Data Min Knowl Disc (2012) 25:298–324
DOI 10.1007/s10618-012-0280-z

Tensor factorization using auxiliary information

Atsuhiro Narita · Kohei Hayashi · Ryota Tomioka · Hisashi Kashima

Received: 1 November 2011 / Accepted: 8 June 2012 / Published online: 11 July 2012

Abstract Most of the existing analysis methods for tensors (or multi-way arrays) only assume that tensors to be completed are of low rank. However, for example, when they are applied to tensor completion problems, their prediction accuracy tends to be significantly worse when only a limited number of entries are observed. In this paper, we propose to use relationships among data as auxiliary information in addition to the low-rank assumption to improve the quality of tensor decomposition. We introduce two regularization approaches using graph Laplacians induced from the relationships, one for moderately sparse cases and the other for extremely sparse cases. We also give present two kinds of iterative algorithms for approximate solutions: one based on an EM-like algorithms which is stable but not so scalable, and the other based on gradient-based optimization which is applicable to large scale datasets. Numerical experiments on tensor completion using synthetic and benchmark datasets show that the use of

Responsible editor: Dimitrios Gunopulos, Donato Malerba and Michalis Vazirgiannis.

A. Narita · K. Hayashi · R. Tomioka · H. Kashima (✉)
Department of Mathematical Informatics, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
e-mail: kashima@mist.i.u-tokyo.ac.jp

A. Narita
e-mail: atsuhiro_narita@mist.i.u-tokyo.ac.jp

K. Hayashi
e-mail: kohei_hayashi@mist.i.u-tokyo.ac.jp

R. Tomioka
e-mail: tomioka@mist.i.u-tokyo.ac.jp

H. Kashima
Basic Research Programs PRESTO, Synthesis of Knowledge for Information Oriented Society, Tokyo, Japan

auxiliary information improves completion accuracy over the existing methods based only on the low-rank assumption, especially when observations are sparse.

Keywords Tensors · Multi-way arrays · CP-decomposition · Tucker decomposition · Side information

1 Introduction

In real data analysis applications, we often have to face handling multi-object relationships. For example, in on-line marketing scenarios, we analyze relationships among customers, items, and time to capture temporal dynamics of customers' interests and utilize them for recommendation. In social network analysis, interactions among people and the types of interaction are the focus of interest. Similar situations arise in bio- and chemo-informatics as protein-protein interactions and drug-target interactions under various conditions. Tensors (or multi-way arrays) (Kolda and Bader 2009) are highly suitable representation for such multi-object relationships (Fig. 1). Tensor analysis methods, especially, models and efficient algorithms for low-rank tensor decompositions have been extensively studied and applied to many real-world problems. The CANDECOMP/PARAFAC(CP) decomposition and Tucker decomposition are two widely used low-rank decomposition methods for tensors (Fig. 2).

Tensor completion is one of important applications of tensor analysis methods. Given a tensor with some elements missing, the task is to impute the missing values. In the context of the previous on-line marketing scenarios, given the observations for some (customer, item, time)-tuples, we can make recommendations by imputing unobserved combinations of them. Tensor completion is also used for link prediction (Dunlavy et al. 2011; Kashima et al. 2009) and tag recommendation (Rendle and Thieme 2010). Similar to the other tensor analysis methods, low-rank assumption of tensors is often used for imputing missing values. However, when observations are sparse, that is, when the fraction of unobserved elements is high, predictive accuracy of

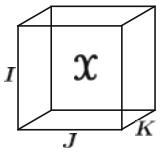
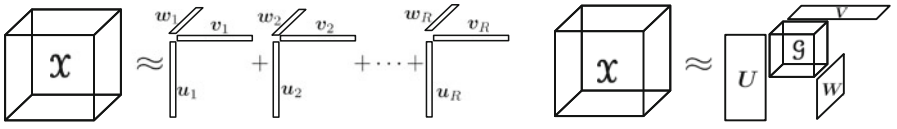


Fig. 1 A third-order tensor (or, a three-way array) $\mathcal{X}$ of size $I \times J \times K$ represents relationships among three sets of objects, S_1, S_2 and S_3, each of which size is I, J, and K, respectively



(a) CP decomposition. **(b)** Tucker decomposition.

Fig. 2 Two widely used low-rank tensor decompositions: CP-decomposition and Tucker decomposition

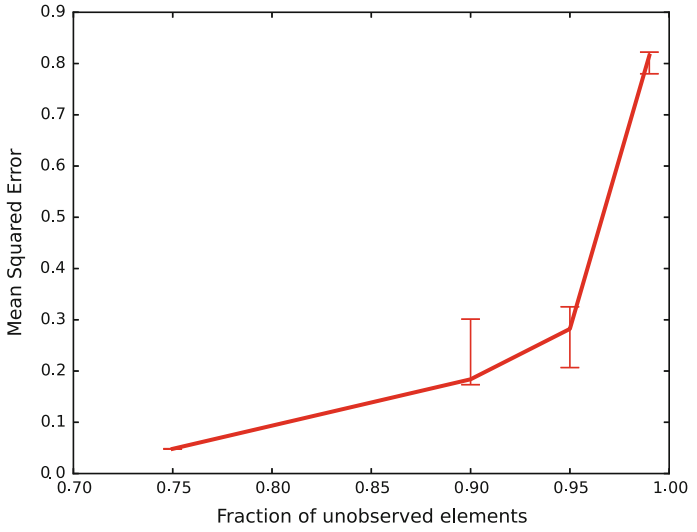


Fig. 3 The tensor completion performance by CP-decomposition for the 'Flow injection' dataset. Prediction accuracy severely degenerates when observations are sparse

tensor completion methods only with the low-rank assumption tends to be worse. For example, Fig. 3 shows the prediction errors by CP-decomposition against the fraction of unobserved elements for a particular dataset. (The experimental setting is the same as that for Fig. 4c, which will be described in detail in the experimental section.) We can see the accuracy of tensor completion severely degrades when observations are sparse. This fact implies that the low-rank assumption by itself is not sufficient and we need other assumptions to introduce more prior knowledge of subjects.

In many cases, we have not only relational information among objects, but also information on the objects themselves. For example, in the on-line marketing scenarios, each customer has his or her demographic information, and each item has its product information. We consider exploiting these auxiliary information for improving the prediction accuracy of tensor decomposition, especially in sparse cases. Inspired by the work by Li and Yeung (2009) which incorporates object similarity into matrix factorization, we exploit the auxiliary information given as similarity matrices in a regularization framework for tensor factorization. We propose two specific regularization methods, one of which we call "within-mode regularization" is a natural extension of the method proposed by Li et al. for matrix factorization. It uses the graph Laplacians induced by the similarity matrices to force two similar objects in each mode to behave similarly, in other words, to have similar factors. The second method we call "cross-mode regularization" exploits the similarity information more aggressively to address extremely sparse cases. We apply the two proposed regularization methods to each of CP-decomposition and Tucker decomposition. To best of our knowledge, our work is the first to incorporates auxiliary information into tensor decomposition.

To obtain tensor decompositions using the two regularization methods, we first give approximate iterative decomposition algorithms for fully-observed tensors. In each iteration of the algorithms, we solve a particular Sylvester equation for

CP-decomposition, or an eigen-problem for the Tucker decomposition. Next, to handle partially observed tensors, we give two tensor completion algorithms, one based on an EM-like algorithm and the other based on gradient-based optimization. In the former, unobserved elements are imputed with estimates by current decomposition, and we decompose the filled tensors with the decomposition algorithms for fully observed tensors. The EM-like algorithm is not a proper EM algorithm, but we also give its justification as a constrained optimization problem. The EM-like algorithm is stable, but not so efficient for large tensors because they work on completely filled tensors, therefore we devise a gradient-based optimization algorithm which decomposes tensors only with their observed parts. Decompositions are optimized by using L-BFGS updates for both of CP-decomposition and Tucker decomposition.

Finally, we show experimental results on missing value imputation using both synthetic and real benchmark datasets. We test two kinds of assumptions on missing elements. The first one is element-wise missing where each element is missing independently. The second one is slice-wise missing (in other words, object-wise missing), where missing values occur in a more bursty manner, and all of the elements related to some objects are completely missing. The experimental results demonstrate that the use of auxiliary information improves imputation accuracy in both cases when observations are sparse.

The rest of the paper is organized as follows. Section 2 reviews the existing low-rank tensor decomposition methods, and introduces the tensor completion problem with auxiliary information that we focus on in this paper. In Sect. 3, we propose two regularization strategies, within-mode regularization and cross-mode regularization, for incorporating auxiliary information in tensor decomposition. Section 4 gives approximation algorithms to obtain decompositions for fully observed tensors. Section 5 discusses decomposition algorithms for partially-observed tensors. We first give an EM-like approach to apply the decomposition algorithms proposed in Sect. 4 to partially-observed tensors, and its justification as a constrained optimization problem. We also introduce another family of algorithms using gradient-based optimization to address scalability issue in the EM-like approach. Section 6 shows the experimental results using several datasets to demonstrate the proposed methods work well especially when observations are sparse. Section 7 reviews related work, and Sect. 8 concludes this paper.

2 Tensor completion problem with auxiliary information

We first review the existing low-rank tensor decomposition methods, and then formulate the tensor completion problem with auxiliary information.

2.1 Tensor analysis using low-rank decomposition

Let $\mathcal{X}$ be third-order tensor (i.e. a three-way array) with $I \times J \times K$ real-valued elements.[1] The third-order tensor $\mathcal{X}$ models relationships among objects from three sets

[1] For simplicity, we focus on third-order tensors in this paper. However, the discussion can be directly applied to higher-order tensors.

