

15.097: Statistical Learning via a Modern Optimization Lens

Lectures 20-22: Matrix Completion

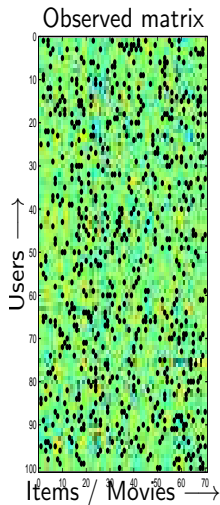
The Netflix DataSet

	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	?	?	...
⋮	⋮	⋮	⋮	⋮	⋮

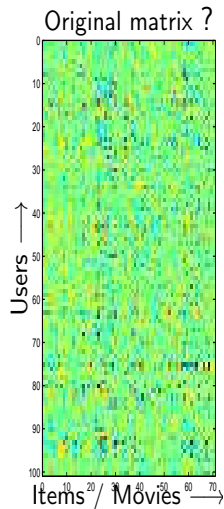
- ▶ **Training Data:**
480K users, 18K movies,
100 M ratings
ratings 1-5
(99 % ratings missing)
- ▶ **Goal:**
\$ 1 M prize for 10 % reduction
in RMSE over Cinematch
- ▶ **BellKor's Pragmatic Chaos**
declared winners on 9/21/2009

used ensemble of models, an
important ingredient being
low-rank factorization

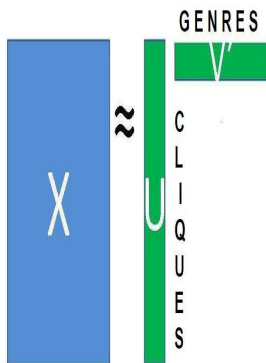
Definition: Matrix Completion/ Collaborative Filtering



- ▶ Large Matrices
 $\# \text{ rows}/\# \text{ columns} \approx 10^5, 10^6$
- ▶ Very Under-determined
(often 1 – 2 % observed)
- ▶ Exploit Matrix structure,
Row/Column Interactions
- ▶ Task “Fill-In” missing entries
- ▶ Applications: Recommender
systems; Image Processing;
Micro-array imputation.



Model Assumption : Low Rank + Noise



- ▶ Under-determined – assume **Low-Rank**

- ▶ **Meaningful?**

Interpretation – User & Item factors induce collaboration

Empirical – Netflix

Theoretical – “Reconstruction” possible under Low-rank & regularity conditions

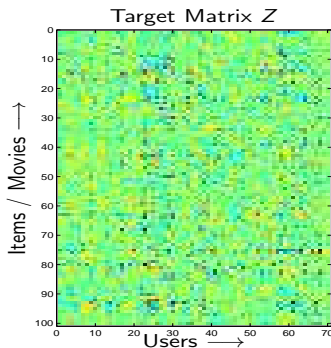
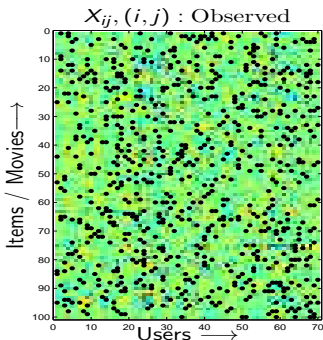
Srebro et al (2005); Candes and Recht (2008); Candes and Tao (2009); Keshavan et. al. (2009); Negahban and Wainwright '12

Objective criterion

Find $Z_{n \times m}$ of rank r **small**, such that training error is small.

$$\begin{array}{ll} \text{minimize} & \text{rank}(Z) \\ \text{subject to} & \sum_{(i,j): \text{Observed}} (Z_{ij} - X_{ij})^2 \leq \delta \end{array}$$

Impute missing X_{ij} with Z_{ij}



Our Approach: Nuclear Norm Relaxation

- ▶ The $\text{rank}(Z)$ term makes the problem non-convex
— combinatorially *very* hard
- ▶ $\|Z\|_* = \sum_j \lambda_j(Z)$: sum of singular values of $Z_{m \times n}$ is convex in Z
- ▶ $\|Z\|_*$ **tightest convex relaxation** of $\text{Rank}(Z)$
(Fazel, Boyd, 2002)

Notation

Define $P_{\Omega}(X)_{n \times m}$: projection onto the observed entries

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

Criterion rewritten as:

