Bayesian CP Factorization of Incomplete Tensors with Automatic Rank Determination

Qibin Zhao, Member, IEEE, Liqing Zhang, Member, IEEE, and Andrzej Cichocki Fellow, IEEE

Abstract—Tensor factorization of incomplete data is a powerful technique for imputation of missing entries (also known as tensor completion) by explicitly capturing the latent multilinear structure. One of the most well-known model is CANDECOMP/PARAFAC (CP) style tensor factorization that can be achieved by various optimization algorithms. However, as either the missing ratio or the noise level increases, most of existing CP factorizations are prone to overfitting since the tensor rank, as a tuning parameter, is required to be manually specified. Unfortunately, the determination of tensor rank is a challenging problem especially in the presence of both missing data and noisy measurements. In addition, the existing approaches can only provide the point estimation of latent factors as well as missing entries, which do not take into account the uncertainty information. To address these issues, we formulate CP factorization by a hierarchical probabilistic model and employ a fully Bayesian treatment by incorporating a sparsity inducing prior over multiple latent factors and the appropriate hyperpriors over all hyperparameters, resulting in an automatic model selection (i.e., rank determination) and noise detection. To learn the model, we develop an elegant deterministic algorithm under the variational Bayesian inference framework as well as the corresponding solution for efficient computation. These simulations on synthetic data illustrate the intrinsic capability of recovering the ground truth of tensor rank and preventing the overfitting problem even with extremely large amount of missing entries. Moreover, the results on several real-world applications, including image inpainting and facial image synthesis, demonstrate that our method significantly outperforms state-of-the-art approaches of both tensor factorization and tensor completion in terms of predictive performance.

Index Terms—Tensor factorization, tensor completion, tensor rank determination, variational Bayesian inference, missing data, image inpainting

1 INTRODUCTION

Tensors (i.e., multiway arrays) provide an effective and faithful representation of structural properties of the data, especially for multidimensional data or data ensemble affected by multiple factors. For instance, a video sequence can be represented by a third order tensor with dimensionality of height × width × time; an image ensemble measured under multiple conditions can be represented by a higher order tensor with dimensionality of pixel × person × pose × illumination. To model these data, tensor factorization, an extension of matrix factorization to multiway arrays, enables us to explicitly take into account the structure information that is available as a priori information on the data nature, and to effectively capture the underlying interactions across multiple dimensions. Therefore its theory and algorithms have been an active area of study within the past decade, see e.g. [1], [2], and have been successfully applied to various fields of application such as face recognition, social network analysis, image and video completion, and brain signal processing. The two most popular tensor factorization framework are Tucker [3] and CANDECOMP/PARAFAC (CP) that is also known as canonical polyadic decomposition (CPD) [4], [5], [6].

Most of existing tensor factorizations assume that the tensor is complete while the missing data can arise in a variety of real-world applications, which attracts a great deal of research interest on tensor completion in recent years. Tensor factorization of incomplete data aims to capture the underlying multilinear factors from partially observed entries, which can in turn predict the missing entries. In [7], CP factorization with missing data is formulated as a weighted least squares problem, termed as CP weighted optimization (CPWOPT). A structured CPD using nonlinear least squares (CPNLS) is proposed in [8]. In [9], Riemannian optimization techniques on the manifold of tensors of fixed multilinear rank, termed as geometric nonlinear conjugate gradient (geomCG), is presented. However, as the number of missing entries increases, tensor factorization schemes tend to overfit the model due to the incorrectly specified tensor rank, resulting in severe deterioration of predictive performance. In
contrast, another popular technique is to exploit low-rank assumption for recovering the missing entries, while the rank minimization can be formulated as a convex optimization problem on nuclear norm. This technique has been extended to higher order tensors by defining the nuclear norm of a tensor, yielding the tensor completion [10]. Some variants related to the nuclear norm were also proposed such as a weighted sum of nuclear norm of mode-$n$ matricizations is not related to tensor rank defined in a general sense. These completion-based techniques generally cannot explicitly interpret the underlying model factors and do not take into account the noisy observations. Thus, a simultaneous tensor decomposition and completion (STDC) was introduced to combine a rank minimization technique with Tucker decomposition [13]. In addition, the auxiliary information is also exploited in [13], [14] to improve completion accuracy, while such auxiliary information is greatly related to the specific application. It is also worth to note that the rank minimization based on convex optimization of nuclear norm is affected by the tuning parameters, which may tend to over/under-estimate the truth.

It is important to emphasize that our knowledge about the properties of tensor rank, defined by the minimum number of rank-one terms in CP decompositions, is surprisingly limited. There is no straightforward algorithm to determine the rank of a given tensor, and the problem has been shown to be NP-complete [15]. Many questions about tensor rank can be much more difficult to answer than their matrix analogues, and many open problems remain [16]. The lower and upper bound of tensor rank has been studied in [17], [18]. The ill-posedness of the best low-rank approximation of a tensor is investigated in [19]. Therefore, determining or even bounding the rank of an arbitrary tensor is quite difficult in contrast to matrix rank, and this difficulty would be significantly enhanced in the presence of large amount of missing entries.

Recently, probabilistic models for matrix/tensor factorization attracted a lot of interests in collaborative filtering and matrix/tensor completion. The probabilistic matrix factorization has been proposed in [20] while its fully Bayesian treatment using Markov chain Monte Carlo (MCMC) inference was shown in [21] and using variational Bayesian inference in [22], [23]. Further extensions to nonparametric and robust variants were presented in [24], [25]. The probabilistic framework of tensor factorization was presented in [26], [27], [28]. The other variants include extensions to the exponential family model [29] and to the nonparametric Bayesian model [30]. However, the tensor rank or model complexity are often given by a tuning parameter selected by either maximum likelihood or cross-validations that are computational expensive and inaccurate. Another issue is the inference of latent factor matrices is performed by either point estimation that is prone to overfitting or MCMC inference that tends to converge very slowly.

To address these issues, we propose a fully Bayesian probabilistic tensor factorization model according to CP factorization framework. Our objective is to capture the underlying multilinear factors from a noisy and incomplete tensor and to infer the predictive distribution of missing entries, while rank of the true latent tensor can be automatically determined to avoid overfitting. To this end, we specify a sparsity inducing prior over multiple factor matrices with shared hyperparameters as well as the appropriate hyperpriors. All model parameters including noise precision are considered as latent variables on which the corresponding priors are placed. Due to sparsity inducing property of the hyperpriors, the dimensionality of latent space is constraint to be minimum and can be learned automatically. The complex interactions among multiple factors and fully Bayesian treatment results in that learning the model is analytically intractable. Therefore, we resort to the variational Bayesian approximate inference and derive a deterministic solution to approximate the posteriors of all model parameters and hyperparameters. In addition, a computationally efficient algorithm is developed using the multilinear operations. Therefore, our method is a fully tuning parameter free approach that enables us to infer the latent factors with a low-rank constraint from an incomplete and noisy tensor data, while providing the predictive distributions over latent factors and predictions of missing entries. We conduct extensive experiments and comparisons on synthetic data to illustrate the advantages of our approach in terms of rank determination, predictive capability, and robustness to overfitting. Finally, several real-world applications including object/text removal, image completion, image restoration, and facial image synthesis are presented, which demonstrate that our method significantly outperforms state-of-the-art approaches, including both tensor factorizations and tensor completion, in terms of the predictive performance.