S_1, S_2, and S_3. For example, in the context of on-line marketing, S_1, S_2, and S_3 represent sets of customers, items, and time stamps, respectively. The (i, j, k)th element $[\mathcal{X}]_{i,j,k}$ indicates the ith user's rating of the jth item at time k.

We often assume that the tensor is of "low-rank" when we analyze tensor-represented data. In contrast to matrices (that are special cases of tensors), definitions of the "low-rank" tensor are not unique. The CANDECOMP/PARAFAC(CP) decomposition and the Tucker decomposition are often used as definitions of low-rank tensors.

The CP-decomposition is a natural extension of the matrix rank, and it approximates a tensor by the sum of P rank-1 tensors. The CP-decomposition $\hat{\mathcal{X}}$ of $\mathcal{X}$ is defined as

$$\hat{\mathcal{X}} \equiv \sum_{p=1}^{P} \mathbf{u}_p \circ \mathbf{v}_p \circ \mathbf{w}_p,$$

where $\circ$ indicates the outer product operation. Alternatively, it can also be represented by using mode-i multiplications as

$$\hat{\mathcal{X}} = \mathcal{J} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}, \tag{1}$$

where $\mathcal{J} \in \mathbb{R}^{P \times P \times P}$ is a unit tensor with all superdiagonal elements having the value 1 and the other elements having the value 0. $\mathbf{U} \in \mathbb{R}^{I \times P}$, $\mathbf{V} \in \mathbb{R}^{J \times P}$, $\mathbf{W} \in \mathbb{R}^{K \times P}$ are factor matrices, and $\times_i$ is the mode-i multiplication (Kolda and Bader 2009). When the left-hand side is equal to the right-hand side in the above relation, we say $\mathcal{X}$ is of rank P.

The Tucker decomposition approximates a tensor with a small "core tensor" and factor matrices, which is defined as

$$\hat{\mathcal{X}} \equiv \mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}, \tag{2}$$

where $\mathcal{G}$ is a (P, Q, R)-tensor and $\mathbf{U} \in \mathbb{R}^{I \times P}$, $\mathbf{V} \in \mathbb{R}^{J \times Q}$, $\mathbf{W} \in \mathbb{R}^{K \times R}$ are factor matrices. In this case, we say $\mathcal{X}$ is of rank (P, Q, R).

For most of realistic case, observations are perturbed by noise, and the strict low-rank decompositions do not hold even when the "true" $\mathcal{X}$ is actually of low-rank. Therefore, we try to find a decomposition $\hat{\mathcal{X}}$ that best approximates the original tensor $\mathcal{X}$ in terms of the squared loss by the following optimization problem,

$$\text{minimize}_{\hat{\mathcal{X}}} \ \frac{1}{2}\|\mathcal{X} - \hat{\mathcal{X}}\|_F^2, \tag{3}$$

where $\|\cdot\|_F$ indicates the tensor Frobenius norm (Kolda and Bader 2009), and $\hat{\mathcal{X}}$ is defined by Eq. (1) for CP-decomposition, or by Eq. (2) for Tucker decomposition. It is generally hard to obtain the optimal solution, so we use approximation methods which optimize $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ alternately.

2.2 Tensor completion with auxiliary information

Among various tensor-related problems, the tensor completion problem is one of the important problems, where the task is to impute the missing values of a given tensor with missing values. The low-rank assumption is usually used as a heuristic for inferring the missing parts. Since not all of the elements are observed in the optimization problem (3) in this case, EM-like algorithms are often applied to this purpose (Srebro 2004; Walczak 2001). First, we fill in the missing values with some initial estimates (such as the average of the observed elements), and apply tensor decomposition to the filled tensor. We then obtain new estimates by assembling the decomposed tensor. We continue the decomposition step and the reconstruction step until convergence to obtain final estimates.

Since the EM-like algorithm uses unobserved elements for its computation, it is not efficient enough for large-scale data. Therefore, another approach modifies the objective function (3) to focus only on observed parts (Acar et al. 2010). In this paper, we give two kinds of algorithms based on the above two approaches.

The low-rank assumption of tensors makes it possible to impute missing values. However, when observations are sparse, that is the fraction of unobserved elements is high, predictive accuracy of tensor completion methods only with the low-rank assumption severely degrades. (See Fig. 3 showing the predictive errors against the fraction of unobserved elements for a dataset.) Therefore, the low-rank assumption by itself is not sufficient, and we need other assumptions for obtaining a satisfactory prediction accuracy.

In many realistic cases, we have not only relational information represented as tensors, but also information on the objects forming the relationships. For example, in the (customer, item, time)-relationships, each customer has his/her demographic information, and each item has its product information. We also know that time is continuous and can assume temporal smoothness. Therefore, we assume that we have similarity measures for S_1, S_2, and S_3, each of which corresponds to the sets of objects for each of the three modes. We define a non-negative symmetric matrix $\mathbf{A}_1$ for representing the similarity between two arbitrary objects in S_1. $\mathbf{A}_2$ and $\mathbf{A}_3$ are defined similarly.[2]

We consider exploiting these auxiliary information for improving the prediction accuracy by tensor decomposition, especially for sparse observations. The tensor completion problem that we focus on in this paper is summarized as follows.

Problem: (Third-order) tensor completion with auxiliary information

– **INPUT:**

- A third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$ with some elements having been observed and the others remaining unobserved.
- Three non-negative symmetric similarity matrices $\mathbf{A}_1 \in \mathbb{R}^{+I \times I}$, $\mathbf{A}_2 \in \mathbb{R}^{+J \times J}$, and $\mathbf{A}_3 \in \mathbb{R}^{+K \times K}$, each corresponding to one of the three modes of $\boldsymbol{\mathcal{X}}$.

[2] An alternative way to represent auxiliary information is to use object-attribute matrices. We use the similarity matrices since we sometimes want to use non-linear similarities created from the object-attribute matrices such as the Gaussian kernel, and since sometimes side information is readily represented as object networks (which are used as the similarity matrices under the assumption that two adjacent objects behave similarly.)

– **OUTPUT:** A decomposition $\hat{\mathcal{X}}$ defined by either Eq. (1) for CP-decomposition or Eq. (2) for Tucker decomposition, which is used to fill in the unobserved parts of $\mathcal{X}$.

3 Proposed methods: within-mode and cross-mode regularization

In this section, we propose two regularization methods for incorporating auxiliary information into tensor factorization. Both of the CP-decomposition and the Tucker decomposition are generalized with the regularization framework.

3.1 Regularization using auxiliary similarity matrices

Given three object similarity matrices $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{A}_3$ besides $\mathcal{X}$, how can we use them for improving tensor decompositions? Probably, one of natural assumptions we can make is that "two similar objects behave similarly". We implement this idea as regularization terms for the optimization problem (3). Namely, instead of the objective function in Eq. (3), we minimize the following regularized objective function with a regularization term $R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)$.

$$f(\hat{\mathcal{X}}) \equiv \frac{1}{2}\|\mathcal{X} - \hat{\mathcal{X}}\|_F^2 + \frac{\alpha}{2} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3). \tag{4}$$

In Eq. (4), α is a positive regularization constant.

We propose two specific choices of the regularization term $R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)$. The first method we call "within-mode regularization" is a natural extension of the method proposed by Li and Yeung (2009) for matrix factorization. It uses the graph Laplacians induced by the three similarity matrices to force two similar objects in each mode to behave similarly, in other words, to have similar factors. The second method we call "cross-mode regularization" exploits the similarity information more aggressively to address extremely sparse cases. It uses the graph Laplacian induced by the Kronecker product of the three similarity matrices to regularize factors for all the modes at the same time, while also taking interactions across different modes into account.

3.2 Method 1: Within-mode regularization

The first regularization term we propose regularizes factor matrices for each mode using the similarity matrices. The "within-mode" regularization term is defined as

$$R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) \equiv \mathrm{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right), \tag{5}$$

where $\mathbf{L}_1$ is the Laplacian matrix induced from the similarity matrix $\mathbf{A}_1$ for the object set S_1. The Laplacian matrix is defined as

$$\mathbf{L}_1 \equiv \mathbf{D}_1 - \mathbf{A}_1,$$

where $\mathbf{D}_1$ is the diagonal matrix whose ith diagonal element is the sum of all of the elements in the ith row of $\mathbf{A}_1$. The Laplacian matrices for the other two sets, $\mathbf{L}_2 \equiv \mathbf{D}_2 - \mathbf{A}_2$ and $\mathbf{L}_3 \equiv \mathbf{D}_3 - \mathbf{A}_3$, are defined similarly.

To interpret the regularization term, we note $\mathrm{tr}(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U})$ can be rewritten as

$$\mathrm{tr}(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}) = \sum_{i,j=1}^{I} [\mathbf{A}_1]_{i,j} \sum_{p=1}^{P} \left([\mathbf{U}]_{i,p} - [\mathbf{U}]_{j,p} \right)^2, \tag{6}$$

where $[\cdot]_{i,j}$ denotes the (i, j)th element of a matrix. This term implies that, if two objects (say, $s_i, s_j \in S_1$) are similar to each other (that is, $[\mathbf{A}_1]_{i,j}$ is large), the corresponding factor vectors ($[\mathbf{U}]_{i*}$ and $[\mathbf{U}]_{j*}$) should be similar to each other.

3.3 Proposed method 2: Cross-mode regularization

The within-mode regularization scheme regularizes only the elements inside each factor matrix, because each element of $\mathbf{U}$ interacts only with at most $I - 1$ elements within $\mathbf{U}$. This fact sometimes limits the effect of the regularization when we have bursty missing values, for example, slice-level missing situations where no observations are given for some objects often occurs in the context of recommender systems as the "cold-start" problem. In such cases, the within-mode regularization can sometimes be too conservative.

