SCMF: Sparse Covariance Matrix Factorization for Collaborative Filtering

Jianping Shi[†] **Naiyan Wang**[‡] **Yang Xia**[†] **Dit-Yan Yeung**[‡] **Irwin King**[†] **Jiaya Jia**[†]
[†]Department of Computer Science and Engineering, The Chinese University of Hong Kong
[‡]Department of Computer Science and Engineering
Hong Kong University of Science and Technology
jpshi@cse.cuhk.edu.hk winsty@gmail.com yxia@cse.cuhk.edu.hk
dyyeung@cse.ust.hk {king,leojia}@cse.cuhk.edu.hk

Abstract

Matrix factorization (MF) is a popular collaborative filtering approach for recommender systems due to its simplicity and effectiveness. Existing MF methods either assume that all latent features are uncorrelated or assume that all are correlated. To address the important issue of what structure should be imposed on the features, we investigate the covariance matrix of the latent features learned from real data. Based on the findings, we propose an MF model with a sparse covariance prior which favors a sparse yet non-diagonal covariance matrix. Not only can this reflect the semantics more faithfully, but imposing sparsity can also have a side effect of preventing overfitting. Starting from a probabilistic generative model with a sparse covariance prior, we formulate the model inference problem as a *maximum a posteriori* (MAP) estimation problem. The optimization procedure makes use of stochastic gradient descent and majorization-minimization. For empirical validation, we conduct experiments using the MovieLens and Netflix datasets to compare the proposed method with two strong baselines which use different priors. Experimental results show that our sparse covariance prior can lead to performance improvement.

1 Introduction

Nowadays, recommender systems complement search engines for information filtering on the Internet. They provide personalized recommendation for items such as products or services by taking into consideration such sources of information as characteristics of the items and preferences of similar users. As two well-known examples, Amazon and Netflix use recommender systems to recommend to their users books and movies, respectively. For this approach to work effectively for personalized advertising which leads to commercial benefit, it is essential that user preference can be predicted as accurately as possible. This explains why research in recommender systems has drawn so much interest from both the academic and commercial worlds.

While recommender systems can use either the content-based or *collaborative filtering* (CF) approach (or a hybrid approach incorporating the two), the CF approach is considered particularly promising in this era of the social web (or called Web 2.0) in which social relations are often exploited extensively for a variety of applications. Unlike content-based filtering which defines user or item similarity based on content-based features extracted from user profiles or item descriptions, the CF approach predicts user preference by analyzing the previous rating data of many users (hence the qualifier 'collaborative').

Memory-based CF methods [Breese *et al.*, 1998; Linden *et al.*, 2003] directly compute user or item similarity using the rating data. These methods have a longer history and have been used by many commercial systems. On the other hand, model-based CF methods make use of machine learning and data mining models to explore patterns in the rating data. They include the clustering model [Xue *et al.*, 2005], aspect model [Hofmann, 2004], latent factor model [Canny, 2002], restricted Boltzmann machine [Salakhutdinov *et al.*, 2007], etc. Over the past few years, extensive empirical studies conducted on large-scale datasets show that model-based CF methods generally have higher accuracy than memory-based methods [Töscher *et al.*, 2009].

The most popular and successful model-based CF methods are based on low-rank *matrix factorization* (MF). The rationale behind the MF approach is that the observed rating data can be explained holistically by a small number of latent factors for both users and items, which, for instance, may be related to user groups and item topics. Given an incomplete and highly sparse rating matrix representing the observed rating data, a typical MF method factorizes the matrix into two low-rank matrices, one for latent user features and the other for latent item features. Multiplying the two matrices thus allows us to complete the original incomplete matrix by filling in the missing entries.

Among these MF methods, *probabilistic matrix factorization* (PMF) [Salakhutdinov and Mnih, 2008b] is arguably the most representative one due to its high efficiency and accuracy. Many other MF methods are variants of PMF. For example, [Hu *et al.*, 2008] utilized the rated-or-not indicator for implicit feedback to improve the results; [Koren, 2008] unified the latent factor MF model with the neighborhood model under the same framework; and [Rendle and Schmidt-Thieme, 2008] generalized the standard inner product operation for user features and item features to various kernel functions.

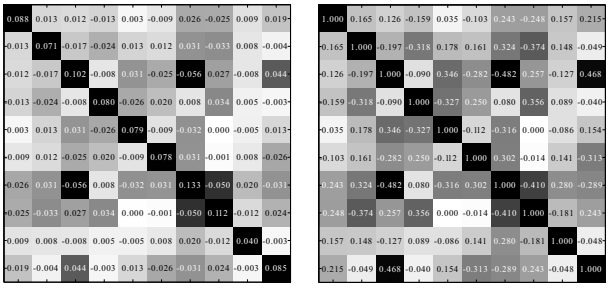


Figure 1: The left panel is the feature covariance matrix learned with 10 latent features. The right panel shows the corresponding correlation matrix.

