Collaborative Rating Allocation

Yali Du[†], **Chang Xu**[‡], **Dacheng Tao**[‡]
[†]Center for Artificial Intelligence, FEIT, University of Technology Sydney
[‡]UBTech Sydney AI Institute, The School of IT, FEIT, The University of Sydney
yali.du@student.uts.edu.au, {c.xu, dacheng.tao}@sydney.edu.au

Abstract

This paper studies the collaborative rating allocation problem, in which each user has limited ratings on all items. These users are termed "energy limited". Different from existing methods which treat each rating independently, we investigate the geometric properties of a user's rating vector, and design a matrix completion method on the simplex. In this method, a user's rating vector is estimated by the combination of user profiles as basis points on the simplex. Instead of using Euclidean metric, a non-linear pull-back distance measurement from the sphere is adopted since it can depict the geometric constraints on each user's rating vector. The resulting objective function is then efficiently optimized by a Riemannian conjugate gradient method on the simplex. Experiments on real-world data sets demonstrate our model's competitiveness versus other collaborative rating prediction methods.

1 Introduction

Collaborative prediction aims to predict users' preference based on their previous choices and connections with other users. It has wide applications in industry; for example, in an Amazon book recommendation system, if the system inputs are user ratings on previously read books, predictions of a user's preference of new and unread books are accomplished using patterns discovered from the partially observed rating matrix.

Most collaborative filtering works under the underlying assumption that if user A has the same interest in an item as user B , then A is more likely to have more similar opinions on other items with B than with an arbitrarily chosen user [Breese *et al.*, 1998]. Memory-based methods refer to algorithms that explore similarities among users or items. Of these, neighborhood-based methods are most popular [Lee *et al.*, 2012] and can be divided into two categories: user-based and item-based methods. User-based methods [Breese *et al.*, 1998] identify similar users to the queried user, and the desired rating is then estimated by averaging the ratings of these similar users. Conversely, item-based methods [Sarwar *et al.*, 2001] discover similar items to the queried item and evaluate the desired rating by the average of ratings of these similar items. Different similarity measurements are employed to compute the averaging weights including Pearson correlations [Herlocker *et al.*, 1999], vector cosine [Sarwar *et al.*, 2001], and mean-squared-difference [Papagelis *et al.*, 2005]. Other researchers employ a model-based approach by constructing a parametric model and fit parameters to training data to make predictions of unseen data [Lee *et al.*, 2012]. In addition to these traditional methods, a fairly large body of work describes collaborative prediction from the perspective of matrix factorization. A rating matrix is factorized into a product of two low rank matrices (user profiles and item profiles), which are used to estimate missing entries. Different factorization techniques can be applied such as non-negative matrix factorization [Lee and Seung, 1999], probabilistic matrix factorization [Salakhutdinov and Mnih, 2011; Lawrence and Urtasun, 2009; Fang *et al.*, 2014], maximum margin matrix factorization [Rennie and Srebro, 2005], and non-linear principal component analysis [Yu *et al.*, 2009].

Existing collaborative prediction algorithms are effective and perform well in diverse applications including book, movie, and music recommendation systems. However, their success is mainly seen in isolated situations where items are rated independently by a user without consideration on their connections. In practice, a user may be required to allocate limited credits to a set of items. In other words, the limited credits owned by the user are split among all items. For example, in cumulative voting, a voter is given an explicit number of points (or votes) to distribute among candidates, while in family financing, a user distributes his (or her) limited money among different financial products based on his (or her) preference or confidence. We refer to this type of user as "energy limited". Thus, existing collaborative prediction algorithms are not applicable to collaborative rating allocation problem, which additionally involves constraints among ratings.

For an energy-limited user, N different items will occupy parts of the energy and together they will use up all the user's energy. The deep-seated truth in this scenario is that the elements in the N-dimensional vector of ratings are not independent. With the observation of the limited budget of a user's ratings, we are motivated to consider users' rating vectors lying on the simplex, i.e., the multinomial manifold [Sun *et al.*, 2016].

In this paper, we investigate the method that emphasizes the connections among ratings of a user. We seat each user's rating vector on the simplex and develop a simplex constrained matrix completion algorithm-referred to as MC-S. In this way, the relations among ratings are intrinsically implied in the model. Rather than using the intuitive distance metric in Euclidean space, a non-linear pull-back metric from sphere is employed to investigate the geometric properties of the data on the simplex. The resulting objective function can be efficiently optimized using a Riemannian conjugate gradient method. Experimental results on real world datasets demonstrate the effectiveness of the proposed method on excavating data's geometric properties.

2 Related Work

We first provide a brief review of existing collaborative filtering algorithms, which can be divided into three main categories: memory-based methods, model-based methods, and a new class of matrix factorization methods.

Memory-based methods [Linden *et al.*, 2003] predict new entries according to users who are similar to the queried user. These methods memorize the rating matrix and make recommendations by calculating an aggregate of some similar users ratings of the item [Lee *et al.*, 2012]. Of these memory-based methods, neighborhood-based methods achieve good performance by identifying similar users to the queried user or by identifying similar items to the predicted item. The main similarity metrics include cosine similarity [Sarwar *et al.*, 2001] and Pearson correlation [Herlocker *et al.*, 1999]. However, they have some shortcomings; when there are only a few observed entries, the calculated similarity might be inaccurate, compromising algorithm effectiveness.

