# Probabilistic Low-Rank Matrix Completion with Adaptive Spectral Regularization Algorithms

Adrien Todeschini†, François Caron⋆, Marie Chavent†

†Inria Bordeaux, ⋆Univ. Oxford

*January* 2014

# Summary

# Summary

# Matrix completion

## Objective

- Complete a matrix of potentially large dimension based on a small (and potentially noisy) subset of its entries [Srebro et al., 2005, Candès and Plan, 2010].

## Popular application: collaborative filtering

- To build automatic recommender systems, where the rows correspond to users, the columns to items and entries may be ratings or binaries (like/dislike).
- The objective is then to predict user preferences from a subset of the entries.
- e.g. Netflix, Amazon, Google...

# Model

- $Z$ an $m \times n$ unknown matrix of preferences
- Low rank assumption:

$$\underbrace{Z}_{m\times n} \simeq \underbrace{A}_{m\times k}\underbrace{B^T}_{k\times n}$$

  with $k \ll \min(m, n)$.
- Likelihood: we typically observe a noisy version $X_{ij}$ of some entries $(i, j) \in \Omega$ where $\Omega \subset \{1, \dots, m\} \times \{1, \dots, n\}$.

$$X_{ij} = Z_{ij} + \varepsilon_{ij}, \;\; \varepsilon_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2) \;\forall (i, j) \in \Omega, \qquad (1)$$

# Further notations

- Frobenius norm:

$$||X||_F^2 = \sum_{(i,j)} X_{ij}^2$$

- Subset operators:

$$P_\Omega(X)(i,j) = \begin{cases} X_{ij} & \text{if } (i,j) \in \Omega \\ 0 & \text{otherwise} \end{cases}$$

$$P_\Omega^\perp(X)(i,j) = \begin{cases} 0 & \text{if } (i,j) \in \Omega \\ X_{ij} & \text{otherwise} \end{cases}$$

- $r = \min(m, n)$
- $X = \widetilde{U}\widetilde{D}\widetilde{V}^T$ is the singular value decomposition (SVD) of $X$ with $\widetilde{D} = \text{diag}(\widetilde{d}_1, \dots, \widetilde{d}_r)$ and $\widetilde{d}_1 \geq \widetilde{d}_2 \geq ... \geq \widetilde{d}_r \geq 0$
- Nuclear norm: $||X||_* = \sum_{i=1}^r \widetilde{d}_i$

# Optimization problem

$$\begin{aligned} \underset{Z}{\text{minimize}} \quad & \text{rank}(Z) \\ \text{subject to} \quad & \frac{1}{2\sigma^2} \sum_{(i,j)\in\Omega} (X_{ij} - Z_{ij})^2 \leq \delta \end{aligned} \tag{2}$$

$$\Leftrightarrow \underset{Z}{\text{minimize}} \quad \underbrace{\frac{1}{2\sigma^2} ||P_\Omega(X) - P_\Omega(Z)||_F^2}_{\text{- log likelihood}} + \underbrace{\lambda \ \text{rank}(Z)}_{\text{penalty}} \tag{3}$$

- Rank penalty: non convex problem
- Computationally hard for general subset $\Omega$
- Nuclear norm penalty: convex relaxation [Fazel, 2002, Candès et al., 2008, Mazumder et al., 2010]

$$\underset{Z}{\text{minimize}} \quad \frac{1}{2\sigma^2} ||P_\Omega(X) - P_\Omega(Z)||_F^2 + \lambda \ ||Z||_* \tag{4}$$

# Summary

# Rank penalty

- Non convex problem

$$\underset{Z}{\text{minimize}} \quad \frac{1}{2\sigma^2}||X - Z||_F^2 + \lambda \, \text{rank}(Z) \tag{5}$$

- Global solution given by a hard-thresholded (truncated) SVD

$$\hat{Z} = \mathbf{H}_{\lambda\sigma^2}(X) \tag{6}$$

where $\mathbf{H}_\lambda(X) = \widetilde{U}\widetilde{D}^\lambda \widetilde{V}^T$ with $\widetilde{D}^\lambda = \text{diag}((\widetilde{d}_1)_{\lambda+}, \ldots, (\widetilde{d}_r)_{\lambda+})$ and $t_{\lambda+} = \begin{cases} t & \text{if } t \geq \lambda \\ 0 & \text{otherwise} \end{cases}$.

