Statistics and Its Interface Volume 17 (2024) 199–217

Bayesian tensor-on-tensor regression with efficient computation

Kunbo Wang and Yanxun Xu*

We propose a Bayesian tensor-on-tensor regression approach to predict a multidimensional array (tensor) of arbitrary dimensions from another tensor of arbitrary dimensions, building upon the Tucker decomposition of the regression coefficient tensor. Traditional tensor regression methods making use of the Tucker decomposition either assume the dimension of the core tensor to be known or estimate it via cross-validation or some model selection criteria. However, no existing method can simultaneously estimate the model dimension (the dimension of the core tensor) and other model parameters. To fill this gap, we develop an efficient Markov Chain Monte Carlo (MCMC) algorithm to estimate both the model dimension and parameters for posterior inference. Besides the MCMC sampler, we also develop an ultra-fast optimization-based computing algorithm wherein the maximum a posteriori estimators for parameters are computed, and the model dimension is optimized via a simulated annealing algorithm. The proposed Bayesian framework provides a natural way for uncertainty quantification. Through extensive simulation studies, we evaluate the proposed Bayesian tensor-on-tensor regression model and show its superior performance compared to alternative methods. We also demonstrate its practical effectiveness by applying it to two real-world datasets, including facial imaging data and 3D motion data.

Keywords and phrases: Fractional Bayes Factor, Markov chain Monte Carlo, Tensor-on-tensor regression, Tucker decomposition.

1. INTRODUCTION

Multi-dimensional arrays, also called tensors, are widely used to represent data with complex structures in different fields such as genomics, neuroscience, computer vision, and graph analysis. For example, a multi-tissue experiment (Wang et al., 2019) collects gene expression data in different tissues from different individuals, leading to three-dimensional arrays ($Genes \times Tissues \times Individuals$). Other notable examples include magnetic resonance imaging data (MRI, three-dimensional arrays), functional MRI (fMRI) data (four-dimensional arrays), and facial images (four-dimensional arrays) (Vasilescu and Terzopoulos, 2002; Hasan et al., 2011; Guhaniyogi and Spencer, 2021). In this paper, we focus on the task of tensor-on-tensor regression that predicts one multi-dimensional tensor from another multi-dimensional tensor, e.g., predicting gene expression across multiple tissues for multiple individuals from their clinical/omics data with tensor structures.

One simple approach dealing with tensor-on-tensor regression is to turn tensors into vectors, and then apply classic regression methods. However, such a treatment introduces high-dimensional unstructured vectors and destroys the correlation structure of data, resulting in a huge number of parameters to be estimated and potentially significant loss of information. For example, to predict a response tensor of dimensions $N \times Q_1 \times Q_2$ from a predictor tensor of dimensions $N \times P_1 \times P_2$, the classic linear regression method requires estimating $P_1 \times P_2 \times Q_1 \times Q_2$ parameters, which may cause overfitting or computational issues, especially when the number of parameters is larger than the sample size N.

To reduce the number of free parameters while preserving the correlation structure in modeling tensor data, tensor decomposition techniques have been widely applied (Kolda and Bader, 2009). The two most commonly-used tensor decomposition methods are the PARAFAC/CANDECOMP (CP) decomposition (Harshman, 1970) and Tucker decomposition (Tucker, 1966). The CP decomposition reconstructs a tensor as a linear combination of rank-1 tensors, each one of which is represented as the outer product of a number of vectors. On the other hand, the Tucker decomposition factorizes a tensor into a small core tensor and a set of matrices along each dimension. Both decomposition methods are able to reduce model dimensionality to a manageable size and make parameter estimation more efficient. Compared to CP decomposition, Tucker decomposition allows a more flexible correlation structure processed by the core tensor and the freedom in choosing different orders, making it useful in estimating data with skewed dimensions (Li et al., 2013). In fact, CP decomposition is a special case of Tucker decomposition with the core tensor being superdiagonal.

There is a rich literature on regression methods treating tensors as either predictors or responses in both frequentist and Bayesian statistics. Guo et al. (2012) and Zhou et al. (2013) proposed tensor regression models to predict scalar outcomes from tensor predictors by assuming that the coefficient tensor has a low rank CP decomposition. Li et al.

*Corresponding author.

(2013) later extended the framework by employing Tucker decomposition for the coefficient tensor, and demonstrated that Tucker decomposition is more suitable to deal with tensor predictors of skewed dimensions and gains better accuracy in neuroimaging data analysis. Guhaniyogi et al. (2017) proposed a Bayesian approach to regression with a scalar response on tensor predictors by developing a multiway Dirichlet generalized double Pareto prior on tensor margins after applying CP decomposition to the coefficient tensor. Miranda et al. (2018) developed a Bayesian tensor partition regression model using a generalized linear model with a sparse inducing normal mixture prior to learn the relationship between a matrix response (clinical outcomes) and a tensor predictor (imaging data). Li and Zhang (2017) proposed a parsimonious regression model with tensor response and vector predictors adopting a generalized sparsity principle based on Tucker decomposition. To detect neuronal activation in fMRI experiments, Guhaniyogi and Spencer (2021) developed a Bayesian regression approach with a tensor response on scalar predictors by introducing a novel multiway stick breaking shrinkage prior distribution on tensor structured regression coefficients.

There exist many scientific applications that require methods for predicting a tensor response from another tensor predictor. One typical example in fMRI studies is to detect brain regions activated by an external stimulus or condition (Zhang et al., 2015). Hoff (2015) proposed a tensor-on-tensor bilinear regression framework to handle a special case where the tensor predictor has the same dimension as the tensor response making use of Tucker decomposition. Billio et al. (2018) introduced a Bayesian tensor autoregressive model to tackle tensor-on-tensor regression, and used CP decomposition to provide parsimonious parametrization. Lock (2018) proposed to predict a tensor response from another tensor predictor by assuming that the coefficient tensor has a low-rank CP factorization. Gahrooei et al. (2021) extended the work of Lock (2018) to allow multiple tensor inputs under the Tucker decomposition framework.

Despite advances in methods development for dealing with tensor data, there are some limitations in the aforementioned methods. First, tensor-on-tensor regression methods based on CP decomposition, e.g., Lock (2018), require both the response tensor and the predictor tensor to have the same rank in CP decomposition, making them restrictive when the response and predictor tensors have different ranks. Second, the rank in CP decomposition and the dimension of the core tensor in Tucker decomposition (i.e., model dimension) are essential for statistical inference in tensor-on-tensor regression models. However, they are either assumed known or estimated via cross-validation (Gahrooei et al., 2021) or some model selection criteria, such as Bayesian information criterion (Guhaniyogi and Spencer, 2021). To our best knowledge, there is no existing method that can simultaneously estimate the model dimension and parameters.

In this paper, we develop a novel Bayesian approach for tensor-on-tensor regression based on Tucker decomposition of the coefficient tensor. The main contributions of this work are threefold. First, our Bayesian framework is built upon the flexible Tucker decomposition so that the response tensor and the predictor tensor can have different dimensions in the core tensor. Second, we propose an efficient Markov chain Monte Carlo (MCMC) algorithm to simultaneously estimate the model dimension (the dimension of the core tensor) and parameters. The resulting posterior inference naturally offers us characterization of uncertainty in parameter estimation and prediction. Third, as an alternative to MCMC, we develop an ultra-fast computing algorithm, in which the maximum a posteriori (MAP) estimators for parameters are computed and meanwhile the dimension of the core tensor is optimized via a simulated annealing (SA) algorithm (Kirkpatrick et al., 1983).

The rest of the article is organized as follows. We start with introducing some preliminaries in Section 2. Section 3 describes the proposed Bayesian tensor-on-tensor regression model. We develop an efficient MCMC algorithm to simultaneously estimate the model dimension and parameters in Section 4. An optimization-based ultra-fast computational algorithm for inference is described in Section 5. Section 6 evaluates the proposed approach via simulation studies and comparisons to alternative methods. Section 7 provides real data analyses on facial imaging data and 3D motion data. Section 8 concludes with a discussion.

2. PRELIMINARIES

2.1 Notations

We begin with introducing notations and operations that will be used throughout the paper. We use uppercase blackboard bold characters ($\mathbb{X}$) to denote tensors, bold uppercase characters ($\mathbf{X}$) to denote matrices, and bold lowercase characters ($\mathbf{a}$) to denote vectors. The *order* of a tensor is the number of dimensions. For example, $\mathbb{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ denotes an Nth order tensor, where I_n denotes the dimension of the nth mode, $n = 1, \ldots, N$. The ith entry of a vector $\boldsymbol{a}$ is denoted as a_i; the element (i, j) of a matrix $\mathbf{X}$ is denoted as X_{ij}; and the entries of a tensor are defined by indices enclosed in square brackets: $\mathbb{X}_{[i_1,\cdots,i_N]}$, where $i_n \in \{1, \cdots, I_n\}$ for $n \in \{1, \cdots N\}$. The nth element in a sequence of matrices or vectors is denoted by a subscript in parenthesis. For example, $\mathbf{X}_{(n)}$ denotes the nth matrix in a sequence of matrices, and $\mathbf{x}_{(n)}$ denotes the nth vector in a sequence of vectors.

The *vectorization* of a tensor $\mathbb{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ transforms an Nth order tensor into a column vector $vec\mathbb{X}$ such that the entry $\mathbb{X}_{[i_1,\cdots,i_N]}$ maps to the jth entry of $vec\mathbb{X}$, that is

$$\mathbb{X}_{[i_1,\cdots,i_N]} = vec\mathbb{X}_j\,, \tag{1}$$

where $j = 1 + \sum_{k=1}^{N}(i_k - 1)\prod_{l=1}^{k-1} I_l$. Similarly, $vec\mathbf{X}$ is used to denote the *vectorization* of a matrix $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$ when

$N = 2$ in (1). *Matricization*, also known as unfolding, is the process of transforming a tensor into a matrix. The mode-n matricization of a tensor $\mathbb{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoted by $\mathbb{X}_{(n)} \in \mathbb{R}^{I_n \times J}$ where $J = \prod_{k \neq n} I_k$. The entry $\mathbb{X}_{[i_1, \cdots, i_N]}$ of $\mathbb{X}$ maps to the (i_n, j) element of the resulting matrix $\mathbb{X}_{(n)}$, where

$$j = 1 + \sum_{\substack{k=1 \\ k \neq n}}^{N} (i_k - 1) J_k \quad \text{with} \quad J_k = \prod_{\substack{l=1 \\ l \neq n}}^{k-1} I_l.$$

A more general treatment of the tensor matricization is defined as follows. Let $\mathcal{R} = \{r_1, \cdots, r_L\}$ and $\mathcal{C} = \{c_1, \cdots, c_M\}$ be two sets of indices such that $\mathcal{R} \cup \mathcal{C} = \{1, \cdots, N\}$ and $\mathcal{R} \cap \mathcal{C} = \emptyset$. Then the matricized tensor can be specified by $\mathbb{X}_{(\mathcal{R} \times \mathcal{C})} \in \mathbb{R}^{J \times K}$, where $J = \prod_{n \in \mathcal{R}} I_n$ and $K = \prod_{n \in \mathcal{C}} I_n$. And the entry $\mathbb{X}_{[i_1, \cdots, i_N]}$ maps to the (j, k) element of the matrix $\mathbb{X}_{(\mathcal{R} \times \mathcal{C})}$, that is

$$\mathbb{X}_{[i_1, \cdots, i_N]} = \left(\mathbb{X}_{(\mathcal{R} \times \mathcal{C})}\right)_{jk}, \tag{2}$$

where

$$j = 1 + \sum_{l=1}^{L} \left[(i_{r_l} - 1) \prod_{l'=1}^{l-1} I_{r_{l'}} \right],$$

and

$$k = 1 + \sum_{m=1}^{M} \left[(i_{c_m} - 1) \prod_{m'=1}^{m-1} I_{c_{m'}} \right].$$

The Kronecker product of matrices $\mathbf{U} \in \mathbb{R}^{I \times J}$, and $\mathbf{V} \in \mathbb{R}^{K \times L}$ is denoted by $\mathbf{U} \otimes \mathbf{V}$ with the detailed definition and properties shown in A.1. The product of a tensor and a matrix in mode n is defined as the *n-mode product*. The n-mode product of $\mathbb{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ is denoted by $\mathbb{X} \times_n \mathbf{U}$, resulting in a new tensor $\mathbb{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}$ where the $[i_1, \cdots i_{n-1}, j, i_{n+1}, \cdots i_N]$ entry is defined by

$$\mathbb{Y}_{[i_1, \cdots, i_{n-1}, j, i_{n+1}, \cdots, i_N]} = \sum_{i_n=1}^{I_n} \mathbb{X}_{[i_1, \cdots, i_N]} U_{j i_n}.$$

An important fact regarding the n-mode product is that given matrices $\mathbf{U} \in \mathbb{R}^{J_1 \times I_n}$, $\mathbf{V} \in \mathbb{R}^{J_2 \times I_m}$ with $m \neq n$, and tensor $\mathbb{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, then

$$\mathbb{X} \times_n \mathbf{U} \times_m \mathbf{V} = (\mathbb{X} \times_n \mathbf{U}) \times_m \mathbf{V} = (\mathbb{X} \times_m \mathbf{V}) \times_n \mathbf{U}.$$