Recently, incorporating social network information into MF models has emerged as a hot research direction [Ma *et al.*, 2008; 2009; 2011].

Since the rating matrix contains many missing entries, MF methods usually impose an ℓ_2 penalty on both the latent user features and latent item features to avoid overfitting. However, restricting all features to the same regularization level limits the flexibility of the model. One variant [Lakshminarayanan *et al.*, 2011] uses a more general diagonal covariance matrix whose diagonal entries do not have to be the same to give different regularization levels to different features, if we assume that the user and item features are generated from multivariate normal distributions. Nevertheless, it still assumes that the features are uncorrelated. An even more flexible variant uses a full covariance matrix, as in *Bayesian probabilistic matrix factorization* (BPMF) [Salakhutdinov and Mnih, 2008a]. Although BPMF gives good results, its high computational requirement, due to the use of sampling-based inference techniques, makes it challenging to be applied to very large datasets.

An open question is whether there is indeed the need for a full covariance matrix. To answer this question, let us first take a look at the covariance matrix for the user and item features learned by PMF. Figure 1 shows the covariance matrix and correlation matrix for a movie recommendation dataset (ML-10M with 80% training data). Not only are the diagonal entries in the covariance matrix significantly positive, some non-diagonal entries also differ from 0 significantly although such entries are sparse. Note that the latent features are generally found to be semantically meaningful. For example, the latent features may correspond to such topics as biography, comedy, history, and romance. If two features, e.g., those corresponding to biography and history, are highly correlated, we expect that the corresponding entries in the (symmetric) covariance matrix will also be significantly non-zero. Nevertheless, most other pairs of features are uncorrelated or at most weakly correlated. This explains why the majority of the non-diagonal entries are close to 0. Similar observations have also been noted in some other datasets. This finding inspires us to propose later in this paper a sparse covariance prior which favors a sparse yet non-diagonal covariance matrix for MF models.

Besides its role in reflecting the semantics more properly, we anticipate that imposing sparsity on the covariance matrix also has a side effect of preventing overfitting. When there exist outliers in the very sparse rating matrix, two otherwise uncorrelated features may exhibit high correlation incorrectly. Imposing sparsity on the covariance matrix will likely help to alleviate this problem.

Based on the inspiration above, we propose a probabilistic generative MF model with a novel sparse covariance prior. The same sparse covariance prior is imposed on both the user and item feature vectors. For model inference, we take the *maximum a posteriori* (MAP) approach for its efficiency and scalability. Our model is compared empirically with two baseline MF methods on some large datasets.

2 Brief Review of PMF

Since our model builds on PMF [Salakhutdinov and Mnih, 2008b], let us first give a quick review of the model formulation and learning algorithm of PMF.

For the sake of presentation, we assume that the recommender system is for movie recommendation. The same model may also be used for recommendation of other items. Let there be M movies and N users. The rating of movie j by user i is denoted by R_{ij}. Like many other MF methods, PMF uses two latent feature matrices $U \in \mathbb{R}^{D\times N}$ and $V \in \mathbb{R}^{D\times M}$, with column vectors U_i and V_j corresponding to the latent feature vectors of user i and movie j, respectively. It is assumed that the observed ratings are governed by a normal distribution to give the likelihood as follows:

$$p(R \mid U, V, \sigma^2) = \prod_{i=1}^{N}\prod_{j=1}^{M} \left[\mathcal{N}(R_{ij} \mid U_i^T V_j, \sigma^2)\right]^{Z_{ij}}, \quad (1)$$

where Z_{ij} is an indicator variable introduced to indicate whether R_{ij} is observed, and $\mathcal{N}(X \mid \mu, \sigma^2)$ means that X satisfies a normal distribution with mean μ and variance σ^2 (the same notation is also used for the multivariate normal distribution with mean vector μ and covariance matrix Σ). Zero-mean spherical normal distributions are imposed as priors on the column vectors of U and V independently:

$$\begin{aligned} p(U \mid \sigma_U^2) &= \prod_{p=1}^{N} \mathcal{N}(U_p \mid 0, \sigma_U^2 I) \\ p(V \mid \sigma_V^2) &= \prod_{p=1}^{M} \mathcal{N}(V_p \mid 0, \sigma_V^2 I), \end{aligned} \quad (2)$$