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

Context: bias variance trade-off

- ▶ SVT algorithm of Cai et. al. (2010) consider

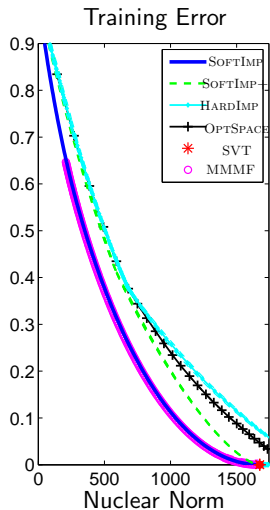
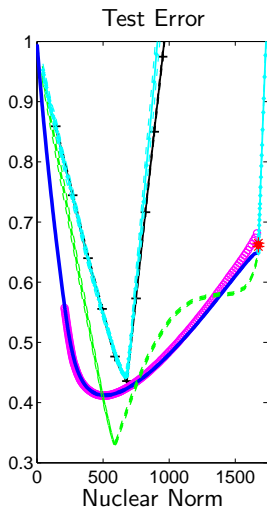
$$\begin{aligned} &\text{minimize } \|Z\|_* \\ &\text{subject to } P_\Omega(Z) = P_\Omega(X) \end{aligned}$$

- ▶ We consider

$$\begin{aligned} &\text{minimize } \|Z\|_* \\ &\text{subject to } \|P_\Omega(Z - X)\|_F^2 \leq \delta \end{aligned}$$

- ▶ First order algorithm
— scalable to large matrices via **sparse SVD**
- ▶ Assumes noise is zero – **reconstruction** problem
This is too rigid...
- ▶ In real-life, there is **noise** – fitting training data **exactly** incurs added **variance**
- ▶ Introduce **bias** to decrease **variance**
- ▶ Need **more** than a **sparse SVD**

Bias Variance Trade-Off



Convex Optimization Program

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

- ▶ (1) is a Convex Problem — SemiDefinite Program (SDP)
- ▶ Complexity of existing off the shelf solvers:
 - Interior Point Methods: $O(n^4) \dots O(n^6) \dots$
 - First Order Methods complexity: $O(n^3)$
- ▶ We compute regularization path for (1) using sequence of λ 's with warm-restarts
 - SVD per-iteration cost:
Linear in **matrix dimensions + # Observed entries**

Main ingredient – Nuclear Norm Regularization

Fully Observed Problem

$$\underset{Z}{\text{minimize}} \quad \frac{1}{2} \|Z - X\|_F^2 + \lambda \|Z\|_* \quad (2)$$

has solution given by **soft-thresholded SVD**

$$\mathbf{S}_\lambda(X) := U \cdot \text{diag}((\sigma_1 - \lambda)_+, \dots, (\sigma_m - \lambda)_+) \cdot V'$$

where the full SVD is

$$X = U \cdot \text{diag}(\sigma_1, \dots, \sigma_m) \cdot V'$$

Compute largest **few** singular vectors/ values of X

Cost depends on structure of X , **computationally difficult** when X is large

Algorithm: SOFT-IMPUTE

Minimize w.r.t. Z objective function:

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

- ▶ Fully **observed** admits analytic solution...but **unobserved** is hard
- ▶ **Impute** missing values, compute **truncated SVD** and then **Re-Impute**
- ▶ Compute entire Regularization path with warm-starts...

[Mazumder, Hastie, Tibshirani (2012), Journal of Machine Learning Research]

SOFT-IMPUTE: Path Algorithm

- 1 Initialize $Z^{\text{old}} = 0$ and create a decreasing grid Λ of values $\lambda_{\max} = \lambda_1 > \dots > \lambda_K$.
- 2 For every fixed $\lambda = \lambda_1, \lambda_2, \dots \in \Lambda$ iterate 2a-2b till convergence:
 - (2a) Compute $Z^{\text{new}} \leftarrow \mathbf{S}_\lambda(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$
 - (2b) Assign $Z^{\text{old}} \leftarrow Z^{\text{new}}$ and go to step (2a)
 - (2c) Assign $\hat{Z}_\lambda \leftarrow Z^{\text{new}}$ and go to 2
- 3 Output the sequence of solutions $\hat{Z}_{\lambda_1}, \dots, \hat{Z}_{\lambda_K}$.

HARD-IMPUTE: Path Algorithm

- 1 Initialize $Z^{\text{old}} = 0$ and create a decreasing grid Λ of values $\lambda_{\max} = \lambda_1 > \dots > \lambda_K$.
- 2 For every fixed $\lambda = \lambda_1, \lambda_2, \dots \in \Lambda$ iterate 2a-2b till convergence:
 - (2a) Compute $Z^{\text{new}} \leftarrow \mathbf{H}_\lambda(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$
 - (2b) Assign $Z^{\text{old}} \leftarrow Z^{\text{new}}$ and go to step (2a)
 - (2c) Assign $\hat{Z}_\lambda \leftarrow Z^{\text{new}}$ and go to 2
- 3 Output the sequence of solutions $\hat{Z}_{\lambda_1}, \dots, \hat{Z}_{\lambda_K}$.

