Tensor Completion with Side Information: A Riemannian Manifold Approach

Tengfei Zhou, Hui Qian*, Zebang Shen, Chao Zhang, Congfu Xu
College of Computer Science and Technology, Zhejiang University, China
{zhoutengfei,qianhui,shenzebang,zczju,xucongfu}@zju.edu.cn

Abstract

By restricting the iterate on a nonlinear manifold, the recently proposed Riemannian optimization methods prove to be both efficient and effective in low rank tensor completion problems. However, existing methods fail to exploit the easily accessible side information, due to their format mismatch. Consequently, there is still room for improvement. To fill the gap, in this paper, a novel Riemannian model is proposed to tightly integrate the original model and the side information by overcoming their inconsistency. For this model, an efficient Riemannian conjugate gradient descent solver is devised based on a new metric that captures the curvature of the objective. Numerical experiments suggest that our method is more accurate than the state-of-the-art without compromising the efficiency.

1 Introduction

Low Rank Tensor Completion (LRTC) problem, which aims to recover a tensor from its linear measurements, arises naturally in many artificial intelligence applications. In hyperspectral image inpainting, LRTC is applied to interpolate the unknown pixels based on the partial observation [Xu et al., 2015]. In recommendation tasks, LRTC helps users find interesting items [Liu et al., 2015]. In computational phenotyping, one adopts LRTC to discovery phenotypes in heterogeneous electronic health records [Wang et al., 2015].

Euclidean Models: LRTC can be formulated by a variety of optimization models over the Euclidean space. Amongst them, convex models that encapsulate LRTC as a regression problem penalized by a tensor nuclear norm are the most popular and well-understood [Romera-Paredes and Pontil, 2013; Zhang et al., 2014]. Though most of them have sound theoretical guarantees [Zhang and Aaron, 2016; Chen et al., 2013; Yuan and Zhang, 2015], in general, their solvers are ill-suited for large tensors because these procedures involve Singular Value Decomposition (SVD) of huge matrices per iteration [Liu et al., 2013]. Another class of Euclidean models is formulated as the decomposition problem that factorizes a low rank tensor into small factors [Jain and Oh, 2014; Filipović and Jukić, 2015; Xu et al., 2015]. Many solvers for such decomposition based model have been proposed, and low per-iteration computational cost is illustrated [Beutel et al., 2014; Liu et al., 2014; Smith et al., 2016].

Riemannian Models: LRTC can also be modeled by optimization constrained on Riemannian manifolds [Kressner et al., 2014; Kasai and Mishra, 2016], which is easily handled by many manifold based solvers [Absil et al., 2009]. Empirical comparison has shown that Riemannian solvers use significantly less CPU time to recover the underlying tensor in contrast to the Euclidean solvers [Kasai and Mishra, 2016]. The main reason resides in that they avoid SVD of huge matrices by explicitly exploiting the geometrical structure of LRTC, which makes them more suitable for massive problem.

Of all the Riemannian models, two search spaces, fix multi-linear rank manifold [Kressner et al., 2014] and Tucker manifold [Kasai and Mishra, 2016], are usually employed. The former is a sub-manifold of Euclidean space, and the latter is a quotient manifold induced by the Tucker decomposition. Generally, quotient manifold based solvers have higher convergence rates because it is usually easier to design a pre-conditioner for them [Kasai and Mishra, 2016; Mishra and Sepulchre, 2016].

Side Information: In the Euclidean models of LRTC, side information has been proved to be helpful in improving the accuracy [Narita et al., 2011; Acar et al., 2011; Beutel et al., 2014]. One common form of the side information is the feature matrix, which measures the statistical properties of tensor modes [Kolda and Bader, 2009]. For example, in Netflix tasks, feature matrix can be built from the demography of users [Bell and Koren, 2007]. Another form is the similarity matrix, which quantifies the resemblance between two entities of a tensor mode. For instance, the social network generates the similarity matrix by utilizing the correspondence between users [Rai et al., 2015]. In practice, these two matrices can be transformed to each other, and we only consider the feature matrix case.