where I is an identity matrix and σ_U^2 and σ_V^2 are variance parameters. Using a MAP estimation approach for model inference, PMF solves a log-posterior maximization problem which is equivalent to the following minimization problem:

$$\min_{U,V} \sum_{i=1}^{N}\sum_{j=1}^{M} Z_{ij}(R_{ij} - U_i^T V_j)^2 + \lambda_U \sum_{i=1}^{N} \|U_i\|_2^2 + \lambda_V \sum_{i=1}^{M} \|V_i\|_2^2, \quad (3)$$

where $\lambda_U = \sigma^2/\sigma_U^2$ and $\lambda_V = \sigma^2/\sigma_V^2$ are regularization parameters. The graphical model of PMF is shown in plate notation in the left panel of Figure 2. To solve the optimization problem in Eq. (3) efficiently, PMF uses stochastic gradient descent to update U and V iteratively.

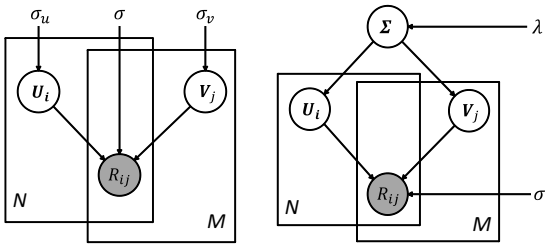


Figure 2: Graphical models in plate notation for PMF (left panel) and SCMF (right panel).

3 Sparse Covariance Matrix Factorization

We are now ready to present our MF model. We first present it as a probabilistic generative model with a sparse covariance prior. For this reason, we refer to it as *sparse covariance matrix factorization* (SCMF). As in PMF, model inference in SCMF is formulated as a MAP estimation problem which corresponds to an equivalent minimization problem. We then present an extension of the basic model as well as its complexity analysis.

3.1 Probabilistic Model

The likelihood term of SCMF is exactly the same as that of PMF as given in Eq. (1). The main difference lies in the prior distributions of the latent user and item feature vectors. Specifically, the priors of the feature vectors U_p and V_p are zero-mean multivariate normal distributions with a full covariance matrix:

$$\begin{aligned} p(U \mid \Sigma) &= \prod_{p=1}^{N} \mathcal{N}(U_p \mid 0, \Sigma) \\ p(V \mid \Sigma) &= \prod_{p=1}^{M} \mathcal{N}(V_p \mid 0, \Sigma). \end{aligned} \tag{4}$$

Unlike PMF, using a full covariance matrix allows the features to be correlated. Having the same covariance matrix Σ for both U and V in Eq. (4) provides a bridge to connect the two feature matrices.

As discussed above, it is desirable to impose sparsity on the non-diagonal entries of the covariance matrix both for reflecting the underlying semantics properly and for preventing overfitting. To achieve this, we impose a Laplace prior on every element of Σ:

$$p(\Sigma \mid \lambda) = \begin{cases} \prod_{i<j} \mathcal{L}(\Sigma_{ij} \mid \lambda), & \text{if } \Sigma \text{ is a p.d. matrix;} \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

where λ is a parameter that controls the degree of sparsity, p.d. stands for *positive definite*, which is a property required for all covariance matrices,[1] and $\mathcal{L}$ is the Laplace distribution defined as

$$\mathcal{L}(\Sigma_{ij} \mid \lambda) = \frac{\lambda}{2} \exp(-\lambda\,|\Sigma_{ij}|), \tag{6}$$

[1]Although covariance matrices may be positive semi-definite (p.s.d.), we impose the stricter p.d. requirement here because the probability density will be undefined for p.s.d. covariance matrices. Nevertheless, the p.d. requirement is easy to satisfy because the latent dimensionality is typically low.

where $|a|$ denotes the absolute value of the scalar a. The generative model of SCMF is shown in the right panel of Figure 2 to allow us to reveal the similarities and differences between it and PMF.

To understand how model inference is done, we first take a look at the joint posterior distribution of the latent variables and parameters U, V, Σ given the hyperparameters $\theta = (\sigma, \lambda)$:

$$\begin{aligned} & p(U, V, \Sigma \mid R, \theta) \propto\ p(U, V, \Sigma, R \mid \theta) \\ &= p(R \mid U, V, \sigma^2)\, p(U \mid \Sigma)\, p(V \mid \Sigma)\, p(\Sigma \mid \lambda) \\ &= \prod_{i=1}^{N} \prod_{j=1}^{M} \left[\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(R_{ij} - U_i^T V_j)^2}{2\sigma^2} \right) \right]^{Z_{ij}} \\ &\quad \times \prod_{p=1}^{N} \left[(2\pi)^{-\frac{D}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2} U_p^T \Sigma^{-1} U_p \right) \right] \\ &\quad \times \prod_{p=1}^{M} \left[(2\pi)^{-\frac{D}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2} V_p^T \Sigma^{-1} V_p \right) \right] \\ &\quad \times \prod_{i<j} \frac{\lambda}{2} \exp(-\lambda |\Sigma_{ij}|), \end{aligned} \tag{7}$$

subject to the requirement that the p.d. property is satisfied.