For two tensors $\mathbb{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N \times P_1 \times \cdots \times P_L}$, and $\mathbb{Y} \in \mathbb{R}^{P_1 \times \cdots \times P_L \times J_1 \times \cdots \times J_M}$, the *contracted tensor product* $\langle \mathbb{X}, \mathbb{Y} \rangle_L$ is defined as

$$\mathbb{Z} = \langle \mathbb{X}, \mathbb{Y} \rangle_L \in \mathbb{R}^{I_1 \times \cdots \times I_N \times J_1 \times \cdots \times J_M}$$

with

$$\mathbb{Z}_{[i_1, \cdots, i_N, j_1, \cdots, j_M]} =$$

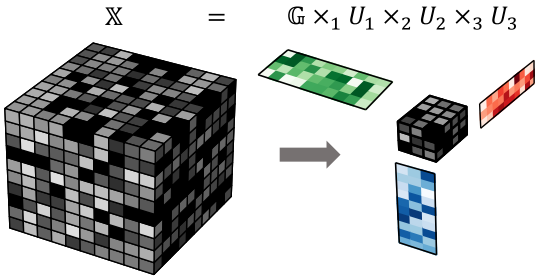


Figure 1. Illustration of Tucker decomposition. Here, the core tensor is of dimension $(4, 3, 3)$.

$$\sum_{p_1=1}^{P_1} \cdots \sum_{p_L=1}^{P_L} \mathbb{X}_{[i_1, \cdots, i_N, p_1, \cdots, p_L]} \mathbb{Y}_{[p_1, \cdots, p_L, j_1, \cdots, j_M]}.$$

It can be shown that for two matrices $\mathbf{U} \in \mathbb{R}^{I \times P}$ and $\mathbf{V} \in \mathbb{R}^{P \times J}$, the contracted product $\langle \mathbf{U}, \mathbf{V} \rangle_1$ is equivalent to the standard matrix product $\mathbf{UV}$. Therefore, the contracted product of two tensors can be regarded as an extension of the usual matrix product to higher-order operands.

2.2 Tucker decomposition

The proposed Bayesian tensor-on-tensor model is built upon Tucker decomposition (Tucker, 1966), which decomposes a tensor $\mathbb{B} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ into a core tensor $\mathbb{G}$ and a set of factor matrices $\mathbf{A}_{(n)}$, $n = 1, \ldots, N$, denoted by

$$\mathbb{B} = [\![\mathbb{G}; \mathbf{A}_{(1)}, \mathbf{A}_{(2)}, \cdots, \mathbf{A}_{(N)}]\!].$$

Or equivalently,

$$\mathbb{B} = \mathbb{G} \times_1 \mathbf{A}_{(1)} \times_2 \mathbf{A}_{(2)} \cdots \times_N \mathbf{A}_{(N)},$$

with $\mathbb{G} \in \mathbb{R}^{J_1 \times \cdots \times J_N}$ being the core tensor and $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times J_n}$ being the factor matrix in mode n, for $n = 1, \cdots, N$, forming a sequence of matrices. The order of $\mathbb{G}$ can be the same as the order of $\mathbb{B}$, but more often, we are interested in compressing the information of $\mathbb{B}$ to a smaller size of $\mathbb{G}$ than $\mathbb{B}$. CP decomposition (Harshman, 1970) is a special case of Tucker decomposition wherein the core tensor is superdiagonal.

3. A BAYESIAN TENSOR-ON-TENSOR REGRESSION MODEL

Our task is to predict a tensor response $\mathbb{Y} \in \mathbb{R}^{N \times Q_1 \times \cdots \times Q_M}$ from a tensor predictor $\mathbb{X} \in \mathbb{R}^{N \times P_1 \times \cdots \times P_L}$. We propose a Bayesian tensor-on-tensor regression framework by extending the standard multivariate linear regression model from matrices to tensors:

$$\mathbb{Y} = \langle \mathbb{X}, \mathbb{B} \rangle_L + \mathbb{E}, \tag{3}$$

where $\mathbb{B} \in \mathbb{R}^{P_1 \times \cdots \times P_L \times Q_1 \times \cdots \times Q_M}$ denotes the coefficient tensor. In this paper, we assume that each element of

$\mathbb{E} \in \mathbb{R}^{N \times Q_1 \times \cdots \times Q_M}$ follows $N(0, \sigma^2)$ independently for illustration simplicity. More flexible covariance structures are discussed in Appendix A.6. The first L modes of $\mathbb{B}$ contract the dimensions of $\mathbb{X}$ and the last M modes of $\mathbb{B}$ match the modes of $\mathbb{Y}$. For each of the N observations, there are a total of $(\prod_{m=1}^{M} Q_m)$ responses and $(\prod_{l=1}^{L} P_l)$ predictors. The model can be reformulated into a matrix form as follows,

$$\mathbb{Y}_{(1)} = \mathbb{X}_{(1)} \mathbb{B}_{(\mathcal{P} \times \mathcal{Q})} + \mathbb{E}_{(1)}, \tag{4}$$

where $\mathcal{P} = \{1, \cdots, L\}$, $\mathcal{Q} = \{L+1, \cdots, L+M\}$, and each row of $\mathbb{E}_{(1)}$ independently follows $N(0, \sigma^2 \mathbf{I}_d)$ with $\mathbf{I}_d$ being an identity matrix of dimension $(\prod_{m=1}^{M} Q_m)$. From the equivalence of (3) and (4), it is clear that our model (3) supports linear relations between responses and predictors for each observation.

If $\mathbb{B}$ is unconstrained, the estimation of $\mathbb{B}$ can be obtained by conducting separate ordinary least squares (OLS) regressions for each of the $\prod_{m=1}^{M} Q_m$ responses in $\mathbb{Y}_{(1)}$ over $\mathbb{X}_{(1)}$ by equation (4) in the frequentist framework. However, the solution is not well-defined if the number of observations N is less than the number of responses $\prod_{m=1}^{M} Q_m$. Even if the solution is well-defined, separate OLS of equation (4) does not consider the correlation structure within the response $\mathbb{Y}$ and predictor $\mathbb{X}$ and between them. Moreover, the total number of parameters in this case is $\prod_{l=1}^{L} P_l \prod_{m=1}^{M} Q_m$, which can be computationally challenging due to its gigantic size.

In our model, we assume that $\mathbb{B}$ follows the Tucker decomposition defined by

$$\mathbb{B} = [\![\mathbb{G}; \mathbf{U}_{(1)}, \cdots, \mathbf{U}_{(L)}, \mathbf{V}_{(1)}, \cdots, \mathbf{V}_{(M)}]\!], \tag{5}$$

where $\mathbb{G}$ denotes the core tensor of dimensions $R_1 \times \cdots \times R_L \times S_1 \times \cdots \times S_M$, $\mathbf{U}_{(l)} \in \mathbb{R}^{P_l \times R_l}$ denotes the factor matrix corresponding to the mode l in $\mathbb{B}$ for $l = 1, \cdots, L$, and $\mathbf{V}_{(m)} \in \mathbb{R}^{Q_m \times S_m}$ denotes the factor matrix corresponding to the mode $L+m$ in $\mathbb{B}$ for $m = 1, \cdots, M$.

We complete the proposed Bayesian tensor-on-tensor regression model by assigning priors to the core tensor $\mathbb{G}$, $\{\mathbf{U}_{(l)}\}_{l=1}^{L}$, $\{\mathbf{V}_{(m)}\}_{m=1}^{M}$, and σ^2. For the core tensor $\mathbb{G}$, we consider a normal prior, that is $vec\mathbb{G} \sim N(\boldsymbol{\mu}_G, \boldsymbol{\Sigma}_G)$ where $\boldsymbol{\Sigma}_G$ is diagonal. For factor matrices $\{\mathbf{U}_{(l)}\}_{l=1}^{L}$ and $\{\mathbf{V}_{(m)}\}_{m=1}^{M}$, taking $\mathbf{U}_{(l)}$ and $\mathbf{V}_{(m)}$ as examples, we assign normal priors for $vec\mathbf{U}^{(l)}$ and $vec\mathbf{V}^{(m)}$. That is, $vec\mathbf{U}_{(l)} \sim N(\boldsymbol{\mu}_{U_l}, \boldsymbol{\Sigma}_{U_l})$, and $vec\mathbf{V}_{(m)} \sim N(\boldsymbol{\mu}_{V_m}, \boldsymbol{\Sigma}_{V_m})$, where $\boldsymbol{\Sigma}_{U_l}$ and $\boldsymbol{\Sigma}_{V_m}$ are diagonal matrices. Usually we choose $\boldsymbol{\mu}_{U_l} = \boldsymbol{\mu}_U$, $\boldsymbol{\mu}_{V_m} = \boldsymbol{\mu}_V$, $\boldsymbol{\Sigma}_{U_l} = \boldsymbol{\Sigma}_U$, and $\boldsymbol{\Sigma}_{V_m} = \boldsymbol{\Sigma}_V$ for all $l = 1, \cdots, L$ and $m = 1, \cdots, M$. Lastly we assign an inverse gamma prior distribution for σ^2: $\sigma^2 \sim IG(\alpha, \beta)$. A discussion on other choices of prior distributions and covariance structures is provided in Appendix A.6.

4. POSTERIOR INFERENCE

We conduct posterior inference using Markov chain Monte Carlo (MCMC) simulations. Given the dimension $\boldsymbol{\theta} = (R_1, \ldots, R_L, S_1, \cdots, S_M)$ of the core tensor, we update $\mathbf{U}_{(l)}$, $\mathbf{V}_{(m)}$, $\mathbb{G}$, and σ^2 using Gibbs sampling transition probabilities for posterior updates, the details of which will be given in Section 4.1. The posterior update of the core tensor dimension $\boldsymbol{\theta}$ is challenging since the dimensions of $\mathbf{U}_{(l)}$, $\mathbf{V}_{(m)}$, and $\mathbb{G}$ change when $\boldsymbol{\theta}$ varies. A reversible jump (RJ) MCMC (Green, 1995) algorithm is a natural choice for such a trans-dimensional update, however, it is difficult to construct a practicable RJ scheme due to the high-dimensionality of the problem. To address this challenge, we will develop an efficient Metropolis-Hastings (MH) algorithm to update $\boldsymbol{\theta}$ building upon the idea of fractional Bayes factor (Lee et al., 2016; O'Hagan, 1995) in Section 4.2. The R code for the proposed algorithms with demonstrating examples can be found at GitHub.

4.1 Posterior inference given the dimension of the core tensor

Given the dimension $\boldsymbol{\theta} = (R_1, \cdots, R_L, S_1, \cdots, S_M)$ of the core tensor $\mathbb{G}$, we derive the full conditional posterior distributions of $\{\mathbf{U}_{(l)}\}_{l=1}^{L}$, $\{\mathbf{V}_{(m)}\}_{m=1}^{M}$, $\mathbb{G}$, and σ^2 in closed forms. Without loss of generality, we first derive the full conditional posterior distribution of $\mathbf{U}_{(1)}$. The full conditional posterior distributions of $\{\mathbf{U}_{(2)}, \cdots, \mathbf{U}_{(L)}\}$ can be derived in the same manner.

By properties of n-mode product of tensor and Tucker decomposition, we have

$$\mathbb{B} = \mathbb{G} \times_2 \mathbf{U}_{(2)} \cdots \times_L \mathbf{U}_{(L)} \times_{L+1} \mathbf{V}_{(1)} \cdots \times_{L+M} \mathbf{V}_{(M)} \times_1 \mathbf{U}_{(1)}.$$

Let $\mathbb{B}_{(-)}$ denote $\mathbb{G} \times_2 \mathbf{U}_{(2)} \cdots \times_L \mathbf{U}_{(L)} \times_{L+1} \mathbf{V}_{(1)} \cdots \times_{L+M} \mathbf{V}_{(M)}$, then $\mathbb{B}_{(-)} \in \mathbb{R}^{R_1 \times P_2 \times \cdots \times P_L \times Q_1 \times \cdots \times Q_M}$, and

$$\mathbb{B} = \mathbb{B}_{(-)} \times_1 \mathbf{U}_{(1)}.$$

We denote the contracted product of $\langle \mathbb{B}_{(-)}, \mathbb{X} \rangle_{P_2, \cdots, P_N}$ by a new tensor called $\mathbb{C}$, then tensor $\mathbb{C} \in \mathbb{R}^{R_1 \times N \times P_1 \times Q_1 \times \cdots Q_M}$. By tensor matricization of $\mathbb{C}$ into $\mathbb{C}_{(\mathcal{R} \times \mathcal{C})} \in \mathbb{R}^{N \prod_{m=1}^{M} Q_m \times R_1 P_1}$, where $\mathcal{R} = \{N, Q_1, \cdots, Q_M\}$, and $\mathcal{C} = \{R_1, P_1\}$, we can rewrite our model (3) as follows:

$$vec\mathbb{Y} = \mathbb{C}_{(\mathcal{R} \times \mathcal{C})} \times vec\mathbf{U}_{(1)} + vec\mathbb{E}. \tag{6}$$

The proof for equation (6) is given in Appendix A.2.