Model-based collaborative filtering algorithms [Breese *et al.*, 1998] use the input rating data to learn or estimate a parametric model and then make predictions for unseen entries. Data mining and machine learning algorithms are applied to find the input rating pattern based on the data's statistical properties. Bayesian networks [Su and Khoshgoftaar, 2006] train a Bayesian classifier based on the given training ratings and make predictions on new items [Miyahara and Pazzani, 2002; Song and Zhu, 2016], while clustering-based collaborative models [Chee *et al.*, 2001; Wang *et al.*, 2015] partition either the set of items or the set of users [OConnor and Herlocker, 1999] based on the rating data as the basis for future predictions. Markov decision process-based collaborative filtering systems [Shani *et al.*, 2002; Su and Khoshgoftaar, 2009] provide better performance than those that do not deploy the system. In addition to the algorithms discussed above, some hybrid methods have been developed that combine the above collaborative filtering algorithms [Ghazanfar *et al.*, 2012]. By combining the strengths of memory and model-based collaborative algorithms, they can result in increased computational complexity.

Recently, plentiful collaborative filtering algorithms are based on low-rank matrix factorizations and they are even further generalized to multi-label and multi-class learning problems [Xu *et al.*, 2016]. In this case, the rating matrix is factorized into a product of two low-rank matrices, thus filling the missing entries. Apart from the basic non-negative matrix factorization [Lee and Seung, 1999], variations include

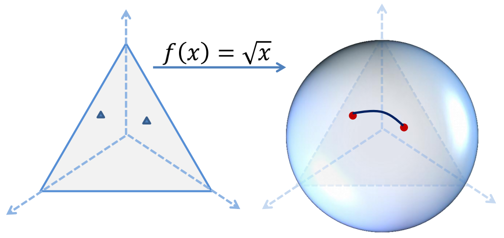


Figure 1: An illustration of the distance on the simplex.

manhattan matrix factorization [Liu and Tao, 2016], probabilistic matrix factorization [Salakhutdinov and Mnih, 2011], maximum margin matrix factorization [Srebro *et al.*, 2004; Rennie and Srebro, 2005], and non-linear principal component analysis [Yu *et al.*, 2009].

3 Problem Formulation

Take m users and their ratings on $n+1$ items as a matrix $Y \in R^{(n+1)\times m}$, where the j-th column represents user j's ratings on $n+1$ items and the i-th row corresponds to the ratings on item i from m users with $i \leq n+1$ and $j \leq m$. Indices of observed entries constitute a set Ω. Valid ratings usually settle within a range $[0, k]$. In the rating allocation problem, the budget of a user's ratings on all items is fixed. Hence, the goal of collaborative rating allocation can be interpreted as filling an incomplete matrix Y with a matrix X whose columns $X_{\cdot j}$ lie on the simplex. An $n-$simplex $\mathbb{P}_n$ is defined as

$$\mathbb{P}_n = \{x \in \mathbb{R}^{n+1} | \forall i, x_i \geq 0 \text{ and} \sum_{i=1}^{n+1} x_i = 1\}.$$

The simplex $\mathbb{P}_n$ is described as a subset of $\mathbb{R}^{n+1}$, but it is actually an $n-$dimensional manifold. In fact, any point on the simplex is a parameter vector x, which itself happens to be a probability vector.

The incomplete matrix $Y_{(n+1)\times m}$ can be recovered through a matrix $X_{(n+1)\times m}$ with a lower latent dimension

$$\begin{aligned} \min_{X} \quad & \sum_{j=1}^{m} \textbf{dist}(X_{\cdot j}, Y_{\cdot j}) \\ \text{s.t.} \quad & P_\Omega(X - Y) = 0, X_{\cdot j} \in \mathbb{P}_n, \end{aligned} \tag{1}$$

where $\textbf{dist}(\cdot,\cdot)$ is a distance measurement and P_Ω is the sampling operator defined as:

$$[P_\Omega(Y)] = \begin{cases} Y_{ij} & \text{if } (i,j) \in \Omega, \\ 0 & \text{else.} \end{cases}$$

Considering a maximal latent dimension of $X \in R^{(n+1)\times m}$ as r, we can approximate X by a matrix product $X = UV$ where $U \in R^{(n+1)\times r}$ acts as the basis on the simplex and $V \in R^{r\times m}$ is the corresponding coefficient matrix. We give the constraints on U as

$$U_{\cdot k}^T \mathbf{1}_{n+1} = 1, \forall k,$$

where $\mathbf{1}_{n+1}$ is an (n+1)-dimensional column matrix of all 1s.