# Nuclear norm penalty

- Convex relaxation

$$\underset{Z}{\text{minimize}} \quad \frac{1}{2\sigma^2}||X - Z||_F^2 + \lambda\, ||Z||_* \tag{7}$$

- Global solution given by a soft-thresholded SVD [Cai et al., 2010, Mazumder et al., 2010]

$$\widehat{Z} = \mathbf{S}_{\lambda\sigma^2}(X)$$

where $\mathbf{S}_\lambda(X) = \widetilde{U}\widetilde{D}_\lambda\widetilde{V}^T$ with $\widetilde{D}_\lambda = \text{diag}((\widetilde{d}_1 - \lambda)_+, \dots, (\widetilde{d}_r - \lambda)_+)$ and $t_+ = \max(t, 0)$.

- The solution to (7) can be interpreted as the Maximum A Posteriori (MAP) estimate

$$\widehat{Z} = \arg\max_Z \left[\log p(X|Z) + \log p(Z)\right]$$

under the likelihood (1) and prior

$$p(Z) \propto \exp\left(-\lambda \left\|Z\right\|_*\right)$$

# MAP interpretaton

Assuming $Z = UDV^T$, with $D = \text{diag}(d_1, d_2, \dots, d_r)$ this can be further decomposed as

$$p(Z) = p(U)p(V)p(D)$$

where

- $U$ and $V$ follow a uniform Haar prior distribution on the unitary matrices
- the singular values $d_i$ follow an exponential distribution

$$p(D) = p(d_1, \dots, d_r) = \prod_{i=1}^{r} \text{Exp}\left(d_i; \lambda\right) \quad (8)$$

The exponential distribution has a mode at 0, hence favoring sparse solution.

# Hierarchical adaptive spectral penalty [Todeschini et al., 2013]

- Idea: to bridge the gap between the nuclear norm and the rank penalty
- We consider the following hierarchical prior for the low rank matrix $Z$.

$$p(d_1, \ldots, d_r | \gamma_1, \ldots \gamma_r) = \prod_{i=1}^{r} p(d_i|\gamma_i) = \prod_{i=1}^{r} \text{Exp}(d_i; \gamma_i)$$

$$p(\gamma_1, \ldots, \gamma_r) = \prod_{i=1}^{r} p(\gamma_i) = \prod_{i=1}^{r} \text{Gamma}(\gamma_i; a, b)$$

- Marginal distribution over $d_i$:

$$p(d_i) = \int_0^\infty \text{Exp}(d_i; \gamma_i)\, \text{Gamma}(\gamma_i; a, b) d\gamma_i = \frac{ab^a}{(d_i + b)^{a+1}} \qquad (9)$$

It is a Pareto distribution with heavier tails than exponential distribution

# Hierarchical adaptive spectral penalty

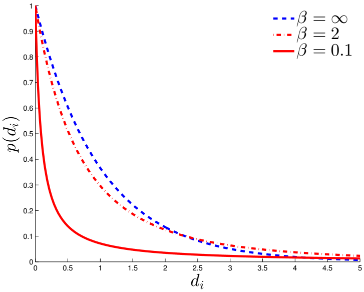
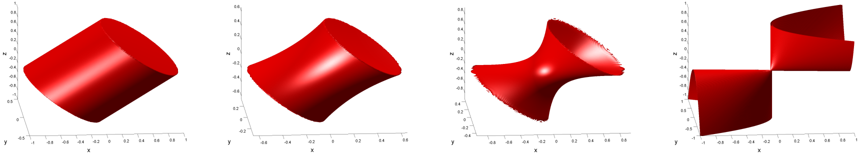



Figure : Marginal distribution $p(d_i)$ with $a = b = \beta$

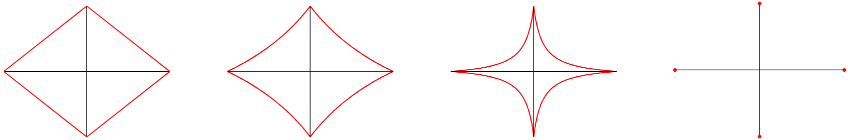
- HASP penalty: admits as special cases the nuclear norm penalty $\lambda||Z||_*$ when $a = \lambda b$ and $b \to \infty$.