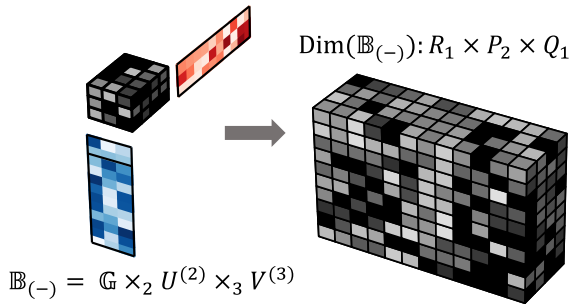
Following (6), we can easily derive that the full conditional posterior distribution of $vec\mathbf{U}_{(1)}$ is normally distributed:

$$p(vec\mathbf{U}_{(1)} \mid vec\mathbb{Y}, \mathbb{X}, \sigma^2, \mathbf{V}_{(m)}, \mathbf{U}_{(l)}\ l \neq 1) \sim N(\boldsymbol{\mu}'_U, \boldsymbol{\Sigma}'_U), \tag{7}$$

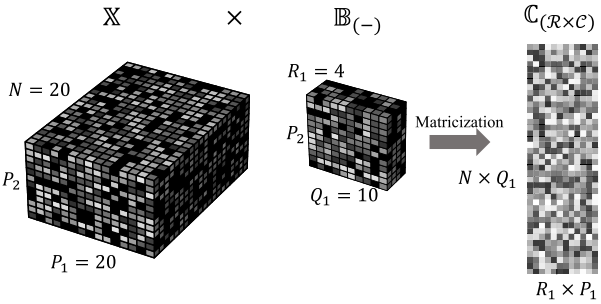
where

$$\boldsymbol{\Sigma}'_U = \left(\frac{\mathbb{C}^T_{(\mathcal{R} \times \mathcal{C})} \mathbb{C}_{(\mathcal{R} \times \mathcal{C})}}{\sigma^2} + \boldsymbol{\Sigma}_U^{-1} \right)^{-1},$$

$$\boldsymbol{\mu}'_U = \boldsymbol{\Sigma}'_U \left(\frac{\mathbb{C}^T_{(\mathcal{R} \times \mathcal{C})} vec\mathbb{Y}}{\sigma^2} + \boldsymbol{\Sigma}_U^{-1} \boldsymbol{\mu}_U \right).$$



(a) Calculate $\mathbb{B}_{(-)}$.



(b) Calculate $\mathbb{C}_{(\mathcal{R}\times\mathcal{C})}$.

Figure 2. Illustration of updating $\mathbf{U}_{(1)}$.

Figure 2 presents an illustration of updating $\mathbf{U}_{(1)}$.

We then derive the conditional distributions of $\mathbf{V}_{(m)}$ given σ^2, $\{\mathbf{U}_{(l)}\}_{l=1}^{L}$, $\mathbf{V}_{(k)}$ for $k \neq m$, and $\mathbb{G}$. Without loss of generality, we derive the full conditional posterior distribution of $\mathbf{V}_{(1)}$ below.

Denote the contracted product of the tensor $\mathbb{G} \times_1 \mathbf{U}_{(1)} \cdots \times_L \mathbf{U}_{(L)} \times_{L+2} \mathbf{V}_{(2)} \cdots \times_{L+M} \mathbf{V}_{(M)}$ and the tensor $\mathbb{X}$ by a new tensor $\mathbb{D}$, where $\mathbb{D} \in \mathbb{R}^{N \times S_1 \times Q_2 \times \cdots \times Q_M}$. We then matricize $\mathbb{D}$ into a matrix $\mathbb{D}_{(\mathcal{R}\times\mathcal{C})} \in \mathbb{R}^{N \prod_{m=2}^{M} Q_m \times S_1}$ and write

$$\mathbb{Y}_{(2)} = \mathbf{V}_{(1)} \times (\mathbb{D}_{(\mathcal{R}\times\mathcal{C})})^T + \mathbb{E}_{(2)}, \tag{8}$$

where $\mathbb{Y}_{(2)} \in \mathbb{R}^{Q_1 \times N \prod_{m=2}^{M} Q_m}$ is the mode-2 matricization of tensor $\mathbb{Y}$. The proof of equation (8) can be found in Appendix A.3. Let $\tilde{\mathbf{Y}} = \mathbb{Y}_{(2)}^T$. Given that $\mathbf{V}_{(1)}$ follows a normal distribution with a diagonal covariance matrix, we can rewrite (8) as

$$vec\tilde{\mathbf{Y}} = \left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})}\right) \times vec\mathbf{V}_{(1)}^T + vec\left(\mathbb{E}_{(2)}\right)^T,$$

where $\mathbf{I}_{Q_1}$ denotes an identity matrix of size Q_1. Given that the prior distribution of $vec\mathbf{V}_{(1)}$ is a normal $N(\boldsymbol{\mu}_V, \boldsymbol{\Sigma}_V)$ with diagonal $\boldsymbol{\Sigma}_V$, the prior distribution of $vec\mathbf{V}_{(1)}^T$ is also a normal distribution $N(\tilde{\boldsymbol{\mu}}_V, \tilde{\boldsymbol{\Sigma}}_V)$ with a diagonal covariance matrix. Then the full conditional posterior distribution of $vec\mathbf{V}_{(1)}{}^T$ is normally distributed:

$$p(vec\mathbf{V}_{(1)}^T \mid vec\tilde{\mathbf{Y}}, \mathbb{X}, \sigma^2, \mathbf{U}_{(l)}, \mathbf{V}_{(m)}\ m \neq 1) \sim\ N(\tilde{\boldsymbol{\mu}}_V^{'}, \tilde{\boldsymbol{\Sigma}}_V^{'}), \tag{9}$$

where

$$\tilde{\boldsymbol{\Sigma}}_V^{'} = \left(\frac{\left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})}\right)^T \left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})}\right)}{\sigma^2} + \tilde{\boldsymbol{\Sigma}}_V^{-1}\right)^{-1},$$

$$\tilde{\boldsymbol{\mu}}_V^{'} = \tilde{\boldsymbol{\Sigma}}_V^{'} \left(\frac{\left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})}\right)^T vec\tilde{\mathbf{Y}}}{\sigma^2} + \tilde{\boldsymbol{\Sigma}}_V^{-1}\tilde{\boldsymbol{\mu}}_V\right). \tag{10}$$

The posterior distribution of the core tensor $\mathbb{G}$ is more complex than the posterior of $\mathbf{U}_l's$ and $\mathbf{V}_m's$. First, we have

$$\begin{aligned}\mathbb{Y}_{(1)} &= \mathbb{X}_{(1)}\mathbb{B}_{(\mathcal{P}\times\mathcal{Q})} + \mathbb{E}_{(1)} \\ &= \mathbb{X}_{(1)}\left(\mathbf{U}_{(L)} \otimes \cdots \otimes \mathbf{U}_{(1)}\right)\mathbb{G}_{(\mathcal{R}\times\mathcal{C})} \\ &\quad \times \left(\mathbf{V}_{(M)} \otimes \cdots \otimes \mathbf{V}_{(1)}\right)^T + \mathbb{E}_{(1)}.\end{aligned} \tag{11}$$

Let $\mathbf{U}$ denote $\left(\mathbf{U}_{(L)} \otimes \cdots \otimes \mathbf{U}_{(1)}\right)$, and $\mathbf{V}$ denote $\left(\mathbf{V}_{(M)} \otimes \cdots \otimes \mathbf{V}_{(1)}\right)$, we have

$$\mathbb{Y}_{(1)} = (\mathbb{X}_{(1)}\mathbf{U})\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}\mathbf{V}^T + \mathbb{E}_{(1)}. \tag{12}$$

If we further let $\tilde{\tilde{\mathbf{Y}}}$ denote $\mathbb{Y}_{(1)}\mathbf{V}(\mathbf{V}^T\mathbf{V})^{-1}$, and $\tilde{\tilde{\mathbf{E}}}$ denote $\mathbb{E}_{(1)}\mathbf{V}(\mathbf{V}^T\mathbf{V})^{-1}$, we can rewrite (12) as

$$\tilde{\tilde{\mathbf{Y}}} = (\mathbb{X}_{(1)}\mathbf{U})\mathbb{G}_{(\mathcal{R}\times\mathcal{C})} + \tilde{\tilde{\mathbf{E}}},$$

or

$$vec\tilde{\tilde{\mathbf{Y}}} = (\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U}))vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})} + vec\tilde{\tilde{\mathbf{E}}},$$

where $\mathbf{I}_S$ denotes an $S \times S$ identity matrix with $S = \prod_{m=1}^{M} S_m$, and $vec\tilde{\tilde{\mathbf{E}}}$ is normally distributed with mean $\mathbf{0}$ and block diagonal covariance matrix $\sigma^2(\mathbf{V}^T\mathbf{V})^{-1} \otimes \mathbf{I}_N$. Then $vec\tilde{\tilde{\mathbf{Y}}} \sim N(\boldsymbol{\mu}_{\tilde{\tilde{Y}}}, \boldsymbol{\Sigma}_{\tilde{\tilde{Y}}})$ with

$$\begin{aligned}\boldsymbol{\mu}_{\tilde{\tilde{Y}}} &= (\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U})) \times vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}, \\ \boldsymbol{\Sigma}_{\tilde{\tilde{Y}}} &= \sigma^2(\mathbf{V}^T\mathbf{V})^{-1} \otimes \mathbf{I}_N.\end{aligned}$$

Given that the prior distribution of $vec\mathbb{G}$ is $N(\boldsymbol{\mu}_G, \boldsymbol{\Sigma}_G)$ with a diagonal covariance $\boldsymbol{\Sigma}_G$, $vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}$ is also normally distributed with $\tilde{\boldsymbol{\mu}}_G$ and diagonal covariance $\tilde{\boldsymbol{\Sigma}}_G$ by rearranging elements of $\boldsymbol{\mu}_G$ and $\boldsymbol{\Sigma}_G$. Then the full conditional posterior distribution of $vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}$ is a normal distribution with

$$\begin{aligned}&\tilde{\boldsymbol{\mu}}_G^{'} = \tilde{\boldsymbol{\Sigma}}_G^{'}\left((\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U}))^T(\boldsymbol{\Sigma}_{\tilde{\tilde{\mathbf{Y}}}})^{-1}vec\tilde{\tilde{\mathbf{Y}}} + \tilde{\boldsymbol{\Sigma}}_G^{-1}\tilde{\boldsymbol{\mu}}_G\right), \\ &\tilde{\boldsymbol{\Sigma}}_G^{'} = \\ &\left((\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U}))^T(\boldsymbol{\Sigma}_{\tilde{\tilde{\mathbf{Y}}}})^{-1}(\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U})) + (\tilde{\boldsymbol{\Sigma}}_G)^{-1}\right)^{-1}.\end{aligned} \tag{13}$$

Lastly, deriving the full conditional posterior distribution of σ^2 is straightforward:

$$(14)\quad p(\sigma^2 \mid \mathbb{Y}, \mathbb{X}, \{\mathbf{U}_{(l)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}\}_{m=1}^{M}, \mathbb{G}) \;\sim\; IG(\alpha', \beta'),$$

where $\alpha' = \alpha + \frac{NQ}{2}$, $\beta' = \beta + \frac{\|\mathbb{Y} - \langle \mathbb{X}, \mathbb{B} \rangle_L\|_F^2}{2}$ with $\mathbb{B}$ defined in (5), and $Q = \prod_{m=1}^{M} Q_m$.

4.2 Updating the model dimension

In this subsection, we show how to simultaneously update the dimension of the core tensor and estimate model parameters. Denote $\boldsymbol{\theta} = (R_1, \ldots, R_L, S_1, \cdots, S_M)$, and we assign a prior distribution $\pi(\boldsymbol{\theta})$ to $\boldsymbol{\theta}$, in this study, a uniform distribution over all candidates of $\boldsymbol{\theta}$. Since conditional posterior distribution of $\boldsymbol{\theta}$ is not in closed form, we employ a trans-dimensional Metropolis-Hastings (MH) sampler to update $\boldsymbol{\theta}$. The most challenging task is to design a good proposal distribution that can result in a reasonable acceptance rate given the fact that the dimensions of $\{\mathbf{U}_{(l)}\}_{l=1}^{L}$, $\{\mathbf{V}_{(m)}\}_{m=1}^{M}$, and $\mathbb{G}$ change when $\boldsymbol{\theta}$ varies.