The rest of this paper is organized as follows. Section 2 presents preliminary multilinear operations and notations. Section 3 introduces a fully Bayesian CP model specification and the variational Bayesian inference for learning the model. The advantages of proposed method over both tensor factorization and tensor completion are discussed in Section 4. Section 5 presents the extensive experimental results on both synthetic data and real-world applications, followed by conclusions in Section 6.
2 Preliminaries and Notations

The order of a tensor is the number of dimensions, also known as ways or modes. Vectors (first order tensors) are denoted by boldface lowercase letters, e.g., \( \mathbf{a} \). Matrices (second order tensors) are denoted by boldface capital letters, e.g., \( \mathbf{A} \). Higher-order tensors (order \( \geq 3 \)) are denoted by boldface calligraphic letters, e.g., \( \mathcal{A} \). Given an \( N \)th order tensor \( \mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \), its \((i_1, i_2, \ldots, i_N)\)th entry is denoted by \( X_{i_1,i_2,\ldots,i_N} \) where the indices typically range from 1 to their capital version, e.g., \( i_n = 1, 2, \ldots, I_n, \forall n \in [1,N] \). The \((i,j)\)th entry of a matrix \( \mathbf{A} \) is denoted by \( A_{ij} \).

The inner product of two tensors \( \{ \mathbf{A}, \mathbf{B} \} \in \mathbb{R}^{I_1 \times \cdots \times I_N} \) is defined by \( \langle \mathbf{A}, \mathbf{B} \rangle = \sum_{i_1i_2\ldots i_N} A_{i_1i_2\ldots i_N} B_{i_1i_2\ldots i_N} \), and the squared Frobenius norm by \( \| \mathbf{A} \|_F^2 = \langle \mathbf{A}, \mathbf{A} \rangle \).

For a more general case, the inner product of a set of matrices or vectors is defined as a sum of element-wise products. For example, given \( \{ \mathbf{A}^{(n)} | n = 1, \ldots, N \} \), we define
\[
\langle \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)} \rangle = \sum_{i,j} \prod_{n} A_{ij}^{(n)}
\]

The Hadamard product is an entrywise product of two vectors, matrices or tensors of the same dimensions. For instance, given two matrices \( \mathbf{A} \in \mathbb{R}^{I \times J} \) and \( \mathbf{B} \in \mathbb{R}^{I \times J} \), their Hadamard product is a matrix of size \( I \times J \) and is denoted by \( \mathbf{A} \odot \mathbf{B} \). Without loss of generality, the Hadamard product of a set of matrices \( \{ \mathbf{A}^{(n)} | n = 1, \ldots, N \} \) is simply denoted and defined by
\[
\bigotimes_{n} \mathbf{A}^{(n)} = \mathbf{A}^{(1)} \odot \mathbf{A}^{(2)} \odot \cdots \odot \mathbf{A}^{(N)}
\]

The Kronecker product \( \bigotimes \) of matrices \( \mathbf{A} \in \mathbb{R}^{I \times J} \) and \( \mathbf{B} \in \mathbb{R}^{K \times L} \) is a matrix of size \( IK \times JL \), denoted by \( \mathbf{A} \bigotimes \mathbf{B} \). The Khatri Rao product of matrices \( \mathbf{A} \in \mathbb{R}^{I \times K} \) and \( \mathbf{B} \in \mathbb{R}^{I \times K} \) is a matrix of size \( IJ \times K \) defined by a columnwise Kronecker product, and denoted by \( \mathbf{A} \odot \mathbf{B} \). In particular, the Khatri Rao product of a set of matrices \( \{ \mathbf{A}^{(n)} | n = 1, \ldots, N \} \) in a reverse order is defined by
\[
\bigodot_{n} \mathbf{A}^{(n)} = \mathbf{A}^{(N)} \odot \mathbf{A}^{(N-1)} \odot \cdots \odot \mathbf{A}^{(1)}
\]

while the Khatri Rao product of a set of matrices except the \( n \)th matrix, denoted by \( \mathbf{A}^{(n)} \), is defined by
\[
\bigodot_{k \neq n} \mathbf{A}^{(k)} = \mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)}
\]

3 Bayesian Tensor Factorization

3.1 Probabilistic Model and Priors

Let \( \mathbf{Y} \) be an incomplete \( N \)th-order tensor of size \( I_1 \times I_2 \times \cdots \times I_N \) with missing entries. The element \( Y_{i_1,i_2,\ldots,i_N} \) is observed if \( (i_1, i_2, \ldots, i_N) \in \Omega \) where \( \Omega \) denotes a set of indices. We define an indicator tensor \( \mathcal{O} \) with same size as \( \mathbf{Y} \), whose entry \( O_{i_1,i_2,\ldots,i_N} \) is equal to 1 if \( (i_1, i_2, \ldots, i_N) \in \Omega \) otherwise is equal to 0.

We assume \( \mathbf{Y} \) is a noisy observation of a true latent tensor \( \mathbf{X} \), that is \( \mathbf{Y} = \mathbf{X} + \mathbf{e} \) where the noise term is assumed to be an i.i.d. Gaussian distribution, i.e., \( \mathbf{e} \sim \prod_{i_1,\ldots,i_N} \mathcal{N}(0, \tau^{-1}) \), and the latent tensor \( \mathbf{X} \) can be exactly represented by a CP model, giving by
\[
\mathbf{X} = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \odot \cdots \odot \mathbf{a}_r^{(N)} = \left[ \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)} \right]
\]

where \( \odot \) denotes the outer product of vectors and \( \cdot \) denotes a Kruskal operator of a set of matrices having the same number of columns. \( \{ \mathbf{A}^{(n)} | n = 1, \ldots, N \} \) are latent factor matrices corresponding to each of \( N \) modes, respectively. CP model can be interpreted as a sum of \( R \) rank-one tensors, which is related to a precise definition of tensor rank that is the smallest integer \( R \) for which the above representation holds. In the following parts, mode-\( n \) factor matrix \( \mathbf{A}^{(n)} \) of size \( I_n \times R \) is denoted by two different representations using row-wise and column-wise vectors, that is,
\[
\mathbf{A}^{(n)} = \begin{bmatrix} a_{1}^{(n)} & \cdots & a_{I_n}^{(n)} \end{bmatrix}^T = \begin{bmatrix} a_{1}^{(n)} & \cdots & a_{R}^{(n)} \end{bmatrix}
\]

The CP generative model together with noise assumption directly give rise to the observation model, the probability density of partially observed tensor \( \mathbf{Y}_{\Omega} \) (i.e., \( \mathbf{Y}_{(\Omega=1)} \) given the parameters, which is factorized over tensor elements
\[
p \left( \mathbf{Y}_{\Omega} | \{ \mathbf{A}^{(n)} \}_{n=1}^{N}, \tau \right) = \prod_{i_1=1}^{I_1} \cdots \prod_{i_N=1}^{I_N} \mathcal{N} \left( Y_{i_1,i_2,\ldots,i_N} \left| \langle \mathbf{A}^{(1)}_{i_1}, \mathbf{A}^{(2)}_{i_2}, \ldots, \mathbf{A}^{(N)}_{i_N} \rangle, \tau^{-1} \right\rangle_{i_1\cdots i_N} \right) \]

where the parameter \( \tau \) denotes the noise precision, and \( \cdot \) denotes an inner-product of \( N (N \geq 3) \) vectors, which is defined in a similar form to matrix case shown in \( \langle \cdot \rangle \), i.e., \( \langle \mathbf{a}_{i_1}, \mathbf{a}_{i_2}, \ldots, \mathbf{a}_{i_N}^{(N)} \rangle = \sum_{i_N} a_{i_1,\ldots,i_N}^{(n)} \).

The likelihood model in \( \tau \) indicates that \( Y_{i_1\cdots i_N} \) can be generated from multiple \( R \)-dimensional latent
variables \[ \{ a_{i n}^{(n)} \mid n = 1, \ldots, N \} \], whereas each latent variable \( a_{i n}^{(n)} \) corresponds to a set of observations rather than one, i.e., a subtensor whose mode-\( n \) index is \( i_n \). Another interpretation is that each element of \( \mathbf{Y} \) is the product of its corresponding \( N \) latent variables, which is measured by an inner product of multiple vectors. The essential difference between matrix factorization and tensor factorization is that the inner product of \( N \geq 3 \) latent variables allows us to model multilinear interaction structure, which in turn leads to much more difficulties in model learning.