$$pen(Z) = -\log p(Z) = -\sum_{i=1}^{r} \log(p(d_i)) = \sum_{i=1}^{r} (a+1)\log(b + d_i) \qquad (10)$$

# Hierarchical adaptive spectral penalty



(a) Nuclear norm (b) HASP ($\beta = 1$) (c) HASP ($\beta = 0.1$) (d) Rank penalty



(e) $\ell_1$ norm (f) HAL ($\beta = 1$) (g) HAL ($\beta = 0.1$) (h) $\ell_0$ norm

Figure : Top: Manifold of constant penalty, for a symmetric $2 \times 2$ matrix $Z = [x, y; y, z]$ for (a) the nuclear norm, hierarchical adaptive spectral penalty with $a = b = \beta$ (b) $\beta = 1$ and (c) $\beta = 0.1$, and (d) the rank penalty. Bottom: contour of constant penalty for a diagonal matrix $[x, 0; 0, z]$, where one recovers the classical (e) lasso, (f-g) hierarchical lasso and (h) $\ell_0$ penalties.

# EM algorithm for MAP estimation

We derive an Expectation Maximization (EM) algorithm to obtain a MAP estimate

$$\widehat{Z} = \arg\max_{Z} \left[\log p(X|Z) + \log p(Z)\right]$$

i.e. to minimize

$$L(Z) = \frac{1}{2\sigma^2} \|X - Z\|_F^2 + \sum_{i=1}^{r} (a+1) \log(b + d_i) \tag{11}$$

# EM algorithm for MAP estimation

- Latent variables: $\gamma = (\gamma_1, \ldots, \gamma_r)$
- E step:

$$
\begin{aligned}
Q(Z, Z^*) &= \mathbb{E}\left[\log(p(X, Z, \gamma))|Z^*, X\right] \\
&= C - \frac{1}{2\sigma^2}\|X - Z\|_F^2 - \sum_{i=1}^{r} \mathbb{E}[\gamma_i|d_i^*]d_i \\
&= C - \frac{1}{2\sigma^2}\|X - Z\|_F^2 - \sum_{i=1}^{r} \omega_i d_i
\end{aligned}
$$

where $\omega_i = \mathbb{E}[\gamma_i|d_i^*] = \frac{a+1}{b+d_i^*}$.

# EM algorithm for MAP estimation

- M step:

$$\underset{Z}{\text{minimize}} \quad \frac{1}{2\sigma^2}\|X - Z\|_F^2 + \sum_{i=1}^{r} \omega_i d_i \qquad (12)$$

(12) is an adaptive spectral penalty regularized optimization problem, with weights $\omega_i = \frac{a+1}{b+d_i^*}$.

$$d_1^* \geq d_2^* \geq \ldots \geq d_r^*$$
$$\Rightarrow 0 \leq \omega_1 \leq \omega_2 \leq \ldots \leq \omega_r \qquad (13)$$

Given condition (13), the solution is given by a weighted soft-thresholded SVD [Gaïffas and Lecué, 2011]

$$\widehat{Z} = \mathbf{S}_{\sigma^2\omega}(X) \qquad (14)$$

where $\mathbf{S}_\omega(X) = \widetilde{U}\widetilde{D}_\omega\widetilde{V}^T$ with $\widetilde{D}_\omega = \text{diag}((\widetilde{d}_1 - \omega_1)_+, \ldots, (\widetilde{d}_r - \omega_r)_+)$.

# EM algorithm for MAP estimation

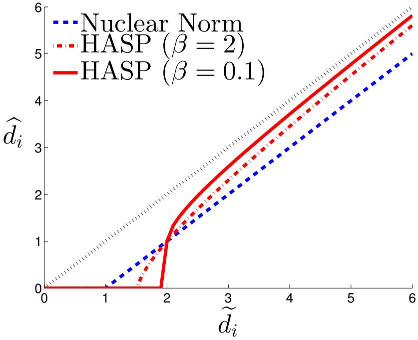



Figure : Thresholding rules on the singular values $\widetilde{d}_i$ of $X$

The weights will penalize less heavily higher singular values, hence reducing bias.