To address the challenge, we construct our proposal distribution building upon the idea of fractional Bayes factor (O'Hagan, 1995; Lee et al., 2016). Assuming that at iteration $t-1$ of the MCMC sampler $\boldsymbol{\theta}^{(t-1)} = (R_1^{(t-1)}, \cdots, R_L^{(t-1)}, S_1^{(t-1)} \cdots, S_M^{(t-1)})$, at iteration t we generate a candidate $\tilde{\boldsymbol{\theta}}$ from the "neighbor" of $\boldsymbol{\theta}^{(t-1)}$ defined as $O(\boldsymbol{\theta}^{(t-1)}) := \{\tilde{\boldsymbol{\theta}} \in \boldsymbol{\Theta} : \|\boldsymbol{\theta}^{(t-1)} - \tilde{\boldsymbol{\theta}}\|_{L_1} = 1\}$, where $\boldsymbol{\Theta}$ is the parameter space for $\boldsymbol{\theta}$. In this work, we propose to generate $\tilde{\boldsymbol{\theta}}$ uniformly over all candidates in $O(\boldsymbol{\theta}^{(t-1)})$, denoted by $q(\tilde{\boldsymbol{\theta}} \mid \boldsymbol{\theta}^{(t-1)})$. To calculate the acceptance rate of the proposed $\tilde{\boldsymbol{\theta}}$ in the MH step, we denote $\boldsymbol{\xi} = \left(\{\mathbf{U}_{(l)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}\}_{m=1}^{M}, \mathbb{G}, \sigma^2\right)$, and write the likelihood function as the multiplication of two parts:

$$p(\mathbb{Y} \mid \boldsymbol{\xi}, \boldsymbol{\theta}) = \underbrace{p(\mathbb{Y} \mid \boldsymbol{\xi}, \boldsymbol{\theta})^{b}}_{\text{training}} \times \underbrace{p(\mathbb{Y} \mid \boldsymbol{\xi}, \boldsymbol{\theta})^{(1-b)}}_{\text{testing}},$$

where b is small and $0 < b < 1$. The key idea here is to utilize a fraction b of the data as the training data to propose new parameters $\boldsymbol{\xi}$ associated with $\tilde{\boldsymbol{\theta}}$ so that the new values can be accepted with a reasonable acceptance probability. In practice, we usually choose a small value of b, e.g., between 0.01 and 0.1, to obtain a reasonable acceptance rate. In particular, we update $\boldsymbol{\theta}^{(t)}$ as follows:

- Generate $\tilde{\boldsymbol{\theta}}$ from $q(\cdot \mid \boldsymbol{\theta}^{(t-1)})$.
- Generate $\tilde{\boldsymbol{\xi}}$ from a distribution proportional to

$$p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^{b} \times p(\tilde{\boldsymbol{\xi}} \mid \tilde{\boldsymbol{\theta}}),$$

  which is the posterior of $\boldsymbol{\xi}$ based on the training portion conditional on $\tilde{\boldsymbol{\theta}}$. The detailed conditional posterior distributions are shown in Appendix A.4.
- Generate $\tilde{\tilde{\boldsymbol{\xi}}}$ from a distribution proportional to

$$p(\mathbb{Y} \mid \tilde{\tilde{\boldsymbol{\xi}}}, \boldsymbol{\theta}^{(t-1)})^{b} \times p(\tilde{\tilde{\boldsymbol{\xi}}} \mid \boldsymbol{\theta}^{(t-1)}),$$

  which is the posterior of $\boldsymbol{\xi}$ based on the training portion conditional on $\boldsymbol{\theta}^{(t-1)}$.
- Accept $\tilde{\boldsymbol{\theta}}$ with probability $\min\left(1, A(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta}^{(t-1)})\right)$ where

$$(15)\quad A(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta}^{(t-1)}) = \frac{\pi(\tilde{\boldsymbol{\theta}}) q(\boldsymbol{\theta}^{(t-1)} \mid \tilde{\boldsymbol{\theta}})}{\pi(\boldsymbol{\theta}^{(t-1)}) q(\tilde{\boldsymbol{\theta}} \mid \boldsymbol{\theta}^{(t-1)})} \times \underbrace{\frac{p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^{(1-b)}}{p(\mathbb{Y} \mid \tilde{\tilde{\boldsymbol{\xi}}}, \boldsymbol{\theta}^{(t-1)})^{(1-b)}}}_{(*)}.$$

The (*) part in equation (15) coincides with the fractional Bayes factor given in O'Hagan (1995). The detailed proof is given in Appendix A.5.

Algorithm 1 MCMC Sampler

1: Input data $\mathbb{X}, \mathbb{Y}$.
2: Initialize the core tensor dimension $\boldsymbol{\theta}^{(0)}$.
3: **for** $t = 1, \cdots, T$ **do**
4: Propose $\tilde{\boldsymbol{\theta}} \in O(\boldsymbol{\theta}^{(t-1)})$ from $q(\cdot \mid \boldsymbol{\theta}^{(t-1)})$.
5: Sample $\left(\{\tilde{\mathbf{U}}_{(l)}\}_{l=1}^{L}, \{\tilde{\mathbf{V}}_{(m)}\}_{m=1}^{M}, \tilde{\mathbb{G}}, \tilde{\sigma}^2\right)$ from full posterior based on the training portion b conditional on $\tilde{\boldsymbol{\theta}}$ according to Appendix A.4.
6: Sample $\left(\{\tilde{\tilde{\mathbf{U}}}_{(l)}\}_{l=1}^{L}, \{\tilde{\tilde{\mathbf{V}}}_{(m)}\}_{m=1}^{M}, \tilde{\tilde{\mathbb{G}}}, \tilde{\tilde{\sigma}}^2\right)$ from full posterior based on the training portion b conditional on $\boldsymbol{\theta}^{(t-1)}$ according to Appendix A.4.
7: Given $\left(\{\tilde{\mathbf{U}}_{(l)}\}_{l=1}^{L}, \{\tilde{\mathbf{V}}_{(m)}\}_{m=1}^{M}, \tilde{\mathbb{G}}, \tilde{\sigma}^2, \tilde{\boldsymbol{\theta}}\right)$ and $\left(\{\tilde{\tilde{\mathbf{U}}}_{(l)}\}_{l=1}^{L}, \{\tilde{\tilde{\mathbf{V}}}_{(m)}\}_{m=1}^{M}, \tilde{\tilde{\mathbb{G}}}, \tilde{\tilde{\sigma}}^2, \boldsymbol{\theta}^{(t-1)}\right)$, calculate the acceptance probability according to (15).
8: Update and save $\boldsymbol{\theta}^{(t)}$ given the acceptance probability.
9: Sample from full conditional posterior distributions:
10: Sample $\{vec\mathbf{U}_{(l)}^{(t)}\}_{l=1}^{L}$ according to (7).
11: Sample $\{vec\left(\mathbf{V}_{(m)}^{T}\right)^{(t)}\}_{m=1}^{M}$ according to (9).
12: Sample $vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}^{(t)}$ according to (11).
13: Get $\{\mathbf{U}_{(l)}^{(t)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}^{(t)}\}_{m=1}^{M}, \mathbb{G}^{(t)}$ according to (1), (2).
14: Sample $\left(\sigma^2\right)^{(t)}$ according to (14).
15: Calculate $\mathbb{B}^{(t)}$ given by

$$\mathbb{B}^{(t)} = [\![\mathbb{G}^{(t)}; \{\mathbf{U}_{(l)}^{(t)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}^{(t)}\}_{m=1}^{M}]\!].$$

16: Save $\left(\{\mathbf{U}_{(l)}^{(t)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}^{(t)}\}_{m=1}^{M}, \mathbb{G}^{(t)}, \mathbb{B}^{(t)}, \left(\sigma^2\right)^{(t)}\right)$.
17: **end for**

We summarize the full MCMC sampler in Algorithm 1. For predictive inference, it is straightforward based on the posterior samples obtained from Algorithm 1. Given new data $\mathbb{X}_{new}$ of $\tilde{N}$ samples, we can easily sample $\mathbb{Y}_{new}$ predictions according to

$$(16)\qquad vec\hat{\mathbb{Y}}_{new} \;\sim\; N(vec\left(\langle \mathbb{X}_{new}, \hat{\mathbb{B}} \rangle_L\right), \hat{\sigma}^2 \mathbf{I}_{\tilde{N}Q}),$$

where $Q = \prod_{m=1}^{M} Q_m$, and $\hat{\mathbb{B}}$ is calculated by (5) using post-burn-in samples of $\mathbb{G}, \{\mathbf{U}_{(l)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}\}_{m=1}^{M}$, and σ^2.

5. FAST COMPUTING ALGORITHM

In practice, the proposed MCMC sampler involves generating samples from high-dimensional conditional posterior distributions at each iteration, which can be time-consuming. In this section, we propose an ultra-fast optimization-based computing algorithm as an alternative for posterior inference using the maximum a posteriori probability (MAP) estimators. Given the dimension of the core tensor, the MAP estimators of $\{\mathbf{U}_{(l)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}\}_{m=1}^{M}, \mathbb{G}$, and σ^2 can be computed in closed forms. Specifically, the MAP estimator of $\mathbf{U}_{(1)}$ is

(17)
$$vec\mathbf{U}_{(1)}^{MAP} = \left(\mathbb{C}_{(\mathcal{R}\times\mathcal{C})}^{T}\mathbb{C}_{(\mathcal{R}\times\mathcal{C})} + \sigma^2\boldsymbol{\Sigma}_U^{-1}\right)^{-1}\mathbb{C}_{(\mathcal{R}\times\mathcal{C})}^{T}vec\mathbb{Y},$$

when $\boldsymbol{\mu}_U = 0$. And if we further set $\boldsymbol{\Sigma}_U$ to be an identity matrix, the result is exactly the solution to the ridge linear regression problem

$$\arg\min_{\mathbf{U}_{(1)}} \|\mathbb{Y} - \langle\mathbb{X},\mathbb{B}\rangle_L\|_F^2 - \lambda\left\|\mathbf{U}_{(1)}\right\|_2^2,$$

with $\lambda = \sigma^2$ and $\mathbb{B} = [\![\mathbb{G}; \mathbf{U}_{(1)}, \cdots, \mathbf{U}_{(L)}, \mathbf{V}_{(1)}, \cdots, \mathbf{V}_{(M)}]\!]$. Similarly, assuming $\boldsymbol{\mu}_V = 0$, and $\boldsymbol{\mu}_G = 0$, the MAP estimators of $\mathbf{V}_{(1)}$ and the core tensor $\mathbb{G}$ are given below:

(18)
$$\begin{aligned} vec\mathbf{V}_{(1)}^{T\ MAP} = & \\ & \left(\mathbf{I}_{Q_1}\otimes\left(\mathbb{D}_{(\mathcal{R}\times\mathcal{C})}^{T}\mathbb{D}_{(\mathcal{R}\times\mathcal{C})}\right) + \sigma^2\tilde{\boldsymbol{\Sigma}}_V^{-1}\right)^{-1} \\ & \times\left(\mathbf{I}_{Q_1}\otimes\mathbb{D}_{(\mathcal{R}\times\mathcal{C})}^{T}\right)vec\tilde{\mathbf{Y}}, \end{aligned}$$

and

(19)
$$\begin{aligned} vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}^{MAP} = & \\ & \left(\left(\mathbf{V}^T\mathbf{V}\right)\otimes\left((\mathbb{X}_{(1)}\mathbf{U})^T(\mathbb{X}_{(1)}\mathbf{U})\right) + \sigma^2\tilde{\boldsymbol{\Sigma}}_G^{-1}\right)^{-1} \\ & \times\left(\left(\mathbf{V}^T\mathbf{V}\right)\otimes(\mathbb{X}_{(1)}\mathbf{U})^T\right)vec\tilde{\tilde{\mathbf{Y}}}, \end{aligned}$$

with the same notations given in Section 4.1. And the MAP estimator of σ^2 is given by

(20)
$$\left(\sigma^2\right)^{MAP} = \frac{\beta'}{\alpha'-1}$$

with $\alpha' = \alpha + \frac{NQ}{2}$, $\beta' = \beta + \frac{\|\mathbb{Y}-\langle\mathbb{X},\mathbb{B}\rangle_L\|_F^2}{2}$, and $Q = \prod_{m=1}^{M} Q_m$. We remark that the unregularized least square results in Lock (2018) is a special case of our MAP results above, when flat priors are given to $\mathbf{U}_{(l)}$'s and $\mathbf{V}_{(m)}$'s, and the core tensor is fixed to be a superdiagonal tensor.

By using MAP estimators of $\{\mathbf{U}_{(l)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}\}_{m=1}^{M}$ and $\mathbb{G}$ instead of generating samples from high dimensional posterior distributions, the problem of choosing the optimal dimension of the core tensor becomes a discrete optimization problem to find the tuple of parameters over an $(L+M)$-dimensional grid of parameters $\boldsymbol{\theta} := (R_1, \cdots, R_L, S_1, \cdots, S_M) \in \boldsymbol{\Theta}$ that minimizes some loss function. In this work, we use the Bayesian information criterion (BIC) as the loss function.