HARD-IMPUTE

HARD-IMPUTE leads to upper bounds for the following rank-regularized problem:

$$\underset{Z}{\text{minimize}} \quad \frac{1}{2} \|P_{\Omega}(X) - P_{\Omega}(Z)\|_F^2 + \lambda \text{rank}(Z)$$

with the notation $\mathbf{H}_{\lambda}(M)$:

$$\underset{Z}{\text{minimize}} \quad \frac{1}{2} \|Z - M\|_F^2 + \lambda \text{rank}(Z)$$

(Can also consider, an integral sequence of ranks $r \in \{r_1, r_2, \dots\}$:

$$\underset{Z}{\text{minimize}} \quad \frac{1}{2} \|Z - M\|_F^2 \quad \text{subject to} \quad \text{rank}(Z) \leq r$$

SOFT-IMPUTE: Properties

The sequence Z_λ^{k+1}

$$Z_\lambda^{k+1} = \arg \min_Z Q_\lambda(Z|Z_k)$$

$$Q_\lambda(Z|Z_k) = \frac{1}{2} \|\{P(X) + P^\perp(Z_k)\} - Z\|_F^2 + \lambda \|Z\|_*$$

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

satisfies:

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

SOFT-IMPUTE: Properties

Theorem

Take $\lambda > 0$. The sequence of estimates $\{Z_k\}_k$ given by:

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

converges to Z_∞ , a fixed point of

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

Hence, Z_∞ minimizes $f_\lambda(Z)$.

SOFT-IMPUTE: 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* $Z^* \in \operatorname{argmin}_Z f_{\lambda}(Z)$

- Furthermore: 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 is effectively *linear* i.e. $O(\gamma^k)$ in presence of warm-starts and large λ)

SOFT-IMPUTE : Computational Bottleneck

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

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

Computational bottleneck — low-rank SVD of $P_\Omega(X) + P_\Omega^\perp(Z_k)$

Trick:

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

Computational Bottleneck: low-rank SVD

- ▶ Compute largest **few** singular vectors/ values of W
 - Direct “SVD” factorizations **infeasible**
 - Iterative Methods: utilize Wb , $W'a$
 - other randomized/ mini-batch methods.
- ▶ **smart linear algebra**: Fast iterative SVD for:

Sparse + Low Rank

Our Algorithm updates have this structure...we exploit it

- ▶ **LBPRO**: Lanczos bidiagonalization with partial reorthogonalization (PROPACK, Larsen 2000)

Computational Bottleneck: low-rank SVD

- ▶ Compute largest **few** singular vectors/ values of W
 - Direct “SVD” factorizations **infeasible**
 - Iterative Methods: utilize Wb , $W'a$
 - other randomized/ mini-batch methods.
- ▶ **smart linear algebra**: Fast iterative SVD for:

Sparse + Low Rank

Our Algorithm updates have this structure...we exploit it

- ▶ **LBPRO**: Lanczos bidiagonalization with partial reorthogonalization (PROPACK, Larsen 2000)

Total cost **per iteration** for SOFT-IMPUTE:

$$\underbrace{O(|\Omega|)}_{\text{Sparse}} + \underbrace{O((\#Users + \#Movies) \cdot r)}_{\text{Low-rank}}$$

Experimental Studies

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

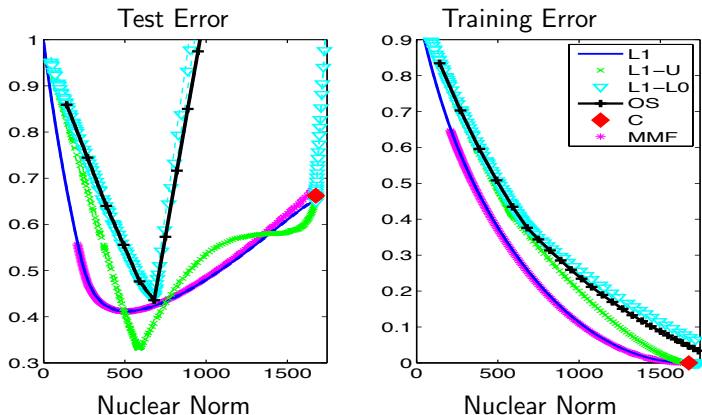


Figure: (L1-U) SOFT-IMPUTE+, (L1)SOFT-IMPUTE, (L1-L0)HARD-IMPUTE, (OS)OPTSPACE, (C) SVT, (MMF) MAX-MARGIN FACTORIZATION

MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS

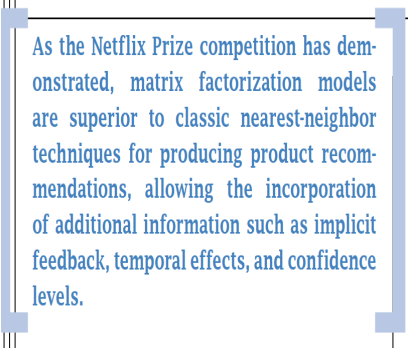
Yehuda Koren, *Yahoo Research*

Robert Bell and Chris Volinsky, *AT&T Labs—Research*

As the Netflix Prize competition has demonstrated, matrix factorization models are superior to classic nearest-neighbor techniques for producing product recommendations, allowing the incorporation of additional information such as implicit feedback, temporal effects, and confidence levels.