However, as far as we know, side information has not been incorporated in any Riemannian model for LRTC. The first difficulty lies in the model design. Fusing the side information into the Riemannian model inevitably compromises the integrity of the low rank tensor due to the compactness of the manifold. The second difficulty results from the solver design. Incorporating the side information may aggravate the
ill-conditioning of LRTC problem and degenerates the convergence significantly.

**Contributions:** To address these difficulties, a novel Riemannian LRTC method is proposed from the perspective of both model and solver designs. By exploring the relation between the subspace spanned by the tensor fibers and the column space of the feature matrix, we explicitly integrate the side information in a compact way. Meanwhile, a first order solver is devised under the manifold optimization framework. To ease the ill-conditioning, we design a novel metric based on an approximated Hessian of the cost function. The metric implicitly induce an adaptive preconditioner for our solver. Empirical studies illustrate that our method achieves much more accurate solutions within comparable processing time than the state-of-the-art.

2 Notations and Preliminaries

In this paper, we only focus on the 3rd order tensor, but generalizing our method to higher order is straightforward. We use the notation $X \in \mathbb{R}^{n \times n \times n}$ to denote a matrix, and the notation $X \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ to denote a $d$-th order tensor. We also denote by $X(i_1, \cdots, i_d)$ the element in position $(i_1, \cdots, i_d)$ of $X$. For many cases, we use abbreviation like $\{O_i\}_{i=1}^3$ to denote the sequence $O_1, O_2, O_3$.

**Mode-k Fiber and Matricization:** A fiber of a tensor is obtained by varying one index while fixing the others, i.e., $X(i_1, \cdots, k_{-1}, 1, k_{+1}, \cdots, i_d)$ is the mode-$k$ fiber of a $d$-th order tensor $X$. A mode-$k$ matricization $X(k)_j \in \mathbb{R}^{n_{k+1} \times (n_1 \cdots n_{k-1} n_{k+2} \cdots n_d)}$ of a tensor is obtained by arranging the mode-$n$ fibers of $X$ so that each of them is a column of $X(k)_j$ [Kolda and Bader 2009]. The mode-$k$ product of tensor $X$ and matrix $A$ is denoted by $X \times_k A$, whose mode-$k$ matricization can be expressed as $(X \times_k A)(k)_j = A \cdot X(k)_j$.

**Inner Product and Norm:** The inner product of two tensors with the same size is defined by $\langle \mathbf{X}, \mathbf{Y} \rangle = \sum_{i_1, \cdots, i_d} X(i_1, \cdots, i_d) \cdot Y(i_1, \cdots, i_d)$. The Frobenius norm of a tensor $X$ is defined by $\|X\|_F = \sqrt{\langle X, X \rangle}$.

**Multi-linear rank and Tucker decomposition:** The multi-linear rank $\text{rank}(X) = (r_1, r_2, r_3)$, Tucker decomposition factorizes $X$ into a small core tensor $G \in \mathbb{R}^{r_1 \times r_2 \times r_3}$ and three matrices $U_i \in \mathbb{R}^{n_i \times r_i}$ with orthogonal columns, that is $X = G \times_1 U_1 \times_2 U_2 \times_3 U_3$. Note that, the Tucker decomposition of a tensor is not unique. In fact, if $X = G \times_1 U_1$, we can easily obtain $X = H \times_1 V_1$, with $H = G \times_3 O_3^\top$, $V_1 = U_1 O_1$, where $O_1 \in \mathbb{R}^{n_1 \times r_1}$ is any orthogonal matrix. Thus, we obtain the equivalent class

$$\{G, \{U_i\}_{i=1}^3\} \triangleq \{(G \times_3 O_3^\top, \{U_i O_i\}_{i=1}^3)\} O_3^\top O_1 = I_1\}.$$

We denote $\{G, \{U_i\}_{i=1}^3\}$ by $[X]$, when $X = G \times_3 U_3$. Usually, $[X]$ is called the Tucker representation of $X$, while $X$ is called the tensor representation of $[X]$. We also use $\overline{X}$ to denote a specific decomposition of $X$, additionally $\overline{X} \in [X]$.

