$$

Exact and Noisy Matrix Completion

- SVT algorithm of Cai et. al. (2010) solves

$$\min_{Z} \|Z\|_* \quad \text{s.t. } P_\Omega(Z) = P_\Omega(X)$$

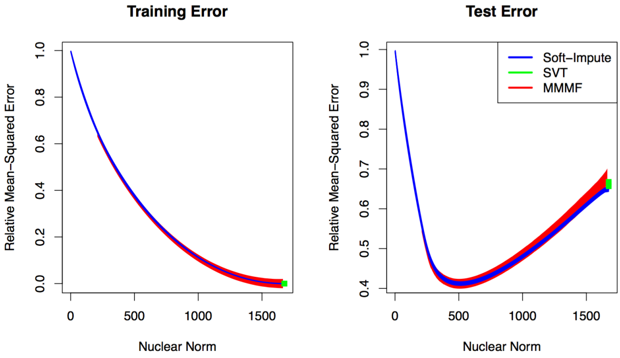
  - First order algorithm - scalable to large matrices via **sparse SVD**
  - No-noise reconstruction model seems too rigid

- Rephrasing our criterion

$$\min_{Z} \|Z\|_* \quad \text{s.t. } \|P_\Omega(X) - P_\Omega(Z)\|_F \leq \delta$$

  - In real-life, there is noise – fitting training data exactly incurs added variance
  - Introduce bias to decrease variance
  - Computation requires more than a sparse SVD

Bias-Variance Trade-Off



50% missing entries with SNR=1, true rank 6, 50 simulations

Soft SVD

Let (fully observed) $X_{n\times m}$ have SVD

$$X = U\text{diag}[\sigma_1, \ldots, \sigma_m]V^T$$

Consider the convex optimization problem

$$\min_Z \frac{1}{2}\|X - Z\|_F^2 + \lambda\|Z\|_*$$

Solution is **soft-thresholded SVD**

$$\mathbf{S}_\lambda(X) = U\text{diag}[(\sigma_1 - \lambda)_+, \ldots, (\sigma_m - \lambda)_+]V^T$$

Like lasso for SVD: singular values are shrunk to zero, with many set to zero. Smooth version of best-rank approximation.

Singular Value Thresholding

- An approximation algorithm
  - $Z_+ = \mathbf{S}_\lambda(Y)$
  - $Y_+ = Y + \delta_k P_\Omega(X - Z_+)$
- On the theoretical side, the authors provide a convergence analysis showing that the sequence of iterates converges
- On the practical side, the authors provide numerical examples in which 1000×1000 matrices are recovered in less than a minute on a modest desktop computer.

Convex Optimization Problem

Back to the missing data problem, in Lagrange form:

$$\min_{Z} \frac{1}{2}\|P_\Omega(X) - P_\Omega(Z)\|_F^2 + \lambda\|Z\|_*$$

- This is a semi-definite program (SDP), convex in Z.
- Existing off-the-shelf solvers:
  - Interior-point methods
  - (Black box) first-order methods
- We solve using an iterative soft SVD (next slide), with cost per soft SVD $O[(m+n)\cdot r + |\Omega|]$ where r is rank of solution.

Gradient Descent for the Composite Model
(Nesterov, 2007; Beck and Teboulle, 2009)

- Optimization objective

$$\min_{Z} f(Z) = \mathcal{L}(Z) + \lambda \|Z\|_*$$

Gradient Descent for the Composite Model
(Nesterov, 2007; Beck and Teboulle, 2009)

- Optimization objective

$$\min_{Z} f(Z) = \mathcal{L}(Z) + \lambda \|Z\|_*$$

- At each iteration we construct a model

$$\mathcal{M}(Z_i, \gamma_i) = [\mathcal{L}(Z_i) + \langle \nabla \mathcal{L}(Z_i), (Z - Z_i) \rangle] + \frac{1}{2\gamma_i} \|Z - Z_i\|_F^2 + \lambda \|Z\|_*$$

- Optimization algorithm
  - Repeat
  - $x_{i+1} = \arg\min \mathcal{M}(x_i, \gamma_i)$
  - Until convergence