Such systems are particularly useful for entertainment products such as movies, music, and TV shows. Many customers will view the same movie, and each customer is likely to view numerous different movies. Customers have proven willing to indicate their level of satisfaction with particular movies, so a huge volume of data is available about which movies appeal to which customers. Companies can analyze this data to recommend movies to particular customers.

Connections with Matrix Factorization



As the Netflix Prize competition has demonstrated, matrix factorization models are superior to classic nearest-neighbor techniques for producing product recommendations, allowing the incorporation of additional information such as implicit feedback, temporal effects, and confidence levels.

Connections with Matrix Factorization

user preferences using *implicit feedback*, which indirectly reflects opinion by observing user behavior including purchase history, browsing history, search patterns, or even mouse movements. Implicit feedback usually denotes the presence or absence of an event, so it is typically represented by a densely filled matrix.

A BASIC MATRIX FACTORIZATION MODEL

Matrix factorization models map both users and items to a joint latent factor space of dimensionality f , such that user-item interactions are modeled as inner products in that space. Accordingly, each item i is associated with a

known ratings:

$$\min_{q_i, p_u} \sum_{(u,i) \in \mathcal{K}} (r_{ui} - q_i^T p_u)^2 + \lambda (\|q_i\|^2 + \|p_u\|^2) \quad (2)$$

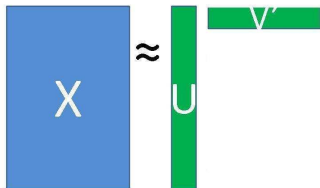
Here, \mathcal{K} is the set of the (u,i) pairs for which r_{ui} is known (the training set).

The system learns the model by fitting the previously observed ratings. However, the goal is to generalize those previous ratings in a way that predicts future, unknown ratings. Thus, the system should avoid overfitting the observed data by regularizing the learned parameters, whose magnitudes are penalized. The constant λ controls

Connections with Matrix Factorization

Fix $\text{Rank}(Z) = r$ (small) ie $Z = U_{m \times r} V'_{n \times r}$

$$\underset{U, V}{\text{minimize}} \sum_{(i,j): \text{Observed}} ([UV']_{ij} - X_{ij})^2 + \lambda(\|U\|_F^2 + \|V\|_F^2)$$



- ▶ Maximum-Margin Matrix Factorization (MMMF), Regularized SVD
- ▶ **bi-convex** criterion, alternating ridge regression... (Non-convex)

Connections between MMMF & SOFT-IMPUTE

Recall:

$$\text{MMMF : } \underset{U_{n \times r}, V_{m \times r}}{\text{minimize}} \quad \frac{1}{2} \|P_{\Omega}(X - UV')\|_F^2 + \frac{\lambda}{2} (\|U\|_F^2 + \|V\|_F^2)$$

$$\text{Nuclear} \quad \underset{Z}{\text{minimize}} \quad \frac{1}{2} \|P_{\Omega}(X - Z)\|_F^2 + \lambda \|Z\|_*$$

Lemma

For any matrix W , the following holds:

$$\|W\|_* = \min_{U, V: W=UV^T} \frac{1}{2} (\|U\|_F^2 + \|V\|_F^2). \quad (3)$$

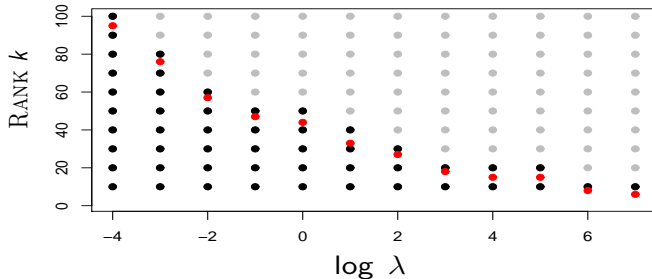
If $\text{rank}(W) = k \leq \min\{m, n\}$, then the minimum above is attained at a factor decomposition $W = U_{m \times k} V_{n \times k}^T$.

Connections between MMMF & SOFT-IMPUTE

Consequences of the Lemma:

- ▶ Solution-space of MMMF **contains** solutions of Nuclear.
- ▶ For *large* r : $\text{MMMF} \equiv \text{Nuclear}$.

However, MMMF is **non-convex** in (U, V) .



Improved SOFT-IMPUTE: SOFT-IMPUTE-ALS

[Ref.: Hastie, M., Lee, Zadeh (2015), Journal of Machine Learning Research]

Improving SOFT-IMPUTE?

- ▶ SOFT-IMPUTE works quite well, but the main bottleneck are the “black-box” SVDs
- ▶ Need to rely on software for large scale SVDs which has no clue that we are using it for SOFT-IMPUTE.
- ▶ Recall for a fixed λ , the sequence $Z_{k+1} \rightarrow Z_\infty$. In addition, $Z_{k+1} - Z_k \rightarrow 0$.
- ▶ SVDs do not exploit this information. How do we fix this?
- ▶ Idea: combine ideas from MMMF and SOFT-IMPUTE to solve nuclear norm regularized problem.