Maximizing the (log-)posterior in Eq. (7) is equivalent to the following minimization problem:

$$\begin{aligned} & \min_{U,V,\Sigma} f(U, V, \Sigma \mid R, \theta) \\ &= \frac{1}{2\sigma^2} \sum_{i=1}^{N} \sum_{j=1}^{M} Z_{ij} (R_{ij} - U_i^T V_j)^2 + \frac{N+M}{2} \log\det(\Sigma) \\ &\quad + \frac{1}{2} \mathrm{tr}\Big(\Sigma^{-1} \sum_{p=1}^{N} U_p U_p^T \Big) + \frac{1}{2} \mathrm{tr}\Big(\Sigma^{-1} \sum_{p=1}^{M} V_p V_p^T \Big) \\ &\quad + \frac{\lambda}{2} \|P \otimes \Sigma\|_1, \end{aligned} \tag{8}$$

where Σ should be a p.d. matrix, $\det(\cdot)$ denotes the determinant of a matrix, $\mathrm{tr}(\cdot)$ denotes the trace of a matrix, P is a matrix with zeros in the diagonal entries and ones in the non-diagonal entries to prevent the shrinking of diagonal entries in Σ, and $\otimes$ denotes element-wise multiplication.

3.2 Optimization Problem

To solve the optimization problem in Eq. (8), we now present a stochastic gradient descent algorithm which alternates the update of U, V and Σ.

Optimizing with respect to U and V

To optimize with respect to U while keeping V and Σ fixed, the problem in Eq. (8) becomes

$$\begin{aligned} \min_{U} f(U \mid R, V, \Sigma, \theta) = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \sum_{j=1}^{M} Z_{ij} (R_{ij} - U_i^T V_j)^2 \\ + \frac{1}{2} \mathrm{tr}\Big(\Sigma^{-1} \sum_{p=1}^{N} U_p U_p^T \Big). \end{aligned} \tag{9}$$

For each observed R_{ij}, the gradient with respect to U_i is

$$\frac{\partial f}{\partial U_i} = -\frac{1}{\sigma^2}(R_{ij} - U_i^T V_j)V_j + \Sigma^{-1}U_i. \quad (10)$$

Similarly, the gradient with respect to V_j is

$$\frac{\partial f}{\partial V_j} = -\frac{1}{\sigma^2}(R_{ij} - U_i^T V_j)U_i + \Sigma^{-1}V_j. \quad (11)$$

With these gradients, we can use stochastic gradient descent to update U_i and V_j iteratively.

Optimizing with respect to Σ

To optimize with respect to Σ while keeping U and V fixed, the problem in Eq. (8) becomes

$$\min_{\Sigma} f(\Sigma \mid R, U, V, \theta) = \log\det(\Sigma) + \mathrm{tr}(\Sigma^{-1}S) + \frac{\lambda}{M+N}\|P \otimes \Sigma\|_1, \quad (12)$$

where

$$S = \frac{1}{M+N}\left(\sum_{p=1}^{N} U_p U_p^T + \sum_{p=1}^{M} V_p V_p^T\right) \quad (13)$$

is the sample covariance matrix for the column vectors of U and V. We note that the p.d. property of Σ is preserved since the log-determinant term acts as a barrier function [Boyd and Vandenberghe, 2004].

While it appears challenging to solve the optimization problem in Eq. (12) since the objective function is both non-convex and non-smooth, it is in fact well structured because $\mathrm{tr}(\Sigma^{-1}S) + \alpha\|P \otimes \Sigma\|_1$ is convex with respect to Σ while $\log\det(\Sigma)$ is concave. Problems of this type have been widely studied and can be solved using the *majorization-minimization* (MM) approach [Hunter and Li, 2005].

Theorem 1. *The objective function value of Eq. (12) is non-increasing under the following update rule:*

$$\Sigma_{t+1} \leftarrow \mathcal{T}\Big\{\Sigma_t - \alpha\big(\Sigma_t^{-1} - \Sigma_t^{-1} S\, \Sigma_t^{-1}\big), \frac{\alpha\lambda}{M+N}P\Big\}, \quad (14)$$

where α is a step size and $\mathcal{T}$ is an element-wise soft-thresholding operator defined by $\mathcal{T}_{ij}\{X, Y\} = \max(0, |X_{ij}| - Y_{ij})$. The objective function value remains unchanged under the update rule if and only if Σ is at a stationary point.