Unlike the matrix case, the estimation of tensor rank is a challenging problem. In practice, the effective dimensionality of the latent space, i.e., \( R \), is a tuning parameter selected by a specific criteria such as fitting error or generalization error. Such parameter controls the model complexity and thus its selection can be considered as a problem of model selection. In our study, we seek an elegant way for automatic model selection, which can not only infer rank of the true latent tensor \( \mathbf{X} \) but also effectively avoid overfitting problem. To achieve this, a set of continuous hyperparameters are employed to control the precision related to each dimensionality of the latent space, respectively. Since the minimum \( R \) is desired in the sense of low rank approximation, a sparsity inducing prior is specified over these hyperparameters, resulting in that automatic rank determination can be achieved as a part of the process of Bayesian inference. This technique is known as automatic relevance determination (ARD) [31] or sparse Bayesian learning [32]. However, unlike the traditional method that place ARD prior over either latent variables or weight parameters such as Bayesian principle component analysis [33], we consider all model parameters as latent variables over which a sparsity inducing prior is placed with shared hyperparameters.

More specifically, we place a hierarchical prior over the latent factors, governed by a \( R \)-dimensional hyperparameters \( \lambda = [\lambda_1, \ldots, \lambda_R] \), where each hyperparameter controls one of latent components in \( \mathbf{A}^{(n)} \). Thus, the prior of mode-\( n \) factor matrix is given by

\[
p(\mathbf{A}^{(n)} \mid \lambda) = \prod_{i_n} \mathcal{N}(a_{i_n}^{(n)} | 0, \Lambda^{-1}), \forall n \in [1, N] \tag{8}
\]

where \( \Lambda = \text{diag}(\lambda) \) denotes an inverse variance matrix and is shared by latent factor matrices in all modes. We can further define a hyperprior over hyperparameters \( \lambda \), which is factorized over latent dimensionality due to the independent assumption

\[
p(\lambda) = \prod_{r=1}^{R} \text{Ga}(\lambda_r | c_0^r, d_0^r) \quad \tag{9}
\]

where \( \text{Ga}(x|a,b) \) denotes a Gamma distribution of the form

\[
\text{Ga}(x|a,b) = \frac{b^a x^{a-1} e^{-bx}}{\Gamma(a)} \quad \tag{10}
\]

and \( \Gamma(a) \) is the Gamma function. Since the sparsity is enforced in the latent dimensions, initialization of the dimensionality of latent space (i.e., \( R \)) is usually set to its maximum possible value, while the effective dimensionality can be inferred automatically under Bayesian inference framework. For instance, if a particular \( \lambda_r \) has a posterior distribution concentrated at large values, the corresponding \( \{ a_{i_n}^{(n)} \mid n \in [1, N] \} \) will tend to be zero and effectively be pruned out. Note that the priors are shared across \( N \) latent matrices, our framework can learn the same sparsity pattern for all factor matrices, yielding the minimum latent components. Therefore, our model can effectively infer the rank of tensors while performing the tensor factorization, which can be treated as a Bayesian low-rank tensor factorization.

To complete the model with a fully Bayesian treatment, we also place a hyperprior over the noise precision \( \tau \), that is

\[
p(\tau) = \text{Ga}(\tau | a_0, b_0). \tag{11}
\]

The probabilistic graph structure of the model is illustrated in Fig. 1.

For simplicity of notations, all unknowns including both latent variables and hyperparameters are collected and denoted together by \( \Theta = \{ \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}, \lambda, \tau \} \). From Fig. 1, we can easily write the joint distribution of observed data and all model parameters as

\[
p(\mathbf{Y}_\Omega, \Theta) = p(\mathbf{Y}_\Omega \mid \{ \mathbf{A}^{(n)} \}_{n=1}^N, \tau) \prod_{n=1}^N p(\mathbf{A}^{(n)} \mid \lambda) \, p(\lambda) \, p(\tau) \tag{12}
\]

By combing likelihood in (7), the priors of model parameters in (8), and hyperpriors in (9)(11), the logarithm of the joint distribution is given by (see Appendix for a detailed derivation)

\[
\ell(\Theta) = -\frac{\tau}{2} \left\| \mathcal{O} \otimes (\mathbf{Y} - [\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]) \right\|_F^2 \\
- \frac{1}{2} \text{Tr} \left( \Lambda \sum_n \mathbf{A}^{(n)T} \mathbf{A}^{(n)} \right) + \left( \frac{M}{2} + a_0 - 1 \right) \ln \tau \\
+ \sum_r \left[ \left( \frac{\sum_n I_{n_r}}{2} + (c_0^r - 1) \right) \ln \lambda_r \right] \\
- \sum_r d_0^r \ln \lambda_r - b_0 \tau + \text{const} \tag{13}
\]

where \( M = \sum_{i_1, \ldots, i_N} \mathcal{O}_{i_1, \ldots, i_N} \) denotes the total number of observations, ‘\( \otimes \)’ denotes Hadamard product of tensors. Without loss of generality, we can perform the maximum a posterior (MAP) estimation of latent matrices and hyperparameters by maximizing (13), which is, to some extent, equivalent to optimize a squared error function with regularized constrains imposed on factor matrices and additional constraints imposed on the regularization parameters. Since this is non-convex, we can only find a locally optimal
MAP solution by using stochastic gradient descent methods.

However, in this study, in contrast to the point estimation, we aim to compute the full posterior distribution of all variables in $\Theta$ given the observed data, that is

$$ p(\Theta | Y_{\Omega}) = \frac{p(\Theta, Y_{\Omega})}{\int p(\Theta, Y_{\Omega}) \, d\Theta} $$

(14)

Based on the posterior distribution of model parameters in $\Theta$, the predictive distribution over missing entries, denoted by $Y_{\Omega \setminus \Omega}$, can be inferred by

$$ p(Y_{\Omega \setminus \Omega} | Y_{\Omega}) = \int p(Y_{\Omega \setminus \Omega} | \Theta) \, p(\Theta | Y_{\Omega}) \, d\Theta $$

(15)

where the likelihood term w.r.t. missing entries are similar to that of observed entries shown in (7).

### 3.2 Model Learning via Bayesian Inference

An exact Bayesian inference in (14) and (15) would integrate over all latent variables as well as hyperparameters, which is obviously analytically intractable. Hence, we must resort to the approximate inference. In this section, we develop a deterministic approximate inference method under variational Bayesian (VB) framework [34], [35] to learn the probabilistic CP factorization model, which can infer the posterior over the whole set of latent variables and hyperparameters.

We therefore seek a distribution $q(\Theta)$ to approximate the true posterior distribution $p(\Theta | Y_{\Omega})$ by minimizing the KL divergence, that is

$$ \text{KL}(q(\Theta) || p(\Theta | Y_{\Omega})) = \int q(\Theta) \ln \left\{ \frac{q(\Theta)}{p(\Theta | Y_{\Omega})} \right\} \, d\Theta $$

$$ = \ln p(Y_{\Omega}) - \int q(\Theta) \ln \left\{ \frac{p(Y_{\Omega} | \Theta)}{q(\Theta)} \right\} \, d\Theta $$

(16)

where $\ln p(Y_{\Omega})$ represents the model evidence, and the second term is defined as lower bound $\mathcal{L}(q) = \int q(\Theta) \ln \left\{ \frac{p(Y_{\Omega} | \Theta)}{q(\Theta)} \right\} \, d\Theta$ together with $\mathcal{F}(q) = -\mathcal{L}(q)$ known as variational free energy. Since the model evidence is a constant, then the maximum of the lower bound occurs when the KL divergence vanishes, which implies that $q(\Theta) = p(\Theta | Y_{\Omega})$.