Improving SOFT-IMPUTE?

1. Recall, SOFT-IMPUTE solves:

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

2. The above is equivalent to:

$$(A_{k+1}, B_{k+1}) \in \operatorname{argmin}_{A, B} \left(\frac{1}{2} \|AB' - (P_\Omega(X + P_\Omega^\perp(Z_k)))\|_F^2 + \lambda \|AB\|_* \right)$$

$$(A_{k+1}, B_{k+1}) \in \operatorname{argmin}_{A, B} \left(\frac{1}{2} \|AB' - (P_\Omega(X + P_\Omega^\perp(Z_k)))\|_F^2 + \frac{\lambda}{2} (\|A\|_F^2 + \|B\|_F^2) \right)$$

[This can be done via a low-rank SVD]

- Where, $Z_k = A_k B_k'$ for all k and the rank of A, B is large enough.

Improving SOFT-IMPUTE?

- Idea: Replace a full minimization wrt A, B in Step 2.
- Perform the following iterations (for $k \geq 1$):

$$A_{k+1} \in \operatorname{argmin}_A \left(\frac{1}{2} \|AB'_k - (P_\Omega(X + P_\Omega^\perp(A_k B'_k)))\|_F^2 + \frac{\lambda}{2} \|A\|_F^2 \right)$$

$$B_{k+1} \in \operatorname{argmin}_B \left(\frac{1}{2} \|A_{k+1}B' - (P_\Omega(X + P_\Omega^\perp(A_{k+1} B'_k)))\|_F^2 + \frac{\lambda}{2} \|B\|_F^2 \right)$$

- Track the convergence of $Z_k := A_k B'_k$ with k . This will *converge* to a solution of Nuclear.
(imprecise statement: more on this later...)

SOFT-IMPUTE-ALS

$$F(A, B) := \frac{1}{2} \|P_{\Omega}(X - AB^T)\|_F^2 + \frac{\lambda}{2} \|A\|_F^2 + \frac{\lambda}{2} \|B\|_F^2.$$

We define the surrogate functions

$$\begin{aligned} Q_A(Z_1|A, B) &:= \frac{1}{2} \|P_{\Omega}(X - Z_1 B^T) + P_{\Omega}^{\perp}(AB^T - Z_1 B^T)\|_F^2 \quad (4) \\ &\quad + \frac{\lambda}{2} \|Z_1\|_F^2 + \frac{\lambda}{2} \|B\|_F^2 \end{aligned}$$

$$\begin{aligned} Q_B(Z_2|A, B) &:= \frac{1}{2} \|P_{\Omega}(X - AZ_2^T) + P_{\Omega}^{\perp}(AB^T - AZ_2^T)\|_F^2 \quad (5) \\ &\quad + \frac{\lambda}{2} \|A\|_F^2 + \frac{\lambda}{2} \|Z_2\|_F^2. \end{aligned}$$

SOFT-IMPUTE-ALS

Repeat until Convergence

1. $k \leftarrow k + 1$.
2. $X^* \leftarrow P_{\Omega}(X) + P_{\Omega}^{\perp}(AB^T) = P_{\Omega}(X - AB^T) + AB^T$
3. $A \leftarrow X^*B(B^TB + \lambda I)^{-1} = \operatorname{argmin}_{Z_1} Q_A(Z_1|A, B)$.
4. $X^* \leftarrow P_{\Omega}(X) + P_{\Omega}^{\perp}(AB^T)$
5. $B \leftarrow X^{*T}A(A^TA + \lambda I)^{-1} = \operatorname{argmin}_{Z_2} Q_B(Z_2|A, B)$.

Observe that the softImpute-ALS algorithm can be described as the following iterative procedure:

$$A_{k+1} \in \operatorname{argmin}_{Z_1} Q_A(Z_1|A_k, B_k) \quad (6)$$

$$B_{k+1} \in \operatorname{argmin}_{Z_2} Q_B(Z_2|A_{k+1}, B_k). \quad (7)$$

SOFT-IMPUTE-ALS: Work done per iteration

Consider the A update.

- ▶ Need to form: $P_{\Omega}(X - AB^T) + AB^T$:
 - Cost to compute $P_{\Omega}(X - AB^T)$ is $O(r|\Omega|)$ flops.
 - We do not compute AB^T , but directly obtain the minimizer.
- ▶ The matrix $B(B^T B + \lambda I)^{-1}$ requires $O(2nr^2 + r^3)$ flops to form.
- ▶ The multiplication $X^* B(B^T B + \lambda I)^{-1}$ requires $O(r|\Omega| + mr^2 + nr^2)$ flops, using the sparse plus low-rank structure of X^* .
- ▶ Total cost per iteration: $O(2r|\Omega| + mr^2 + 3nr^2 + r^3)$.