2.1 Search Space of Riemannian Models

The Tucker manifold that we used in our Riemannian model is a quotient manifold induced by the Tucker decomposition. In order to lay the ground for Tucker manifold, we first describe its counterpart, the fix multi-rank manifold, which will be helpful in understanding the whole derivation.

A fixed multi-linear rank manifold $\mathcal{F}_r$ consists of tensors with the same fixed multi-linear rank. Specifically

$$\mathcal{F}_r = \{X \in \mathbb{R}^{n_1 \times n_2 \times n_3} \mid \text{rank}(X) = r\}.$$

To define the Tucker manifold, we first define a total space $\mathcal{M}_r = \mathbb{R}^{r_1 \times r_2 \times r_3} \times S(r_1, n_1) \times S(r_2, n_2) \times S(r_3, n_3)$, (1) in which $S(r_i, n_i)$ is the Stiefel manifold of $r_i \times n_i$ matrices with orthogonal columns. Then, we can depict the Tucker manifold of multi-linear rank $r$ as follows.

$$\mathcal{M}_r / \sim \triangleq \{(G, \{U_i\}_{i=1}^3) \mid (G, \{U_i\}_{i=1}^3) \in \mathcal{M}_r\}.$$ (2)

The Tucker manifold is a quotient manifold of the total space (1). We use the abstract quotient manifold, rather than the concrete total space, as search space because the non-uniqueness of the Tucker decomposition is undesirable for optimization. Note that such non-uniqueness will introduce more local optima into the minimization. The relation of manifold $\mathcal{F}_r$ and $\mathcal{M}_r / \sim$ is characterized as follows.

**Proposition 1.** The quotient manifold $\mathcal{M}_r / \sim$ is diffeomorphic to the fix multi-linear rank manifold $\mathcal{F}_r$, with diffeomorphism $\mu(\cdot)$ from $\mathcal{F}_r$ to $\mathcal{M}_r / \sim \sim$ defined by $\mu(X) = (G, \{U_i\}_{i=1}^3)$ where $\{G, \{U_i\}_{i=1}^3\}$ is the Tucker representation of $X$.

This proposition says that each tensor $X \in \mathcal{F}_r$ can be represented by a unique equivalent class $\{G, \{U_i\}_{i=1}^3\} \in \mathcal{M}_r / \sim$ and vice-versa.

2.2 Vanilla Riemannian Tensor Completion

The purest incarnation of Riemannian tensor completion model is the Riemannian model over the fix multi-linear rank manifold. Let $\mathcal{R} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ be a partially observed tensor. Let $\Omega$ be the set which contains the indices of observed entries. The model can be expressed as:

$$\min_{X \in \mathcal{F}_r} \frac{1}{2} \|P_{\Omega}(X - \mathcal{R})\|_F^2 \quad \text{s.t. } X \in \mathcal{F}_r,$$ (3)

with $P_{\Omega}$ maps $X$ to the sparsified tensor $P_{\Omega}(X)$, where $P_{\Omega}(X)(i_1, i_2, i_3) = X(i_1, i_2, i_3)$ if $(i_1, i_2, i_3) \in \Omega$, and $P_{\Omega}(X)(i_1, i_2, i_3) = 0$ otherwise.

Another popular model, Tucker model, is based on the quotient manifold $\mathcal{M}_r / \sim$, which can be expressed as:

$$\min_{X \in \mathcal{M}_r / \sim} \frac{1}{2} \|P_{\Omega}(\mu^{-1}(X) - \mathcal{R})\|_F^2 \quad \text{s.t. } X \in \mathcal{M}_r / \sim.$$ (4)

with $\mu$ defined in Prop. 1.