Since each individual column of $X_{(n+1)\times m}$ lies on the simplex $\mathbb{P}_n$ and $X_{\cdot j} = UV_{\cdot j}$, we get

$$(UV_{\cdot j})^T \mathbf{1}_{n+1} = V_{\cdot j}^T U^T \mathbf{1}_{n+1} = V_{\cdot j}^T \mathbf{1}_r = 1,$$

which implies $V_{\cdot j}^T \mathbf{1}_r = 1, \forall j$. Hence, problem (1) can be reformulated as

$$\begin{aligned} \min_{X,U,V} \quad & \sum_{i=1}^{m} \mathbf{dist}_{\mathcal{M}}(X_{\cdot j}, UV_{\cdot j}) \\ \text{s.t.} \quad & P_\Omega(X - Y) = 0, X_{\cdot j}^T \mathbf{1}_{n+1} = 1, \\ & U_{\cdot k}^T \mathbf{1}_{n+1} = 1, V_{\cdot j}^T \mathbf{1}_r = 1, \end{aligned} \tag{2}$$

where $\mathbf{dist}_{\mathcal{M}}(\cdot,\cdot)$ refers to the distance on the simplex. Since an $n-$simplex $\mathbb{P}_n$ is a subset of R^{n+1}, Euclidean metric is an intuitive choice. However, it has been shown to be inefficient [Le and Cuturi, 2015], since it cannot depict the geometric constraints. Therefore, we propose using a pull-back metric from the sphere to specify the distance on the simplex.

In information geometry, Fisher information metric is a particular Riemannian metric defined on the simplex. Given the diffeomorphism mapping $H : \mathbb{P}_n \longmapsto \mathbb{S}_n^+$, we consider the pull-back metric from the positive orthant from the sphere $\mathbb{S}_n^+$, where

$$H(x) = \sqrt{x},$$

$$\mathbb{S}_n^+ = \{x \in \mathbb{R}^{n+1} | \forall i, x_i \geq 0 \text{ and} \sum_{i=1}^{n+1} x_i^2 = 1\}.$$

The pull-back Fisher information metric (inner product) on the simplex $\mathbb{P}_n$ takes the form

$$g_x(\xi_x, \eta_x) = \sum_{i=1}^{n+1} (\xi_x)_i (\eta_x)_i / x_i,$$

which derives the geodesic distance on the simplex

$$d(x,z) = \cos^{-1}(\langle H(x), H(z) \rangle) = \cos^{-1}(\sum_{i=1}^{n+1} \sqrt{x_i z_i}).$$

Here $\langle \cdot,\cdot \rangle$ is the Euclidean inner product (more details about simplex can be found in [Sun *et al.*, 2016]).

Based on the aforementioned knowledge on the simplex, we can proceed to specialize Eq. (2) to the simplex $\mathbb{P}_n$ to achieve the following model:

$$\begin{aligned} \min_{X,U,V} \quad & \sum_{j=1}^{m} \cos^{-1}(\sum_{i=1}^{n+1} \sqrt{X_{ij}} \cdot \sqrt{U_{i\cdot} V_{\cdot j}}) \\ \text{s.t.} \quad & P_\Omega(X - Y) = 0, X_{\cdot j} \in \mathbb{P}_n, \forall j, \\ & U_{\cdot k} \in \mathbb{P}_n, \forall k, V_{\cdot j} \in \mathbb{P}_{r-1}, \forall j. \end{aligned} \tag{3}$$

By solving the above objective function, we expect to obtain user profiles U and item profiles V, which then produce X to approximate unseen entries in the partially observed Y. In contrast to existing matrix completion methods, the proposed model thoroughly investigates the geometric properties of the data, and thus it can lead to an optimal solution for the challenging collaborative rating allocation problem.

4 Optimization on the Manifold

Optimization of the objective Eq. (3) is split into two subproblems: (i) optimize U and V while X is fixed; and (ii) optimize X while U and V are fixed. This method refers to the alternating direction method (see [Parikh and Boyd, 2014] for further details). Since X can be updated by $X = UV + P_\Omega(Y - UV)$, we only focus on the optimization of (i), which leads to the following subproblem

$$\begin{aligned} \min_{U,V} \quad & F(U,V) = \sum_{j=1}^{m} \cos^{-1}(\sum_{i=1}^{n+1} \sqrt{X_{ij}} \cdot \sqrt{U_{i\cdot} V_{\cdot j}}) \\ \text{s.t.} \quad & U_{\cdot k} \in \mathbb{P}_n, \forall k, V_{\cdot j} \in \mathbb{P}_{r-1}, \forall j. \end{aligned} \tag{4}$$

Eq. (4) is a non-convex function with regard to U and V, so we proceed to update U and V alternately. The objective function Eq. (4) can be decomposed over $U_{i\cdot}$ row-wise and $V_{\cdot j}$ column-wise. Taking derivatives to $U_{i\cdot}$ and $V_{\cdot j}$, we get

$$\frac{\partial F}{\partial U_{i\cdot}} = -\frac{1}{2} \sum_{j=1}^{m} \frac{\sqrt{X_{ij}} / \sqrt{U_{i\cdot} V_{\cdot j}}}{\sqrt{1 - S_j^2}} \cdot V_{\cdot j}^T, \tag{5a}$$

$$\frac{\partial F}{\partial V_{\cdot j}} = -\frac{1}{2\sqrt{1 - S_j^2}} \sum_{i=1}^{n+1} \frac{\sqrt{X_{ij}}}{\sqrt{U_{i\cdot} V_{\cdot j}}} \cdot U_{i\cdot}^T, \tag{5b}$$

where $S = (S_j)_{1 \times m} = (\sum_{i=1}^{n+1} \sqrt{X_{ij}} \cdot \sqrt{U_{i\cdot} V_{\cdot j}})_{j=1,2,\ldots,m}$. Since both $X_{\cdot j}$ and $UV_{\cdot j}$ are on $\mathbb{P}_n$, S_j is guaranteed to be not bigger than 1.