For the initial derivation, it will be assumed that the variational distribution is factorized w.r.t. each variable $\Theta_j$ and so can be written as

$$ q(\Theta) = q_\lambda(\lambda) q_r(\tau) \prod_{n=1}^{N} q_n(\mathbf{A}^{(n)}) $$

(17)

Note that this is the only assumptions about the distribution, while the particular functional forms of the individual factors $q_j(\Theta_j)$ can be explicitly derived in turn. The optimised form of the $j$th factor based on maximisation of lower bound $\mathcal{L}(q)$ is given by

$$ \ln q_j(\Theta_j) = \mathbb{E}_{q(\Theta \setminus \Theta_j)} [ \ln p(Y, \Theta) ] + \text{const} $$

(18)

where $\mathbb{E}_{q(\Theta \setminus \Theta_j)}[\cdot]$ denotes an expectation w.r.t. the $q$ distributions over all variables except $\Theta_j$. Since the distributions of all variables are drawn from the exponential family and are conjugate w.r.t. the distributions over their parent variables (see Fig. [1]), we can derive the posterior distributions of model parameters using [18] and [13].

#### 3.2.1 Posterior distribution of factor matrices

From the graphical model shown in Fig. [1], the inference of mode-$n$ factor matrix $\mathbf{A}^{(n)}$ can be performed by receiving the messages from observed data and its co-parents including other factors $\mathbf{A}^{(k)}, k \neq n$ and the hyperparameter $\tau$, which are expressed by the likelihood term (7), and incorporating the messages from their parents, which are expressed by the prior term (8). By applying (18), it has been shown that their posteriors can be factorized as independent distributions of their rows that are also Gaussian (see Appendix for a detailed derivation), i.e., $\forall n \in [1, N]$,

$$ q_n(\mathbf{A}^{(n)}) = \prod_{i_n=1}^{I_n} \mathcal{N}(\mathbf{a}^{(n)}_{i_n}, \mathbf{V}^{(n)}_{i_n}) $$

(19)

where the posterior parameters can be updated by

$$ \mathbf{a}^{(n)}_{i_n} = \mathbb{E}_{q[\mathbf{A}^{(n)}]} \mathbf{V}^{(n)}_{i_n} \mathbb{E}_{q[\mathbf{A}^{(n)^T}]} \vec{\mathbf{y}}_{i_n=1} = \mathbf{V}^{(n)}_{i_n} \mathbb{E}_{q[\mathbf{A}^{(n)}]} \mathbf{A}^{(n)}_{i_n} + \mathbb{E}_{q[\mathbf{A}]}^{-1} $$

(20)

where $\mathbf{y}_{i_n=1}$ is a sample function denoting a subset of the observed entries $Y_{\Omega_{i_n}}$ whose mode-$n$ index is $i_n$. In another word, they are the observed entries that are affected by the latent factor $\mathbf{a}^{(n)}_{i_n}$. In (20), the most complex term is related to

$$ \mathbf{A}^{(n)^T} = \left( \bigotimes_{k \neq n} \mathbf{A}^{(k)} \right)^T $$

(21)

where $(\bigotimes_{k \neq n} \mathbf{A}^{(k)})^T$ is of size $R \times \prod_{k \neq n} I_k$, and each column is computed by $\otimes_{k \neq n} \mathbf{a}^{(k)}_{i_k}$ with varying mode-$k$ index $i_k$. The symbol $(\cdot)_{i_n=1}$ denotes a subset of columns sampled according to the subtensor $O_{i_n} = 1$. Hence, $\mathbb{E}_{q[\mathbf{A}^{(n)^T}]} \mathbf{A}^{(n)}_{i_n}$ denotes the posterior covariance matrix of the Khatri-Rao product of latent factors in all modes except $n$th-mode, and it only contains the columns corresponding to the observed entries whose mode-$n$ index is $i_n$. In order to evaluate this posterior covariance matrix, we need to firstly introduce the following results.

**Theorem 3.1.** Given a set of independent random matrices $\left\{ \mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R} \mid n = 1, \ldots, N \right\}$, $\mathbf{a}^{(n)}_{i_n}$ denotes $i_n$th-row random vector and these random vectors are assumed independent. If $\forall n, \forall i_n$, $\mathbb{E}[\mathbf{a}^{(n)}_{i_n}]$ and $\text{Var}([\mathbf{a}^{(n)}_{i_n}])$ are known, then

$$ \mathbb{E} \left[ \left( \bigotimes_{n} \mathbf{A}^{(n)^T} \right) \left( \bigotimes_{n} \mathbf{A}^{(n)} \right) \right] = \sum_{i_1, \ldots, i_N} \mathbb{E} \left[ \mathbf{a}^{(n)}_{i_n} \mathbf{a}^{(n)^T}_{i_n} \right] $$

(22)
where \( E \left[ a_{in}^{(n)} a_{in}^{(n)T} \right] = E[a_{in}^{(n)}] E[a_{in}^{(n)T}] + \text{Var}(a_{in}) \).

Proof: See appendix for details.

For simplicity, we attempt to compute (22) by multilinear operations in a tensor or matrix form. Hence, \( \forall n \), let \( B^{(n)} \) of size \( I_n \times R^2 \) denote an expectation of a quadratic form related to \( A^{(n)} \) by defining \( i_n \)th-row vector \( b_{in}^{(n)} = \text{vec} \left[ E \left[ a_{in}^{(n)} a_{in}^{(n)T} \right] \right] \), then we have

\[
\text{vec} \left( \sum_{i_1, \ldots, i_N} \bigotimes_{n \in \Omega} \left( E \left[ a_{in}^{(n)} a_{in}^{(n)T} \right] \right) \right) = \left( \bigotimes_n B^{(n)} \right)^T 1_{\prod I_n}
\]

(23)

where \( 1_{\prod I_n} \) denotes a vector with length of \( \prod I_n \) and all elements are equal to one.

According to the theorem 3.1 and the computation form in (23), the term \( E_q[A_{in}^{(n)}, A_{in}^{(n)}] \) in (20) can be evaluated efficiently by

\[
\text{vec} \left( E_q[A_{in}^{(n)T} A_{in}^{(n)}] \right) = \sum_{(i_1, \ldots, i_{n-1}, i_n+1, \ldots, i_N) \in \Omega} \otimes_{k \neq n} \left( E_q[a_{ik}^{(k)} a_{ik}^{(k)T}] \right) = \left( \bigotimes_n B^{(k)} \right)^T \text{vec}(O_{i_1 \ldots i_N})
\]

(24)

where \( B^{(k)} \) denote a posterior quadratic form of \( A^{(k)} \), and \( O_{i_1 \ldots i_N} \) denotes a subtensor with model-\( n \) index fixed to \( i_n \). Note that in (24), the Khatri-Rao product is performed on \( n - 1 \) indices that interacts with model-\( n \) index \( i_n \) yielding the observed entries, which is indicated by the tensor \( O_{i_1 \ldots i_N} \).

In (20), another complicated part can be also simplified by using multilinear operations, given by

\[
E_q[A_{in}^{(n)T}] \text{vec}(Y|I_{O_{in}=1}) = \sum_{(i_1, \ldots, i_{n-1}, i_n+1, \ldots, i_N) \in \Omega} \left( \bigotimes_{k \neq n} E_q[a_{ik}^{(k)}] \right) \text{vec}(O_{i_1 \ldots i_N})
\]

(25)

Finally, the variational posterior approximation of factor matrices can be obtained by evaluating (25). Based on the approximated posterior, we can easily evaluate the posterior moments including \( \forall n \), \( \forall i_n \), \( E_q[a_{in}^{(n)}] \), \( \text{Var}(a_{in}^{(n)}) \), \( E_q[A^{(n)}] \), and \( E_q[A_{in}^{(n)T}] \), \( E_q[A_{in}^{(n)T}] \), which are required for learning other latent variables in \( \Theta \).