# HAST algorithm

**Hierarchical Adaptive Soft Thresholded (HAST) algorithm for low rank estimation of complete matrices**

Initialize $Z^{(0)}$. At iteration $t \geq 1$

- For $i = 1, \ldots, r$, compute the weights $\omega_i^{(t)} = \frac{a+1}{b+d_i^{(t-1)}}$
- Set $Z^{(t)} = \mathbf{S}_{\sigma^2 \omega^{(t)}}(X)$
- If $\frac{L(Z^{(t-1)}) - L(Z^{(t)})}{L(Z^{(t-1)})} < \varepsilon$ then return $\widehat{Z} = Z^{(t)}$

This algorithm admits the soft-thresholded SVD operator as a special case when $a = b\lambda$ and $b = \beta \to \infty$.

# Settings

- Parametrization:
  - We set $b = \beta$ and $a = \lambda\beta$ where $\lambda$ and $\beta$ are tuning parameters that can be chosen by cross-validation.
  - It is possible to estimate $\sigma$ within the EM algorithm. In our experiments, we have found the results not very sensitive to the setting of $\sigma$, and set it to 1.
- Initialization:
  - As $\lambda$ is the mean value of the regularization parameter $\gamma_i$, we initialize the algorithm with the soft thresholded SVD with parameter $\sigma^2\lambda$.

# Summary

# Matrix completion

- Only a subset $\Omega \subset \{1, \ldots, m\} \times \{1, \ldots, n\}$ of the entries of the matrix $X$ is observed.
- Relies on imputing missing values
- Assuming the same prior (9), the MAP estimate is obtained by minimizing

$$L(Z) = \frac{1}{2\sigma^2} \|P_\Omega(X) - P_\Omega(Z)\|_F^2 + (a+1) \sum_{i=1}^{r} \log(b + d_i) \qquad (15)$$

# EM algorithm for MAP estimation

- Latent variables: $\gamma$ and $P_\Omega^\perp(X)$
- E step:

$$
\begin{aligned}
Q(Z, Z^*) &= \mathbb{E}\left[\log(p(P_\Omega(X), P_\Omega^\perp(X), Z, \gamma)) | Z^*, P_\Omega(X)\right] \\
&= C_2 - \frac{1}{2\sigma^2}\left\{\left\|P_\Omega(X) + P_\Omega^\perp(Z^*) - Z\right\|_F^2\right\} - \sum_{i=1}^r \mathbb{E}[\gamma_i | d_i^*] d_i
\end{aligned}
$$

- M step:

$$
\underset{Z}{\text{minimize}} \frac{1}{2\sigma^2}\|X^* - Z\|_F^2 - \sum_{i=1}^r \omega_i d_i \qquad (16)
$$

where $\omega_i = \mathbb{E}[\gamma_i | d_i^*]$ and $X^* = P_\Omega(X) + P_\Omega^\perp(Z^*)$ is the observed matrix, completed with entries in $Z^*$.
We now have a complete matrix problem whose solution is obtained with a weighted soft-thresholded SVD.

# EM algorithm for MAP estimation

**Hierarchical Adaptive Soft Impute (HASI) algorithm for matrix completion**

Initialize $Z^{(0)}$. At iteration $t \geq 1$

- For $i = 1, \dots, r$, compute the weights $\omega_i^{(t)} = \frac{a+1}{b+d_i^{(t-1)}}$
- Set $Z^{(t)} = \mathbf{S}_{\sigma^2 \omega^{(t)}} \left( P_\Omega(X) + P_\Omega^\perp(Z^{(t-1)}) \right)$
- If $\frac{L(Z^{(t-1)}) - L(Z^{(t)})}{L(Z^{(t-1)})} < \varepsilon$ then return $\widehat{Z} = Z^{(t)}$

- HASI algorithm admits the Soft-Impute algorithm of [Mazumder et al., 2010] as a special case when $a = \lambda b$ and $b = \beta \to \infty$. In this case, one obtains at each iteration $\omega_i^{(t)} = \lambda$ for all $i$.
- On the contrary, when $\beta < \infty$, our algorithm adaptively updates the weights so that to penalize less heavily higher singular values.