Note that since the dawn of Riemannian framework for LRTC, a quandary exists: on one hand, sparse measurement limits the capacity of the solution; on the other hand, rich side information can not be incorporated into this framework. In many artificial intelligence applications, demands for high accuracy further exacerbates such dilemma.
3 Riemannian Model with Side Information

We focus on the case that the side information is encoded in feature matrices $\mathbf{P}_i \in \mathbb{R}^{n_i \times k_i}$. Suppose $\mathcal{R} \in \mathcal{F}$ has Tucker factors $(\mathcal{G}, \{\mathbf{U}_i\}_{i=1}^3)$. Without loss of generality, we assume that $k_i \geq r_i$ and $\mathbf{P}_i$ has orthogonal columns.

In the ideal case, we assume that
\[
\text{span}(\mathbf{U}_i) \subset \text{span}(\mathbf{P}_i).
\]
Such relation means that the feature matrices contain all the information in the latent space of the underlying tensor. Equivalently, there exists a matrix $\mathbf{W}_i$ such that $\mathbf{U}_i = \mathbf{P}_i \mathbf{W}_i$. However, in practice, due to the existence of noise, one can only expect such relation to hold approximately, i.e. $\mathbf{U}_i \approx \mathbf{P}_i \mathbf{W}_i$. Incorporating such relation to a tensor completion model via penalization, we have the following formulation
\[
\min_{\mathcal{G}, \{\mathbf{U}_i\}} \frac{1}{2} \norm{\mathbf{L} \mathcal{G} \{\mathbf{U}\}_{i=1}^3} + \frac{3}{2} \sum_{i=1}^3 \frac{\alpha_i}{2} \norm{\mathbf{U}_i - \mathbf{P}_i \mathbf{W}_i}^2.
\]

Remark 1. In Riemannian optimization literature, problem (8) is called the lifted representation of problem (9) over the total space [Absil et al., 2009]. This model is closely related to the Laplace regularization model [Narita et al., 2011]. Concretely, they share the same form:
\[
\min_{\mathcal{G}, \{\mathbf{U}_i\}} \frac{1}{2} \norm{\mathbf{L} \mathcal{G} \{\mathbf{U}\}_{i=1}^3} + \frac{3}{2} \sum_{i=1}^3 \frac{\alpha_i}{2} \norm{\mathbf{U}_i - \mathbf{P}_i}^2.
\]

The difference lies in that $\mathbf{L}_i$ is a projection matrix in our case, while, in the Laplace regularization model, $\mathbf{L}_i$ is a Laplacian matrix.

4 Riemannian Conjugate Gradient Descent

We depict the optimization framework for quotient manifolds in Fig. 1. Under this framework, we solve the proposed problem (9) by Riemannian Conjugate Gradient descent (CG). With the details specified later, we list our CG solver for problem (9) in Alg. 1, where the CG direction is composed in the Polak-Ribiere+ manner with the momentum weight $\beta^{(k)}$ computed by Fletcher-Reeves formula [Absil et al., 2009], and $T_k(\cdot)$ is the projector of horizontal space $\mathcal{H}_k$. The convergence property of CGSI (Alg. 1) to a stationary point follows the general analysis of CG method [Sato and Iwai, 2015]. To represent CGSI in concrete tensor formulations, four items must be specified: the Riemannian metric $\langle \cdot, \cdot \rangle_{\mathcal{X}}$, the Riemannian gradient $\nabla \mathcal{F}(\mathcal{X})$, the retraction $R_{\mathcal{X}}(\cdot)$, and the projector onto horizontal space $T_{\mathcal{X}}$.
Algorithm 1 CGSI: a Riemannian CG method