The second regularization function we propose exploits the given auxiliary information more aggressively. It combines the given similarity matrices to co-regularize combinations of elements across different modes as

$$R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) \equiv \mathrm{tr}\left((\mathbf{W} \otimes \mathbf{V} \otimes \mathbf{U})^\top \mathbf{L} \, (\mathbf{W} \otimes \mathbf{V} \otimes \mathbf{U}) \right), \tag{7}$$

where the $IJK \times IJK$-Laplacian matrix $\mathbf{L}$ is defined as

$$\mathbf{L} \equiv \mathbf{D}_3 \otimes \mathbf{D}_2 \otimes \mathbf{D}_1 - \mathbf{A}_3 \otimes \mathbf{A}_2 \otimes \mathbf{A}_1.$$

The regularization term Eq. (7) is rewritten with the matrix elements as

$$R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) = \sum_{i,j,k=1}^{I,J,K} \sum_{\ell,m,n=1}^{I,J,K} [\mathbf{A}_1]_{i,\ell} [\mathbf{A}_2]_{j,m} [\mathbf{A}_3]_{k,n} \sum_{p,q,r=1}^{P,Q,R} \left([\mathbf{U}]_{i,p} [\mathbf{V}]_{j,q} [\mathbf{W}]_{k,r} - [\mathbf{U}]_{\ell,p} [\mathbf{V}]_{m,q} [\mathbf{W}]_{n,r} \right)^2,$$

which regularizes the combinations of elements from three different factors unlike the within-mode regularization (6) considering each mode independently.

The cross-mode regularization (7) can be seen as a natural variant of the within-mode regularization (5). Because, if we use the Kronecker sum $\oplus$ instead of the

Kronecker product $\otimes$ in Eq. (7), it is reduced to Eq. (5) under the orthonormality constraints. The intuition behind the cross-mode regularization is "two similar object triplets behave similarly", which eventually induces more dense similarity among tensor elements to address "cold-start" situations.

4 Algorithms for fully observed tensors

We propose algorithms for minimizing the objective function (4) with the within-mode or cross-mode regularization for fully observed $\mathcal{X}$. The algorithms are not directly applicable to the tensor completion problem, and their modification for application to partially observed tenors will be discussed in Sect. 5.

4.1 Algorithms for within-mode regularization

4.1.1 CP-decomposition

The objective function for CP-decomposition with the within-mode regularization is written as

$$f(\mathcal{J}, \mathbf{U}, \mathbf{V}, \mathbf{W}) \equiv \frac{1}{2}\|\mathcal{X} - \mathcal{J} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 + \frac{\alpha}{2}\mathrm{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right). \tag{8}$$

Equation (8) is not a convex function for $(\mathbf{U}, \mathbf{V}, \mathbf{W})$, but is convex for each of $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$. Therefore, we optimize one of $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ with fixing the others to the current values, and alternately update them by changing the factor matrix to optimize.

Suppose we want to optimize $\mathbf{U}$ by fixing $\mathbf{V}$ and $\mathbf{W}$. Unfolding Eq. (8) by the first mode (i.e. making the mode-1 matricization), we obtain

$$\begin{aligned} f(\mathcal{J}, \mathbf{U}, \mathbf{V}, \mathbf{W}) =& \frac{1}{2}\|\mathbf{X}_{(1)} - \mathbf{U}\,(\mathbf{W} \odot \mathbf{V})^\top\|_F^2 \\ &+ \frac{\alpha}{2}\mathrm{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right) \\ =& \frac{1}{2}\mathrm{tr}\left(\left(\mathbf{X}_{(1)} - \mathbf{U}(\mathbf{W} \odot \mathbf{V})^\top\right)^\top \left(\mathbf{X}_{(1)} - \mathbf{U}(\mathbf{W} \odot \mathbf{V})^\top\right)\right) \\ &+ \frac{\alpha}{2}\mathrm{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right), \end{aligned}$$

where $\mathbf{X}_{(n)}$ denotes the mode-n matricization of $\mathcal{X}$, and $\odot$ denotes the Khatri-Rao product (Kolda and Bader 2009). Differentiating this with respect to $\mathbf{U}$, and setting it to be zero gives the Sylvester equation,

$$\mathbf{U}(\mathbf{W} \odot \mathbf{V})^\top (\mathbf{W} \odot \mathbf{V}) + \alpha \mathbf{L}_1 \mathbf{U} = \mathbf{U}\left(\mathbf{V}^\top \mathbf{V} * \mathbf{W}^\top \mathbf{W}\right) + \alpha \mathbf{L}_1 \mathbf{U} = \mathbf{X}_{(1)}(\mathbf{W} \odot \mathbf{V}),$$

where $*$ denotes the Hadamard product (i.e. element-wise product). The Sylvester equation can be solved by several numerical approaches such as the one implemented as the `dlyap` function in MATLAB®.

4.1.2 Tucker decomposition

In the case of the Tucker decomposition, the objective function becomes

$$f(\mathcal{G}, \mathbf{U}, \mathbf{V}, \mathbf{W}) \equiv \frac{1}{2}\|\mathcal{X} - \mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 + \frac{\alpha}{2}\left(\operatorname{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}\right) + \operatorname{tr}\left(\mathbf{V}^\top \mathbf{L}_2 \mathbf{V}\right) + \operatorname{tr}\left(\mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right) \right). \quad (9)$$

We minimize this objective function (9) under the orthonormality constraints, $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$, $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$, and $\mathbf{W}^\top\mathbf{W} = \mathbf{I}$. Noting the core tensor $\mathcal{G}$ is obtained as the closed form solution,

$$\mathcal{G} = \mathcal{X} \times_1 \mathbf{U}^\top \times_2 \mathbf{V}^\top \times_3 \mathbf{W}^\top,$$

the first term of Eq. (9) can be rewritten as

$$\begin{aligned}\frac{1}{2}\|\mathcal{X} - \mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 &= \frac{1}{2}\left(\|\mathcal{X}\|_F^2 - \|\mathcal{G}\|_F^2\right) \\ &= \frac{1}{2}\left(\|\mathcal{X}\|_F^2 - \|\mathcal{X} \times_1 \mathbf{U}^\top \times_2 \mathbf{V}^\top \times_3 \mathbf{W}^\top\|_F^2\right).\end{aligned}$$

When we optimize Eq. (9) with respect to $\mathbf{U}$, by ignoring the terms unrelated to $\mathbf{U}$, we obtain an equivalent maximization problem,

$$\tilde{f}(\mathbf{U}) \equiv \|\mathcal{X} \times_1 \mathbf{U}^\top \times_2 \mathbf{V}^\top \times_3 \mathbf{W}^\top\|_F^2 - \alpha \operatorname{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}\right). \quad (10)$$

Unfolding Eq. (10) by the first mode, we have

$$\tilde{f}(\mathbf{U}) = \|\mathbf{U}^\top \mathbf{X}_{(1)} (\mathbf{W} \otimes \mathbf{V})\|_F^2 - \alpha \operatorname{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}\right).$$

Setting $\mathbf{S} \equiv \mathbf{X}_{(1)} (\mathbf{W} \otimes \mathbf{V}) = (\mathcal{X} \times_2 \mathbf{V} \times_3 \mathbf{W})_{(1)}$, $\tilde{f}(\mathbf{U})$ is further rewritten as

$$\tilde{f}(\mathbf{U}) = \|\mathbf{U}^\top \mathbf{S}\|_F^2 - \alpha \operatorname{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}\right) = \operatorname{tr}\left(\mathbf{U}^\top \left(\mathbf{S}\mathbf{S}^\top - \alpha \mathbf{L}_1\right) \mathbf{U}\right). \quad (11)$$

The maximizer of Eq. (11) satisfying the orthonormality constraint $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ is obtained as the P leading eigenvectors of $\mathbf{S}\mathbf{S}^\top - \alpha\mathbf{L}_1$.

4.2 Algorithms for cross-mode regularization

4.2.1 CP-decomposition

The objective function for the cross-mode regularized CP-decomposition is defined as

$$f(\mathbf{U},\mathbf{V},\mathbf{W}) \equiv \frac{1}{2}\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{J}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 + \frac{\alpha}{2}\mathrm{tr}\left((\mathbf{W}\otimes\mathbf{V}\otimes\mathbf{U})^\top \mathbf{L}\,(\mathbf{W}\otimes\mathbf{V}\otimes\mathbf{U})\right).$$

Noting that Eq. (7) is simplified as

$$R(\hat{\boldsymbol{\mathcal{X}}};\mathbf{A}_1,\mathbf{A}_2,\mathbf{A}_3) = \mathrm{tr}\left(\mathbf{W}^\top\mathbf{D}_3\mathbf{W}\right)\mathrm{tr}\left(\mathbf{V}^\top\mathbf{D}_2\mathbf{V}\right)\mathrm{tr}\left(\mathbf{U}^\top\mathbf{D}_1\mathbf{U}\right) - \mathrm{tr}\left(\mathbf{W}^\top\mathbf{A}_3\mathbf{W}\right)\mathrm{tr}\left(\mathbf{V}^\top\mathbf{A}_2\mathbf{V}\right)\mathrm{tr}\left(\mathbf{U}^\top\mathbf{A}_1\mathbf{U}\right),$$

similar to the within-mode regularization, we obtain the Sylvester equation for $\mathbf{U}$ as

$$\mathbf{U}(\mathbf{W}\odot\mathbf{V})^\top(\mathbf{W}\odot\mathbf{V}) + (D_{VW}\mathbf{D}_1 - A_{VW}\mathbf{A}_1)\,\mathbf{U} = \mathbf{X}_{(1)}(\mathbf{W}\odot\mathbf{V}),$$

where D_{VW} and A_{VW} are defined as follows:

$$D_{VW} \equiv \mathrm{tr}\left(\mathbf{W}^\top\mathbf{D}_3\mathbf{W}\right)\mathrm{tr}\left(\mathbf{V}^\top\mathbf{D}_2\mathbf{V}\right) \tag{12}$$

$$A_{VW} \equiv \mathrm{tr}\left(\mathbf{W}^\top\mathbf{A}_3\mathbf{W}\right)\mathrm{tr}\left(\mathbf{V}^\top\mathbf{A}_2\mathbf{V}\right) \tag{13}$$