We can optimize $V_{\cdot j}$ column-by-column on the simplex $\mathbb{P}_{r-1}$. As for $U_{i\cdot}$, however, row-by-row optimization may violate the constraints among the rows; for example, U_{ij} and U_{kj} with $i \neq k$ are not independent given $U_{\cdot j}^T \mathbf{1}_{n+1} = 1$. In this case, we cannot optimize $U_{i\cdot}$ and $U_{k\cdot}$ independently. To tackle this problem, we consider the product manifold of the multiple simplex manifold

$$\mathbb{P}_n^r = \{U = [U_{ik}] \in \mathbb{R}^{(n+1) \times r}) : U_{ik} \geq 0, U^T \mathbf{1}_{n+1} = \mathbf{1}_r\}.$$

We consider stacking the derivatives of $\frac{\partial F}{\partial U_{i\cdot}}$, obtain the derivative of U

$$\frac{\partial F}{\partial U} = ((\frac{\partial F}{\partial U_{1\cdot}})^T, (\frac{\partial F}{\partial U_{2\cdot}})^T, ..., (\frac{\partial F}{\partial U_{(n+1)\cdot}})^T)^T,$$

and then perform manifold optimization on the product manifold.

This approach is convenient because: (i) optimizing on the product manifold $\mathbb{P}_n^r$ intrinsically implies the constraints among the rows of U; and (ii) optimizing the rows of U jointly can parallelize computation and improve numerical efficiency. Although we do not need to jointly optimize $V_{\cdot j}$, we process it similar to the optimization U to exploit numerical efficiency.

Optimization on a Riemannian manifold is comparable to that on a Euclidean space, that is: (i) finding a descent direction; and (ii) performing a line search to obtain a sufficient decrease and to ensure convergence. In the following section, we discuss the optimization of U in detail to illustrate our optimization method on the simplex. The optimization of V is similar, which is omitted here.

4.1 Conjugate Gradients with the Armijo Line Search Method

The conjugate gradient method is chosen to solve the model in Eq. (4) due to its well-recognized efficiency when applied to large-scale problems.

The conjugate gradient method incorporates gradients at the current point with descent directions from previous points to obtain a new direction. Let $f(U)$ and $g(V)$ denote $F(U,V)$ with V and U fixed respectiveely, in Euclidean space, a conjugate gradient method generates a sequence U_k by the following recurrence:

$$U_{k+1} = U_k + \alpha_k \xi_{U_k}, \tag{6}$$

where α_k is the step size generated by a line search algorithm, and in Euclidean space ξ_{U_k} is the descent direction generated by the following rule:

$$\xi_{U_k} = -\frac{\partial f}{\partial U} + \beta_{k-1}\xi_{U_{k-1}}.$$

On a Riemannian manifold, the descent direction is computed on the tangent space. This space varies smoothly as one moves along the manifold. At a point U, the tangent space $T_U\mathbb{P}_n^r$ is a vector space and is actually a subspace of its embedded Euclidean space,

$$T_U\mathbb{P}_n^r = \{\xi_U \in R^{(n+1)\times r} : \xi_U^T \mathbf{1}_{n+1} = \mathbf{0}_r\}. \tag{7}$$

Given a descent direction $\xi_U \in T_U\mathbb{P}_n^r$, the line-search is performed along a smooth curve on the manifold. The derivative of this curve at U equals the descent direction ξ_U. After one step movement on tangent space T_U, we need to retract it back to the manifold. The retraction operation is defined as a mapping from the tangent space to the manifold. The updating step is concluded by $U_{k+1} = R_{U_k}(\alpha_k\xi_{U_k})$.

Unfortunately, the tangent space $T_U\mathbb{P}_n^r$ is defined locally at point U, which means $\xi_{U_k} \in T_{U_k}\mathbb{P}_n^r$ and $\xi_{U_{k+1}} \in T_{U_{k+1}}\mathbb{P}_n^r$ are in different tangent spaces and might not coincide. This property makes the direct addition of tangent vector ξ_{U_k} and $\xi_{U_{k+1}}$ impossible. To adapt the conjugate gradient method to manifolds, vector transport needs to be defined to transport tangent vectors from one tangent space to another.

To specify this recurrence to the simplex, we need to define: (i) a projection of the matrix in ambient space into the tangent space; and (ii) a retracting mapping from tangent space to the manifold $R_{U_k}(\alpha_k\xi_{U_{k+1}})$.

(i) Projection: the linear operation that projects a matrix $Z \in R^{n\times r}$ into the tangent space $T_U\mathbb{P}_n^r$, is defined as

$$\Pi_U(Z) = Z - \mathbf{1}_{n+1}\mathbf{1}_{n+1}^T Z \otimes U, \tag{8}$$

where $\otimes$ is the element-wise matrix multiplication operation.