### 3.2.2 Posterior distribution of hyperparameters \( \lambda \)

The prior of factor matrices is given by hyperparameters \( \lambda \) in which \( \lambda \) corresponds to \( r \)th component in the latent space. It should be noted that, instead of point estimation via optimizations, our model aims to infer the full posterior of \( \lambda \) from given data, which is crucial for automatic model selection. From the graphical model shown in Fig. 1, the inference of \( \lambda \) can be performed by receiving messages from \( N \) factor matrices and incorporating the messages from its hyperprior. By applying (18), we can identify the posteriors of \( \lambda_r \) \( \forall r \in [1, R] \) are independent Gamma distribution (see Appendix for details),

\[
q_\lambda(\lambda) = \prod_{r=1}^{R} \text{Ga}(\lambda_r | c_r, d_r)
\]

(26)

where \( c_r, d_r \) denote the posterior parameters learned from \( M \) observations, and can be updated by

\[
c_r = c_r + \frac{N}{2} \sum_{n=1}^{N} I_n
\]

\[
d_r = d_r + \frac{N}{2} \sum_{n=1}^{N} E_q[a_{in}^{(n)T} a_{in}^{(n)}]
\]

(27)

The expectation of inner product of \( r \)th component in mode-\( n \) matrix w.r.t. \( q \) distribution can be evaluated using the posterior parameters in (19) thus we have

\[
E_q[a_{in}^{(n)T} a_{in}^{(n)}] = a_{in}^{(n)T} a_{in}^{(n)} + \sum_{i_n} \left( Y_{in}^{(n)} \right)_{rr}
\]

(28)

Combining (27) and (28), we can further simplify the computation of \( d_M = [d_1^{M}, \ldots, d_R^{M}]^T \) as

\[
d_M = \sum_{n=1}^{N} \left\{ \text{diag} \left( \tilde{A}^{(n)T} \tilde{A}^{(n)} + \sum_i \tilde{Y}_{in}^{(n)} \right) \right\}
\]

(29)

where \( \tilde{A} = E_q[A^{(n)}] \). Due to the property of Gamma distribution, the posterior expectation of \( \lambda \) w.r.t. \( q \) distribution can be obtained by \( E_q[\lambda] = [c_r/d_r, \ldots, c_R/d_R]^T \) and thus \( E_q[\lambda] = \text{diag}(E_q[\lambda]) \).

### 3.2.3 Posterior distribution of hyperparameter \( \tau \)

The inference of noise precision \( \tau \) can be performed by receiving the messages from observed data and its co-parents including \( N \) factor matrices, and incorporating the messages from its hyperprior. By applying (18), the variational posterior is a Gamma distribution (see Appendix for details), given by

\[
q_\tau(\tau) = \text{Ga}(\tau | a_M, b_M)
\]

(30)

where the posterior parameters can be updated by

\[
a_M = a_0 + \frac{1}{2} \sum_{i_1, \ldots, i_N} \Theta_{i_1, \ldots, i_N}
\]

\[
b_M = b_0 + \frac{1}{2} E_q \left[ \left\| \Theta \otimes (Y - [A^{(1)}, \ldots, A^{(N)}]) \right\|_F^2 \right]
\]

(31)

However, the posterior expectation of model error in the above expression can not be computed straightforward, we need to introduce the following results.
Theorem 3.2. Assume we have a set of independent R-dimensional random vector \( \{x^{(n)}|n = 1, \ldots, N\} \), if \( \forall n \), the \( E[x^{(n)}] \) and \( \text{Var}[x^{(n)}] \) are known, then
\[
E \left[ \left( x^{(1)}, \ldots, x^{(N)} \right)^2 \right] = E \left[ x^{(1)}x^{(1)^T} \right] \ldots E \left[ x^{(N)}x^{(N)^T} \right]
\] (32)
where the left term denotes the expectation of square of inner products of \( N \) vectors, and the right term denotes the inner products of \( N \) matrices where each matrix of size \( R \times R \) denotes an expectation of outer product of \( n \)th vector respectively.

Proof: See appendix for details.

\[\square\]

Theorem 3.3. Given a set of independent random matrices \( \{A^{(n)}|n = 1, \ldots, N\} \), \( a^{(n)}_i \) denotes the \( i \)-th random vector of \( A^{(n)} \). If \( \forall n \), \( a^{(n)}_i \) are independent and \( E[a^{(n)}_i] \), \( \text{Var}(a^{(n)}_i) \) are known, then
\[
E \left[ \left\| A^{(1)}, \ldots, A^{(N)} \right\|^2_F \right] = \sum_{i_1, \ldots, i_N} E \left[ a^{(1)}_{i_1} a^{(1)^T}_{i_1} \right] \ldots E \left[ a^{(N)}_{i_N} a^{(N)^T}_{i_N} \right]
\] (33)
Let \( B^{(n)} \) denotes the expectation of a quadratic form related to matrix \( A^{(n)} \) by defining \( i \)-th row vector \( b^{(n)}_i = \text{vec} \left( E \left[ a^{(n)}_i a^{(n)^T}_i \right] \right) \), thus (33) can be computed by
\[
E \left[ \left\| A^{(1)}, \ldots, A^{(N)} \right\|^2_F \right] = E \left[ \left\| B^{(n)} \right\|^2 \right] = 1_{\Omega} 1_{\Omega} \left( \bigotimes_n B^{(n)} \right) 1_{R^2}
\] (34)

Proof: See appendix for details.

\[\square\]

From Theorem 3.2 and 3.3, the posterior expectation term in (31) can be evaluated explicitly, however, since there are missing entries in \( Y \), the evaluation form can be finally written as (see Appendix for details)
\[
E_q \left[ \left\| Q \otimes \left( Y - [A^{(1)}, \ldots, A^{(N)}] \right) \right\|^2_F \right]
= \left\| Y^{(1)} \right\|^2_F - 2\text{vec}^T(Y^{(1)})\text{vec} \left( \left[ \tilde{A}^{(1)}, \ldots, \tilde{A}^{(N)} \right] \right)
+ \text{vec}^T \left( \bigotimes_n B^{(n)} \right) 1_{R^2}
\] (34)
where \( \tilde{A}^{(n)} = E_q [A^{(n)}] \), and \( B^{(n)} \) denotes the posterior expectation of a quadratic form of \( A^{(n)} \), i.e., \( b^{(n)}_i = \text{vec} \left( a^{(n)}_i a^{(n)^T}_i + V^{(n)}_i \right) \). Finally, the posterior approximation of \( \tau \) can be obtained by (31) together with the posterior expectation w.r.t. \( q \) distribution, i.e.,
\[
E_q [\tau] = a_M/b_M.
\]

3.2.4 Lower bound of model evidence

We can also evaluate the variational lower bound in (16) for our model. Since at each step of the iterative re-estimation procedure the value of this bound should not decrease, we can monitor the bound in order to test for convergence. The lower bound on the log marginal likelihood can be also written as
\[
\mathcal{L}(q) = E_{q(\Theta)} \left[ \ln p(Y_{\Theta}) + H(q(\Theta)) \right]
\] (35)
where the first term denotes the posterior expectation of joint probability density, and the second term denotes the entropy of \( q \) distribution.

The various terms in the bound are evaluated and derived by taking parametric forms of \( q \) distribution, giving the following results (see Appendix for details)
\[
\mathcal{L}(q) =
- a_M \frac{\mathbb{E}_q}{2b_M} \left\| Q \otimes \left( Y - [A^{(1)}, \ldots, A^{(N)}] \right) \right\|^2_F
- \frac{1}{2} \text{Tr} \left\{ \Lambda \sum_n \left( \tilde{A}^{(n)} \tilde{A}^{(n)} + \sum_i V^{(n)}_i \right) \right\}
+ \frac{1}{2} \sum_n \sum_i \left( \ln V^{(n)}_i \right) + \sum_r \left( \ln \Gamma (\epsilon^{(r)}_M) \right)
+ \sum_r \left( c^{(r)}_M \left( 1 - \ln d^{(r)}_M - \frac{a^{(r)}_M}{d^{(r)}_M} \right) \right) + \ln \Gamma (a_M)
+ a_M \left( 1 - \ln b_M - \frac{b_0}{b_M} \right) + \text{const}
\] (36)

In these expressions, \( \Gamma (\cdot) \) denotes the Gamma function. The posterior expectation of model error denoted by \( E_q[\cdot] \) can be computed using [34].