# Initialization

- The objective function (15) is in general not convex and different initializations may lead to different modes.
- As in the complete case, we suggest to set $a = \lambda b$ and $b = \beta$ and to initialize the algorithm with the Soft-Impute algorithm with regularization parameter $\sigma^2\lambda$.

# Scaling

- Similarly to the Soft-Impute algorithm, the computationally demanding part of HASI is $\mathbf{S}_{\sigma^2\omega^{(t)}}\left(P_\Omega(X) + P_\Omega^\perp(Z^{(t-1)})\right)$ which requires calculating a low rank truncated SVD.
- For large matrices, one can resort to the PROPACK algorithm [Larsen, 2004]. This sophisticated linear algebra algorithm can efficiently compute the truncated SVD of the "sparse + low rank" matrix

$$P_\Omega(X) + P_\Omega^\perp(Z^{(t-1)}) = \underbrace{P_\Omega(X) - P_\Omega(Z^{(t-1)})}_{\text{sparse}} + \underbrace{Z^{(t-1)}}_{\text{low rank}}$$

and can thus handle large matrices.

# Summary

# Simulated data

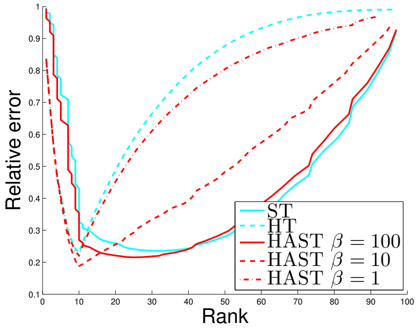
## Procedure

- We generate Gaussian matrices $A$ and $B$ respectively of size $m \times q$ and $n \times q$, $q \leq r$ so that the matrix $Z = AB^T$ is of low rank $q$. A Gaussian noise of variance $\sigma^2$ is then added to the entries of $Z$ to obtain the matrix $X$.
- The signal to noise ratio is defined as $\text{SNR} = \sqrt{\frac{\text{var}(Z)}{\sigma^2}}$.
- We set $m = n = 100$ and $\sigma = 1$.
- We run all the algorithms with a precision $\epsilon = 10^{-9}$ and a maximum number of $t_{max} = 200$ iterations (initialization included for HASI).
- For the HASP penalty, we set $a = \lambda\beta$ and $b = \beta$.
- We compute the solutions over a grid of 50 values of the regularization parameter $\lambda$ linearly spaced from $\lambda_0$ to 0, where $\lambda_0 = ||P_\Omega(X)||_2$ is the largest singular value of the input matrix $X$, padded with zeros. This is done for three different values $\beta = 1, 10, 100$.
- We compute *err*, the relative error between the estimated matrix $\widehat{Z}$ and the true matrix $Z$ in the complete case, and $err_{\Omega^\perp}$ in the incomplete case, where

$$err = \frac{||\widehat{Z} - Z||_F^2}{||Z||_F^2} \quad \text{and} \quad err_{\Omega^\perp} = \frac{||\widehat{P}_\Omega^\perp(\widehat{Z}) - P_\Omega^\perp(Z)||_F^2}{||P_\Omega^\perp(Z)||_F^2}$$

# Simulated data

## Complete case



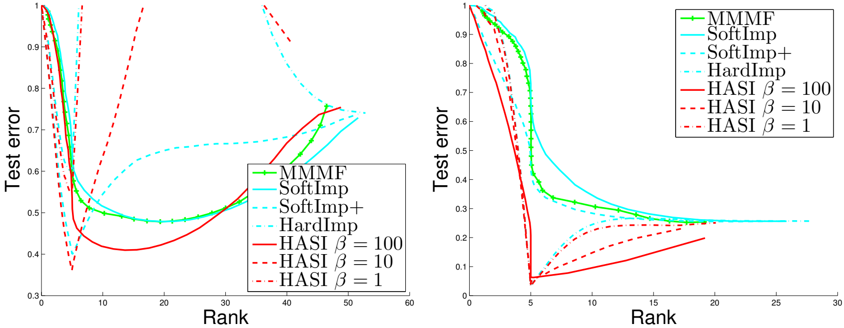

(a) SNR=1; Complete; rank=10

Figure : Test error w.r.t. the rank obtained by varying the value of the regularization parameter $\lambda$.