(ii) Retraction: the mapping that locates the next iteration to the manifold along the tangent vector $R_U : T_U\mathbb{P}_n^r \to \mathbb{P}_n^r$ is

$$\begin{aligned} U_+ &= R_U(\xi_U) \qquad (9)\\ &= U \otimes \exp(\xi_U U^*) \cdot (\mathbf{1}_{n+1}\mathbf{1}_{n+1}^T(U \otimes \exp(\xi_U U^*)))^* \end{aligned}$$

where $()^*$ denotes the element-wise matrix inversion and $\exp(\cdot)$ denotes the element-wise exponential operator.

Let $\text{grad}f(U)$ denote the Riemannian gradient in tangent space $T_U\mathbb{P}_n^r$ at U; it is calculated by projecting a normalized Euclidean gradient into the tangent space $T_U\mathbb{P}_n^r$,

$$\text{grad}f(U) = \Pi_U(\frac{\partial f}{\partial U} \otimes U). \tag{10}$$

Algorithm 1 Alternating Direction Method for Eq. (3)

```
Input: $Y$: Incomplete matrix, $\Omega$: sampling set, $\epsilon$: stopping criteria,
    $k$: latent dimension, $T$: max iteration
Output: $X$: the optimal approximation of $Y$
 1: Initialize $U_0, V_0$ and $X_0 = O_{(n+1)\times m}$
 2: for $t = 1, 2, ..., T$ do
 3: $X_t = U_{t-1}V_{t-1} + P_\Omega(Y - U_{t-1}V_{t-1})$
 4: Update $U_t$ :
      4a: choose a proper step size $\tau_t$ from Eq.(14)
          $U_{init} = -\tau_t \text{grad}f(U_{t-1})$
      4b: use $U_{init}$ as an initial guess and call RSCG
          $U_t = \text{RSCG}(U_{init})$
 5: Update $V_t$ :
      5a: choose a proper step size $\upsilon_t$ from Eq.(14)
          $V_{init} = -\upsilon_t \text{grad}g(V_{t-1})$
      5b: use $U_{init}$ as an initial guess and call RSCG
          $V_t = \text{RSCG}(V_{init})$
 6: If 1-$F(U_t, V_t)/F(U_{t-1}, V_{t-1}) < \epsilon$; break; end
 7: end for
```

The vector transport from the previous tangent space to the current tangent space

$$\mathcal{T}_{U_k \to U_{k+1}} : T_{U_k}\mathbb{P}_n^r \to T_{U_{k+1}}\mathbb{P}_n^r, \xi_{U_k} \mapsto \Pi_{U_{k+1}}(\xi_{U_k}) \tag{11}$$

is specified by the projection defined in Eq.(8).

Hence the conjugate gradient method can be specified to the simplex

$$U_{k+1} = R_{U_k}(\alpha_k \xi_{U_k}) \tag{12}$$

with $\xi_{U_k} = -\text{grad}f(U_k) + \beta_{k-1}\mathcal{T}_{U_{k-1}\to U_k}\xi_{U_{k-1}}$.

4.2 Conjugate Gradients Update Parameter

Several options of β_k can be used in the conjugate gradient algorithm. Here we choose the Polak-Ribiere(PR+) method to calculate the CG update parameter. Within the Riemannian optimization framework, β_k takes the following form:

$$\beta_k = \frac{\langle \text{grad}f(U_k), \text{grad}f(U_k) - \mathcal{T}_{\boldsymbol{U}_{k-1}\to \boldsymbol{U}_k}(\text{grad}f(U_{k-1}))\rangle}{\langle \text{grad}f(U_{k-1}), \text{grad}f(U_{k-1})\rangle}.$$

Note that the PR+ updating rule has the auto restart property when the descent direction is almost orthogonal to the gradient direction. Specifically, the algorithm will abandon the previous CG descent direction and initialize the CG algorithm according to the current gradient direction.

4.3 Armijo Step Size

The step size α_k is chosen with the Armijo line search algorithm. Given the initialized step size $\overline{\alpha}$ and $t, \sigma \in (0,1)$, the condition is given by:

$$f(U) - f(R_U(t^m\overline{\alpha}\xi)) \geq -\sigma\langle \text{grad}f(U), t^m\overline{\alpha}\xi\rangle_U,$$

where m is the number of backtrackings in finding the current Armijo point. Thus the step size is given by $\alpha = t^m\overline{\alpha}$.

The initialization of step size $\overline{\alpha}$ is very important since it can greatly enhance the efficiency of the line search algorithm. In our method, we observe that an exact minization on the tangent space alone without retraction is a good initial guess. For U, the exact minimization takes the form

$$\min_t \frac{1}{2}\|X - (U + t\xi)V\|_F^2, \tag{13}$$

Table 1: Dataset information

Dataset	#Samp.	#Class	Feature	Rep	#Dim
MIT Scene	2080	8	SIFT	BoF	600
UIUC Scene	3000	15	SIFT	BoF	600

V takes a similar form so it is omitted here. The minimizer of Eq. (13) takes a closed-form solution:

$$t_* = \langle X - UV, \xi V\rangle / \langle \xi V, \xi V\rangle. \qquad (14)$$

As ξ is the search direction of U, it is always non-zero when the algorithm has not converged. Therefore, there are no worries about the denominator being zero.