3.2.5 Initialization of model parameters

Note that the variational Bayesian inference is only guaranteed to converge to a local minimum. To alleviate getting stuck in poor local solutions, it is important to choose an initialization point. In our model, the top level hyperparameters including \( c_0, d_0, a_0, b_0 \) are set to \( 10^{-5} \), resulting in a noninformative prior within the conditionally conjugate family. Thus the expectation of precision hyperparameters can be initialized to \( E[A] = I \), and \( E[\tau] = 1 \). For the factor matrices, the expectation of \( E[A^{(n)}] \), \( \forall n \in [1, N] \) can be initialized by two different schemes. One is randomly drawn from \( \mathcal{N}(0, I) \) for each row vector \( a^{(n)}_i \), \( \forall n \in [1, N] \) is simply set to \( \text{U}^{(n)} \Sigma^{(n)} \Gamma^{(n)} \), where \( \text{U}^{(n)} \) denotes the left singular vectors and \( \Sigma^{(n)} \) denotes the diagonal singular values matrix, obtained from SVD of mode-\( n \) matricization of tensor \( Y \). The covariance matrix \( V^{(n)} \) is simply set to \( \text{I} \). The tensor rank \( R \) is usually initialized by the weak upper bound on its maximum rank, i.e., \( R \leq \min_n P_n \), where \( P_n = \prod_{i \neq n} I_i \). In practice, for simplicity, we can also manually define the initialization value of tensor rank.

Theoretically, the posterior factors in (17) can be updated in any order. However, the update order can also affect the stationary solution to which the algorithm converges, even if the initialization is unchanged. In our method, we generally follow the order that from bottom to top (see Fig. 1), which
indicates that the message passing is started from observed data.

3.2.6 Interpretation of automatic rank determination

The whole procedure of model inference is summarized in Algorithm 1. It should be noted that tensor rank is determined automatically and implicitly. More specifically, updating parameter $\lambda$ in each iteration results in a new prior over $\{A^{(n)}\}$, then $\{A^{(n)}\}$ can be updated by such a new prior in the subsequent iteration, which in turn affects $\lambda$. Hence, if any $\lambda_n$ becomes very large, the $r$th components in all $\{A^{(n)}\}$, $\forall n \in [1, N]$, are enforced to be zero due to its prior information and the tensor rank can be obtained by $R = \max_n \text{Rank}(A^{(n)})$ or by simply count the number of non-zero components in any of $\{A^{(n)}\}$. For implementation of the algorithm, we keep the size of $\{A^{(n)}\}$ unchanged during iterations, while an alternative way is to remove the zero components of $\{A^{(n)}\}$ after each iteration.

3.3 Predictive Distribution

The predictive distributions over missing entries, given observed entries, are also intractable. However, we can approximate the predictive distribution by replacing the true posterior distribution with its variational posterior, that is

$$p(Y_{i_1, \ldots, i_N} | Y_{\Omega}) = \int p(Y_{i_1, \ldots, i_N} | \Theta) p(\Theta | Y_{\Omega}) \, d\Theta$$

$$\simeq \int \int p(Y_{i_1, \ldots, i_N} | \{a^{(n)}_{i_n}\}, \tau) q \left( \{a^{(n)}_{i_n}\} \right) q(\tau) \, d\tau \, d\Theta$$

We can now approximate these integrations giving an Student’s t-distribution (see Appendix for details)

$$p(Y | \Omega) = \prod_{i_1, \ldots, i_N} T(Y_{i_1, \ldots, i_N} | \tilde{Y}_{i_1, \ldots, i_N}, S_{i_1, \ldots, i_N}, \nu_y)$$

with its parameters given by

$$\tilde{Y}_{i_1, \ldots, i_N} = \langle \tilde{a}^{(1)}_{i_1}, \ldots, \tilde{a}^{(n)}_{i_n} \rangle$$

$$S_{i_1, \ldots, i_N} = \left\{ \frac{b_M}{a_M} + \sum_n \left( \bigotimes \tilde{a}^{(k)}_{i_n} \right)^T V^{(n)}_{i_n} \left( \bigotimes \tilde{a}^{(k)}_{i_n} \right) \right\}^{-1}$$

Thus, the predictive variance can be obtained by

$$\text{Var}(Y_{i_1, \ldots, i_N}) = \frac{\nu_y}{\nu_y - 2} S_{i_1, \ldots, i_N}^{-1}.$$  

3.4 Computational Complexity

The computation cost of $N$ factor matrices in [20] is $O(NR^2M + R^3 \sum I_n)$, where $N$ is the order of tensor; $M = P \prod I_n$ denotes the input data size in which $P$ is the ratio between number of observed entries and the tensor size. $R$ is the number of latent components in each $A^{(n)}$, i.e., model complexity or tensor rank, and it is generally much smaller than the data size, i.e., $R \ll M$. Hence, it has linear complexity w.r.t. the data size while polynomial complexity w.r.t. the model complexity. Note that due to the automatic model selection, the excessive latent components are pruned out in the first few iterations such that $R$ reduces rapidly in practice. The computation of hyperparameter $\lambda$ in [27] costs $O(R^2 \sum I_n)$ that is dominated by model complexity, while the computation of noise precision $\tau$ in [31] costs $O(R^2M)$. Therefore, the overall complexity of our algorithm is $O(NR^2M + R^3)$, which scales linearly with the tensor size but polynomially with the tensor rank. For each unobserved entry, the time complexity of prediction is only $O(NR^2)$ that is related to the order and rank of a tensor, but is independent with the tensor size.

4 Discussion of Advantages

Since our method is based on a hierarchical probabilistic model framework and fully Bayesian treatment, several significant advantages can be gained, which are discussed as follows:

- The capability of automatic determination of tensor rank enables us to effectively obtain the low-rank approximation of a tensor. Moreover, it has shown to be effective even on a highly noisy and incomplete tensor.
- Our method is a tuning parameter free approach and all model parameters can be inferred only based on observed data, which avoids the computational expensive parameter selection procedure. In contrast, the existing tensor factorization methods require predefined rank while the tensor completion methods using nuclear norm require several regularization parameters.
- The posteriors are inferred by integrating all irrelevant variables, which can effectively avoid the overfitting problem. This is a significant advantage.
over most existing tensor factorization algorithms that are based on optimization methods.

- The uncertainty information over both latent factors and predictions of missing entries can be inferred by our method, while most existing tensor factorization and completion methods only provide point estimation of model parameters.
- An efficient and deterministic algorithm is developed for Bayesian inference, which empirically shows a fast convergence and its computational complexity scales linearly with the data size.

5 EXPERIMENTAL RESULTS

We conducted extensive experiments on both synthetic data and real-world applications, and compared our method with seven state-of-the-art methods based on either tensor factorization or tensor completion schemes. For synthetic data, we aim to validate our method from several aspects: i) the capability of rank determination; ii) reconstruction performance given a complete tensor; iii) predictive performance over missing entries given an incomplete tensor; iv) the sensitivity of performance w.r.t. the noise level and missing ratio. For real-world data, one typical application is image inpainting such as completion of an image with an extremely large amount of randomly missing pixels, restoration of a scratched image and object removal from an image; Another application, newly introduced in this study, is facial image synthesis under multiple conditions by modeling given images under other multiple conditions.

5.1 Validation on Synthetic Data

The synthetic tensor data is generated by the following procedure: $N$ factor matrices $\{A^{(n)}\}_{n=1}^N$ of size $I_n \times R$ are drawn from a standard normal distribution, i.e., $\forall n$, $\forall i_n$, $a^{(n)}_{i_n} \sim \mathcal{N}(0, I_R)$, then the true latent tensor is constructed by $X = [A^1, \ldots, A^{(N)}]$, which is used to generate an observed tensor by $Y = X + \epsilon$ where $\epsilon \sim \prod_{i_1, \ldots, i_N} \mathcal{N}(0, \sigma^2)$ denotes an i.i.d. additive noise whose parameter controls the noise level. The missing entries are chosen uniformly and marked by an indicator tensor $O$ of the same size with $Y$.