Require: Initializer \(X^{(0)} = (G^{(0)}, \{U_i^{(0)}\}_{i=1}^3)\) and tolerance \(\epsilon\)
1: \(k = 0;\)
2: \(\eta^{(-1)} = (0, (0)^3)_{i=1};\)
3: repeat
4: compute current Riemannian gradient \(\xi^{(k)} = \nabla f(X^{(k)});\)
5: compose CG direction \(\eta^{(k+1)} = -\xi^{(k)} + \beta^{(k)}T_k(\eta^{(k-1)});\)
6: choose a step size \(t_k > 0;\)
7: update by retraction \(X^{(k+1)} = R_{X^{(k)}}(t_k \eta^{(k)});\)
8: \(k = k + 1;\)
9: until \((\xi^{(k-1)}, \xi^{(k-1)})_{X^{(k-1)}} \leq \epsilon;\)
10: return \(X^{(k)}\).

a well-tuned metric, because (1) the metric determines the differential structure of the quotient manifold, and more importantly (2) it implicitly endows the solver with a preconditioner, which heavily affects the convergent rate [Mishra and Sepulchre, 2014, Mishra, 2014].

From the perspective of preconditioning, it seems that the best candidate is the Newton metric \(\langle \eta, \xi \rangle_{X} = D^2 f(X)[\eta, \xi]_{X} = \|
abla f(X)\|^2 + \sum_{i=1}^3 \alpha_i \|\nabla U_i f\|^2; \chi_{\text{sym}}(\alpha_i \|\nabla U_i f\|^2)\) \(\eta, \xi \in T_XM_r\).

where \(g(X)\) is a scaled approximation to the original cost function, and \(D^2 g(X)\) is the block approximation of its second derivative, specifically \(g(X) \triangleq \frac{1}{2} \|\nabla X \|_{Y}^2 + \sum_{i=1}^3 \alpha_i \|U_i \|^2; \chi_{\text{sym}}(\alpha_i \|U_i \|^2)\) \(\eta, \xi \in T_XM_r\).

Our metric is more scalable than Newton metric. The following Proposition indicates that the scale induced by this metric can be computed with \(O(\sum_{i=1}^3 n_i k_i r_i + r_i^3)\) additional operations.

**Proposition 3.** Suppose that the cost function \(f(\cdot)\) has Euclidean gradient \(\nabla f(X) = (\nabla g f; \nabla U_i f)_{i=1}^3.\) Then its scaled gradient \(\nabla f(X)\) under the metric (11) can be computed by:

\[
\nabla g f(X) = \nabla g f(X)
\]

\[
\nabla U_i f(X) = E_i G_i^{-1} + F_i G_i + N_{i} L_i^{-1}
\]

where \(E_i = P_i \nabla U_i f; F_i = \nabla U_i f - E_i; \) and \(G_i = G_i G_i^{-T}.\)

The final proposition suggests that the proposed metric makes the representation of solvers in the total space possible.

**Proposition 4.** The quotient manifold \(M_r/ \sim\) admits a structure of Riemannian quotient manifold, if \(M_r\) is endowed with the Riemannian metric defined in (11).

---

Table 1: Expressions of Projectors. We define the following matrices: \(V_i := P_i P_i^T; U_i := X_i - X_r; G_i := G_i G_i^T; G_{aa} := \alpha_i I_i + G_i G_i^T.\) \(\xi_i = \max\{k|k \in \{1, 2, 3\}, k \neq i\}\) and \(k_i = \min\{k|k \in \{1, 2, 3\}, k \neq i\}.\) And the operator \(\text{sym}()\) and \(\text{skw}()\) extract the symmetric and skew components of a matrix respectively, i.e. \(\text{sym}(A) = (A + A^T)/2\) and \(\text{skw}(A) = (A - A^T)/2.\) Note that the above linear systems can be solved by MATLAB command \(\text{pcg}\) in \(O(\sum_{i=1}^3 (n_i k_i^2 + r_i^3))\) flops.