We name our algorithm the "Riemannian Simplex Conjugate Gradient" (RSCG). The convergence of RSCG can be guaranteed by [Absil *et al.*, 2009; Hager and Zhang, 2006] when β is chosen by the PR+ rule and α is chosen by the Armijo condition. The overall algorithm is summarized in Algorithm 1.

5 Experiments

In this section, we evaluate the proposed MC-S on three real-world datasets and compare it with PMF [Salakhutdinov and Mnih, 2007], NPCA [Yu *et al.*, 2009], LGeomCG [Vandereycken, 2013] and LMaFit [Wen *et al.*, 2012]. PMF is a probabilistic matrix factorization model for collaborative filtering that models users' preference by a conditional distribution on user and item matrices ; LMaFit performs baisc matrix factorization while employing non-linear successive over-relaxation algorithm for advancing the effectiveness; NPCA refers to a non-parametric probabilistic PCA model; and LGeomCG performs matrix completion on the manifold of fixed-rank matrices.

Following Marlin's setup [Marlin, 2004], we test our algorithm under "weak" and "strong" generalization. The weak generalization is a single step process that involves directly predicting missing data in the rating matrix. The strong generalization refers to a two-stage process where the models are trained on one set of users and tested on another disjoint set of users. Since LGeomCG cannot explicitly factorize a rating matrix into user profiles and item profiles, it is not applicable for strong generalization. The measurements for matrix recovery are root mean square error (RMSE) and normalized mean absolute error (NMAE):

$$\text{RMSE} = \|P_\Omega(Y - X)\|_F / \sqrt{|\Omega|},$$
$$\text{NMAE} = |P_\Omega(Y - X)| / (Y_{\max} - Y_{\min})|\Omega|,$$

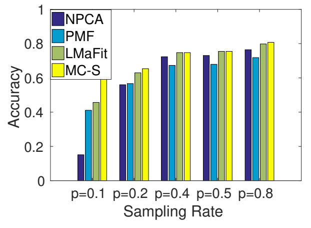
where Y represents ground truth ratings and X shows estimated values.

5.1 Image Data

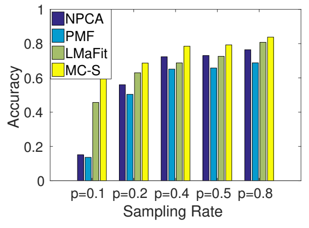
We consider two image datasets MIT Scene and UIUC Scene whose properties and parameters are displayed in Table 1. We extract SIFT descriptors on 16×16 patches in images and then convert them to bag-of-features (BoF). The number of visual words in the codebook, $|V|$, is set to be 600 and the implementation is based on *vlfeat toolbox* [Vedaldi and Fulkerson, 2010].

Table 2: Comparisons of different collaborative filtering methods in terms of RMSE ($\times 10^{-3}$) and NMAE ($\times 10^{-3}$)

Methods	MIT Scene		UIUC Scene	
	W. RMSE	S. RMSE	W. RMSE	S. RMSE
LMaFit	10.897	8.804	8.802	8.238
LGeomCG	9.691	—	11.415	—
PMF	9.271	9.129	8.508	8.508
NPCA	4.028	8.926	3.224	6.587
MC-S	2.929	4.194	2.673	4.201
	W. NMAE	S. NMAE	W. NMAE	S. NMAE
LMaFit	2.078	2.383	2.175	1.833
LGeomCG	1.561	—	1.738	—
PMF	2.476	3.185	2.426	2.426
NPCA	1.155	2.648	0.830	2.305
MC-S	1.197	1.721	1.310	1.929



(a) MIT Scene



(b) UIUC Scene

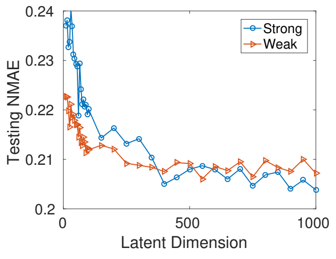
Figure 2: Classification results vary with sampling fraction on MIT Scene and UIUC Scene.

The BoF representation of each image is the histogram of visual words. Each dimension of a histogram represents the relative frequency of a visual word in the descriptors of an image and the sum of the histogram equals 1. It is an estimation of the probability distribution and is exactly on a $(|V|-1)$-simplex.

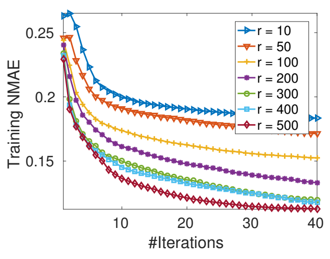
We first extract the BoF features of an image dataset. Define sampling fraction $p = \frac{|\Omega|}{(n+1)m}$ as the ratio of observed entries to total entries. By setting $p = 0.5$, we randomly keep 50% of features for fitting collaborative filtering models. Then we split the dataset into training, validation and test sets. In "weak" generalization, we train the collaborative filtering algorithms on the training set and recover the full matrix. Then we train multi-class Support Vector Machine classifiers on the recovered training set by one-vs-all strategy. In "strong" generalization, we train the collaborative filtering algorithms on test set by using the basis matrix learned from training set and hence estimate missing entries in test set. At last we evaluate the performance of SVM classifier on the recovered test set.