4.2.2 Tucker decomposition

The objective function for the cross-mode regularized Tucker decomposition is defined as

$$f(\boldsymbol{\mathcal{G}},\mathbf{U},\mathbf{V},\mathbf{W}) \equiv \frac{1}{2}\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{G}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 + \frac{\alpha}{2}\mathrm{tr}\left((\mathbf{W}\otimes\mathbf{V}\otimes\mathbf{U})^\top \mathbf{L}\,(\mathbf{W}\otimes\mathbf{V}\otimes\mathbf{U})\right).$$

Again, we alternately minimize the objective function with respect to one of $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$. By a derivation similar to that for the within-mode regularization, the optimal $\mathbf{U}$ with $\mathbf{V}$ and $\mathbf{W}$ fixed is obtained as the P leading eigenvectors of

$$\mathbf{S}\mathbf{S}^\top - \alpha\,(D_{VW}\mathbf{D}_1 - A_{VW}\mathbf{A}_1).$$

Table 1 Comparison of time complexities for various decompositions

Decomposition type	Regularization type	Time complexity
CP decomposition	–	$I^2P + IP^2 + P^3 + IJKP$
	Within-mode	$N_1IP^2 + I^2P^3 + IJKP$
	Cross-mode	$N_1IP^2 + I^2P^3 + IJKP$
Tucker decomposition	–	$IP^2 + I^2QR + IJKP$
	Within-mode	$IP^2 + I^2QR + IJKP$
	Cross-mode	$IP^2 + I^2QR + IJKP$

4.3 Computational complexity

We analyze the impact of using auxiliary information in tensor decomposition on the computational complexity. Let the number of non-zero entries in the ith similarity matrix $\mathbf{A}_i$ be N_i. The time complexities for obtaining the optimal $\mathbf{U}$ given $\mathbf{V}$ and $\mathbf{W}$ are summarized in Table 1. (The time complexities for updating $\mathbf{V}$ or $\mathbf{W}$ are obtained similarly.)

In the case of the CP-decomposition with auxiliary information, we use the conjugate gradient method to solve the Sylvester equations. The time complexity is the same for both of the within-mode regularization and the cross-mode regularization, which is slightly larger than that for the plain CP-decomposition using the LU decomposition.

In the Tucker decomposition, we need to solve the top-R eigen-decomposition problem (of size N), which can be done in $O(RN^2)$ by using the Lanczos method. The time complexity is the same for all approaches with and without auxiliary information.

As for the space complexity, the our methods require $\mathrm{O}(IJK + IP + JQ + KR + N_1 + N_2 + N_3)$ space, which is slightly larger than that for the ordinary decomposition.

5 Tensor completion algorithms for partially observed tensors

The algorithms proposed in the previous section are not directly applicable to the tensor completion problem where some parts of the target tensor $\mathcal{X}$ are missing. In this section, we first modify the previous algorithms by applying an EM-like procedure in which the decomposition algorithms are repeatedly called, so that they can be used to solve the completion problem.

Since the EM-like algorithm is not efficient for large tensors, we also give more efficient algorithms based on gradient-based optimization that which scales linearly with the number of observed elements.

5.1 An EM-like approach

The algorithms proposed in Sect. 4 assume fully-observed tensors, and hence are not directly applicable to the tensor completion problem where some parts of $\mathcal{X}$ are missing. Therefore, EM-like algorithms are often applied to this purpose (Srebro 2004; Walczak 2001).

To apply the algorithms for fully-observed tensors, we first prepare a new tensor $\mathcal{Z}$ as a proxy target, which is a modified tensor of $\mathcal{X}$ by filling its missing parts with some initial estimate ϵ. For example, ϵ can be the average of all observed elements. We apply a tensor decomposition algorithm to the completely filled tensor $\mathcal{Z}$, and obtain a new decomposition, which is then re-assembled to obtain $\hat{\mathcal{X}}$. The $\hat{\mathcal{X}}$ is used to fill in the missing parts of $\mathcal{X}$ again, which results in an updated $\mathcal{Z}$. We continue this procedure until convergence.

The algorithm is summarized in Algorithm 1, where a binary tensor $\mathcal{O}$ indicates indices of the observed elements in $\mathcal{X}$. The (i, j, k)th element $[\mathcal{O}]_{i,j,k}$ represents if the (i, j, k)th element in $\mathcal{X}$ is observed (1) or not (0), namely,

$$[\mathcal{O}]_{i,j,k} = \begin{cases} 1 & (\text{if } [\mathcal{X}]_{i,j,k} \text{ is observed}), \\ 0 & (\text{otherwise}). \end{cases}$$

Algorithm 1 An EM-like tensor completion

Initialize $[\mathcal{Z}]_{i,j,k} \leftarrow \begin{cases} [\mathcal{X}]_{i,j,k} & (\text{if } [\mathcal{O}]_{i,j,k} = 1) \\ \epsilon & (\text{if } [\mathcal{O}]_{i,j,k} = 0) \end{cases}$ for $\forall(i, j, k)$

while convergence **do**

$\hat{\mathcal{X}} \leftarrow \text{Decompose}(\mathcal{Z})$

$[\mathcal{Z}]_{i,j,k} \leftarrow \begin{cases} [\mathcal{X}]_{i,j,k} & (\text{if } [\mathcal{O}]_{i,j,k} = 1) \\ [\hat{\mathcal{X}}]_{i,j,k} & (\text{if } [\mathcal{O}]_{i,j,k} = 0) \end{cases}$ for $\forall(i, j, k)$

end while

Strictly speaking, this EM-like algorithm is not a proper EM-algorithm. Here we give a justification of the algorithm as a block coordinate descent algorithm for a particular constrained optimization problem. Let us consider the following optimization problem with two tensor parameters $\mathcal{Z}$ and $\hat{\mathcal{X}}$.

$$\begin{aligned} &\text{minimize}_{\mathcal{Z},\hat{\mathcal{X}}} \ \frac{1}{2}\|\mathcal{Z} - \hat{\mathcal{X}}\|_F^2 + \frac{\alpha}{2} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) \\ &\text{s.t.} \quad \mathcal{O} * (\mathcal{Z} - \mathcal{X}) = \mathbf{0} \end{aligned} \tag{14}$$

The constraint (14) forces the observed parts of $\mathcal{X}$ to be equal to the corresponding parts of $\mathcal{Z}$. This optimization problem can be approximately solved by using the block coordinate descent. Once $\mathcal{Z}$ is fixed, the optimization problem with respect to $\hat{\mathcal{X}}$ is unconstrained, and we can apply the tensor decomposition algorithms to $\mathcal{Z}$. On the other hand, with fixed $\hat{\mathcal{X}}$, the solution for the optimization problem with respect to $\mathcal{Z}$ is simply given as

$$[\mathcal{Z}]_{i,j,k} = \begin{cases} [\mathcal{X}]_{i,j,k} & (\text{if } [\mathcal{O}]_{i,j,k} = 1) \\ [\hat{\mathcal{X}}]_{i,j,k} & (\text{if } [\mathcal{O}]_{i,j,k} = 0) \end{cases} \text{ for all } (i, j, k).$$

This alternative optimization of $\mathcal{Z}$ and $\hat{\mathcal{X}}$ with some initial estimate of $\mathcal{Z}$ is exactly what Algorithm 1 does.

5.2 Gradient-based optimization approach

The EM-like algorithm is not efficient for large tensors because of two reasons. One is that it works on dense tensors whose missing elements have been imputed with current estimates, and the other is that we need to call tensor decomposition algorithms many times until convergence. To cope with this scalability issue, we give more efficient algorithms that work only on observed parts by following Acar et al. (2010).

5.2.1 CP decomposition

Focusing only on the observed part, we have a new objective function,

$$f(\hat{\mathcal{X}}) \equiv \frac{1}{2}\|\mathcal{O} * \left(\mathcal{X} - \hat{\mathcal{X}}\right)\|_F^2 + \frac{\alpha}{2} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3), \tag{15}$$

instead of the objective function (4) for fully observed cases. Note that $*$ indicates the Hadamard product (that is, element-wise product).

If we assume the CP-decomposition, the objective function (15) is rewritten as

$$\begin{aligned} f(\hat{\mathcal{X}}) &= \frac{1}{2}\|\mathcal{O} * (\mathcal{X} - \mathcal{J} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W})\|_F^2 + \frac{\alpha}{2} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) \\ &= \frac{1}{2} \sum_{i,j,k=1}^{I,J,K} \left([\mathcal{X}]_{ijk} - \sum_{p=1}^{P} [\mathcal{O}]_{i,j,k} [\mathbf{U}]_{i,p} [\mathbf{V}]_{j,p} [\mathbf{W}]_{k,p} \right)^2 + \frac{\alpha}{2} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3). \end{aligned}$$

Similar to the fully observed cases, this objective function is not jointly convex for $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$, while it is convex function with respect to each of them. Thus, we iterate optimization with respect to one of them with the others fixed. Let us assume that we want to optimize this with respect to $\mathbf{U}$. We employ gradient-based optimization approaches such as L-BFGS. Taking the gradient of the objective function, we obtain

$$\begin{aligned} \frac{\partial f}{\partial [\mathbf{U}]_{i,p}} = &- \sum_{j,k=1}^{J,K} \left([\mathcal{O}]_{i,j,k} \left([\mathcal{X}]_{i,j,k} - [\hat{\mathcal{X}}]_{i,j,k} \right) [\mathbf{V}]_{j,p} [\mathbf{W}]_{k,p} \right) \\ &+ \frac{\alpha}{2} \frac{\partial}{\partial [\mathbf{U}]_{i,p}} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3), \end{aligned}$$

or

$$\frac{\partial f}{\partial \mathbf{U}} = -\left(\mathbf{O}_{(1)} * \left(\mathbf{X}_{(1)} - \hat{\mathbf{X}}_{(1)}\right)\right)(\mathbf{W} \odot \mathbf{V}) + \frac{\alpha}{2} \frac{\partial}{\partial \mathbf{U}} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) \tag{16}$$

in a more concise form using matrices. The key to scalability is the computation of the first term of the gradient (16). Unlike previous methods, we can perform the update only with the observed parts by skipping the unobserved elements in the summation.