First Order Optimization

Proximal Gradient

$$\begin{aligned} Z_{i+1} &= \mathcal{L}(Z_i) + \langle \nabla \mathcal{L}(Z_i), (Z - Z_i) \rangle + \frac{1}{2\gamma_i} \|Z - Z_i\|_F^2 + \lambda \|Z\|_* \\ &= \arg\min_x \left\{ \frac{1}{2} \|Z - (Z_i - \gamma_i \nabla \mathcal{L}(Z_i))\|_F^2 + \gamma_i \lambda \|Z\|_* \right\} \\ &\equiv \mathsf{Prox}_{\gamma_i}^{\lambda} (Z_i - \gamma_i \nabla \mathcal{L}(Z_i)) \end{aligned}$$

First Order Optimization

Proximal Gradient

$$
\begin{aligned}
Z_{i+1} &= \mathcal{L}(Z_i) + \langle \nabla \mathcal{L}(Z_i), (Z - Z_i) \rangle + \frac{1}{2\gamma_i} \|Z - Z_i\|_F^2 + \lambda \|Z\|_* \\
&= \arg\min_x \left\{ \frac{1}{2} \|Z - (Z_i - \gamma_i \nabla \mathcal{L}(Z_i))\|_F^2 + \gamma_i \lambda \|Z\|_* \right\} \\
&\equiv \mathsf{Prox}_{\gamma_i}^{\lambda}(Z_i - \gamma_i \nabla \mathcal{L}(Z_i))
\end{aligned}
$$

Proximal Operator

$$
\min_Z \frac{1}{2} \|Z - \hat{Z}\|_F^2 + \lambda \|Z\|_*
$$

admits the closed form solution $Z^* = \mathbf{S}_\lambda(\hat{Z})$.

SOFT-IMPUTE: Path Algorithm

1. Initialize $Z^{\text{old}} = 0$ and create a decreasing grid Λ of values $\lambda_0 > \lambda_1 > \cdots > \lambda_K > 0$, with $\lambda_0 = \sigma_{\max}(P_\Omega(X))$
2. For each $\lambda = \lambda_1, \lambda_2, \cdots \in \Lambda$ iterate the following till convergence:
   (2a) Compute $Z^{\text{new}} \leftarrow \mathbf{S}_\lambda(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$
   (2b) Assign $Z^{\text{old}} \leftarrow Z^{\text{new}}$
   (2c) Assign $Z_\lambda^* \leftarrow Z^{\text{new}}$ and go to 2.
3. Output the sequence of solutions $Z_{\lambda_1}^*, \ldots, Z_{\lambda_K}^*$

Soft-Impute: Convergence Analysis

Theorem (Mazumder et. al., 2010)
Take $\lambda > 0$. The sequence of estimates $\{Z_k\}_k$ given by:

$$Z_{k+1} = \arg\min_{Z} \frac{1}{2} \|P_\Omega(X) + P_\Omega^\perp(X) - Z\|_F^2 + \lambda \|Z\|_*$$

converges to Z_∞, a fixed point of

$$Z = \mathbf{S}_\lambda(P_\Omega(X) + P_\Omega^\perp(X))$$

Hence, Z_∞ minimizes

$$f_\lambda(Z) = \frac{1}{2} \|P_\Omega(X) - P_\Omega(Z)\|_F^2 + \lambda \|Z\|_*$$

SOFT-IMPUTE: Algorithm Properties

- Objective values decrease at every iteration:

$$f_\lambda(Z_{k+1}) \leq f_\lambda(Z_k)$$

- Successive iterates move closer to the set of optimal solutions:

$$\|Z_{k+1} - Z^*\| \leq \|Z_k - Z^*\|$$

  for any $Z^* \in \arg\min_Z f_\lambda(Z)$

- **Theorem** (Mazumder et. al.; 2010)
  Worst rate of convergence is $O(\frac{1}{k})$

$$f_\lambda(Z_k) - f_\lambda(Z_\infty) \leq \frac{2}{k+1}\|Z_0 - Z_\infty\|_F^2$$