To solve the discrete optimization problem, we adopt simulated annealing (SA) algorithm (Kirkpatrick et al., 1983), which is a metaheuristic to approximate global optimum in a large search space. Starting from the current guess $\boldsymbol{\theta}^{(t)}$ at iteration t, we uniformly generate $\tilde{\boldsymbol{\theta}}$ from $O(\boldsymbol{\theta}^{(t)})$. The probability of accepting the new candidate $\tilde{\boldsymbol{\theta}}$ is $A(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta}^{(t)})$ defined as follows:

(21)
$$A(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta}^{(t)}) = \begin{cases} 1 & \text{if } BIC(\tilde{\boldsymbol{\theta}}) < BIC(\boldsymbol{\theta}^{(t)}), \\ \exp\left(\frac{BIC(\boldsymbol{\theta}^{(t)}) - BIC(\tilde{\boldsymbol{\theta}})}{\zeta(t)}\right) & \text{otherwise,} \end{cases}$$

where $\zeta(t)$, a function of the iteration t, is the usual *temperature* parameter in standard SA algorithm. Two commonly-used choices for $\zeta(t)$ are $\zeta(t) = \gamma^t\zeta_0$ (Dosso and Oldenburg, 1991), and $\zeta(t) = \frac{\zeta_0}{\log(1+t)}$ (Geman and Geman, 1984), where ζ_0 is the initial temperature. The detailed procedure of the fast computing algorithm is shown in Algorithm 2.

Algorithm 2 Fast Computing Algorithm

Input data $\mathbb{X} \in \mathbb{R}^{N\times P_1\times\cdots\times P_L}$, and $\mathbb{Y} \in \mathbb{R}^{N\times Q_1\times\cdots\times Q_M}$.
Initialize core tensor dimensions $\boldsymbol{\theta}^{(0)}$.
for $t = 1, \cdots, T$ **do**
  Calculate the temperature of current SA step: $\zeta(t)$.
  Generate $\tilde{\boldsymbol{\theta}}$ from the neighbour of $\boldsymbol{\theta}^{(t)}$.
  for $k = 1, \cdots, K$ **do**
    Calculate $\{\tilde{\mathbf{U}}_{(l)}^{(k)}\}_{l=1}^{L}$ using (17) and (1).
    Calculate $\{\tilde{\mathbf{V}}_{(m)}^{(k)}\}_{m=1}^{M}$ using (18) and (1).
    Calculate $\tilde{\mathbb{G}}^{(k)}$ using (19) and (2).
    Calculate $\left(\tilde{\sigma}^2\right)^{(k)}$ using (20).
  end for
  Given $\{\tilde{\mathbf{U}}_{(l)}^{(K)}\}_{l=1}^{L}, \{\tilde{\mathbf{V}}_{(m)}^{(K)}\}_{m=1}^{M}, \tilde{\mathbb{G}}^{(K)}, \left(\tilde{\sigma}^2\right)^{(K)}$, and $\tilde{\boldsymbol{\theta}}$ calculate BIC.
  Calculate acceptance probability $A(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta}^{(t)})$ given by (21).
  Generate $r \sim \text{Unif}(0,1)$.
  if $r < A(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta}^{(t)})$ **then**
    $\boldsymbol{\theta}^{(t+1)}, \mathbb{G}^{(t+1)}, \left(\sigma^2\right)^{(t+1)} = \tilde{\boldsymbol{\theta}}, \tilde{\mathbb{G}}^{(K)}, \left(\tilde{\sigma}^2\right)^{(K)}$.
    $\{\mathbf{U}_{(l)}^{(t+1)}\}_{l=1}^{L} = \{\tilde{\mathbf{U}}_{(l)}^{(K)}\}_{l=1}^{L}$.
    $\{\mathbf{V}_{(m)}^{(t+1)}\}_{m=1}^{M} = \{\tilde{\mathbf{V}}_{(m)}^{(K)}\}_{m=1}^{M}$.
  else
    $\boldsymbol{\theta}^{(t+1)}, \mathbb{G}^{(t+1)}, \left(\sigma^2\right)^{(t+1)} = \boldsymbol{\theta}^{(t)}, \mathbb{G}^{(t)}, \left(\sigma^2\right)^{(t)}$.
    $\{\mathbf{U}_{(l)}^{(t+1)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}^{(t+1)}\}_{m=1}^{M} = \{\mathbf{U}_{(l)}^{(t)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}^{(t)}\}_{m=1}^{M}$.
  end if
end for
Calculate $\mathbb{B}^{(T)} = [\![\mathbb{G}^{(T)}; \mathbf{U}_{(1)}^{(T)}, \cdots, \mathbf{U}_{(L)}^{(T)}, \mathbf{V}_{(1)}^{(T)}, \cdots, \mathbf{V}_{(M)}^{(T)}]\!]$.

As will be shown in numerical studies of Section 6, the proposed Algorithm 2 performs well in terms of parameter estimation, and saves around 90% of the running time

in most simulation cases. For example, the average running time of the BayTensor MCMC algorithm on a problem with $\mathbb{X} \in \mathbb{R}^{100\times16\times12}$ and $\mathbb{Y} \in \mathbb{R}^{100\times10\times8}$ is around 23 minutes on a personal laptop, whereas the BayTensor Fast algorithm takes less than 3 minutes when executed on the same machine. We also remark that we only obtain a point estimate for parameter estimation from Algorithm 2 since it is an optimization-based method. If one wants uncertainty quantification of parameters, we suggest to use the estimated optimal dimension of the core tensor from Algorithm 2, and conduct the MCMC sampler introduced in Section 4.1, i.e., update model parameters given the estimated dimension of the core tensor. Such a treatment is able to not only save the computing time for updating the dimension of the core tensor in Algorithm 1, but also quantify parameter uncertainties.

6. SIMULATION STUDIES

We evaluate the proposed Bayesian tensor-on-tensor regression model and the computational algorithms through extensive simulation studies and comparisons to alternative methods.

6.1 Simulation setup

Assume that the response tensor is a three-way array $\mathbb{Y} \in \mathbb{R}^{N\times Q_1\times Q_2}$ and the predictor tensor is another three-way array $\mathbb{X} \in \mathbb{R}^{N\times P_1\times P_2}$. The sample size is set to $N = 100$. We set $P_1 = 16$, $P_2 = 12$, $Q_1 = 10$, and $Q_2 = 8$. We generate simulated data as follows:

- Generate $\mathbb{X} \in \mathbb{R}^{N\times P_1\times P_2}$.
- Generate $\mathbf{U}_{(l)}$ and $\mathbf{V}_{(m)}$ for $l = 1, \cdots, L$ and $m = 1, \cdots, M$, each with independent $N(0, 1)$ entries.
- Generate $\mathbb{G}$ with θ^* dimensions, and independent $N(0, 1)$ entries.
- Generate $\mathbb{E} \in \mathbb{R}^{N\times Q_1\times Q_2}$.
- Calculate $\mathbb{Y} = \langle \mathbb{X}, \mathbb{B}\rangle_L + c\mathbb{E}$, where

$$\mathbb{B} = [\![\mathbb{G}; \mathbf{U}_{(1)}, \cdots, \mathbf{U}_{(L)}, \mathbf{V}_{(1)}, \cdots, \mathbf{V}_{(M)}]\!],$$

and c is a scaling parameter for defining the signal-to-noise (SNR) ratio such that

$$\frac{\|\langle \mathbb{X}, \mathbb{B}\rangle_L\|_F^2}{c^2\|\mathbb{E}\|_F^2} = \text{SNR}.$$

We consider three different setups for generating the predictor tensor $\mathbb{X}$ and error $\mathbb{E}$. In the first setup, all entries of $\mathbb{X}$ and $\mathbb{E}$ are independently and identically generated from a normal distribution $N(0, 1)$, called *uncorrelated-normal setup*. To test the performance of our algorithms when the simulated data generation model is different from the data fitting model, in the second setup all entries of $\mathbb{X}$ are independently and identically generated from a normal $N(0, 1)$, and entries of $\mathbb{E}$ are independently and identically generated from a student-t distribution with a degree of freedom 3. We call this *uncorrelated-t setup*. In the third setup, we consider the entries for each observation of tensor $\mathbb{X}$ to have a correlated structure, which can be seen in some real-world applications such as spatial and temporal data (Lankao et al., 2008). We call it *correlated setup*. Specifically, for each of the N observations of $\mathbb{X}$, denoted by $\mathbf{X}_{(n)} \in \mathbb{R}^{P_1\times P_2}$, $n = 1, \cdots, N$, the correlation between the (i, j) entry and (k, l) entry of $\mathbf{X}_{(n)}$ is $e^{-r\sqrt{|i-k|^2+|j-l|^2}}$, where $i, k = 1, \cdots, P_1$, $j, l = 1, \cdots, P_2$, and $r > 0$.

In each of the setups, we consider two cases for the simulated true dimension of the core tensor $\mathbb{G}$: $\theta^* = (3, 3, 3, 3)$ and $(4, 4, 2, 2)$, and two cases for SNR $= 2$ and 5, yielding a total of four cases. We conduct 50 repeated experiments by generating 50 replicated datasets for each case, and apply the proposed Bayesian tensor-on-tensor method using both the MCMC sampler (BayTensor MCMC) in Algorithm 1 and the fast computing Algorithm 2 (BayTensor Fast) for each dataset.

To evaluate the performance of different methods, for each dataset in each case we generate 5 new datasets with $N_{new} = 1000$ observations. The new observations are given by

$$\mathbb{Y}_{new} = \langle \mathbb{X}_{new}, \mathbb{B}\rangle_L + c\mathbb{E}_{new},$$

with $\mathbb{X}_{new}$ and $\mathbb{E}_{new}$ generated in the same way as $\mathbb{X}$ and $\mathbb{E}$. We then calculate the relative prediction error (RPE), defined as the average prediction error for the 5 new datasets:

$$\textbf{RPE} = \frac{1}{5}\sum_{i=1}^{5}\frac{\left\|\mathbb{Y}_{new}^{(i)} - \hat{\mathbb{Y}}_{new}^{(i)}\right\|_F^2}{\left\|\mathbb{Y}_{new}^{(i)}\right\|_F^2},$$

where $\hat{\mathbb{Y}}_{new}^{(i)}$ is the predicted value for the dataset i, $i = 1, \ldots, 5$. For BayTensor MCMC and BayTnesor Fast algorithms, we also calculate the "Dimension Recovery" rate defined by (the number of experiments where recovered dimension equals the true core tensor dimension) / (total number of experiments) for each simulation setup.

For comparison, we consider four alternative methods:

- The tensor-on-tensor regression method based on the CP decomposition from Lock (2018), denoted as the CP method. As the CP method needs to pre-define or estimate the rank R, we run their algorithm with different R's and report the result under the optimal R value, defined as the one yielding the smallest RPE.
- Multivariate linear regression method after turning tensors to vectors to solve $\mathbb{B}_{(\mathcal{P}\times\mathcal{Q})}$ in equation (4), denoted as the OLS method.
- Bayesian multivariate linear regression method with the rescaled spike and slab algorithm prior (Ishwaran and Rao, 2005) for inducing sparsity after turning tensors to vectors, denoted as the SAS (Spike-And-Slab) method.

- Bayesian multivariate linear regression method with the horseshoe prior (Carvalho et al., 2010) for inducing shrinkage after turning tensors to vectors, denoted as the HS (horseshoe) method.

6.2 Simulation results: uncorrelated setup

We first apply the proposed BayTensor MCMC with $b = 0.05$, BayTensor Fast, and four alternative methods (including the CP method, the OLS method, the SAS method, and the HS method) to datasets under the uncorrelated-normal setup. For the CP method, we run the algorithm with $R = 1, \cdots, 6$. The reason why we choose 6 as the upper bound is that when $R = 6$, the total number of parameters in the CP method is 276 which is larger than the true number of parameters 212 for the $\boldsymbol{\theta}^* = (4, 4, 2, 2)$ case and 219 for the $\boldsymbol{\theta}^* = (3, 3, 3, 3)$ case. The CP method with $R = 6$ yields the smallest RPE under both cases in all repeated simulations. Full results from the CP method with different R values are shown in Appendix A.9.

Table 1 reports the means and standard deviations (sd) of RPEs averaging over 50 replicated experiments for each of the four cases under the six methods. No matter when the information in $\mathbb{X}$ and $\mathbb{Y}$ is balanced (i.e., $\theta^* = (3, 3, 3, 3)$) or skewed to some modes (i.e., $\theta^* = (4, 4, 2, 2)$), BayTensor MCMC and BayTensor Fast always yield smaller prediction RPEs than the CP method, while the OLS method, the SAS method and the HS method all yield much larger RPEs. For example, when $\theta^* = (4, 4, 2, 2)$ with SNR $= 2$, BayTensor MCMC and BayTensor Fast have comparable results with mean RPEs being 0.350 and 0.357 respectively, while the CP method has a slightly larger mean RPE of 0.377, and the OLS method, the SAS method, and the HS method all have much larger mean RPE of 1.016, 0.970, and 1.114 respectively. The superior performance of the Bayesian tensor-on-tensor regression method over the CP method comes from the flexibility of Tucker decomposition that permits different orders along different modes of the core tensor. From Table 1, we can also see that a larger signal to noise ratio leads to lower RPEs. In particular, when SNR $= 5$, all methods, except for the HS method, have lower RPEs compared to the results from cases where SNR $= 2$.