5.2.2 Tucker decomposition

When we assume the Tucker-decomposition, the objective function (15) becomes

$$
\begin{aligned}
f(\hat{\mathcal{X}}) =& \frac{1}{2}\|\mathcal{O} * (\mathcal{X} - \mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W})\|_F^2 + \frac{\alpha}{2} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) \\
=& \frac{1}{2} \sum_{i,j,k=1}^{I,J,K} [\mathcal{O}]_{i,j,k} \left([\mathcal{X}]_{i,j,k} - \sum_{p,q,r=1}^{P,Q,R} [\mathcal{G}]_{p,q,r} [\mathbf{U}]_{i,p} [\mathbf{V}]_{j,q} [\mathbf{W}]_{k,r} \right)^2 \\
& + \frac{\alpha}{2} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3).
\end{aligned}
$$

Unlike the case of Tucker-type decompositions for fully-observed tensors, we do not assume orthonormality of $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ that Tucker-type decompositions usually assume. Actually, there are several learning methods that can appropriately handle the orthonormality assumption by using Stiefel and Grassman manifolds (Nishimori and Akaho year; Plumbley 2005). However, we do not employ them because it is simpler to implement without the orthonormality assumption.

To solve the optimization problem, we again resort to the gradient-based approaches. The gradient of the objective function with respect to $\mathbf{U}$ is given as

$$
\begin{aligned}
\frac{\partial f}{\partial [\mathbf{U}]_{ip}} =& - \sum_{j,k=1}^{J,K} \left([\mathcal{O}]_{i,j,k} \left([\mathcal{X}]_{i,j,k} - [\hat{\mathcal{X}}]_{i,j,k} \right) \sum_{q,r=1}^{Q,R} [\mathcal{G}]_{p,q,r} [\mathbf{V}]_{j,q} [\mathbf{W}]_{k,r} \right) \\
& + \frac{\alpha}{2} \frac{\partial}{\partial [\mathbf{U}]_{i,p}} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3),
\end{aligned}
$$

or in the matrix form

$$
\frac{\partial f}{\partial \mathbf{U}} = - \left(\mathbf{O}_{(1)} * \left(\mathbf{X}_{(1)} - \hat{\mathbf{X}}_{(1)} \right) \right) (\mathbf{W} \otimes \mathbf{V}) \mathbf{G}_{(1)}^{\top} + \frac{\alpha}{2} \frac{\partial}{\partial \mathbf{U}} R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3). \quad (17)
$$

Because we do not assume the orthonormality, the optimal solution for the core tensor $\mathcal{G}$ is not given in a closed form. Therefore, we also need the gradient with respect to $\mathcal{G}$, which is given as

$$
\frac{\partial f}{\partial [\mathcal{G}]_{p,q,r}} = - \sum_{i,j,k=1}^{I,J,K} \left([\mathcal{O}]_{i,j,k} \left([\mathcal{X}]_{i,j,k} - [\hat{\mathcal{X}}]_{i,j,k} \right) [\mathbf{U}]_{i,p} [\mathbf{V}]_{j,q} [\mathbf{W}]_{k,r} \right),
$$

or concisely,

$$
\frac{\partial f}{\partial \mathcal{G}} = - \left(\mathcal{O} * \left(\mathcal{X} - \hat{\mathcal{X}} \right) \right) \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}. \quad (18)
$$

Table 2 Comparison of time complexities of computing a gradient for gradient-based optimization

Decomposition type	Regularization type	Time complexity
CP decomposition	–	MR
	Within-mode	$MR + N_1 P$
	Cross-mode	$MR + (N_1 + I)P$
Tucker decomposition	–	$MPQR$
	Within-mode	$MPQR + N_1 P$
	Cross-mode	$MPQR + (N_1 + I)P$

5.2.3 Regularization terms

In the gradients of the objective functions for CP-decomposition (16) and Tucker decomposition (17), we need the gradient of the regularizer $R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)$. When we use the within-mode regularization (5), the gradient of $R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)$ with respect to $\mathbf{U}$ is given as

$$\frac{\partial R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)}{\partial \mathbf{U}} = \mathbf{L}_1 \mathbf{U}.$$

If we use the cross-mode regularization (7), the gradient is given as

$$\frac{\partial R(\hat{\mathcal{X}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)}{\partial \mathbf{U}} = (D_{VW}\mathbf{D}_1 - A_{VW}\mathbf{A}_1)\,\mathbf{U},$$

where D_{VW} and A_{VW} are defined in Eqs. (12) and (13), respectively. The gradients with respect to $\mathbf{V}$ and $\mathbf{W}$ can be obtained similarly, and now we can calculate the gradients (16) and (17) to use gradient-based updates. In our experiments in Sect. 6, we employ the L-BFGS method.

5.2.4 Computational complexity

Table 2 summarizes the time complexities of computing the gradients with respect to $\mathbf{U}$ for gradient-based optimization. The gradients for $\mathbf{V}$ and $\mathbf{W}$ are obtained similarly, and we need $\mathrm{O}(MPQR)$ for computing the gradient for $\mathcal{G}$ in the Tucker decomposition. The important observation is that the time complexity of evaluating the gradients is linear with the number of observed elements M, which is a major advantage over the EM-like approach. The use of auxiliary information slightly increases the computational cost linearly depending on the numbers of non-zero elements in the similarity matrices. The Tucker decomposition requires more computational cost, but it still remains linear with respect to the numbers of the objects.

6 Experiments

We show the results of our numerical experiments of third-order tensor completion problems using synthetic and real benchmark datasets, and demonstrate that introducing auxiliary information improves predictive accuracy especially when observations are sparse.

6.1 Datasets

6.1.1 Synthetic dataset

The first dataset consists of synthetic tensors with correlated objects. We generate CP-decomposed tensors with $\mathbf{U} \in \mathbb{R}^{I \times R}$, $\mathbf{V} \in \mathbb{R}^{J \times R}$ and $\mathbf{W} \in \mathbb{R}^{K \times R}$ with the rank $R \equiv 2$ and $I \equiv J \equiv K \equiv 30$ by using the linear formulae,

$$\begin{aligned}
[\mathbf{U}]_{ir} &\equiv i\epsilon_r + \epsilon'_r \ (1 \leq i \leq I, 1 \leq r \leq R) \\
[\mathbf{V}]_{jr} &\equiv j\zeta_r + \zeta'_r \ (1 \leq j \leq J, 1 \leq r \leq R) \\
[\mathbf{W}]_{kr} &\equiv k\eta_r + \eta'_r \ (1 \leq k \leq K, 1 \leq r \leq R),
\end{aligned}$$

where $\{\epsilon_r, \epsilon'_r, \zeta_r, \zeta'_r, \eta_r, \eta'_r\}_{r=1}^{R}$ are constants generated by using the standard Gaussian distribution. A synthetic tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$ is defined as

$$\boldsymbol{\mathcal{X}} \equiv \boldsymbol{\mathcal{T}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}.$$

Since the columns of each factor matrix are generated by linear functions, the consecutive rows are similar to each other. Therefore, the similarity matrix for the ith mode is naturally defined as the tri-diagonal matrix.

$$\mathbf{A}_i \equiv \begin{bmatrix} 0 & 1 & 0 & \cdots \\ 1 & 0 & 1 & \cdots \\ 0 & 1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

6.1.2 Benchmark dataset 1: flow injection

As a real benchmark dataset with auxiliary information, we used the 'Rank-deficient spectral FIA dataset',[3] which consists of results of flow injection analysis on 12 different chemical substances. The results are represented as a tensor of size 12 (substances)$\times$100 (wavelengths)$\times$89 (reaction times).

We constructed three similarity matrices for the three modes as follows. Since 12 chemical substances differ with regard to the content of three structural isomers of

[3] The datasets are available from http://www.models.life.ku.dk/datasets.

a certain chemical compound, each substance can be represented as a three-dimensional feature vector. We defined the similarity between two substances as the inverse of Euclidean distance between their feature vectors. Also, since wavelength and reaction time have continuous real values, we simply set the similarity of two consecutive wavelength values (or reaction time values) to one since they are equally-spaced.

Actually, the design of the similarity matrix directly affects the quality of decompositions. The main reason for our choice of the similarity matrices for wavelength and reaction time is that we want to make them sparse, since the density of the similarity matrices directly affects the computational time of matrix multiplications. As for the similarity matrix for the chemical substances, we use the dense matrix without any sparsification since it is small.

6.1.3 Benchmark dataset 2: licorice

Another benchmark dataset we use is the 'Three-way electronic nose dataset',[2] which consists of measurements of an odor sensing system applied to licorices for checking their quality, and is represented as a third-order tensor of size 18 (samples)$\times$241 (reaction times)$\times$12 (sensors).