A competing Algorithm: ALS

$$F(A, B) := \frac{1}{2} \|P_{\Omega}(X - AB^T)\|_F^2 + \frac{\lambda}{2} \|A\|_F^2 + \frac{\lambda}{2} \|B\|_F^2.$$

► Repeat the following steps till convergence:

► $A \in \operatorname{argmin}_A F(A, B)$

for (i in 1 to m) do

$$A_i \leftarrow \left(\sum_{j \in \Omega_i} B_j B_j^T \right)^{-1} \left(\sum_{j \in \Omega_i} X_{ij} B_j \right)$$

end

► $B \in \operatorname{argmin}_B F(A, B)$

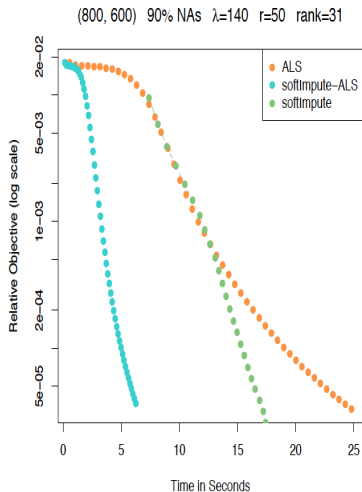
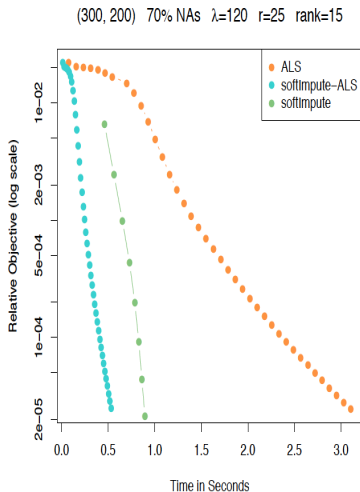
for (j=1 to n) do

$$B_j \leftarrow \left(\sum_{i \in \Omega_j} A_i A_i^T \right)^{-1} \left(\sum_{i \in \Omega_j} X_{ij} A_i \right)$$

end

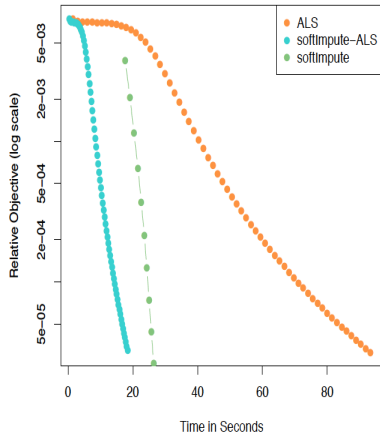
- ▶ ALS may decrease the criterion at each iteration more than `softImpute-ALS`.
- ▶ It tends to be slower because the cost is higher by a factor $O(r)$.
- ▶ Cost per iteration of ALS:
 - ▶ The cost for each ridge regression is $O(|\Omega_j|r^2 + r^3)$, so the cost of one iteration is $O(2|\Omega|r^2 + mr^3 + nr^3)$.
 - ▶ Hence the cost of one iteration of ALS is r times more flops than one iteration of `softImpute-ALS`.

Performance in practice

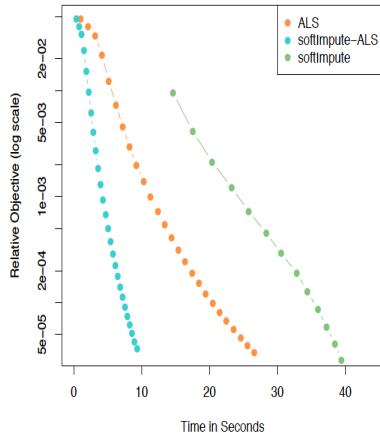


Performance in practice

(1200, 900) 80% NAs $\lambda=300$ $r=50$ rank=27

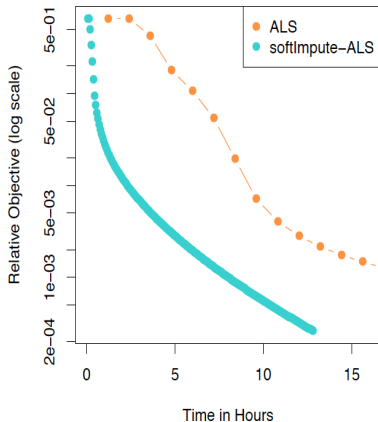


(943, 1682) 93% NAs $\lambda=20$ $r=40$ rank=35

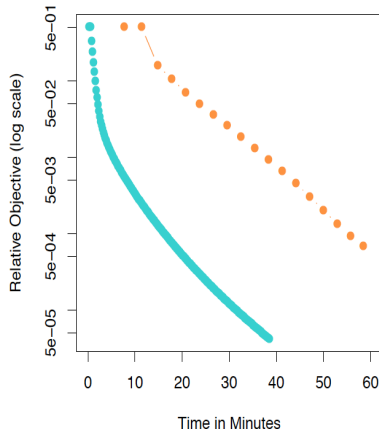


Performance in practice

Netflix (480K, 18K) $\lambda=100$ $r=100$



MovieLens 10M (72K, 10K) $\lambda=50$ $r=100$



Analysis of SOFT-IMPUTE-ALS

$$F(A, B) := \frac{1}{2} \|P_{\Omega}(X - AB^T)\|_F^2 + \frac{\lambda}{2} \|A\|_F^2 + \frac{\lambda}{2} \|B\|_F^2.$$