4.2 Other Optimization Items

**Projectors:** To derive the optimization related items, two orthogonal projectors, \(\Psi_{X_r}(\cdot)\) and \(\Pi_{X_r}(\cdot)\), are required. The former projects a vector onto the tangent space \(T_{X_r}M_r\), and the latter is a projector from the tangent space onto the horizontal space \(\mathcal{H}_{X_r}\). The orthogonality of both projectors is measured by the metric (11). For lack of space, the mathematical derivation is deferred to a long version of this paper.

**Riemannian Gradient:** According to [Abis et al., 2009], the Riemannian gradient can be computed by projecting the scaled gradient onto tangent space, specifically

\[
\nabla f(X) = \Psi_{X_r}(\nabla f(X)).
\]

**Retraction:** We use the retraction defined by

\[
R_{X_r}(\eta_X) = \left(\mathbb{G} + \eta_X, \{\text{uf}(U_i + \eta_i)\}_{i=1}^3\right)
\]

where \(\text{uf}(\cdot)\) extracts the orthogonal component from a matrix. Such retraction is proposed by [Kasai and Mishra, 2016]. In the long version of this paper, we give rigorous analysis to prove that the above retraction is compatible with the proposed metric.

5. **Experiments**

We validate the effectiveness of the proposed solver CGSI by comparing it with the state-of-the-art. The baseline can be partitioned into three classes. The first class contains Riemannian solvers including GeoCG [Kressner et al., 2014], FTC [Kasai and Mishra, 2016], and gHOI [Liu et al., 2016]. The second class consists of Euclidean solvers that take no account of the side information, including AltMin [Romera-Paredes et al., 2013] and HalRTC [Liu et al., 2013]. The third class comprises of two methods that incorporate side information, including RUBIK [Wang et al., 2015] and TFAI.
Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)

Table 2: Performance of the compared methods on hyperspectral images.

<table>
<thead>
<tr>
<th>Data</th>
<th>AllMin</th>
<th>FTC</th>
<th>GeoG/G</th>
<th>pHOT</th>
<th>HalRTC</th>
<th>RUBIK</th>
<th>FCT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SS</td>
<td>TME</td>
<td>SS</td>
<td>TME</td>
<td>SS</td>
<td>TME</td>
<td>SS</td>
</tr>
<tr>
<td>S1</td>
<td>3</td>
<td>0.163</td>
<td>183</td>
<td>0.091</td>
<td>52</td>
<td>0.113</td>
<td>61</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.156</td>
<td>307</td>
<td>0.067</td>
<td>76</td>
<td>0.077</td>
<td>93</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.156</td>
<td>429</td>
<td>0.060</td>
<td>100</td>
<td>0.056</td>
<td>124</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.156</td>
<td>550</td>
<td>0.046</td>
<td>126</td>
<td>0.044</td>
<td>151</td>
</tr>
</tbody>
</table>

| S2   | 3      | 0.173 | 183    | 0.099 | 50     | 0.114 | 61  |
|      | 5      | 0.166 | 306    | 0.082 | 76     | 0.076 | 92  |
|      | 7      | 0.166 | 428    | 0.073 | 101    | 0.064 | 123 |
|      | 9      | 0.166 | 578    | 0.062 | 125    | 0.056 | 154 |

| S3   | 3      | 0.034 | 226    | 0.041 | 88     | 0.044 | 181 |
|      | 5      | 0.033 | 466    | 0.030 | 99     | 0.029 | 251 |
|      | 7      | 0.033 | 486    | 0.023 | 124    | 0.021 | 389 |
|      | 9      | 0.033 | 587    | 0.019 | 156    | 0.021 | 386 |

| S4   | 3      | 0.033 | 238    | 0.031 | 78     | 0.036 | 181 |
|      | 5      | 0.033 | 359    | 0.015 | 108    | 0.015 | 254 |
|      | 7      | 0.033 | 486    | 0.012 | 128    | 0.012 | 391 |
|      | 9      | 0.033 | 600    | 0.012 | 170    | 0.012 | 398 |