Since each of 18 samples is labeled with one of the three quality labels, {'BAD', 'FBAD','GOOD'}, we set the similarity between two samples sharing an identical label to one. The similarity for reaction time is defined in the same way as for the flow injection dataset. Eventually, we obtain two similarity matrices for this dataset.

6.1.4 Benchmark dataset 3: delicious

The third dataset 'hetrec2011-delicious-2k' is a relatively large dataset provided at the Hetrec 2011 workshop.[4] This dataset is a collection of social bookmarks obtained from Delicious social bookmarking system.[5] We used a third-order binary tensor representing (user, URL, tag)-relationships. Its original size is $108,035 \times 107,253 \times 52,995$; however, we eliminate slices with no observed elements and obtain a third-order tensor of size $1,867 \times 69,223 \times 40,897$ with 437, 593 observed elements. Note that, unlike the other datasets whose elements are completely filled, this dataset is originally sparse.

We constructed two similarity matrices, one for URLs, and the other is for tags. We defined the URL similarities as the identity of domains. The tag similarities were defined as similarities in tag names, specifically, the number of common prefix characters.

Note that the EM-like approach is not scalable enough to run on this dataset, and we used this dataset only for the gradient-based approach.

[4] The dataset is available from http://www.grouplens.org/node/462.

[5] http://www.delicious.com.

6.2 Experimental settings

6.2.1 Comparison methods

We compared the following 3 (regularization methods)×2 (decomposition models) ×2 (algorithms) = 12 methods.

1. Ordinary {CP, Tucker}-decomposition using the {EM-like, gradient-based} optimization,
2. Within-mode regularized {CP, Tucker}-decomposition using the {EM-like, gradient-based} optimization,
3. Cross-mode regularized {CP, Tucker}-decomposition using the {EM-like, gradient-based} optimization.

In the EM-like algorithms, we updated the missing value estimates every time one of the factor matrices is updated (Walczak 2001), since it converged faster. We set the initial estimates for the unobserved elements of $\mathcal{X}$ to the average of the observed elements. In all of the methods, we initialized $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ as the leading eigenvectors of $\mathbf{X}_{(1)}$, $\mathbf{X}_{(2)}$, and $\mathbf{X}_{(3)}$, respectively.

On the basis of the results of preliminary experiments, we set the model ranks as $P \equiv Q \equiv R \equiv 2$ for the synthetic dataset, $P \equiv Q \equiv R \equiv 4$ for the flow injection dataset, and $P \equiv Q \equiv R \equiv 3$ for the licorice dataset. For the Delicious dataset, we set $P \equiv 4$ for the CP-decomposition, and $P \equiv Q \equiv R \equiv 2$ for the Tucker decomposition. The hyper-parameter α was selected from $\left\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\right\}$ by using cross-validation. For the gradient-based methods, we also used small regularization terms for factors, namely, $\beta\|\mathbf{U}\|_F^2$ with $\beta \equiv 10^{-11}$ for $\mathbf{U}$ (and for $\mathbf{V}$ and $\mathbf{W}$, similarly).

6.2.2 Element-wise missing versus slice-wise missing

We test two kinds of assumptions on missing elements, that are, element-wise missing and slice-wise missing. In the element-wise missing setting, individual elements $[\mathcal{X}]_{i,j,k}$ are missing. On the other hand, in the slice-wise missing setting, missing values occur at object level, and therefore all of the elements related to some objects are totally missing. In other words, slices such as $\{[\mathcal{X}]_{i,j,k}\}_{j,k}$ for some i, $\{[\mathcal{X}]_{i,j,k}\}_{i,k}$ for some j, and $\{[\mathcal{X}_{i,j,k}]\}_{i,j}$ for some k are missing, which means that missing values occurring in a more bursty manner.

We varied the fraction of unobserved elements among {0.75, 0.9, 0.95, 0.99} for the element-wise missing setting, and among {0.5, 0.75, 0.9, 0.95} for the slice-wise missing setting. We randomly selected elements or objects to be used as the unobserved parts, and evaluated the mean squared errors between true values and predicted values. We continued the evaluation ten times, and recorded the averaged errors and their standard errors.

6.3 Results

6.3.1 Accuracy

Figure 4 shows the accuracy of tensor completion when the EM-like optimization was used for the six methods ({'Ordinary', 'Within-mode', 'Cross-mode'} × {CP-decomposition, Tucker-decomposition})in the element-wise missing setting. Figure 5 shows those for the gradient-based optimization. Figures 6 and 7 show the results for the slice-wise missing setting.

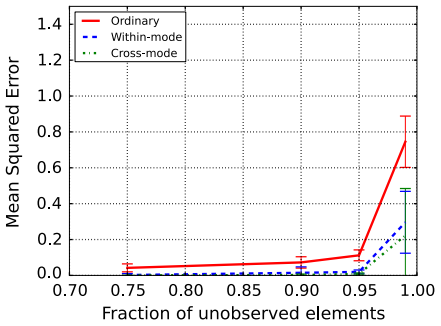
Overall, incorporating auxiliary information improves the predictive accuracy, which implies that the use of auxiliary information is effective in mitigating the performance degration resulting from observation sparsity. The EM-like methods are more accurate and stable than the gradient-based methods. Therefore, generally, our recommendation is to use the EM-like method in sparse cases if computational resources allow. However, for large-scale datasets, it is not realistic to use the EM-like methods, and the gradient-based methods are the exclusive option. We guess the difference of stability of the two approaches as follows; in sparse cases, the degree of freedom of the model is too large against the number of available observations, which leads to the instability of results. The EM-based method mitigates the instability by giving the initial estimates for the unobserved elements as the average of the observed elements.

In the element-wise missing setting, the difference between the within-mode and the cross-mode regularizations is not significant in moderately sparse cases. However, in extremely sparse cases (where 99 % of the data tensor are missing), the cross-mode regularization performs well because it induces dense similarities among tensor elements. This is more significant in the "cold-start" situation such as slice-wise missing cases. In particular, the cross-mode regularization seems compatible with the EM-like methods.

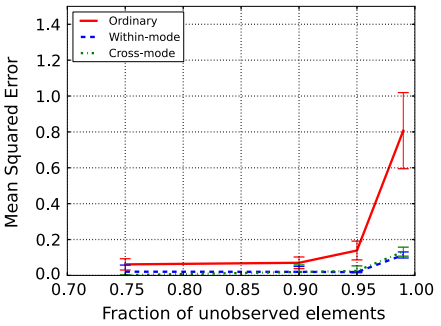
6.3.2 Computation time

Figure 8 shows a comparison between the total computation times in the EM-like approach and those in the gradient-based approach. The computation times were evaluated as the average of five runs by using the flow injection dataset with 99 % missing values in the element-wise missing setting. Even with this small-sized dataset, we can see the advantage of the gradient-based updates over the EM-based ones.

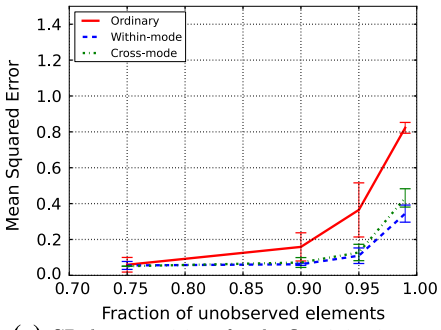
It is interesting to see the use of auxiliary information does not increase the overall computation time. In particular, the computation time even decreases with the EM-like methods. Actually, the additional information surely increases the computational time for each step in the EM-like method. Nevertheless, the number of iterations needed for convergence decreases with the within- or cross-mode regularization, which eventually results in the lower total computational time of the EM-like methods. Similarly, the increase of the cost of computing gradients is cancelled by the decrease of the number of iterations. They are probably because adding the additional regularization terms makes the landscape of the objective function smoother.



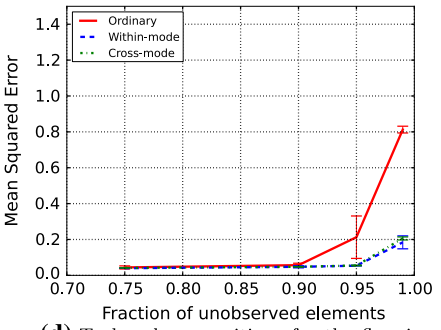
(a) CP-decompositions for the synthetic dataset



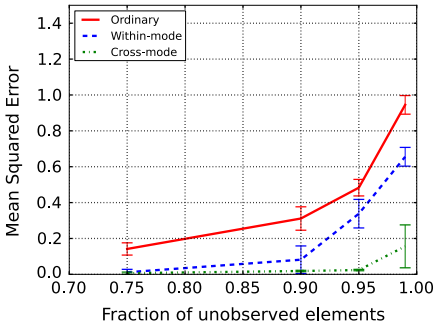
(b) Tucker decompositions for the synthetic dataset



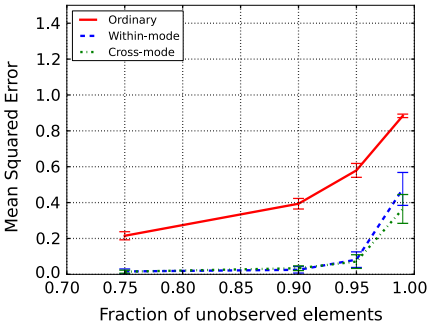
(c) CP-decompositions for the flow injection dataset



(d) Tucker decompositions for the flow injection dataset



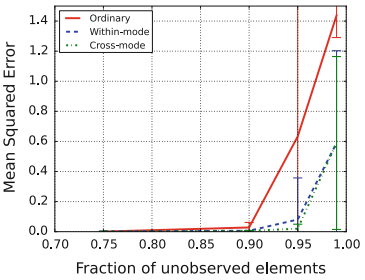
(e) CP-decompositions for the licorice dataset



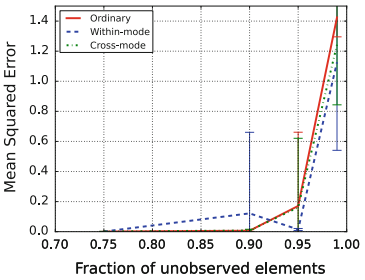
(f) Tucker decompositions for the licorice dataset

Fig. 4 Accuracy of tensor completion for three datasets in the *element-wise* missing setting by the EM-like algorithm. The proposed methods perform well when observations are sparse

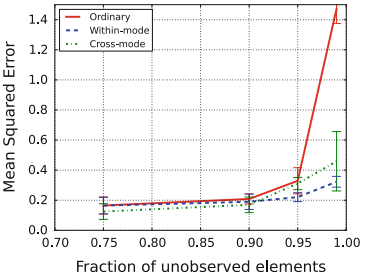
Although we employed the L-BFGS method by following Acar et al. (2010) in our experiments, the gradients we derived in Sect. 5.2 can also be used in stochastic gradient descent-type updates, which will further accelerate the computation with less memory requirements.