(Rate can be itghtened to linear with warm-start and large λ)

SOFT-IMPUTE: Algorithm Properties

Obtain the sequence $\{Z_k\}$, where Z_k is current guess ...

$$Z_{k+1} = \arg\min_{Z} \frac{1}{2}\|P_\Omega(X) + P_\Omega^\perp(X) - Z\|_F^2 + \lambda\|Z\|_*$$

Computational bottleneck – soft SVD requires (low-rank) SVD of **completed** matrix after k iterations:

$$X_k = P_\Omega(X) + P_\Omega^\perp(X)$$

SOFT-IMPUTE: Algorithm Properties

Obtain the sequence $\{Z_k\}$, where Z_k is current guess . . .

$$Z_{k+1} = \arg\min_{Z} \frac{1}{2}\|P_\Omega(X) + P_\Omega^\perp(X) - Z\|_F^2 + \lambda\|Z\|_*$$

Computational bottleneck – soft SVD requires (low-rank) SVD of **completed** matrix after k iterations:

$$X_k = P_\Omega(X) + P_\Omega^\perp(X)$$

Trick:

$$P_\Omega(X) + P_\Omega^\perp(X) = \underbrace{\{P_\Omega(X) - P_\Omega(Z_k)\}}_{\text{Sparse}} + \underbrace{Z_k}_{\text{Low Rank}}$$

Hard-Impute

- Consider the rank constraint problem

$$\min_{Z} \|P_\Omega(X) - P_\Omega(Z)\|_F^2, \quad \text{s.t. } \operatorname{rank}(Z) = r.$$

HARD-IMPUTE

- Consider the rank constraint problem

$$\min_{Z} \|P_\Omega(X) - P_\Omega(Z)\|_F^2, \quad \text{s.t. } \mathrm{rank}(Z) = r.$$

- This is not convex in Z, but by analogy with Soft-Impute, an iterative algorithm gives good solutions.
- Replace step:
  (2a) Compute $Z^{\text{new}} \leftarrow \mathbf{S}_\lambda(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$
  with
  (2a') Compute $Z^{\text{new}} \leftarrow \mathbf{H}_r(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$
- Here $\mathbf{H}_r(X)$ is the best rank-r approximation to X

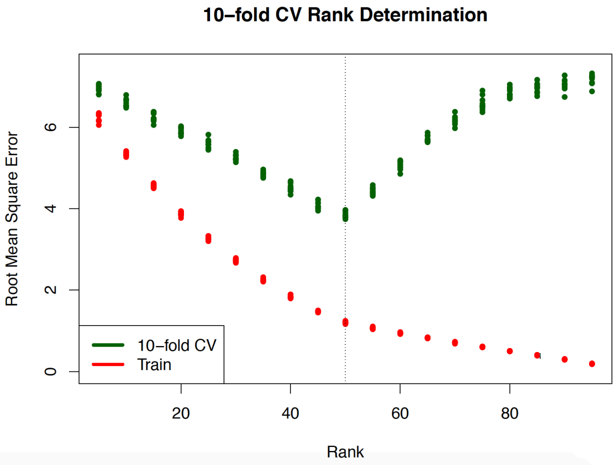
Hard-Impute

- Consider the rank constraint problem

$$\min_Z \|P_\Omega(X) - P_\Omega(Z)\|_F^2, \quad \text{s.t. } \mathrm{rank}(Z) = r.$$

- This is not convex in Z, but by analogy with Soft-Impute, an iterative algorithm gives good solutions.
- Replace step:
  (2a) Compute $Z^{\text{new}} \leftarrow \mathbf{S}_\lambda(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$
  with
  (2a') Compute $Z^{\text{new}} \leftarrow \mathbf{H}_r(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$
- Here $\mathbf{H}_r(X)$ is the best rank-r approximation to X
  - i.e., the rank-r truncated SVD approximation.