As the proposed BayTensor MCMC method and BayTensor Fast method can simultaneously estimate the dimension of the core tensor and other model parameters, we next report the empirical probabilities that the true dimension of the core tensor can be recovered by them in 50 replicated experiments for all 4 cases, as shown in Table 2. BayTensor MCMC has higher recovering rates than BayTensor Fast in all cases. And we can better recover the core tensor dimension with a higher SNR. Another observation is that both BayTensor MCMC and BayTensor Fast have higher recovering rates when $\theta^* = (3, 3, 3, 3)$ compared with the cases where $\theta^* = (4, 4, 2, 2)$.

We also present the average numbers of parameters required by BayTensor MCMC and BayTensor Fast in 50 replicated experiments in Table 2. BayTensor MCMC and BayTensor Fast usually require smaller numbers of parameters than the CP method. For example, when $\theta^* = (4, 4, 2, 2)$ with SNR $= 2$, BayTensor MCMC requires 207 parameters on average, and BayTensor Fast requires 196 parameters on average. In contrast, the CP method with $R = 6$ has 276 parameters. When $\theta^* = (3, 3, 3, 3)$ with SNR $= 2$, the average numbers of parameters are 218, 193, and 276 for BayTensor MCMC, BayTensor Fast, and the CP method respectively. We can see that both BayTensor MCMC and BayTensor Fast yield higher dimension recovery rates than the CP method with smaller numbers of parameters.

To quantify the prediction uncertainty, for each test dataset under each simulation setup, we collect 1000 post-burn-in samples from BayTensor MCMC and generate the empirical posterior predictive distribution for each element of $\mathbb{Y}_{new}$. The symmetric credible intervals are then calculated. For cases with $\boldsymbol{\theta}^* = (4, 4, 2, 2)$, the empirical coverage rates for 95% credible interval are 0.944 (0.013) and 0.945 (0.012) for SNR $= 2$ and SNR $= 5$ respectively. And for cases with $\boldsymbol{\theta}^* = (3, 3, 3, 3)$, the empirical coverage rates for 95% credible interval are 0.950 (0.011) and 0.950 (0.011) for SNR $= 2$ and SNR $= 5$ respectively. Figure 3(a) plots the predictive posterior estimations with 95% credible intervals for a randomly-selected sample of $\mathbb{Y}_{new}$ under a randomly-selected test case when $\boldsymbol{\theta}^* = (3, 3, 3, 3)$ and SNR $= 5$. Figure 3(b) plots the empirical predictive distributions from BayTensor MCMC for 8 randomly selected elements from the same sample of $\mathbb{Y}_{new}$ shown in Figure 3(a).

We then apply all six methods to datasets under the uncorrelated-t setup. The results are very similar to those under the uncorrelated-normal setup, suggesting that our proposed algorithms have reasonably good performance under model mis-specification when the data fitting model is different from the data generating model. The RPEs and dimension recovery results are presented in Table A.1 and Table A.2 of Appendix A.7, respectively.

6.3 Simulation results: correlated setup

We then apply the proposed BayTensor MCMC method with $b = 0.05$, the BayTensor Fast method, the CP method, the OLS method, the SAS method, and the HS method to datasets under the correlated setup. For the CP method, we run the algorithm with $R = 1, \cdots, 6$. The CP method with $R = 6$ yields the smallest RPE under all cases except for the case of $\boldsymbol{\theta}^* = (4, 4, 2, 2)$ and SNR $= 2$ where the optimal $R = 5$. Full results from the CP method with different R values are shown in Appendix A.9. For each replicated dataset, we compute the RPEs under all four methods. The means and standard deviations (sd) of RPEs averaging over the 50 replicated datasets for all 4 cases are presented in Table 3.

The prediction RPE results under the correlated setup are analogous to the results under the uncorrelated setup. To summarize, the RPE results from Baytensor MCMC and BayTensor Fast are comparable in all cases. And both of

Table 1. Mean RPE with SD on uncorrelated-normal data

RPE (SD)	BayTensor MCMC	BayTensor Fast	CP Method	OLS Method	SAS Method	HS Method
$\theta^* = (4,4,2,2)$, SNR=2	**0.350 (0.013)**	0.357 (0.026)	0.377 (0.014)	1.016 (0.034)	0.970 (0.025)	1.114 (0.310)
$\theta^* = (3,3,3,3)$, SNR=2	**0.343 (0.004)**	0.351 (0.015)	0.391 (0.016)	1.023 (0.031)	0.963 (0.020)	1.079 (0.153)
$\theta^* = (4,4,2,2)$, SNR=5	**0.173 (0.001)**	0.185 (0.034)	0.209 (0.046)	0.751 (0.030)	0.946 (0.038)	1.182 (0.410)
$\theta^* = (3,3,3,3)$, SNR=5	**0.172 (0.001)**	0.181 (0.037)	0.222 (0.016)	0.748 (0.023)	0.934 (0.031)	1.136 (0.240)

Table 2. Core tensor dimension recovery and number of model parameters on uncorrelated-normal data

	BayTensor MCMC		BayTensor Fast	
	Dimension Recovery	# Parameters (SD)	Dimension Recovery	# Parameters (SD)
$\theta^* = (4,4,2,2)$, SNR=2	80%	207 (15)	46%	189 (14)
$\theta^* = (3,3,3,3)$, SNR=2	98%	218 (6)	54%	193 (18)
$\theta^* = (4,4,2,2)$, SNR=5	96%	215 (14)	64%	209 (16)
$\theta^* = (3,3,3,3)$, SNR=5	100%	219 (0)	76%	215 (16)

Table 3. Mean RPE with SD on correlated data

RPE (SD)	BayTensor MCMC	BayTensor Fast	CP Method	OLS Method	SAS Method	HS Method
$\theta^* = (4,4,2,2)$, SNR=2	**0.344 (0.004)**	0.350 (0.006)	0.370(0.011)	0.886 (0.086)	0.542 (0.115)	1.380 (0.540)
$\theta^* = (3,3,3,3)$, SNR=2	**0.344 (0.004)**	0.348 (0.006)	0.369(0.007)	0.893 (0.065)	0.532 (0.121)	1.366 (0.783)
$\theta^* = (4,4,2,2)$, SNR=5	**0.173 (0.002)**	0.175 (0.003)	0.188(0.006)	0.492 (0.086)	0.397 (0.141)	1.466 (0.648)
$\theta^* = (3,3,3,3)$, SNR=5	**0.173 (0.008)**	0.173 (0.003)	0.187(0.006)	0.497 (0.059)	0.384 (0.150)	1.479 (1.003)

Table 4. Core tensor dimension recovery and number of model parameters on correlated data

	BayTensor MCMC		BayTensor Fast	
	Dimension Recovery	# Parameters (SD)	Dimension Recovery	# Parameters (SD)
$\theta^* = (4,4,2,2)$, SNR=2	72%	207 (17)	36%	196 (16)
$\theta^* = (3,3,3,3)$, SNR=2	86%	214 (14)	32%	164 (20)
$\theta^* = (4,4,2,2)$, SNR=5	96%	212 (7)	44%	188 (15)
$\theta^* = (3,3,3,3)$, SNR=5	96%	218 (6)	56%	196 (15)

them have better RPEs than the CP method followed by the SAS method, and the OLS method. The HS performs the worst in terms of yielding the highest RPEs in all cases. And when the signal to noise ratio increases from 2 to 5, the prediction accuracies are improved under all methods, except for the HS method.

Table 4 shows the empirical probabilities of dimension recovery and the average numbers of parameters required by BayTensor MCMC and BayTensor Fast in 50 replicated experiments. In all 4 cases, BayTensor MCMC and BayTensor Fast both require a smaller number of parameters than the CP method. And in terms of recovering the dimension of the core tensor, BayTensor MCMC has higher empirical probabilities of recovering the true dimension than BayTensor Fast in all cases. And the recovering probabilities in the cases where $\boldsymbol{\theta}^* = (3,3,3,3)$ are higher than those in cases where $\boldsymbol{\theta}^* = (4,4,2,2)$ for both methods. We also observe that the recovering probabilities under the correlated setup are smaller than those under the uncorrelated setup for both the BayTensor MCMC and BayTensor Fast.

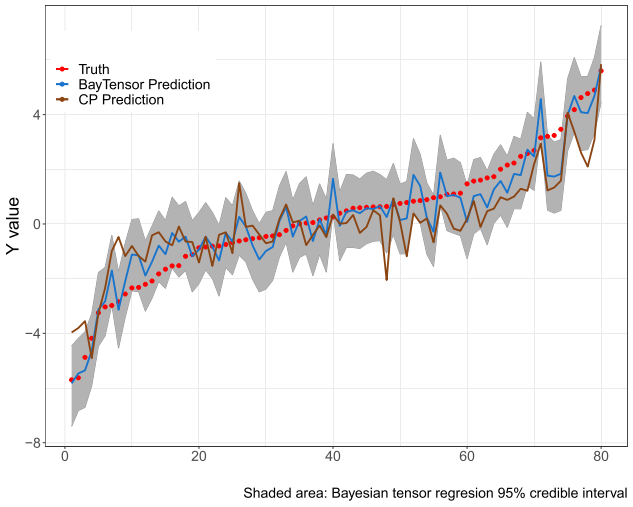
7. REAL DATA ANALYSES

We demonstrate the usefulness of the proposed Bayesian tensor-on-tensor regression approach by applying it to two real-world datasets: labeled faces in the wild database (Huang et al. (2007)) and multi-person motion (UMPM) benchmark (Van der Aa et al., 2011).

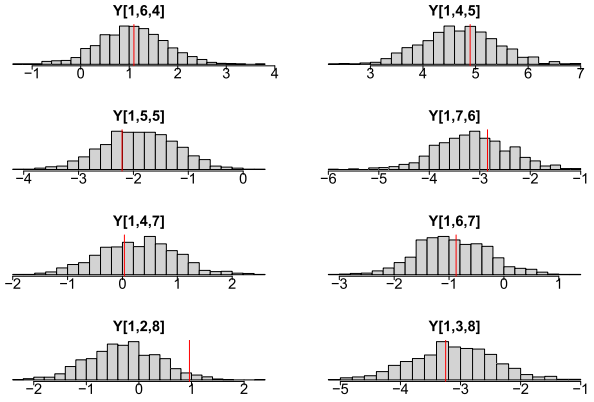
7.1 Facial image data

We apply the proposed Bayesian tensor-on-tensor regression approach, with $b = 0.05$, to predict different attributes of facial images, such as smiling and nose size, using the labeled faces in the wild database (Huang et al. (2007)). The database collects more than 13,000 face images, each of which has been labeled with the name of the person pictured, often a celebrity. There can be multiple images for one person.

We use the estimation results from the attribute classifiers developed in Kumar et al. (2011) for response $\mathbb{Y}$, resulting in a total of 73 describable attributes. These attributes can be categorized into characteristics that describe a person, an expression, or an accessory. The attributes are all



(a)



(b)

Figure 3. Uncertainty quantification for a randomly selected case where $\boldsymbol{\theta}^* = (3,3,3,3)$ and SNR $= 5$. (a): An example of prediction with estimation uncertainty. Elements of $\mathbb{Y}_{[1::]}$ are sorted for visualization. (b): Empirical predictive distributions from BayTensor MCMC for randomly selected elements from $\mathbb{Y}_{new}$ with the simulation truths shown as red lines.

given as continuous variables, with a higher value denoting a more obvious characteristic. The proposed approach is applied to predict the 73 attributes from a given facial image $\mathbb{X}$. In this work, we use the frontalized version of facial images (Hassner et al., 2015), which show only forward-facing faces obtained by rotating, scaling, and cutting original facial images. The frontalized images are highly aligned, allowing for appearances to be easily compared across faces. Each frontalized image contains 90×90 pixels, with each pixel giving color intensities for red, green, and blue, resulting in a $90 \times 90 \times 3$ tensor for each image. We randomly sample 1000 images. Thus the predictor tensor $\mathbb{X}$ is of dimensions $1000 \times 90 \times 90 \times 3$, and the response tensor $\mathbb{Y}$ is of dimensions 1000×73. We center the tensor for each image by subtracting the mean of tensors for all images. Another randomly-sampled 1000 images are used as a validation set, in other words, $\mathbb{X}_{new}$ is of dimensions $1000 \times 90 \times 90 \times 3$, and $\mathbb{Y}_{new}$ is of dimensions 1000×73.

We apply the BayTensor MCMC in Algorithm 1 and BayTensor Fast in Algorithm 2 to the dataset and conduct inference. For comparison, we also apply the CP method proposed by Lock (2018) to the same dataset. For the CP method, we choose $R = 15$ and $\lambda = 10^5$ since these values yielded the best prediction performance, as reported in Lock (2018). BayTensor MCMC estimates the dimension of the core tensor based on the posterior mode to be $(5,2,3,5)$ with a total of 1154 parameters, resulting in an RPE of 0.375. And BayTensor Fast yields an RPE of 0.446. In contrast, the CP method has 3840 parameters and results in a higher RPE, 0.477. To summarize, the proposed Bayesian tensor-on-tensor regression approach is able to reduce predictive errors with a smaller number of parameters due to the flexibility of the Tucker decomposition that permits different orders along different modes.