5.1.1 A toy example

In this section, a toy example was performed for illustration of our model. Firstly, a true latent tensor $X$ of size $10 \times 10 \times 10$ was generated such that tensor rank was $R = 5$; the noise parameter was $\sigma^2 = 0.001$ and 40% entries were missing. Then, we apply our method with initial rank being set to 10. As shown in Fig. 2, three factor matrices are inferred in which 5 components are effectively pruned out, resulting in an automatic determination of tensor rank. The unnecessary components can be identified by the very large values of hyperparameters $\lambda$. The lower bound of marginal likelihood increases monotonically, as shown in Fig. 2 which indicates the effectiveness and convergence of our algorithm. Finally, the posterior of noise precision $\tau \approx 1000$ implies the capability of noise detection and reduction.

Fig. 2. A toy example for illustration of FBCP applied on an incomplete tensor of size $10 \times 10 \times 10$ with rank $R = 5$. The top row shows factor matrices in three modes, while the bottom row shows the posterior of $\lambda$, the lower bound of marginal likelihood, and the posterior of $\tau$ from left to right.
5.1.2 Automatic determination of tensor rank

For automatic determination of tensor rank, extensive simulations were performed under varying experimental conditions related to tensor size, tensor rank, noise level, missing ratio and the initialization method of factor matrices (e.g., SVD or random sample). Each result is evaluated by 50 runs corresponding to 50 different tensors generated under the same criteria. All simulations are divided into four groups: (A) given complete tensors of size $20 \times 20 \times 20$ with true rank $R = 5$, the evaluations were performed under five different noise levels with SNR ranging from $-20$dB to $20$dB, and by two different initialization methods. The results are shown in Fig. 3(a); (B) given incomplete tensors of size $20 \times 20 \times 20$ with true rank $R = 5$ and SNR=$20$dB, the evaluations were performed under five missing ratios ($[0.05, 0.7, 0.9, 0.95]$), and by two different initialization methods. The results are shown in Fig. 3(b); (C) given incomplete tensors with $R = 5$ and SNR=$0$dB, the evaluations were performed under varying missing ratios and two different tensor sizes, i.e., $20 \times 20 \times 20$ and $50 \times 50 \times 50$. The results are shown in Fig. 3(c); (D) given incomplete tensors of size $20 \times 20 \times 20$ and SNR=$20$dB, the evaluations were performed under varying missing ratios and two different true rank (i.e., $R = [10, 15]$). The results are shown in Fig. 3(d).

From the results shown in Fig. 3 we can conclude that SVD initialization is slightly better than random initialization in terms of determination of tensor rank. Our model can exactly detect the true tensor rank with 100% accuracy when SNR$\geq10$dB and without missing entries. If the tensor is complete, the accuracy of exactly detecting the tensor rank is 70% with error deviation only $\pm 1$, even under a high noise level of 0dB. On the other hand, if the tensor is almost noise free, the accuracy is 44% with error deviation only $\pm 1$ even under a highly missing ratio of 90%. As both missing data and high noise level are presented, we observe that our model can achieve 62% accuracy with error deviation only $\pm 1$ under the condition of 0dB noise and 70% missing ratio. Note that when the data size is larger, such as $50 \times 50 \times 50$, our model can achieve 90% accuracy even when SNR=$0$dB and missing ratio is 0.9. If the true rank is larger such as $R = 15$, the model can correctly recover the rank from a complete tensor, but fails when missing ratio is larger than 0.5. These results indicate that determination of tensor rank mainly depends on number of observed entries and the true tensor rank. Generally, more observations are necessary if the tensor rank is larger; however, when high noise occurs, the excessive number of observations may not be helpful for rank determination. The more detailed results can be found from Fig. 3.

Fig. 4. Performance comparisons on complete tensors under varying noise levels. Eight algorithms are generally separated into four groups: FaLRTC, FCSA, HardC. and STDC are in the first group; geomCG is the second group; CPWOPT and CPNLS are in the third group; FBCP is the fourth group which obtains the best performance. Note that as SNR is smaller, the superiority of our method is more significant.

5.1.3 Predictive performance

The predictive performance was investigated by comparing our fully Bayesian CP factorization (FBCP) with several state-of-the-art methods. The tensor factorization methods include CP weighted optimization (CP-WOPT) [7], structured/symmetric CPD by nonlinear least squares (CPNLS) [8],[36], and simultaneous tensor decomposition and completion (STDC) [13], while completion based methods include fast low rank tensor completion (FaLRTC) based on optimization of nuclear norm [10], fast composite splitting algorithms (FCSA) [12], hard-completion (HardC.) based on convex optimization and spectral regularization [11], and geometric nonlinear conjugate gradient (geomCG) based on Riemannian optimization techniques [9]. Several experiments were conducted to evaluate the sensitivity of performance w.r.t. two varying conditions, i.e., noise levels and missing ratios. The relative standard error $RSE = \frac{\| \hat{X} - X \|_F}{\| X \|_F}$, where $X$ denotes the true latent tensor and $\hat{X}$ denotes the estimation, is used to evaluate the performance. To ensure the results are statistically consistent, each evaluation was repeated by 50 runs with tensors generated randomly under the same condition. If $Y$ is incomplete, RSE w.r.t. the whole tensor reflects overall performance including training and testing, while RSE w.r.t. missing entries particularly reflects the generalization ability of model. In principle, a same parameter setting was applied to all algorithms, such as initialization of the tensor rank, and the other additional parameters were set to default values of individual algorithms.

In the first experiment, the complete tensors of size $20 \times 20 \times 20$ generated with true rank $R = 5$
TABLE 1
The performance on complete tensors with SNR=20dB.

<table>
<thead>
<tr>
<th>Factorization</th>
<th>RSE</th>
<th>Completion</th>
<th>RSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBCP</td>
<td>0.0189±7e-4</td>
<td>FaLRTC</td>
<td>0.1001±8e-4</td>
</tr>
<tr>
<td>CPWOPT</td>
<td>0.0346±6e-4</td>
<td>FCSA</td>
<td>0.1001±8e-4</td>
</tr>
<tr>
<td>CPNLS</td>
<td>0.0339±1e-3</td>
<td>HardC.</td>
<td>0.1001±8e-4</td>
</tr>
<tr>
<td>STDC</td>
<td>0.1001±8e-4</td>
<td>geomCG</td>
<td>0.0485±7e-4</td>
</tr>
</tbody>
</table>

were considered under varying noise levels with SNR ranging from -5dB to 20dB. The initial rank was set to 10. As shown in Fig. 4, the performance of eight algorithms can be generally divided into four groups. The completion-based methods including FaLRTC, FCSA, HardC. and STDC obtain the similar results as they do not consider the noise in the model. The second group is geomCG that shows a slightly better performance. The third group consists of tensor factorization methods, i.e., CPWOPT and CPNLS, that are better than geomCG. Finally, the fourth group is FBCP that outperforms all other algorithms under all noise levels. The detailed comparisons when the noise is small (i.e., SNR = 20dB) are shown in Table 1. Observe that as the noise level increases, the superiority of FBCP is more significant.

In the second experiment, we consider incomplete tensors of size $20 \times 20 \times 20$ generated with tensor rank $R = 5$ and SNR=30dB. The initial rank was set to 10 in all algorithms. Our goal is to evaluate the predictive performance in terms of tensor completion under varying missing ratios ranging from 0 to 0.9. As shown in Fig. 5, our method significantly outperforms other algorithms under all missing ratios. Observe that factorization-based methods including CPWOPT and CPNLS show better performance than completion-based methods when missing ratio is relatively small, while they are worse than other methods as missing ratio is large, e.g. 0.9. FaLRTC, FCSA and HardC. provide comparable performances since they are based on the similar optimization objectives. geomCG obtains comparable performance with CWOPT and CPNLS when missing ratio is 0, while it fails as missing ratio becomes large. This is because that geomCG requires large number of observations and precisely defined rank. Note that STDC outperforms all algorithms except FBCP as the missing ratio is extremely high. These results demonstrate that our method is also effective for tensor completion even though the extremely sparse tensor is presented.