| S5   | 3      | 0.059 | 236    | 0.051 | 75     | 0.071 | 180 |
|      | 5      | 0.059 | 362    | 0.041 | 104    | 0.051 | 254 |
|      | 7      | 0.059 | 483    | 0.034 | 137    | 0.037 | 308 |
|      | 9      | 0.059 | 603    | 0.028 | 156    | 0.029 | 400 |

| S6   | 3      | 0.060 | 237    | 0.067 | 76     | 0.070 | 181 |
|      | 5      | 0.060 | 356    | 0.039 | 105    | 0.040 | 251 |
|      | 7      | 0.060 | 489    | 0.039 | 130    | 0.040 | 325 |
|      | 9      | 0.060 | 600    | 0.039 | 166    | 0.040 | 396 |

| S7   | 3      | 0.071 | 245    | 0.075 | 82     | 0.069 | 181 |
|      | 5      | 0.072 | 377    | 0.034 | 102    | 0.032 | 225 |
|      | 7      | 0.072 | 581    | 0.028 | 101    | 0.028 | 336 |
|      | 9      | 0.072 | 706    | 0.027 | 129    | 0.028 | 404 |

| S8   | 3      | 0.059 | 236    | 0.030 | 74     | 0.042 | 181 |
|      | 5      | 0.059 | 354    | 0.018 | 107    | 0.019 | 247 |
|      | 7      | 0.059 | 701    | 0.013 | 102    | 0.013 | 283 |
|      | 9      | 0.059 | 853    | 0.012 | 112    | 0.012 | 502 |

Figure 2: Visual results of the recovered 27th frame of scene 7 when OS is set to 3.

5.1 Hyperspectral Image Inpainting

A hyperspectral image is a tensor whose slices are photographs of the same scene under different wavelets. We adopt the dataset provided in [Foster et al., 2006], which contains images about eight different rural scenes taken under 33 various wavelets. To make all methods in our baseline applicable to the completion problem, we resize each hyperspectral image to a small dimension such that $r_1 = 306$, $r_2 = 402$, and $n_3 = 33$. Empirically, we treat these graphs as tensors of rank $r = (30,30,6)$. The observed pixels, or the training set, are sampled from the tensors uniformly at random. And the sample size is set to $|\Omega| = OS \times p$ in which $OS$ is so-called Over-Sampling ratio and $p = \sum_{i=1}^{3} (n_1 r_i - r_i^2) + r_1 r_2 r_3$ is the number of free parameters in a size $n$ tensor with rank $r$. In the observed entries, the mode-1 feature matrix is constructed by extracting the top-$(r_1+10)$ singular vectors from a matrix of size $n_1 \times 10r_1$ whose columns are sampled from the mode-1 fibers of the hyperspectral graphs. The recovery accuracy is measured by Normalized Root mean Square Error (NRSE) [Kressner et al., 2014]. All the compared methods are terminated when the training NRSE is less than 0.003 or iterate more than 300 epochs. We report the NRSE and CPU time of the compared methods in Tab. 2. From the table, we can see that the proposed method has much higher accuracy than the other solvers in our baseline. The empirical results also indicate that the sparser the observed pixels are the higher CGSI's improvement is on the recovery accuracy. The visual results of the 27th slices of recovered hyperspectral images of scene 7 are illustrated in Fig. 2.

5.2 Recommender System

In recommendation tasks, two datasets are considered: MovieLens 10M (ML10M) and MovieLens 20M (ML20M). Both datasets contain the rating history of users for items at specific moments. For both datasets, we partition the samples into 731 slices in terms of time stamp. Those slices have the identical time intervals. Accordingly, in the completion tasks for the two datasets are of sizes $71567 \times 10681 \times 731$ and $138493 \times 26744 \times 731$ respectively. In addition to the rating history, both datasets contain two extra files: one describes the genres of each movie, and the other contains tags of each
Table 3: Performance of the compared methods on Recommendation Tasks.