- The HASP penalty provides a bridge/tradeoff between the nuclear norm and the rank penalty.
- For example, value of $\beta = 10$ show a minimum at the true rank $q = 10$ as HT, but with a lower error when the rank is overestimated.

# Simulated data

## Incomplete case




(a) SNR=1; 50% missing; rank=5 (b) SNR=10; 80% missing; rank=5

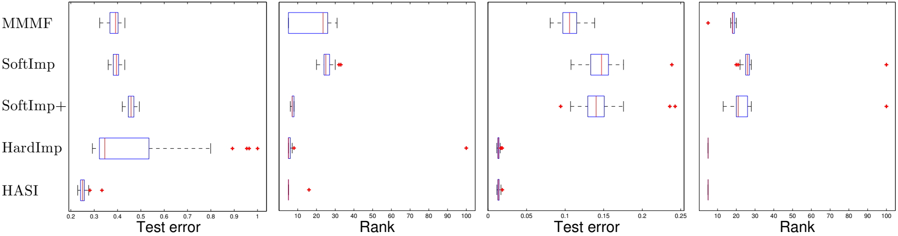
Figure : Test error w.r.t. the rank obtained by varying the value of the regularization parameter $\lambda$, averaged over 50 replications.

- Similar behavior is observed, with the HASI algorithm attaining a minimum at the true rank $q = 5$.

# Simulated data

## Incomplete case

We then remove 20% of the observed entries as a validation set to estimate the regularization parameters. We use the unobserved entries as a test set.




(a) SNR=1; 50% miss. (b) SNR=1; 50% miss. (c) SNR=10; 80% miss. (d) SNR=10; 80% miss.

Figure : Boxplots of the test error and ranks obtained over 50 replications.

- For 50% missing data, HASI is shown to outperform the other methods.
- For 80% missing data, HASI and Hard Impute provide the best performances.
- In both cases, it is able to recover very accurately the true rank of the matrix.

# Collaborative filtering examples (Jester)

## Procedure

- We randomly select two ratings per user as a test set, and two other ratings per user as a validation set to select the parameters $\lambda$ and $\beta$.
- The results are computed over four values $\beta = 1000, 100, 10, 1$.
- We compare the results of the different methods with the Normalized Mean Absolute Error (NMAE)

$$\text{NMAE} = \frac{\frac{1}{card(\Omega_{test})} \sum_{(i,j)\in\Omega_{test}} |X_{ij} - \widehat{Z}_{ij}|}{\max(X) - \min(X)}$$

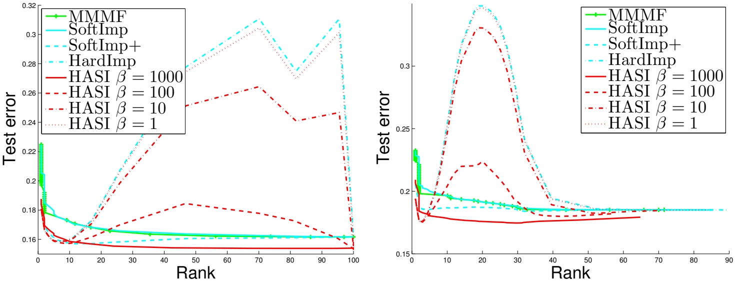
# Collaborative filtering examples (Jester)

Table : Results on the Jester datasets, averaged over 10 replications

| | Jester 1<br>24983 × 100<br>27.5% miss. | | Jester 2<br>23500 × 100<br>27.3% miss. | | Jester 3<br>24938 × 100<br>75.3% miss. | |
|---|---|---|---|---|---|---|
| Method | NMAE | Rank | NMAE | Rank | NMAE | Rank |
| MMMF | 0.161 | 95 | 0.162 | 96 | 0.183 | 58 |
| Soft Imp | 0.161 | 100 | 0.162 | 100 | 0.184 | 78 |
| Soft Imp+ | 0.169 | 14 | 0.171 | 11 | 0.184 | 33 |
| Hard Imp | 0.158 | 7 | 0.159 | 6 | 0.181 | 4 |
| HASI | **0.153** | 100 | **0.153** | 100 | **0.174** | 30 |

- The HASI algorithm provides very good performance on the different Jester datasets, with lower NMAE than the other methods.