In the third experiment, we consider incomplete tensors of size $50 \times 50 \times 50$ generated with true rank $R = 5$ and high level of noises with SNR=0dB. As previously, the initial rank was set to 10. Our goal is to evaluate the predictive performance under varying missing ratios when observations are corrupted by high level noises. As shown in Fig. 6, our method outperforms other algorithms significantly under all missing ratios. Similar to the previous experiment, CPWOPT, CPNLS and geomCG works well when missing ratio is relatively low, while they fail when missing ratio is extremely high, and FaLRTC, FCSA and HardC. provide similar performances. Note that even though the missing ratio is extremely high (e.g., 0.95), FBCP can achieve RSE of 0.46±0.09, while all other algorithms fail in the sense that RSE≥1. These results demonstrate that our method can effectively perform the tensor completion even though both the missing ratio and noise are extremely high.

5.2 Image Inpainting

In this section, several real-world applications of image inpainting are used to evaluate the performance of different methods in terms of tensor completion. Three benchmark images shown in Fig. 7 are used in this study. Each colorful image can be...
Fig. 6. The predictive performance with SNR=0dB under varying missing ratios within the range of [0, 0.95].

TABLE 2
The performance of image inpainting evaluated by RSE based on either total image pixels (T) or on only missing image pixels (M).

<table>
<thead>
<tr>
<th>Method</th>
<th>Case A T</th>
<th>Case A M</th>
<th>Case B T</th>
<th>Case B M</th>
<th>Case C T</th>
<th>Case C M</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBCP</td>
<td>0.13</td>
<td>0.13</td>
<td>0.10</td>
<td>0.13</td>
<td>0.13</td>
<td>0.16</td>
</tr>
<tr>
<td>CPWOPT</td>
<td>0.34</td>
<td>0.34</td>
<td>0.15</td>
<td>0.18</td>
<td>0.16</td>
<td>0.18</td>
</tr>
<tr>
<td>CPNLS</td>
<td>0.86</td>
<td>0.88</td>
<td>0.16</td>
<td>0.22</td>
<td>0.24</td>
<td>0.30</td>
</tr>
<tr>
<td>FaLRTC</td>
<td>0.16</td>
<td>0.16</td>
<td>0.21</td>
<td>0.29</td>
<td>0.23</td>
<td>0.29</td>
</tr>
<tr>
<td>FCSA</td>
<td>0.18</td>
<td>0.19</td>
<td>0.20</td>
<td>0.28</td>
<td>0.22</td>
<td>0.28</td>
</tr>
<tr>
<td>HardC.</td>
<td>0.19</td>
<td>0.20</td>
<td>0.23</td>
<td>0.31</td>
<td>0.24</td>
<td>0.30</td>
</tr>
<tr>
<td>STDC</td>
<td>0.67</td>
<td>0.69</td>
<td>0.16</td>
<td>0.22</td>
<td>0.22</td>
<td>0.28</td>
</tr>
</tbody>
</table>

RSE evaluated based on either the whole image or only missing entries are compared in details in Table 2, where the case of (D) is not shown because the ground truth is not available. Observe that FBCP achieves the best performance among seven different methods in all cases, especially in terms of RSE on missing entries. These results demonstrate that our method outperforms state-of-the-art methods in various image inpainting problems.

5.3 Facial Image Synthesis

For face recognition, the ideal solution is to create a robust classifier that is invariant to some factors such as poses and illuminations. However, the images under all possible conditions are not available for model training. On the other hand, face images captured from surveillance videos have various poses and illuminations, which are difficult to be recognized correctly. Hence, there arises the question whether we can generate novel facial images under a specific multiple conditions given images under other conditions. For example, given a profile image of one person, the goal is to predict the frontal image of that person. To handle this problem, tensor methods are highly suitable for modeling a multifactor image ensemble.

Therefore, we introduce a novel application of facial
image synthesis by employing tensor completion approaches.

For illustration, we use the dataset of 3D Basel Face Model [37] that contains an ensemble of facial images of 10 people, each rendered in 9 different poses under 3 different illuminations. All images were decimated and cropped to $68 \times 68$ pixels, and then were represented by a fourth order tensor of size $4624 \times 10 \times 9 \times 3$. As shown in Fig.9, some images were randomly selected as missing images. Our goal is to model the whole image ensemble by a multilinear model and predict the unknown images from observed images under other conditions. We have performed experiments on this dataset by using FBCP, CPWOPT, FaLRTC and HardC. methods, and the initial rank was set to 100. Other algorithms have not been applied since they are either computational intractable for large dataset or not applicable to a higher order tensor with $N > 3$. The experiments with different missing ratios, i.e., 36, 49, 64 and 81 missing images out of 270 total images, were performed and RSE on the total set of images and on the missing images are given. As we known, the completion based algorithms do not consider noise and output the exactly equivalent values for observed data, thus the RSE evaluated only on the missing images are more faithful for comparisons of generalization ability.

As shown in Fig. 10, the visual effects of image synthesis by FBCP is significantly superior to both CPWOPT and FaLRTC. CPWOPT cannot fit well to the observed images thus resulting in poor predictions of unknown images, while predictions by FaLRTC are too smooth and blurred although it is much better than CPWOPT. The detailed performance comparisons are provided in Table3. Observe that completion based methods are much better than factorization based method. However, FBCP, which is also a factorization based approach, significantly outperforms FaLRTC and HardC. under all different
missing ratios, especially in terms of RSE evaluated on unknown images. These results demonstrate the potential advantages of our method for modeling the data ensemble under multiple conditions and for modeling a higher order tensor with $N \geq 4$. There are many potential applications based on this experiment: one is to generate a complete set of images under varying conditions, which can be used to train a robust classifier for face recognition; the other is to infer a frontal image given the image of a novel person under an arbitrary condition, which is quite promising for robust face recognition in surveillance videos.

6 CONCLUSIONS

In this study, we propose a Bayesian CP factorization under a probabilistic framework, which can naturally handle the incomplete and noisy tensor data. By enforcing appropriate priors and hyperpriors over model parameters and hyperparameters, a fully Bayesian treatment has been employed to derive a deterministic solution for model inference. The most significant advantages are automatic determination of tensor rank and superior predictive ability. Moreover, as a fully tuning parameter free approach, our method solves the parameter selection problem in an elegant way, which can also effectively avoid overfitting. In

<table>
<thead>
<tr>
<th>Method</th>
<th>36/270</th>
<th>49/270</th>
<th>64/270</th>
<th>81/270</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T</td>
<td>M</td>
<td>T</td>
<td>M</td>
</tr>
<tr>
<td>FBCP</td>
<td>0.06</td>
<td>0.10</td>
<td>0.10</td>
<td>0.09</td>
</tr>
<tr>
<td>CPWOPT</td>
<td>0.53</td>
<td>0.65</td>
<td>0.56</td>
<td>0.61</td>
</tr>
<tr>
<td>FaLRTC</td>
<td>0.11</td>
<td>0.28</td>
<td>0.13</td>
<td>0.30</td>
</tr>
<tr>
<td>HardC.</td>
<td>0.37</td>
<td>0.37</td>
<td>0.37</td>
<td>0.40</td>
</tr>
</tbody>
</table>
contrast to existing optimization based algorithms, our model can provide uncertainty information of factor matrices and predictions of missing entries. For computational efficiency, we developed an efficient algorithm based on multilinear operations that scales linearly with the tensor size. Empirical results validate effectiveness of the proposed method in terms of discovering the ground truth of tensor rank and capturing the underlying multilinear structure from an extremely sparse tensor. Several real-world applications, such as image completion and image synthesis, demonstrate the superiorities of our method over state-of-the-art tensor factorization and tensor completion approaches. In summary, due to several interesting properties, our method would be attractive in many potential applications.

REFERENCES