Next we report the prediction uncertainty, which is a natural byproduct of the proposed Bayesian framework. We collect 500 post-burn-in MCMC samples and compute the posterior predictive distribution along with credible intervals for each of the 73 attributes for each image. The empirical coverage rate for 95% credible interval is 0.930, and for 90% credible interval is 0.889. Figure 4 shows an example of a test image, and plots the corresponding posterior predictive values for four randomly-selected characteristics.

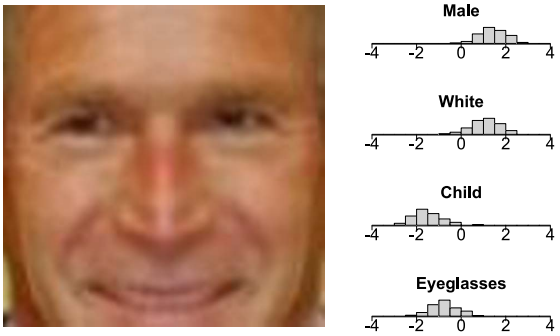


Figure 4. An example of a test image and the corresponding posterior predictive values for four selected characteristics.

7.2 Utrecht multi-person motion (UMPM) data

We then apply the proposed Bayesian tensor-on-tensor regression model, with $b = 0.05$, to the multi-person motion (UMPM) benchmark (Van der Aa et al., 2011) that contains temporally synchronized video sequences from multiple viewpoints and human motion capture data. Each video is of length 30 to 60 seconds, with resolution of 644×484

pixels at 50 fps (frames per second). Motion capture (MoCap) data contain 3D positions of 37 markers at 100 fps for each subject of interest.

To evaluate the performance of our model on 3D motion data, we consider two scenarios, namely 'chair' and 'triangle' from the UMPM dataset. In the scenario 'chair', we have a sequence of 2570 sample images in which the subject of interest starts walking in a circle, finds the chair, sits on the chair and stands up multiple times with different postures. In the scenario 'triangle', we have a sequence of 2471 sample images in which the subject walks by following the path of a triangle within a circular area. In our data analysis, the input data was a grayscale image sequence from the front camera with resolution downsized to 32×24 pixels, forming a 3-order predictor tensor (i.e. *frames* $\times$ *width* $\times$ *height*). The response data containing 3D positions of 37 markers is first downsampled to 50fps to match the video, and is then presented as a 3-order tensor (i.e. *samples* $\times$ *3D position* $\times$ *markers*). For each scenario, we run 10 repeated experiments, and randomly sample 200 images from the sequence in each experiment to form the predictor tensor data $\mathbb{X}$ of dimensions $200 \times 32 \times 24$ and response tensor data $\mathbb{Y}$ of dimensions $200 \times 3 \times 37$. The remaining images are used as testing data $\mathbb{X}_{new}$ and $\mathbb{Y}_{new}$.

We apply BayTensor MCMC, BayTensor Fast, and the CP method to the datasets. For scenario 'chair', the mean prediction RPE of BayTensor MCMC is 0.176 (sd: 0.009) while that of BayTensor Fast and the CP method are 0.206 (sd: 0.053) and 0.254 (sd: 0.064) respectively. The mean number of parameters for BayTensor MCMC is 752 (sd: 58), for BayTensor Fast, the number is 798 (sd: 132), and for the CP method, with $R = 10$, the number of parameters is 960. For scenario 'triangle', BayTensor MCMC yields a mean RPE of 0.243 (sd: 0.049) with the mean number of parameters being 685 (sd: 127). BayTensor Fast, with the mean number of parameters being 798 (sd: 132), has a slightly larger RPE of 0.252 (sd: 0.028). In contrast, the CP method, with $R = 10$, has 960 parameters and results in a worse RPE, 0.346 (sd: 0.049). To summarize, with a smaller number of estimating parameters, the proposed Bayesian tensor-on-tensor regression model is able to yield smaller predictive errors compared to the CP method. And this improvement comes from the flexibility of Tucker decomposition.

To demonstrate the predictive uncertainty, we generate 500 post-burn-in posterior samples, produce the empirical posterior predictive distribution of 3D position for each of the 37 markers, and calculate the symmetric credible intervals. The empirical coverage rate for 95% credible interval is 0.957 for scenario 'chair', and 0.931 for scenario 'triangle'. Figure 5 shows an example of estimation results for 3D position of the #3 marker by BayTensor MCMC method with 95% credible intervals, BayTensor Fast, and the CP method.

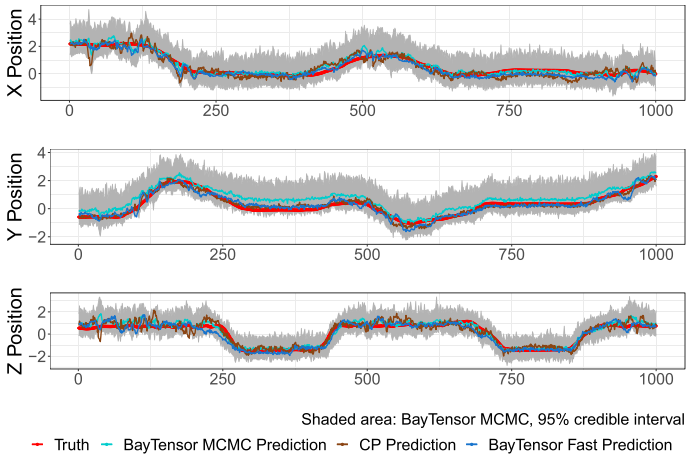


Figure 5. Predictions of x,y,z-position of marker 3 by Bayesian tensor-on-tensor regression method with 95% *credible intervals, the fast computing method, and the CP method. The underlying truths are shown in red.*

8. DISCUSSION

We developed a Bayesian tensor-on-tensor regression model to predict one tensor response from another tensor predictor, building upon the flexible Tucker decomposition of the coefficient tensor so that the response tensor and predictor tensor can have different dimensions in the core tensor. For posterior inference, we proposed an efficient MCMC algorithm to simultaneously estimate the model dimension and model parameters. In addition, we developed an ultra-fast computing algorithm wherein the MAP estimators of model parameters are computed given the model dimension, and then an SA algorithm is used to find the optimal model dimension. Both simulation studies and real data analyses show that the performance of our Bayesian tensor-on-tensor regression model benefits from the flexibility in the core tensor structure compared to alternative methods. And the fast computing algorithm yielded comparable prediction results to the MCMC sampler, meanwhile saved a significant amount of computing time. As the fast computing algorithm is only guaranteed to converge to a local optimum, in practice it can be repeatedly applied with different initial values to search for the global optimum. To our best knowledge, this work represents the first effort in literature to simultaneously estimate the model dimension and parameters in tensor-on-tensor regression setup under a fully Bayesian framework.

There are several interesting future directions. First, instead of assigning normal priors to factor matrices, the proposed framework can easily incorporate other priors, such as sparsity-inducing priors (Guhaniyogi et al., 2017; Miranda et al., 2018), for applications where the sparsity is desired. More general covariance structures for $\mathbb{E}$ will also be considered. Second, this work only considers one tensor

predictor. Some applications may require to predict one tensor response from multiple tensor predictors or mixed-type predictors including tensors, matrices, and vectors. We will extend the proposed model to handle more flexible predictors. Third, the proposed BayTensor Fast algorithm provides a general algorithmic framework for estimating the model dimension as well as model parameters. In this work, we consider the SA algorithm and BIC. As a future direction, we plan to explore alternative optimization methods beyond the SA and target functions beyond the BIC. Finally, the computational cost of the proposed MCMC sampler comes as a price for prediction accuracy and uncertainty quantification compared to the proposed fast computing algorithm. To improve the efficiency of posterior computation, we plan to take advantage of certain advanced MCMC techniques, e.g., stochastic variational inference (Hoffman et al., 2013) and pseudo-marginal Metropolis-Hastings algorithms (Andrieu and Roberts, 2009).

APPENDIX

A.1 A brief review of matrix Kronecker product

The *Kronecker product* is a matrix operation that is important in showing posterior distribution of parameters in this paper, and we will briefly review it here.

The Kronecker product of matrices $\mathbf{A} \in \mathbb{R}^{I\times J}$, and $\mathbf{B} \in \mathbb{R}^{K\times L}$ is denoted by $\mathbf{A}\otimes\mathbf{B}$. And the result is of size $(IK)\times(JL)$ defined by

$$\mathbf{A}\otimes\mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \cdots & a_{IJ}\mathbf{B} \end{bmatrix}.$$

Some of the properties of Kronecker product are proved useful for this paper. See the detailed proofs of these properties in Kolda (2006)

$$(\mathbf{A}\otimes\mathbf{B})(\mathbf{C}\otimes\mathbf{D}) = \mathbf{AC}\otimes\mathbf{BD}.$$

Let $\mathbb{X} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, and $\mathcal{N} = \{1,\cdots,N\}$. Let $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n\times J_n}$ be a sequence matrices for all $n \in \mathcal{N}$. Let the ordered sets $\mathcal{R} = \{r_1,\cdots,r_L\}$ and $\mathcal{C} = \{c_1,\cdots,c_M\}$ be a partition of $\mathcal{N}$, then if

$$\mathbb{X} = \mathbb{Y}\times_1\mathbf{A}_{(1)}\times_2\mathbf{A}_{(2)}\cdots\times_N\mathbf{A}_{(N)}$$

we have

$$\mathbb{X}_{(\mathcal{R}\times\mathcal{C})} = \left(\mathbf{A}_{(r_L)}\otimes\cdots\otimes\mathbf{A}_{(r_1)}\right)\mathbb{Y}_{(\mathcal{R}\times\mathcal{C})} \times\left(\mathbf{A}_{(c_M)}\otimes\cdots\otimes\mathbf{A}_{(c_1)}\right)^T.$$

Consequently, if $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n\times J_n}$ for all $n \in \mathcal{N}$, then for any specific $n \in \mathcal{N}$ if we have

$$\mathbb{X} = \mathbb{Y}\times_1\mathbf{A}_{(1)}\times_2\mathbf{A}_{(2)}\cdots\times_N\mathbf{A}_{(N)},$$

and then

(22)
$$\mathbb{X}_{(n)} = \mathbf{A}_{(n)}\mathbb{Y}_{(n)} \times\left(\mathbf{A}_{(N)}\otimes\cdots\otimes\mathbf{A}_{(n+1)}\otimes\mathbf{A}_{(n-1)}\otimes\cdots\otimes\mathbf{A}_{(1)}\right)^T.$$

A.2 Proof for equation (6)

By properties of n-mode product of tensor and Tucker decomposition, we have

$$\mathbb{B} = \mathbb{G}\times_2\mathbf{U}_{(2)}\cdots\times_L\mathbf{U}_{(L)}\times_{L+1}\mathbf{V}_{(1)}\cdots\times_{L+M}\mathbf{V}_{(M)}\times_1\mathbf{U}_{(1)}.$$

Let $\mathbb{B}_{(-)}$ denote $\mathbb{G}\times_2\mathbf{U}_{(2)}\cdots\times_L\mathbf{U}_{(L)}\times_{L+1}\mathbf{V}_{(1)}\cdots\times_{L+M}\mathbf{V}_{(M)}$, then $\mathbb{B}_{(-)} \in \mathbb{R}^{R_1\times P_2\times\cdots\times P_L\times Q_1\times\cdots Q_M}$, and

$$\mathbb{B} = \mathbb{B}_{(-)}\times_1\mathbf{U}_{(1)}.$$

Therefore,

$$\mathbb{B}_{[p_1,\cdots,p_L,q_1,\cdots,q_M]} = \sum_{r_1=1}^{R_1}\left((\mathbb{B}_{(-)})_{[r_1,p_2,\cdots,p_L,q_1,\cdots,q_M]}\mathbf{U}_{(1)r_1p_1}\right).$$

And

$$\begin{aligned}
\hat{\mathbb{Y}}_{[n,q_1,\cdots,q_M]} &= (\langle\mathbb{X},\mathbb{B}\rangle_L)_{[n,q_1,\cdots,q_M]} \\
&= \sum_{p_1=1}^{P_1}\cdots\sum_{p_L=1}^{P_L}\mathbb{B}_{[p_1,\cdots,p_L,q_1,\cdots,q_M]}\mathbb{X}_{[n,p_1,\cdots,p_L]} \\
&= \sum_{p_1=1}^{P_1}\cdots\sum_{p_L=1}^{P_L}\sum_{r_1=1}^{R_1}\left((\mathbb{B}_{(-)})_{[r_1,p_2,\cdots,p_L,q_1,\cdots,q_M]}\right. \\
&\quad\left.\times\,\mathbf{U}_{(1)r_1p_1}\mathbb{X}_{[n,p_1,\cdots,p_L]}\right) \\
&= \sum_{r_1=1}^{R_1}\sum_{p_1=1}^{P_1} \\
&\left(\sum_{p_2=1}^{P_2}\cdots\sum_{p_L=1}^{P_L}(\mathbb{B}_{(-)})_{[r_1,p_2,\cdots,p_L,q_1,\cdots,q_M]}\mathbb{X}_{[n,p_1,\cdots,p_L]}\right)\mathbf{U}_{(1)r_1p_1} \\
&= \sum_{r_1=1}^{R_1}\sum_{p_1=1}^{P_1}\left(\langle\mathbb{B}_{(-)},\mathbb{X}\rangle_{P_2,\cdots,P_N}\right)_{[n,p_1,r_1,q_1,\cdots,q_M]}\mathbf{U}_{(1)r_1p_1}.
\end{aligned}$$