<table>
<thead>
<tr>
<th>dataset</th>
<th>rank</th>
<th>AltMin</th>
<th>FTC</th>
<th>GeomCG</th>
<th>gHOI</th>
<th>TFAI</th>
<th>CGSI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(4,4,4)</td>
<td>0.982</td>
<td>0.984</td>
<td></td>
<td>1.076</td>
<td>1.011</td>
<td>0.823</td>
</tr>
<tr>
<td>ML10M</td>
<td>(6,6,6)</td>
<td>0.968</td>
<td>0.984</td>
<td></td>
<td>0.835</td>
<td>0.826</td>
<td>0.848</td>
</tr>
<tr>
<td></td>
<td>(8,8,8)</td>
<td>1.010</td>
<td>0.822</td>
<td></td>
<td>1.076</td>
<td>1.011</td>
<td>0.814</td>
</tr>
<tr>
<td></td>
<td>(10,10,10)</td>
<td>1.147</td>
<td>0.824</td>
<td></td>
<td>1.076</td>
<td>1.011</td>
<td>0.810</td>
</tr>
<tr>
<td></td>
<td>(4,4,4)</td>
<td>1.061</td>
<td>0.822</td>
<td></td>
<td>1.050</td>
<td>1.029</td>
<td>0.818</td>
</tr>
<tr>
<td>ML20M</td>
<td>(6,6,6)</td>
<td>1.089</td>
<td>0.808</td>
<td></td>
<td>1.050</td>
<td>1.029</td>
<td>0.805</td>
</tr>
<tr>
<td></td>
<td>(8,8,8)</td>
<td>1.092</td>
<td>0.812</td>
<td></td>
<td>1.050</td>
<td>1.029</td>
<td>0.804</td>
</tr>
<tr>
<td></td>
<td>(10,10,10)</td>
<td>1.092</td>
<td>0.818</td>
<td></td>
<td>1.050</td>
<td>1.029</td>
<td>0.799</td>
</tr>
</tbody>
</table>

Figure 3: Accuracy of compared methods under different size of training set.

Figure 4: Effect of parameter $\alpha$ on the accuracy of CGSI.

movie. We construct a corpus that contains the text description of all movies from the genres descriptions and all the tags. The feature matrix is extracted from the above corpus by the latent semantic analysis (LSA) method. The processing is efficient since LSA is implemented via randomized SVD.

Various empirical studies are conducted to validate the performance of the proposed method. In the first scenario, we record the CPU time and the Root Mean Square Error (RMSE) outputted by the compared algorithms under different choices of multi-linear rank. In this scenario, for both datasets, 80% samples are chosen as training set, and the rest are left for testing. The results are listed in Tab. 3, which suggests that the proposed method outperforms all other solvers in terms of accuracy. For ML10M, our method uses significantly less CPU time than its competitors. In Fig. 3, we report another scenario, in which the percentage of training samples are varied from 10% to 70% and the rank parameter is fixed to (10, 10, 10). Experimental results in this figure indicate that our method has the lowest RMSE.

To show the impact of parameter $\alpha$ on the performance of our method, we depict the relation between RMSE and $\alpha$ in Fig. 4, where the rank parameter is set to (10, 10, 10), and percentage of training samples is set to 80%. From this Figure we can see that our method has higher accuracy than the vanilla Riemannian model’s solver FTC for a wide range of parameter choices.

6 Conclusion

In this paper, we exploit the side information to improve the accuracy of Riemannian tensor completion. A novel Riemannian model is proposed. To solve the model efficiently, we design a new Riemannian metric that implicitly induce an adaptive preconditioner for the solving procedure. Then, we devise a Riemannian conjugate gradient descent method using the well-tuned metric. Empirical results show that our solver outperforms the state-of-the-art.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China (Grant No: 61472347, 61672376, and 61672449).

References