Theorem 1 guarantees the convergence of updating via Eq. (14). In what follows, we will give a detailed proof of Theorem 1. We start by defining an auxiliary function in a way similar to that in [Seung and Lee, 2001].

Definition 1. *$g(x \mid x')$ is an auxiliary function for $f(x)$ if the following conditions are satisfied:*

$$g(x \mid x') \geq f(x), \quad g(x \mid x) = f(x). \quad (15)$$

Lemma 2. *If $g(x \mid x_t)$ is an auxiliary function, then $f(x)$ is non-increasing under the update rule*

$$x_{t+1} = \arg\min_x g(x \mid x_t). \quad (16)$$

Proof. $f(x_{t+1}) \leq g(x_{t+1} \mid x_t) \leq g(x_t \mid x_t) = f(x_t)$. □

Therefore, the equality $f(x_{t+1}) = f(x_t)$ holds only when x_t is a local minimum of $g(x \mid x_t)$. By iteratively updating via Eq. (14), we can reach a local minimum of Eq. (12). We will show in the following that Eq. (14) decreases the value of the auxiliary function of Eq. (12).

Lemma 3. *The following is an auxiliary function of Eq. (12):*

$$g(\Sigma_{t+1} \mid \Sigma_t) = \log\det(\Sigma_t) + \mathrm{tr}\big((\Sigma_t)^{-1}(\Sigma_{t+1} - \Sigma_t)\big) + \mathrm{tr}\big(\Sigma_{t+1}^{-1}S\big) + \frac{\lambda}{M+N}\|P \otimes \Sigma_{t+1}\|_1. \quad (17)$$

Proof. It is trivial to prove directly from the definition that $g(x \mid x) = f(x)$. To prove that $g(x \mid x_t) \geq f(x)$ with Eq. (17) and Eq. (12), we note that

$$\log\det(\Sigma_t) + \mathrm{tr}\big((\Sigma_t)^{-1}(\Sigma_{t+1} - \Sigma_t)\big) \geq \log\det(\Sigma_{t+1}). \quad (18)$$

This is true since $\log\det(\Sigma_{t+1})$ is concave with respect to Σ_{t+1} and $\log\det(\Sigma_t) + \mathrm{tr}\big((\Sigma_t)^{-1}(\Sigma_{t+1} - \Sigma_t)\big)$ is the tangent of $\log\det(\Sigma_{t+1})$ at the point Σ_t. □

Lemma 4. *The update rule in Eq. (14) can guarantee a non-increasing value for the auxiliary function in Eq. (17).*

Proof. Eq. (17) consists of a differentiable convex component and a non-differentiable component with an ℓ_1 penalty. Thus, we can employ a generalized gradient descent algorithm [Beck and Teboulle, 2009], which is a natural extension of gradient descent to non-differentiable objective functions. The objective function in Eq. (17) is then changed to

$$\Sigma = \arg\min_{\Omega}\Big\{\frac{1}{2\alpha}\Big\|\Omega - \Sigma_{t+1} + \alpha\big(\Sigma_t^{-1} - \Sigma_{t+1}^{-1} S\, \Sigma_{t+1}^{-1}\big)\Big\|_F^2 + \frac{\lambda}{M+N}\|P \otimes \Omega\|_1\Big\}. \quad (19)$$

Solving Eq. (19) is equivalent to updating with Eq. (14). □

Proof of Theorem 1. Since Eq. (17) is an auxiliary function, $f(\Sigma)$ is non-increasing, according to Lemma 2, when the function value of Eq. (17) decreases. It follows from Lemma 4 that Eq. (14) minimizes the auxiliary function. This completes the proof.

3.3 Incorporating User Bias and Item Bias

Previous study [Koren, 2008] showed that incorporating user bias and item bias terms can lead to further improvement because it can allow for the intrinsic difference between users and the intrinsic difference between items to be represented. Here we present an extension of our basic model to include such terms.

Specifically, the likelihood in Eq. (1) is modified to

$$p(R \mid U, V, \sigma) = \prod_{i=1}^{N}\prod_{j=1}^{M}\big[\mathcal{N}(R_{ij} \mid U_i^T V_j + b_i^u + b_j^v, \sigma^2)\big]^{Z_{ij}}, \quad (20)$$

where b_i^u, the i^{th} element in a vector $b^u \in \mathbb{R}^{N\times 1}$, represents the bias for user i, and, similarly, b_j^v is the bias for item j.