We define the surrogate functions

$$\begin{aligned} Q_A(Z_1|A, B) &:= \frac{1}{2} \|P_{\Omega}(X - Z_1 B^T) + P_{\Omega}^{\perp}(AB^T - Z_1 B^T)\|_F^2 \\ &\quad + \frac{\lambda}{2} \|Z_1\|_F^2 + \frac{\lambda}{2} \|B\|_F^2 \end{aligned}$$

$$\begin{aligned} Q_B(Z_2|A, B) &:= \frac{1}{2} \|P_{\Omega}(X - AZ_2^T) + P_{\Omega}^{\perp}(AB^T - AZ_2^T)\|_F^2 \\ &\quad + \frac{\lambda}{2} \|A\|_F^2 + \frac{\lambda}{2} \|Z_2\|_F^2. \end{aligned}$$

Also define:

$$g(AB^T) := \frac{1}{2} \|P_{\Omega}(X - AB^T)\|_F^2$$

Analysis of SOFT-IMPUTE-ALS

$$Q_A(Z_1|A, B) \geq F(Z_1, B), \quad Q_B(Z_2|A, B) \geq F(A, Z_2),$$

In addition, the following equality holds:

$$Q_A(A|A, B) = F(A, B) = Q_B(B|A, B)$$

Lemma

Let $\{(A_k, B_k)\}$ be the iterates generated by `softImpute-ALS`. The function values are monotonically decreasing,

$$F(A_k, B_k) \geq F(A_{k+1}, B_k) \geq F(A_{k+1}, B_{k+1}), \quad k \geq 1.$$

Analysis of SOFT-IMPUTE-ALS

Lemma

Let $(A_k, B_k), k \geq 1$ be the sequence generated by the *softImpute-ALS* algorithm. The decreasing sequence of objective values $F(A_k, B_k)$ converges to $F^\infty \geq 0$ (say).

Furthermore, the *softImpute-ALS* algorithm terminates at the rate of $O(\frac{1}{K})$ (where, K denotes the number of iterations).

(Note the above is a statement about the non-convex problem $F(A, B)$)

Analysis of SOFT-IMPUTE-ALS

- ▶ The above statements are about the function $(A, B) \mapsto F(A, B)$.
- ▶ What can be said about Nuclear:

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

- ▶ Given a tuple (A, B) we have:

$$\|AB^T\|_* \leq \frac{1}{2}(\|A\|_F^2 + \|B\|_F^2),$$

with equality holding for a particular parametrization of A, B :

$$A = UD^{1/2}, B = VD^{1/2}, \quad \text{where,} \quad AB^T = UDV^T \quad (\text{SVD}),$$

- ▶ For any tuple A_k, B_k (expressed in the above form) we have the following:

$$F(A_k, B_k) \geq H(A_k B_k^T)$$

Analysis of SOFT-IMPUTE-ALS

Lemma

Let the sequence (A_k, B_k) generated by `softImpute-ALS` be stored in terms of the factored SVD representation.

This results in a decreasing sequence of objective values for problem Nuclear:

$$H(A_k B_k^T) \geq H(A_{k+1} B_{k+1}^T)$$

with $H(A_k B_k^T) = F(A_k, B_k)$, for all k . The sequence $H(A_k B_k^T)$ thus converges to F^∞ .

Analysis of SOFT-IMPUTE-ALS

At stationarity i.e. at a fixed point of softImpute-ALS we have the following:

$$\begin{aligned}A_* &\in \operatorname{argmin}_A \frac{1}{2} \|P_\Omega(X - AB_*^T)\|_F^2 + \frac{\lambda}{2} (\|A\|_F^2 + \|B_*\|_F^2) \\&= \operatorname{argmin}_A \frac{1}{2} \left\| \left(P_\Omega(X) + P_\Omega^\perp(A_* B_*^T) \right) - AB_*^T \right\|_F^2 + \frac{\lambda}{2} (\|A\|_F^2 + \|B_*\|_F^2) \\B_* &\in \operatorname{argmin}_B \frac{1}{2} \|P_\Omega(X - A_* B^T)\|_F^2 + \frac{\lambda}{2} (\|A_*\|_F^2 + \|B\|_F^2) \\&= \operatorname{argmin}_B \frac{1}{2} \left\| \left(P_\Omega(X) + P_\Omega^\perp(A_* B_*^T) \right) - A_* B^T \right\|_F^2 + \frac{\lambda}{2} (\|A_*\|_F^2 + \|B\|_F^2)\end{aligned}$$