Example: choosing a good rank for SVD



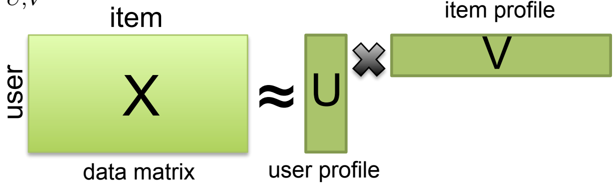
Truth is 200×100 rank-50 matrix plus noise (SNR 3). Randomly omit 10% of entries, and then predict using solutions from Hard-Impute.

Non-Convex Approaches

Matrix Factorization via Direct Search

- Consider rank-r approximation $Z = U_{m\times r}V_{n\times r}^T$, and solve

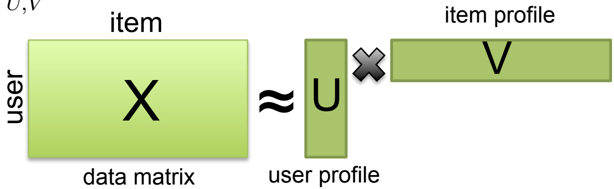
$$\min_{U,V} \|P_\Omega(X) - P_\Omega(UV^T)\|_F^2 + \lambda(\|U\|_F^2 + \|V\|_F^2)$$



Matrix Factorization via Direct Search

- Consider rank-r approximation $Z = U_{m\times r}V_{n\times r}^T$, and solve

$$\min_{U,V} \|P_\Omega(X) - P_\Omega(UV^T)\|_F^2 + \lambda(\|U\|_F^2 + \|V\|_F^2)$$



Lemma (Mazumder et al 2010)
For any matrix W, the following holds:

$$\|W\|_* = \min_{U,V:W=UV^T} \frac{1}{2}\left(\|U\|_F^2 + \|V\|_F^2\right).$$

If $\text{rank}(W) = k \leq \min\{m,n\}$, then the minimum above is attained at a factor decomposition $W = U_{m\times k}V_{n\times k}^T$

Low-Rank Matrix Fitting (LMaFit)

Wen, Yin, and Zhang. 2012

- Consider the following problem:

$$\min_{U,V} \|P_\Omega(X) - P_\Omega(UV^T)\|_F^2$$

Low-Rank Matrix Fitting (LMaFit)

Wen, Yin, and Zhang. 2012

- Consider the following problem:

$$\min_{U,V} \|P_\Omega(X) - P_\Omega(UV^T)\|_F^2$$

- Rewrite the problem into the following:

$$\min_{U,V,Z} \frac{1}{2}\|UV^T - Z\|_F^2 \quad \text{s.t. } P_\Omega(X) = P_\Omega(Z)$$

Low-Rank Matrix Fitting (LMaFit)

Wen, Yin, and Zhang. 2012

- Consider the following problem:

$$\min_{U,V} \|P_\Omega(X) - P_\Omega(UV^T)\|_F^2$$

- Rewrite the problem into the following:

$$\min_{U,V,Z} \frac{1}{2}\|UV^T - Z\|_F^2 \quad \text{s.t. } P_\Omega(X) = P_\Omega(Z)$$

- Alternating solution:
  - $U_+ = \arg\min_U \frac{1}{2}\|UV^T - Z\|_F^2 = ZV(V^TV)^\dagger$
  - $V_+^T = \arg\min_V \frac{1}{2}\|U_+V^T - Z\|_F^2 = (U_+^TU_+)^\dagger(U_+^TZ)$
  - $Z_+ = U_+V_+^T + P_\Omega(X - U_+V_+^T)$

Applications

Application: Image Inpainting

True Image

Application: Image Inpainting

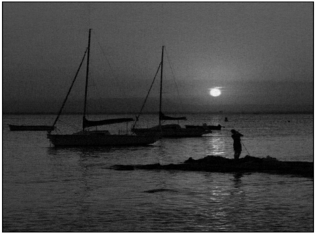
50% Masked/Degraded Noisy Training Image

Application: Image Inpainting

Training

Oracle

SOFT-IMPUTE



SOFT-IMPUTE+

Lena


Full Image
Input Image
SoftImp [Error 0.0634]
LMaFit [Error 0.2103]