The bias vectors b^u and b^v also follow a zero-mean normal distribution, namely,

$$p(b^u \mid \sigma_b^2) = \prod_{i=1}^{N} \mathcal{N}(b_i^u \mid 0, \sigma_b^2)$$
$$p(b^v \mid \sigma_b^2) = \prod_{i=1}^{M} \mathcal{N}(b_i^v \mid 0, \sigma_b^2). \quad (21)$$

Given the probability distribution, the optimization procedure for the extended model can be obtained easily by extending from that in Section 3.2.

3.4 Discussion and Analysis

Positive definite property

The covariance matrix Σ, initialized to the identity matrix I, remains symmetric and p.d. throughout the optimization procedure. For the symmetric property, it is easy to see that every update via Eq. (14) does not change this property. For the p.d. property $\Sigma \succ 0$, it suffices to ensure that $\Sigma \succeq \delta I$ for some $\delta > 0$. Not only can this enhance numerical stability but it can also avoid penalizing some features excessively. Every time when Σ is updated, its eigenvalues are checked. In case the condition $\Sigma \succeq \delta I$ is violated as a result of the update, those (possibly negative) eigenvalues smaller than δ will be rounded up to δ. This corresponds to projecting the solution to the convex set $\mathbb{C} = \{\Sigma \mid \Sigma \succeq \delta I\}$ in order to preserve the p.d. property.

Convergence

Our optimization procedure involves three subproblems in each iteration corresponding to optimization with respect to U, V and Σ. Because the subproblems for U and V are convex, convergence to the optimal solutions is guaranteed. From Theorem 1, the subproblem for Σ, though not convex, is guaranteed to decrease the objective function value in each iteration. Given that the objective function in Eq. (8) is bounded below, it always converges to a local minimum.

Complexity analysis

Algorithm 1 summarizes the optimization procedure for SCMF, where β denotes the step size parameter for the gradient descent update of U and V.

The algorithm has a time complexity of $\mathcal{O}((C + M + N)D^2)$ for each epoch, where C is the number of observed ratings. The complexity is linear to the numbers of users, items, and ratings, but quadratic to the latent feature dimensionality. However, in practice D is very small with $D \ll M$ and $D \ll N$ and hence the algorithm scales linearly with the data size. This gives the algorithm potential to be applied to very large datasets. How to select D is a commonly encountered problem for MF-based methods. Usually it has positive correlation with the sample size and desired accuracy and has negative correlation with computational efficiency.

The actual performance of our method compared to PMF depends on the batch size for updating Σ. In practice, the time requirement of our method is about 1.2-1.5 times that of the baseline PMF method.

Algorithm 1 Optimization procedure for SCMF

input: rating matrix R; hyperparameters α, β, θ;
repeat
  for each observed R_{ij} **do**
    update U_i via $U_i = U_i - \beta \frac{\partial f}{\partial U_i}$;
    update V_j via $V_j = V_j - \beta \frac{\partial f}{\partial V_j}$;
    update b_i^u and b_j^v;
  end for
  update Σ via Eq. (14) until convergence;
until validation error stops decreasing;
return U and V.

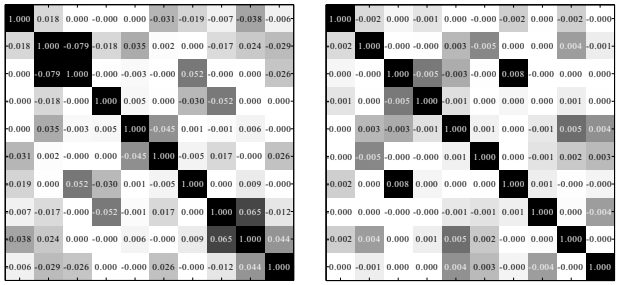


Figure 3: Correlation matrices derived from the covariance matrices learned from the ML-1M (left panel) and ML-10M (right panel) datasets.

4 Experiments

In this section, we will shift our focus to the empirical aspect of the paper. We first conduct an empirical analysis of the covariance sparsity property of SCMF. Then, we compare it with baseline methods PMF [Salakhutdinov and Mnih, 2008b] and *biased matrix factorization* (biased MF) [Koren, 2008] using the MovieLens[2] and Netflix[3] datasets.

4.1 Experimental Settings

There are three MovieLens datasets: ML-100K with 100K ratings, ML-1M with 1M ratings, and ML-10M with 10M ratings. The Netflix dataset is even larger with about 100M ratings. Since our focus in this paper is on CF using rating data, other sources of information such as implicit feedback and tagging information are discarded.