Analysis of SOFT-IMPUTE-ALS

The above fixed point updates are very closely related to the following optimization problem:

$$\underset{A_{m \times r}, B_{m \times r}}{\text{minimize}} \quad \frac{1}{2} \| (P_{\Omega}(X) + P_{\Omega}^{\perp}(A_* B_*^T)) - AB \|_F^2 + \frac{\lambda}{2} (\|A\|_F^2 + \|B\|_F^2)$$

The solution to the above is given by the nuclear norm thresholding operation (with a rank r constraint) on the matrix $P_{\Omega}(X) + P_{\Omega}^{\perp}(A_* B_*^T)$:

$$\underset{Z: \text{rank}(Z) \leq r}{\text{minimize}} \quad \frac{1}{2} \| (P_{\Omega}(X) + P_{\Omega}^{\perp}(A_* B_*^T)) - Z \|_F^2 + \frac{\lambda}{2} \|Z\|_*$$

Analysis of SOFT-IMPUTE-ALS

Lemma Suppose the nuclear norm problem has a solution Z^* with $\text{rank}(Z^*) = r^*$. Then, for $A_* B_*^T$ to be a solution to Nuclear, the following conditions are sufficient:

- (a) $r^* \leq r$
- (b) The outer product $A_* B_*^T$ must be the solution to the *fully observed* nuclear norm regularized problem:

$$A_* B_*^T \in \underset{Z}{\operatorname{argmin}} \quad \frac{1}{2} \| (P_\Omega(X) + P_\Omega^\perp(A_* B_*^T)) - Z \|_F^2 + \lambda \|Z\|_* .$$

The above condition can be verified fairly easily; and requires doing a low-rank SVD of the matrix $P_\Omega(X) + P_\Omega^\perp(A_* B_*^T)$

Broader Perspective

- ▶ In spirit, softImpute-ALS is a non-convex algorithm to solve a convex problem.
- ▶ Related ideas have been proposed in mathematical optimization (Burer, Monteiro '05 (Math. Prog) and Journee et al '10 (SIOPT)). However, these results are general purpose: requires to check a stationary point is a local minimizer (abstract for the problem herein).
- ▶ Some other researchers (Jain, et al '11; Hardt '14) say that the convex problem is hard to solve. They propose ALS for:

$$\min_A \|P_\Omega(X - AB^T)\|_F^2 \quad \overset{\leftarrow}{\underset{\rightarrow}{\quad}} \quad \min_B \|P_\Omega(X - AB^T)\|_F^2$$

and statistically analyze ALS under conditions on the data — results similar (worse) than nuclear norm problem.

Distributed Computing: SOFT-IMPUTE-ALS-//

1. Performed in Apache Spark.
2. Input matrix $P_{\Omega}(X)$ split across (chunks of) rows. The transpose is split across (chunks of) rows across different machines.
- 3a. Recall the A update:
 - ▶ $X^* \leftarrow P_{\Omega}(X) + P_{\Omega}^{\perp}(AB^T) = P_{\Omega}(X - AB^T) + AB^T$
 - ▶ $A \leftarrow X^*B(B^TB + \lambda I)^{-1}$.
- 3b. Step 3a. is computed across different machines (split by chunks of rows of A). The current model (i.e., the current guess for B) and the factorization: $B(B^TB + \lambda I)^{-1}$ is done once and split into different machines. Each machine has its own copy of (chunks of) rows of A .
- 3c. Each machine broadcasts its own copy of A to the master for computing $A(A^TA + \lambda I)^{-1}$.
4. Same thing done for B update.

Distributed Computing: SOFT-IMPUTE-ALS-//

Matrix Size	Number of Nonzeros	Time per iteration (s)
$10^6 \times 10^6$	10^6	5
$10^6 \times 10^6$	10^9	6
$10^7 \times 10^7$	10^9	139

Table: Running times for distributed softimpute-ALS

Summary

- ▶ `SOFT-IMPUTE` is the master algorithm, EM-inspired. Requires black box SVDs.
- ▶ Careful algorithmic thinking leads to `SOFT-IMPUTE-ALS`
- ▶ `SOFT-IMPUTE-ALS-//` admits a parallel implementation.
- ▶ Ideas extend to obtaining good upper bounds for the rank-minimization (not nuclear norm) problem.

Summary

- ▶ Another popular method: Frank-Wolfe/ Conditional Gradient type methods.
(Jaggi '12; Harchaoui, Juditsky, Nemirovski '15)
- ▶ Easier subproblems, but has several issues in practice.
- ▶ Some significant modifications required. Can solve smaller problems.
(Freund, Grigas, M. '15)