# Collaborative filtering examples (Jester)




(a) Jester 1 (b) Jester 3

Figure : NMAE w.r.t. the rank obtained by varying the regularization parameter $\lambda$.

- Low values of $\beta$ exhibit a bimodal behavior with two modes at low rank and full rank.
- High value $\beta = 1000$ is unimodal and outperforms Soft-Impute at any particular rank.

# Collaborative filtering examples (MovieLens)

## Procedure

- We randomly select 20% of the entries as a test set, and the remaining entries are split between a training set (80%) and a validation set (20%).
- For all the methods, we stop the regularization path as soon as the estimated rank exceeds $r_{max} = 100$.
- For the larger MovieLens 1M dataset, the precision, maximum number of iterations and maximum rank are decreased to $\epsilon = 10^{-6}$, $t_{max} = 100$ and $r_{max} = 30$.

# Collaborative filtering examples (MovieLens)

Table : Results on the MovieLens datasets, averaged over 5 replications

| Method | MovieLens 100k<br>943 $\times$ 1682<br>93.7% miss.<br>NMAE | <br><br><br>Rank | MovieLens 1M<br>6040 $\times$ 3952<br>95.8% miss.<br>NMAE | <br><br><br>Rank |
|---|---|---|---|---|
| MMMF | 0.195 | 50 | **0.169** | 30 |
| Soft Imp | 0.197 | 156 | 0.176 | 30 |
| Soft Imp+ | 0.197 | 108 | 0.189 | 30 |
| Hard Imp | 0.190 | 7 | 0.175 | 8 |
| HASI | **0.187** | 35 | 0.172 | 27 |

- For the MovieLens 100k dataset, HASI provides better NMAE than the other methods with a low rank solution.
- For the MovieLens 1M dataset, MMMF provides the best NMAE at maximum rank. HASI provides the second best performances with a slightly lower rank.

# Summary

# Conclusion and perspectives

- Conclusion:
  - The proposed class of methods has shown to provide good results compared to several alternative low rank matrix completion methods.
  - It provides a bridge between nuclear norm and rank regularization algorithms.
  - Although the related optimization problem is not convex, experiments show that initializing the algorithm with the Soft-Impute algorithm of [Mazumder et al., 2010] provides very satisfactory results.
- Perspectives:
  - Investigate a fully Bayesian approach and derive a Gibbs sampler or variational algorithm to approximate the posterior distribution.
  - Application to larger Netflix dataset

# References


Cai, J., Candès, E., and Shen, Z. (2010).
A singular value thresholding algorithm for matrix completion.
*SIAM Journal on Optimization*, 20(4):1956–1982.

Candès, E. and Plan, Y. (2010).
Matrix completion with noise.
*Proceedings of the IEEE*, 98(6):925–936.

Candès, E., Wakin, M., and Boyd, S. (2008).
Enhancing sparsity by reweighted l1 minimization.
*Journal of Fourier Analysis and Applications*, 14(5):877–905.

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

Gaïffas, S. and Lecué, G. (2011).
Weighted algorithms for compressed sensing and matrix completion.
arXiv preprint arXiv:1107.1638.

Larsen, R. M. (2004).
Propack-software for large and sparse svd calculations.
*Available online. URL http://sun. stanford. edu/rmunk/PROPACK*.

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

Srebro, N., Rennie, J., and Jaakkola, T. (2005).
Maximum-Margin Matrix Factorization.
In *Advances in neural information processing systems*, volume 17, pages 1329–1336. MIT Press.

Todeschini, A., Caron, F., and Chavent, M. (2013).
Probabilistic low-rank matrix completion with adaptive spectral regularization algorithms.
In *Advances in Neural Information Processing Systems*, pages 845–853.