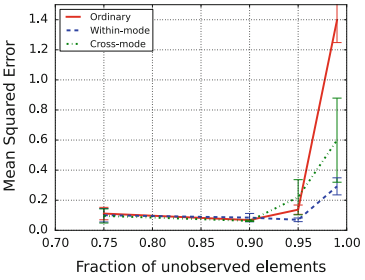
(a) CP-decompositions for the synthetic dataset



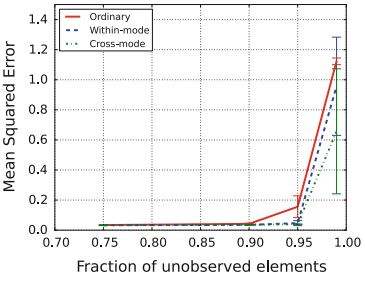
(b) Tucker decompositions for the synthetic dataset



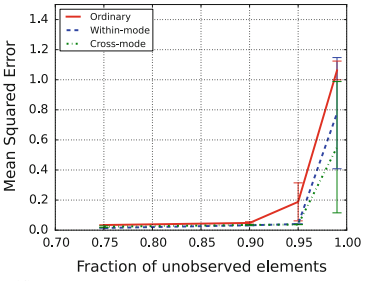
(c) CP-decompositions for the flow injection dataset



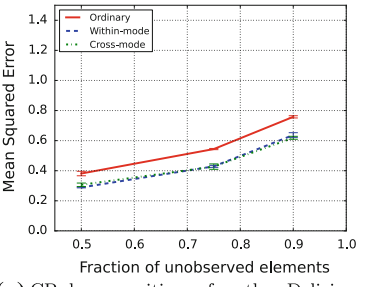
(d) Tucker decompositions for the flow injection dataset



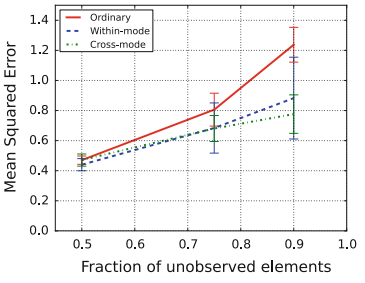
(e) CP-decompositions for the licorice dataset



(f) Tucker decompositions for the licorice dataset

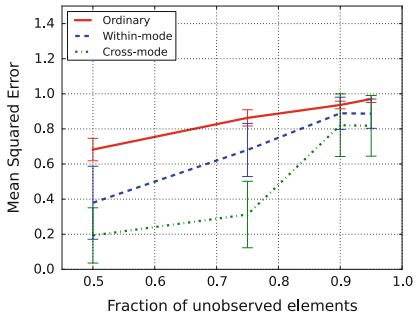


(g) CP-decompositions for the Delicious dataset

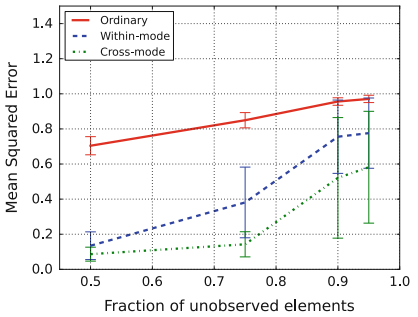


(h) Tucker decompositions for the Delicious dataset

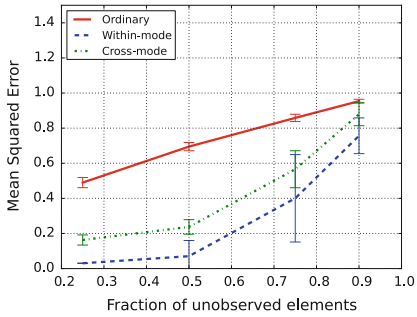
Fig. 5 Accuracy of tensor completion for four datasets in the *element-wise* missing setting by the gradient-based optimization. The proposed methods perform well when observations are sparse



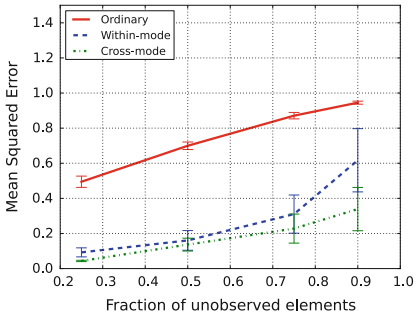
(a) CP-decompositions for the synthetic dataset



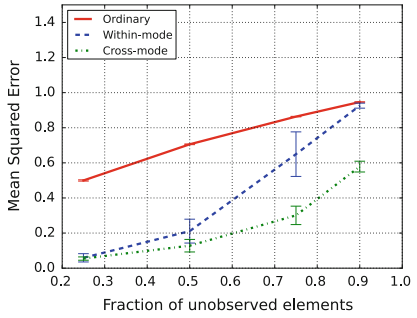
(b) Tucker decompositions for the synthetic dataset



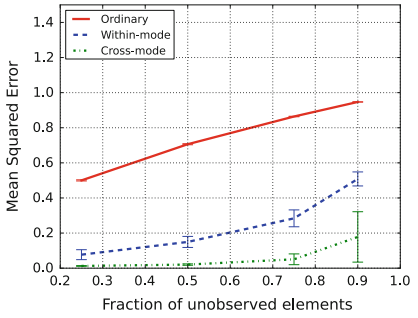
(c) CP-decompositions for the flow injection dataset



(d) Tucker decompositions for the flow injection dataset



(e) CP-decompositions for the licorice dataset

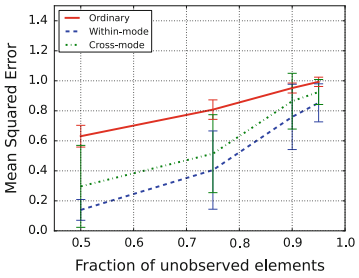


(f) Tucker decompositions for the licorice dataset

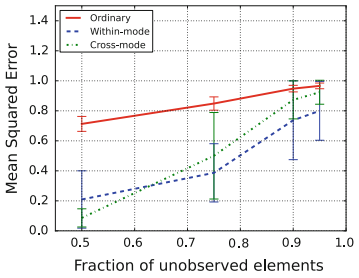
Fig. 6 Accuracy of tensor completion for three datasets in the *slice-wise* missing setting by the EM-like algorithm. The cross-mode regularization method performs especially well when observations are sparse

7 Related work

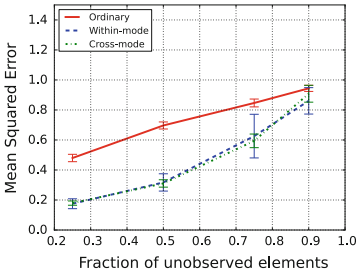
Tensor factorization methods have recently been studied extensively, and widely applied in the data mining communities. The CANDECOMP/PARAFAC(CP) decomposition (Harshman 1970) is a tensor factorization method that can be seen as a special



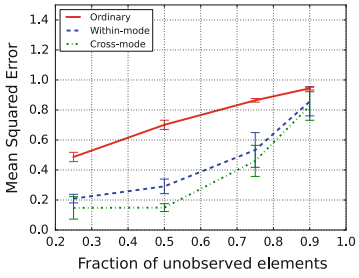
(a) CP-decompositions for the synthetic dataset



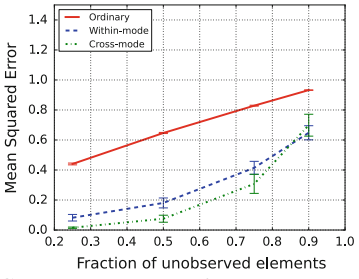
(b) Tucker decompositions for the synthetic dataset



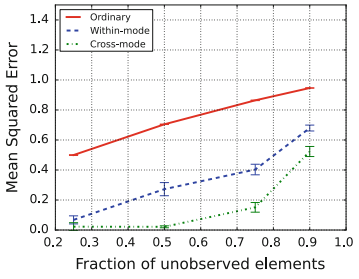
(c) CP-decompositions for the flow injection dataset



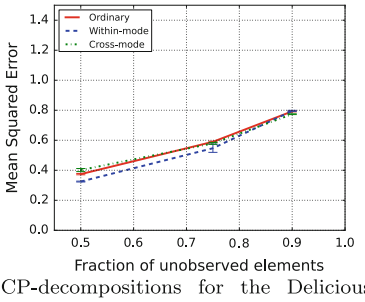
(d) Tucker decompositions for the flow injection dataset



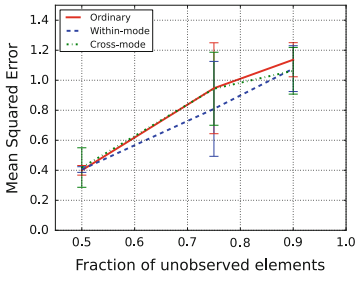
(e) CP-decompositions for the licorice dataset



(f) Tucker decompositions for the licorice dataset



(g) CP-decompositions for the Delicious dataset



(h) Tucker decompositions for the Delicious dataset

Fig. 7 Accuracy of tensor completion for four datasets in the *slice-wise* missing setting by the gradient-based optimization. The cross-mode regularization method performs especially well when observations are sparse

case of the Tucker decomposition (Tucker 1966) in which the core tensor is superdiagonal. The CP-decomposition has been applied to various problems including chemoinformatics (Kolda and Bader 2009). Its variant called pair-wise interaction tensor factorization (Rendle and Thieme 2010) accelerates its computation by using the stochastic gradient descent, and is applied to a large-scale tag recommendation problem.