Evaluation results

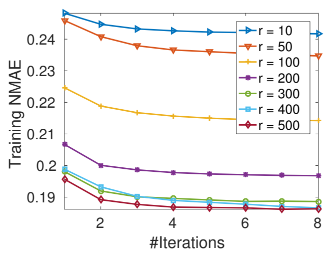
The number of basis points on the simplex r is set to be the number of classes. We random start the training of collaborative filtering algorithms for five times and report averaged RMSE and NMAE in Table 2. We find that MC-S outperforms baselines on two datasets under both "weak" and "strong" generalization. The BoF feature vectors lie exactly on the $(|V|-1)$-simplex. While comparative algorithms ignore this property, we subtly make use of data's geometric properties and constrain the optimization of each user's rating



(a) NMAE



(b) Weak NMAE



(c) Strong NMAE

Figure 3: MovieLens 1M: NMSE and NMAE as objective decreasing with iterations under weak and strong generalization. r refers to number of basis points.

vector on the simplex, which leads to prominent performance of MC-S. The corresponding classification performance is illustrated in Fig. 2 with sampling fraction $p = 0.5$.

When sampling fraction $p = 1$, which means no feature is missing, the averaged classification accuracies of MIT Scene and UIUC Scene are 88.75% and 76.87% respectively. Based on the collaborative filtering results under different sampling fraction p, we repeat the training of SVM for five times, and report averaged accuracies in Fig. 2. With the growing of sampling fractions, all methods witness rising performance. Under different sampling fractions of feature matrix, our method MC-S achieves consistently better performance on UIUC Scene and shows promising efficacy especially when sample size is small. This indicates that our collaborative filtering method implements better approximation to the ground truth feature matrix.

5.2 Recommendation Data

We next explore the proposed MC-S's performance on MovieLens dataset in collaborative filtering [Harper and Konstan, 2015]. ML-1M contains 1,000,209 anonymous ratings of 3,952 movies by 6,040 MovieLens users. 5,000 users are selected for weak generalization and 1,040 users for strong generalization. The input ratings are integers between 1 and 5, and the budget of user j's ratings on items is fixed, i.e., $E = \sum_{i=1}^{n+1} Y_{ij}$. We can normalize the data through $Y_{ij}^{'} = Y_{ij}/E$, $i = 1, 2, ..., n+1$. The pre-processed ratings of user i now lie on the simplex satisfying $\sum_{i=1}^{n+1} Y_{ij}^{'} = 1$. By assuming that sampling of Ω is random and unbiased, the budget of one user's ratings on n items is approximated by $E \approx (\sum_{(i,j)\in\Omega} P_\Omega(Y))(n+1)/\Omega$. The test and validation sets are created by reserving one rating from each user respectively. This process is repeated three times. We compare the averaged NMAE on three random partitions of weak and strong generalization and report the results in Table 3. After the preprocessing, users' rating vectors lie on the simplex and our method MC-S can preserve this geometric property during the learning of user and item matrices, while baseline methods optimize in the free space and hence gain deficient performance.

Latent dimensionality and efficiency

To test the response of MC-S to increasing latent dimension, we report our testing NMAE under latent dimension from 10 to 1000. In Fig. 3, NMAE shows obvious decreasing trends with the increasing of latent dimension and stays stable between [10, 1000], which indicates that MC-S is steady and can achieve good performance in a large parameter range.

Table 3: MovieLens 1M: Comparisons with baselines in terms of NMAE

Methods	Weak NMAE	Strong NMAE
LMaFit	0.6808± 0.001	0.6740±0.0048
LGeomCG	0.6275±0.0032	—
PMF	0.2339±0.0008	0.2339±0.0036
NPCA	0.2267±0.0009	0.2266±0.0043
MC-S	0.2128±0.0022	0.2143±0.0025

To verify the efficiency of MC-S, changes of NMAE with iterations under different latent dimensionalities in training process are reported in Fig. 3. We find that in weak generalization, MC-S reduces NMAE by every iteration and stays stable after 10 iterations while in strong generalization, MC-S becomes steady after only three iterations. We owe this effiency to the representative model learned from weak generalization. The model's representativeness indicates MC-S's capability of learning the geometric properties of users' rating vectors on the simplex.

6 Conclusions

Here we consider collaborative filtering problem with energy limited users. The limited budget of a user's ratings is depicted by the simplex. By seating a user's rating vector on the simplex, we abandon the usual Euclidean metric and employ the pull-back metric from the sphere to specify the distance between points on the simplex, since it can depict geometric constraints of a user's rating vector. We extend conjugate gradient method onto the simplex to optimize our model. Experiments on real data demonstrate our model's effectiveness on learning user and item features.

Acknowledgments

This work is supported by Australian Research Council Projects FT-130101457, DP-140102164, LP-150100671.

References

[Absil *et al.*, 2009] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[Breese *et al.*, 1998] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *UAI*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[Chee *et al.*, 2001] Sonny Han Seng Chee, Jiawei Han, and Ke Wang. Rectree: An efficient collaborative filtering method. In *DWKD*, pages 141–151. Springer, 2001.