We use *root mean squared error* (RMSE) as the evaluation measure. It is defined as

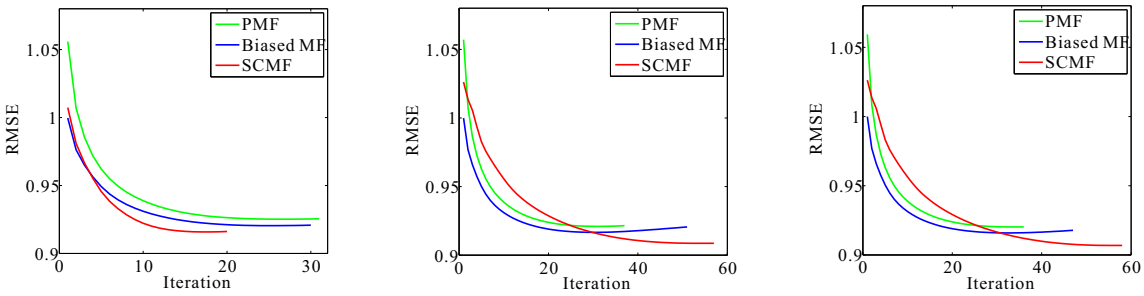
$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i,j} Z_{ij}^{\mathcal{T}} (R_{ij} - \hat{R}_{ij})^2}, \quad (22)$$

where N is the total number of ratings in the test set, R_{ij} is the ground-truth rating of user i for item j, $\hat{R}_{ij}$ denotes the corresponding predicted rating, and $Z_{ij}^{\mathcal{T}}$ is an indicator variable used to select the R_{ij} defined in the test set. For all three methods, we set the regularization parameters via 5-fold cross validation.

[2] http://www.grouplens.org/node/73
[3] http://www.netflixprize.com/

Training data		ML-100K		ML-1M		ML-10M	
		$D = 10$	$D = 20$	$D = 10$	$D = 20$	$D = 10$	$D = 20$
99%	PMF	0.9090±0.0178	0.9065±0.0168	0.8424±0.0071	0.8388±0.0059	0.8038±0.0023	0.7941±0.0021
	Biased MF	0.8953±0.0189	0.8923±0.0150	0.8408±0.0070	0.8367±0.0067	0.8010±0.0027	0.7932±0.0024
	SCMF	**0.8891±0.0146**	**0.8896±0.0198**	**0.8364±0.0065**	**0.8323±0.0065**	**0.7973±0.0026**	**0.7874±0.0027**
80%	PMF	0.9286±0.0027	0.9225±0.0026	0.8559±0.0022	0.8512±0.0017	0.8087±0.0004	0.8008±0.0006
	Biased MF	0.9135±0.0039	0.9087±0.0030	0.8531±0.0019	0.8493±0.0020	0.8057±0.0008	0.7991±0.0007
	SCMF	**0.9092±0.0033**	**0.9068±0.0036**	**0.8496±0.0019**	**0.8465±0.0018**	**0.8011±0.0001**	**0.7944±0.0006**
50%	PMF	0.9595±0.0032	0.9539±0.0025	0.8790±0.0009	0.8745±0.0011	0.8236±0.0006	0.8189±0.0003
	Biased MF	0.9388±0.0029	0.9337±0.0020	0.8766±0.0015	0.8722±0.0012	0.8201±0.0002	0.8165±0.0003
	SCMF	**0.9334±0.0025**	**0.9331±0.0021**	**0.8707±0.0013**	**0.8678±0.0007**	**0.8141±0.0004**	**0.8107±0.0001**

Table 1: Results of comparative study on the MovieLens datasets under two different feature dimensionality values and three different percentages of the training data.



(a). $D = 20$ (b). $D = 50$ (c). $D = 80$

Figure 4: Results of comparative study on the Netflix dataset under three different feature dimensionality values.

4.2 Empirical Analysis of Covariance Sparsity

We use the MovieLens ML-1M and ML-10M datasets and train SCMF on 80% of their data. The latent feature dimensionality is set to 10. Figure 3 depicts the correlation matrices derived from the covariance matrices learned. Among the off-diagonal entries, only a very small fraction show relatively high correlation. Compared with the correlation matrix learned using PMF (Figure 1), incorporating a sparse covariance prior increases the degree of sparsity of the covariance matrix learned.

4.3 Quantitative Results of Comparative Study

For the three MovieLens data sets, we try different settings by using different percentages (99%, 80%, 50%) of the data for model training. The remaining data are used for testing to obtain the RMSE results. The training data are randomly selected from each dataset. This process is repeated five times for each percentage setting and the RMSE reported is the average over these runs.