There also exist probabilistic extensions of tensor factorization methods. Shashua and Hazan (2005) studied the PARAFAC model under the non-negativity constraint with latent variables. Chu and Ghahramani (2009) proposed a probabilistic extension of the Tucker method, known as pTucker.

Although we focus on the squared loss function (3) in this paper, changing the loss function corresponds to non-Gaussian probabilistic models of tensor factorization. In several matrix and tensor factorization studies, non-Gaussian observations have been dealt with. Collins et al. (2002) generalized the likelihood of the probabilistic PCA to the exponential family, which was further extended to tensors by Hayashi et al. (2010).

Recently, convex formulations of matrix factorization using the trace norm constraint instead of the low-rank constraint have been extensively studied (e.g. Srebro et al. 2005; Candes and Tao 2010; Cai et al. 2010). Such formulations have also been extended to tensor cases (e.g. Tomioka et al. 2012; Gandy et al. 2010). Convex programming formulations are another promising approach for mitigating the performance degradation resulting from data sparseness. The use of auxiliary information in tensor decomposition is rather independent of the choice of representation of the low-rank constraint, and combining them is an interesting future direction.

There are several studies to incorporate auxiliary information into matrix factorization. Li and Yeung (2009) introduced a regularizer for one of factor matrices by using a graph Laplacian based on geometry of data distribution. A similar approach is proposed by Cai et al. (2010). Lu et al. (2009) proposed incorporated both spatial

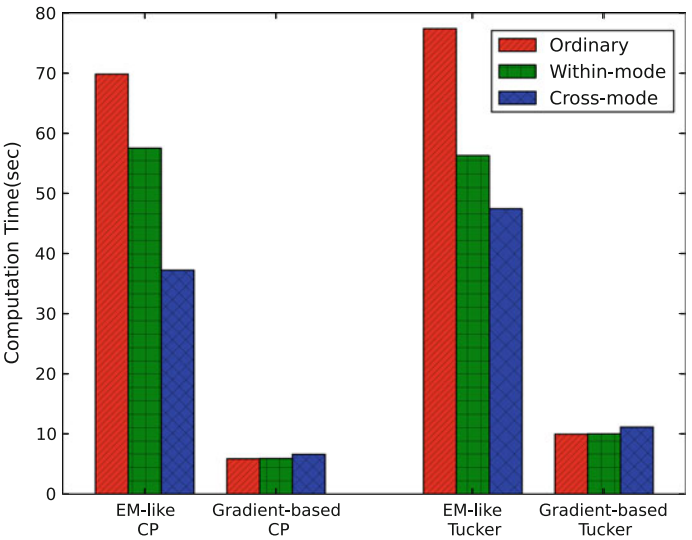


Fig. 8 Comparison of total computation times for the flow injection dataset

and temporal information by using graph Laplacian and Kalman filter. Adams et al. (2010) extended the probabilistic matrix factorization (Salakhutdinov and Mnih 2008) to incorporate side information. In their work, Gaussian process priors are introduced to enforce smoothness to factors. Some work use auxiliary information not in regularization terms but as bias variables added to model parameters (Yu et al. 2009; Porteous et al. 2010). Although not aiming to decomposition or completion of tensors, Banerjee et al. (2007) addresses clustering on multiple relation data from various information sources. To best of our knowledge, our work is the first attempt to incorporate auxiliary information into tensor factorization.

8 Conclusion

In this paper, we proposed the use of relationships among data as auxiliary information in addition to the low-rank assumption to improve the accuracy of tensor factorization. We introduced two regularization approaches using graph Laplacians induced from the relationships, and designed approximate solutions for the optimization problems. Numerical experiments using synthetic and real datasets showed that the use of auxiliary information improved completion accuracy over the existing methods based only on the low-rank assumption, especially when observations were sparse.

Acknowledgments The authors would like to thank the anonymous reviewers of ECML PKDD 2011 for their valuable comments and suggestions to improve the quality of the paper.

References

Acar E, Dunlavy DM, Kolda TG, Mørup M (2010) Scalable tensor factorizations with missing data. In: Proceedings of the 2010 SIAM international conference on data mining, pp 701–712

Adams RP, Dahl GE, Murray I (2010) Incorporating side information into probabilistic matrix factorization using Gaussian processes. In: Grünwald P, Spirtes P (eds) Proceedings of the 26th conference on uncertainty in artificial intelligence, Catalina Island, California, pp 1–9

Banerjee A, Basu S, Merugu S (2007) Multi-way clustering on relation graphs. In: Proceedings of the 2007 SIAM international conference on data mining

Cai D, He X, Han J, Huang TS (2010) Graph regularized non-negative matrix factorization for data representation. IEEE Trans Pattern Anal Mach Intell 33(10):2026–2038

Cai JF, Candes EJ, She Z (2010) A singular value thresholding algorithm for matrix completion. SIAM J Optim 20(4):1956–1982

Candes EJ, Tao T (2010) The power of convex relaxation: near-optimal matrix completion. IEEE Trans Inf Theory 56(5):2053–2080

Chu W, Ghahramani Z (2009) Probabilistic models for incomplete multi-dimensional arrays. In: Proceedings of the 12th international conference on artificial intelligence and statistics

Collins M, Dasgupta S, Schapire RE (2002) A generalization of principal components analysis to the exponential family. In: Dietterich TG, Becker S, Ghahramani Z (eds) Advances in neural information processing systems, vol 14. MIT Press, Cambridge

Dunlavy DM, Kolda TG, Acar E (2011) Temporal link prediction using matrix and tensor factorizations. ACM Trans Knowl Discov Data 5:10–11027

Gandy S, Recht B, Yamada I (2010) Tensor completion and low-n-rank tensor recovery via convex optimization. Inverse Problems 27(2):025010

Harshman RA (1970) Foundations of the PARAFAC procedure: models and conditions for an "explanatory" multi-modal factor analysis. UCLA Work Pap Phonetics 16(1):84

Hayashi K, Takenouchi T, Shibata T, Kamiya Y, Kato D, Kunieda K, Yamada K, Ikeda K (2010) Exponential family tensor factorization for missing-values prediction and anomaly detection. In: Proceedings of the 10th IEEE international conference on data mining, pp 216 –225

Kashima H, Kato T, Yamanishi Y, Sugiyama M, Tsuda K (2009) Link propagation: a fast semi-supervised algorithm for link prediction. In: Proceedings of the 2009 SIAM international conference on data mining

Kolda TG, Bader BW (2009) Tensor decompositions and applications. SIAM Rev 51(3):455–500

Li WJ, Yeung DY (2009) Relation regularized matrix factorization. In: Proceedings of the 21st international joint conference on artificial intelligence, pp 1126–1131

Lu Z, Agarwal D, Dhillon IS (2009) A spatio-temporal approach to collaborative filtering. In: Proceedings of the 3rd ACM conference on recommender systems, pp 13–20

Nishimori Y, Akaho S (2010) Learning algorithms utilizing quasi-geodesic flows on the Stiefel manifold. Neurocomputing 67:106–135

Plumbley MD (2005) Geometrical methods for non-negative ICA: manifolds, Lie groups and toral subalgebras. Neurocomputing 67:161–197

Porteous I, Asuncion A, Welling M (2010) Bayesian matrix factorization with side information and Dirichlet process mixtures. In: Proceedings of the 24th AAAI conference on artificial intelligence, pp 563–568

Rendle S, Thieme LS (2010) Pairwise interaction tensor factorization for personalized tag recommendation. In: Proceedings of the 3rd ACM international conference on web search and data mining, pp 81–90

Salakhutdinov R, Mnih A (2008) Probabilistic matrix factorization. In: Platt JC, Koller D, Singer Y, Roweis S (eds.) Advances in neural information processing systems, vol 20. MIT Press, Cambridge

Shashua A, Hazan T (2005) Non-negative tensor factorization with applications to statistics and computer vision. In: Proceedings of the 22nd international conference on machine learning, pp 792–799

Srebro N (2004) Learning with matrix factorizations. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge

Srebro N, Rennie J, Jaakkola T (2005) Maximum-margin matrix factorization. In: Advances in neural information processing systems, vol 17. MIT Press, Cambridge

Tomioka R, Suzuki T, Hayashi K, Kashima H (2012) Statistical performance of convex tensor decomposition. In: Advances in Neural Information Processing Systems, vol 24. MIT Press, Cambridge

Tucker L (1966) Some mathematical notes on three-mode factor analysis. Psychometrika 31(3):279–311

Walczak B (2001) Dealing with missing data. Part I. Chemom Intell Lab Syst 58(1):15–27

Yu K, Lafferty J, Zhu S, Gong Y (2009) Large-scale collaborative prediction using a nonparametric random effects model. In: Proceedings of the 26th international conference on machine learning, pp 1185–1192