[Fang *et al.*, 2014] Hui Fang, Yang Bao, and Jie Zhang. Leveraging decomposed trust in probabilistic matrix factorization for effective recommendation. In *AAAI*, pages 30–36. AAAI Press, 2014.

[Ghazanfar *et al.*, 2012] Mustansar Ali Ghazanfar, Adam Prügel-Bennett, and Sandor Szedmak. Kernel-mapping recommender system algorithms. *Information Sciences*, 208:81–104, 2012.

[Hager and Zhang, 2006] William W Hager and Hongchao Zhang. A survey of nonlinear conjugate gradient methods. *Pacific journal of Optimization*, 2(1):35–58, 2006.

[Harper and Konstan, 2015] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *ACM TIIS*, 5(4):19, 2015.

[Herlocker *et al.*, 1999] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *ACM SIGIR*, pages 230–237. ACM, 1999.

[Lawrence and Urtasun, 2009] Neil D Lawrence and Raquel Urtasun. Non-linear matrix factorization with gaussian processes. In *ICML*, pages 601–608. ACM, 2009.

[Le and Cuturi, 2015] Tam Le and Marco Cuturi. Unsupervised riemannian metric learning for histograms using aitchison transformations. In *ICML*, pages 2002–2011, 2015.

[Lee and Seung, 1999] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[Lee *et al.*, 2012] Joonseok Lee, Mingxuan Sun, and Guy Lebanon. A comparative study of collaborative filtering algorithms. *arXiv preprint arXiv:1205.3193*, 2012.

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

[Liu and Tao, 2016] Tongliang Liu and Dacheng Tao. On the performance of manhattan nonnegative matrix factorization. *IEEE TNNLS*, 27(9):1851–1863, September 2016.

[Marlin, 2004] Benjamin Marlin. *Collaborative filtering: A machine learning perspective*. PhD thesis, University of Toronto, 2004.

[Miyahara and Pazzani, 2002] Koji Miyahara and Michael J Pazzani. Improvement of collaborative filtering with the simple bayesian classifier. *Information Processing Society of Japan*, 43(11), 2002.

[OConnor and Herlocker, 1999] Mark OConnor and Jon Herlocker. Clustering items for collaborative filtering. In *ACM SIGIR Workshop on Recommender Systems*, volume 128. UC Berkeley, 1999.

[Papagelis *et al.*, 2005] Manos Papagelis, Dimitris Plexousakis, and Themistoklis Kutsuras. Alleviating the sparsity problem of collaborative filtering using trust inferences. In *International Conference on Trust Management*, pages 224–239. Springer, 2005.

[Parikh and Boyd, 2014] Neal Parikh and Stephen P Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

[Rennie and Srebro, 2005] Jasson DM Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *ICML*, pages 713–719. ACM, 2005.

[Salakhutdinov and Mnih, 2007] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Nips*, volume 1, pages 2–1, 2007.

[Salakhutdinov and Mnih, 2011] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. Citeseer, 2011.

[Sarwar *et al.*, 2001] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th WWW*, pages 285–295. ACM, 2001.

[Shani *et al.*, 2002] Guy Shani, Ronen I Brafman, and David Heckerman. An mdp-based recommender system. In *UAI*, pages 453–460. Morgan Kaufmann Publishers Inc., 2002.

[Song and Zhu, 2016] Yang Song and Jun Zhu. Bayesian matrix completion via adaptive relaxed spectral regularization. In *AAAI*, 2016.

[Srebro *et al.*, 2004] Nathan Srebro, Jason Rennie, and Tommi S Jaakkola. Maximum-margin matrix factorization. In *NIPS*, pages 1329–1336, 2004.

[Su and Khoshgoftaar, 2006] Xiaoyuan Su and Taghi M Khoshgoftaar. Collaborative filtering for multi-class data using belief nets algorithms. In *18th ICTAI*, pages 497–504. IEEE, 2006.

[Su and Khoshgoftaar, 2009] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:4, 2009.

[Sun *et al.*, 2016] Yanfeng Sun, Junbin Gao, Xia Hong, Bamdev Mishra, and Baocai Yin. Heterogeneous tensor decomposition for clustering via manifold optimization. *IEEE TPAMI*, 38(3):476–489, 2016.

[Vandereycken, 2013] Bart Vandereycken. Low-rank matrix completion by riemannian optimization. *SIAM Journal on Optimization*, 23(2):1214–1236, 2013.

[Vedaldi and Fulkerson, 2010] Andrea Vedaldi and Brian Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1469–1472. ACM, 2010.

[Wang *et al.*, 2015] Xiangyu Wang, Dayu He, Danyang Chen, and Jinhui Xu. Clustering-based collaborative filtering for link prediction. In *AAAI*, pages 332–338, 2015.

[Wen *et al.*, 2012] Zaiwen Wen, Wotao Yin, and Yin Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 4(4):333–361, 2012.

[Xu *et al.*, 2016] Chang Xu, Dacheng Tao, and Chao Xu. Robust extreme multi-label learning. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA August*, pages 13–17, 2016.

[Yu *et al.*, 2009] Kai Yu, Shenghuo Zhu, John Lafferty, and Yihong Gong. Fast nonparametric matrix factorization for large-scale collaborative filtering. In *ACM SIGIR*, pages 211–218. ACM, 2009.