Table 1 summarizes the results. In practice, SCMF ends within 200 iterations. We can see that SCMF consistently outperforms both PMF and biased MF on all settings of all three datasets. As the main difference between SCMF and the other two methods is in the structure of the covariance matrix of the latent features, this shows that imposing a sparse covariance prior does bring about performance gain as anticipated.

For the Netflix dataset, we use the *training_set* data for training and *probe* data for testing. Since the Netflix dataset is even much larger than the largest MovieLens dataset, we set the latent feature dimensionality to larger values, $D = 20, 50, 80$, following the settings in [Gantner *et al.*, 2011]. The results are shown as convergence curves in Figure 4. Although SCMF converges a bit more slowly than the other two methods when D is larger, it can always reach a lower RMSE and hence the model learned has higher prediction accuracy.

5 Conclusion and Future Work

In this paper, we have proposed an MF model with a sparse covariance prior for CF applications. The model does not take either the extreme of assuming that all latent features are uncorrelated or the other extreme that all are correlated. Instead, it allows a small number of correlated feature pairs. Empirical studies show that imposing this particular prior can lead to performance improvement.

It should be noted that incorporating a sparse covariance prior on the latent features is not limited to CF applications that use rating data only. Other sources of information such as content-based features extracted from user profiles or item descriptions may also be incorporated into the MF model. This is one direction in which the current research will be extended. Moreover, the current MAP estimation approach, as a point estimation approach, may not be robust enough for very sparse observed data. A full Bayesian approach to MF [Salakhutdinov and Mnih, 2008a] provides a potential solution to this problem. However, it is very challenging to

develop efficient and highly scalable full Bayesian models especially when non-conjugate priors are used. As another possible direction in which this research can be extended, some special structure of the problem will be exploited to devise a full Bayesian model with a sparse covariance prior which is a non-conjugate prior. Last but not least, we will apply SCMF to extremely large datasets by exploiting parallel and distributed computing platforms.

Acknowledgments

This research has been supported by General Research Fund No. 621310, No. 413110 and No. 413212 from the Research Grants Council of Hong Kong.

References

[Beck and Teboulle, 2009] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[Boyd and Vandenberghe, 2004] S. Boyd and L. Vandenberghe. *Convex Optimization*. 2004.

[Breese *et al.*, 1998] J.S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[Canny, 2002] J. Canny. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 25th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 238–245, 2002.

[Gantner *et al.*, 2011] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. MyMediaLite: A free recommender system library. In *Proceedings of the 5th ACM Conference on Recommender Systems*, pages 305–308, 2011.

[Hofmann, 2004] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems*, 22(1):89–115, 2004.

[Hu *et al.*, 2008] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 263–272, 2008.

[Hunter and Li, 2005] D.R. Hunter and R. Li. Variable selection using MM algorithms. *Annals of Statistics*, 33(4):1617, 2005.

[Koren, 2008] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 426–434, 2008.

[Lakshminarayanan *et al.*, 2011] Balaji Lakshminarayanan, Guillaume Bouchard, and Cedric Archambeau. Robust bayesian matrix factorisation. In *Proceedings of the 14th International Conference on Articial Intelligence and Statistics (AISTATS)*, 2011.

[Linden *et al.*, 2003] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[Ma *et al.*, 2008] H. Ma, H. Yang, M.R. Lyu, and I. King. SoRec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 931–940, 2008.

[Ma *et al.*, 2009] H. Ma, M.R. Lyu, and I. King. Learning to recommend with trust and distrust relationships. In *Proceedings of the 3rd ACM Conference on Recommender Systems*, pages 189–196, 2009.

[Ma *et al.*, 2011] H. Ma, D. Zhou, C. Liu, M.R. Lyu, and I. King. Recommender systems with social regularization. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, pages 287–296, 2011.

[Rendle and Schmidt-Thieme, 2008] S. Rendle and L. Schmidt-Thieme. Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In *Proceedings of the 2nd ACM Conference on Recommender Systems*, pages 251–258, 2008.

[Salakhutdinov and Mnih, 2008a] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th International Conference on Machine Learning*, pages 880–887, 2008.

[Salakhutdinov and Mnih, 2008b] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems 20*, pages 1257–1264, 2008.

[Salakhutdinov *et al.*, 2007] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning*, pages 791–798, 2007.

[Seung and Lee, 2001] D. Seung and L. Lee. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, pages 556–562, 2001.

[Töscher *et al.*, 2009] A. Töscher, M. Jahrer, and R.M. Bell. The BigChaos solution to the Netflix grand prize. *Netflix Prize Documentation*, 2009.

[Xue *et al.*, 2005] G.R. Xue, C. Lin, Q. Yang, W.S. Xi, H.J. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of the 28th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 114–121, 2005.