Modeling Ratings in Recommender Systems

- Predict rating r_{ui} for user u and item i.
- Given average rating μ, we could assume that the rating comes from both user effects b_u and item effects b_i:

$$b_{ui} = \mu + b_u + b_i,$$

Modeling Ratings in Recommender Systems

- Predict rating r_{ui} for user u and item i.
- Given average rating μ, we could assume that the rating comes from both user effects b_u and item effects b_i:

$$b_{ui} = \mu + b_u + b_i,$$

- Given a dataset of observed ratings Ω, we can solve the least squares

$$\min_{\{b_u\},\{b_i\}} \sum_{(u,i)\in\Omega} (r_{ui} - \mu - b_u - b_i)^2 + \lambda(\sum_u b_u^2 + \sum_i b_i^2)$$

Modeling Ratings in Recommender Systems

- Predict rating r_{ui} for user u and item i.
- Given average rating μ, we could assume that the rating comes from both user effects b_u and item effects b_i:

$$b_{ui} = \mu + b_u + b_i,$$

- Given a dataset of observed ratings Ω, we can solve the least squares

$$\min_{\{b_u\},\{b_i\}} \sum_{(u,i)\in\Omega} (r_{ui} - \mu - b_u - b_i)^2 + \lambda(\sum_u b_u^2 + \sum_i b_i^2)$$

  - Can we recommend items based on this model?

Modeling Ratings in Recommender Systems

- Predict rating r_{ui} for user u and item i.
- Given average rating μ, we could assume that the rating comes from both user effects b_u and item effects b_i:

$$b_{ui} = \mu + b_u + b_i,$$

- Given a dataset of observed ratings Ω, we can solve the least squares

$$\min_{\{b_u\},\{b_i\}} \sum_{(u,i)\in\Omega} (r_{ui} - \mu - b_u - b_i)^2 + \lambda(\sum_u b_u^2 + \sum_i b_i^2)$$

  - Can we recommend items based on this model?
  - Can we achieve personalized recommendation using this model?

SVD++ for Recommender Systems

Koren 2008

- Add interaction model:

$$\hat{r}_{ui} = b_{ui} + p_u^T q_i$$

  where p_u is called user profile and q_i is called item profile.

- SVD++ learns profiles:

$$\min_{\{b_u\},\{b_i\}} \sum_{(u,i)\in\Omega} (r_{ui} - b_{ui} - p_u^T q_i)^2 + \lambda(\|p_u\| + \|q_i\| + b_u^2 + b_i^2)$$

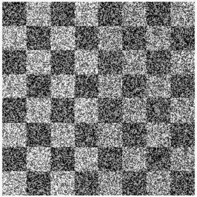
- When $b_{ui} = 0$, we are learning a standard matrix factorization:

$$\min_{P,Q} \|P_\Omega(R) - P_\Omega(PQ^T)\|_F^2 + \lambda(\|P\|_F^2 + \|Q\|_F^2)$$

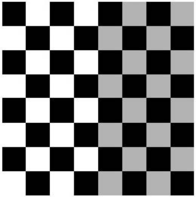
- SVD++ is not a SVD.

The problem of PCA

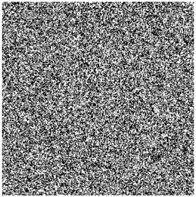
Mixed Image



Low-rank Image



Sparse Image



- Noise within structured data:

$$X = L + S$$

where L is the low rank data matrix and S is a sparse matrix.

Robust PCA (RPCA)

- The general form of the RPCA problem can be formulated as follows:

$$\min_{L,S} \mathsf{rank}(L) + \lambda \|Y\|_0 \quad \text{s.t. } X = L + S$$

Robust PCA (RPCA)

- The general form of the RPCA problem can be formulated as follows:

$$\min_{L,S} \text{rank}(L) + \lambda \|Y\|_0 \quad \text{s.t. } X = L + S$$

- Convex relaxation with theoretical guarantees [Candes, Li, Ma and Write 2009]:

$$\min_{L,S} \|L\|_* + \lambda \|Y\|_1 \quad \text{s.t. } X = L + S$$

  Penalized version can be solved by projected gradient descent.

Robust PCA Performance