And further,

$$vec\hat{\mathbb{Y}} = \mathbb{C}_{(\mathcal{R}\times\mathcal{C})}\times\ vec\mathbf{U}_{(1)}.$$

If we denote the contracted product of $\langle\mathbb{B}_{(-)},\mathbb{X}\rangle_{P_2,\cdots,P_N}$ by a new tensor called $\mathbb{C}$, then tensor $\mathbb{C} \in \mathbb{R}^{R_1\times N\times P_1\times Q_1\times\cdots Q_M}$. And matricize tensor $\mathbb{C}$ to $\mathbb{C}_{(\mathcal{R}\times\mathcal{C})} \in \mathbb{R}^{N\prod_{m=1}^{M}Q_m\times R_1P_1}$, where $\mathcal{R} = \{N,Q_1,\cdots,Q_M\}$, and $\mathcal{C} = \{R_1,P_1\}$. We have

$$vec\mathbb{Y} = \mathbb{C}_{(\mathcal{R}\times\mathcal{C})}\times\ vec\mathbf{U}_{(1)} + vec\mathbb{E}.$$

A.3 Proof for equation (8)

We denote tensor $\mathbb{D}_{(-)} = \mathbb{G} \times_1 \mathbf{U}_{(1)} \cdots \times_L \mathbf{U}_{(L)} \times_{L+2} \mathbf{V}_{(2)} \cdots \times_{L+M} \mathbf{V}_{(M)}$. And then the contracted product of tensor $\mathbb{D}_{(-)}$ and $\mathbb{X}$ is a new tensor denoted as $\mathbb{D}$. Then

$$\mathbb{B} = \mathbb{D}_{(-)} \times_{L+1} \mathbf{V}_{(1)} .$$

Then

$$\mathbb{B}_{[p_1,\cdots,p_L,q_1,\cdots,q_M]} = \sum_{s_1=1}^{S_1} \mathbb{D}_{(-)[p_1,\cdots,p_L,s_1,q_2,\cdots,q_M]} \mathbf{V}_{(1)s_1q_1} ,$$

and

$$\begin{aligned}
\hat{\mathbb{Y}}_{[n,q_1,\cdots,q_M]} &= (\langle \mathbb{X}, \mathbb{B} \rangle_L)_{[n,q_1,\cdots,q_M]} \\
&= \sum_{p_1=1}^{P_1} \cdots \sum_{p_L=1}^{P_L} \mathbb{B}_{[p_1,\cdots,p_L,q_1,\cdots,q_M]} \mathbb{X}_{[n,p_1,\cdots,p_L]} \\
&= \sum_{p_1=1}^{P_1} \cdots \sum_{p_L=1}^{P_L} \sum_{s_1=1}^{S_1} \big((\mathbb{D}_{(-)})_{[p_1,\cdots,p_L,s_1,q_2,\cdots,q_M]} \\
&\times \mathbf{V}_{(1)s_1q_1} \mathbb{X}_{[n,p_1,\cdots,p_L]} \big) \\
&= \sum_{s_1=1}^{S_1} \sum_{p_1=1}^{P_1} \cdots \sum_{p_L=1}^{P_L} \\
&\big((\mathbb{D}_{(-)})_{[p_1,\cdots,p_L,s_1,q_2,\cdots,q_M]} \mathbb{X}_{[n,p_1,\cdots,p_L]} \big) \mathbf{V}_{(1)s_1q_1} .
\end{aligned} \tag{23}$$

By equation (23), we have

$$\hat{\mathbb{Y}} = \mathbb{D} \times_{L+1} \mathbf{V}_{(1)}. \tag{24}$$

Combine (22) and equation (24) resulting in

$$\mathbb{Y}_{(2)} = \mathbf{V}_{(1)} \times (\mathbb{D}_{(\mathcal{R}\times\mathcal{C})})^T + \mathbb{E}_{(2)},$$

where $\mathbb{Y}_{(2)} \in \mathbb{R}^{Q_1 \times N \prod_{m=2}^{M} Q_m}$ is the matricization of tensor $\mathbb{Y}$ and $\mathbb{D}_{(\mathcal{R}\times\mathcal{C})} \in \mathbb{R}^{N \prod_{m=2}^{M} Q_m \times S_1}$ is the matricization of tensor $\mathbb{D}$.

A.4 Conditional posterior distributions given training fraction b

Given the dimension $\boldsymbol{\theta} = (R_1, \cdots, R_L, S_1, \cdots, S_M)$ of the core tensor $\mathbb{G}$, and the training fraction b, we first derive the full conditional posterior distributions of $\{\mathbf{U}_{(l)}\}_{l=1}^{L}$, $\{\mathbf{V}_{(m)}\}_{m=1}^{M}$, $\mathbb{G}$, σ^2 in closed forms. Without loss of generality, we first derive the full conditional posterior distribution of $\mathbf{U}_{(1)}$. The full conditional posterior distributions of $\{\mathbf{U}_{(2)}, \cdots, \mathbf{U}_{(L)}\}$ can be derived in the same manner.

By equation (6), we have

$$vec\mathbb{Y} = \mathbb{C}_{(\mathcal{R}\times\mathcal{C})} \times \ vec\mathbf{U}_{(1)} + vec\mathbb{E} ,$$

where $\mathcal{R} = \{N, Q_1, \cdots, Q_M\}$, and $\mathcal{C} = \{R_1, P_1\}$. And combined with the idea that $\mathbf{U}_{(1)}$ is from a distribution proportional to $p(\mathbb{Y} \mid \{\mathbf{U}_{(l)}\}_{l=1}^{L}, \{\mathbf{V}_{(m)}\}_{m=1}^{M}, \mathbb{G}, \sigma^2, \boldsymbol{\theta})^b \times p(\mathbf{U}_{(1)} \mid \boldsymbol{\theta})$, we have, the posterior distribution of $vec\mathbf{U}_{(1)}$ given all other parameters and b is normal distribution. That is

$$vec\mathbf{U}_{(1)} \ \sim \ N(\boldsymbol{\mu}'_U, \boldsymbol{\Sigma}'_U) \tag{25}$$

where,

$$\begin{aligned}
\boldsymbol{\Sigma}'_U &= \left(\frac{b \times \mathbb{C}^T_{(\mathcal{R}\times\mathcal{C})} \mathbb{C}_{(\mathcal{R}\times\mathcal{C})}}{\sigma^2} + \boldsymbol{\Sigma}_U^{-1} \right)^{-1} , \\
\boldsymbol{\mu}'_U &= \boldsymbol{\Sigma}'_U \left(\frac{b \times \mathbb{C}^T_{(\mathcal{R}\times\mathcal{C})} vec\mathbf{Y}}{\sigma^2} + \boldsymbol{\Sigma}_U^{-1} \boldsymbol{\mu}_U \right) .
\end{aligned}$$

We then derive the conditional posterior distributions of $\mathbf{V}_{(m)}$ given σ^2, $\{\mathbf{U}_{(l)}\}_{l=1}^{L}$, $\mathbf{V}_{(k)}$ for $k \neq m$, and $\mathbb{G}$. Without loss of generality, we derive the full conditional posterior distribution of $\mathbf{V}_{(1)}$ below.

Denote the contracted product of the tensor $\mathbb{G} \times_1 \mathbf{U}_{(1)} \cdots \times_L \mathbf{U}_{(L)} \times_{L+2} \mathbf{V}_{(2)} \cdots \times_{L+M} \mathbf{V}_{(M)}$ and tensor $\mathbb{X}$ by a new tensor $\mathbb{D}$, where $\mathbb{D} \in \mathbb{R}^{N \times S_1 \times Q_2 \times \cdots \times Q_M}$. We then matricize $\mathbb{D}$ into a matrix $\mathbb{D}_{(\mathcal{R}\times\mathcal{C})} \in \mathbb{R}^{N \prod_{m=2}^{M} Q_m \times S_1}$ and write

$$\mathbb{Y}_{(2)} = \mathbf{V}_{(1)} \times (\mathbb{D}_{(\mathcal{R}\times\mathcal{C})})^T + \mathbb{E}_{(2)}, \tag{26}$$

where $\mathbb{Y}_{(2)} \in \mathbb{R}^{Q_1 \times N \prod_{m=2}^{M} Q_m}$ is the matricization of tensor $\mathbb{Y}$. Let $\tilde{\mathbf{Y}} = \mathbb{Y}^T_{(2)}$, given $\mathbf{V}_{(1)}$ follows a normal distribution with a diagonal covariance matrix, we can rewrite (26) as

$$vec\tilde{\mathbf{Y}} = \left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})} \right) \times vec\mathbf{V}^T_{(1)} + vec\left(\mathbb{E}_{(2)} \right)^T ,$$

where $\mathbf{I}_{Q_1}$ denotes an identity matrix of size Q_1. Given the prior distribution of $vec\mathbf{V}_{(1)}$ is a normal $N(\boldsymbol{\mu}_V, \boldsymbol{\Sigma}_V)$ with diagonal $\boldsymbol{\Sigma}_V$, the prior distribution of $vec\mathbf{V}^T_{(1)}$ is also a normal distribution $N(\tilde{\boldsymbol{\mu}}_V, \tilde{\boldsymbol{\Sigma}}_V)$ with a diagonal covariance matrix. Then the full conditional posterior distribution of $vec\mathbf{V}_{(1)}{}^T$ is also normally distributed:

$$p(vec\mathbf{V}^T_{(1)} \mid vec\mathbb{Y}, \mathbb{X}, \sigma^2, \mathbf{U}_{(l)}, \mathbf{V}_{(m)} \ m \neq 1) \sim \ N(\tilde{\boldsymbol{\mu}}'_U, \tilde{\boldsymbol{\Sigma}}'_U), \tag{27}$$

where

$$\begin{aligned}
\tilde{\boldsymbol{\Sigma}}'_V &= \left(\frac{b \times \left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})} \right)^T \left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})} \right)}{\sigma^2} + \tilde{\boldsymbol{\Sigma}}_V^{-1} \right)^{-1} \\
\tilde{\boldsymbol{\mu}}'_V &= \tilde{\boldsymbol{\Sigma}}'_V \left(\frac{b \times \left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})} \right)^T vec\tilde{\mathbf{Y}}}{\sigma^2} + \tilde{\boldsymbol{\Sigma}}_V^{-1} \tilde{\boldsymbol{\mu}}_V \right) .
\end{aligned} \tag{28}$$

By analogous procedures, we have the posterior distribution of $vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}$ is also normal distribution with

$$\begin{aligned}
&\boldsymbol{\mu}'_G = \boldsymbol{\Sigma}'_G \left((\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U}))^T (\boldsymbol{\Sigma}_{\tilde{\mathbf{Y}}})^{-1} (b \times vec\tilde{\mathbf{Y}}) + \boldsymbol{\Sigma}_G^{-1} \boldsymbol{\mu}_G \right) \\
&\boldsymbol{\Sigma}'_G = \\
&\left(b \times (\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U}))^T (\boldsymbol{\Sigma}_{\tilde{\mathbf{Y}}})^{-1} (\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U})) + (\boldsymbol{\Sigma}_G)^{-1} \right)^{-1} ,
\end{aligned} \tag{29}$$

where $\mathbf{I}_S$ denotes an $S \times S$ identity matrix with $S = \prod_{m=1}^{M} S_m$. And for σ^2, the posterior given all other parameters and b also follows inverse gamma distribution. That is

$$\sigma^2 \sim IG(\alpha', \beta') \tag{30}$$

with $\alpha' = \alpha + \frac{b \times NQ}{2}$, $\beta' = \beta + \frac{b \times \|\mathbb{Y} - \langle \mathbb{X}, \mathbb{B} \rangle_L\|_F^2}{2}$, and $Q = \prod_{m=1}^{M} Q_m$.

A.5 Proof for equation (15)

The Fractional Bayes Factor, Eq(11) of O'Hagan (1995), is given by

$$B_b(\mathbb{Y}) = \underbrace{\frac{\int p(\tilde{\boldsymbol{\xi}} \mid \tilde{\boldsymbol{\theta}}) p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}}) d\tilde{\boldsymbol{\xi}}}{\int p(\tilde{\boldsymbol{\xi}} \mid \tilde{\boldsymbol{\theta}}) p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^b d\tilde{\boldsymbol{\xi}}}}_{(**)} \times \frac{\int p(\tilde{\tilde{\boldsymbol{\xi}}} \mid \boldsymbol{\theta}^{(t-1)}) p(\mathbb{Y} \mid \tilde{\tilde{\boldsymbol{\xi}}}, \boldsymbol{\theta}^{(t-1)})^b d\tilde{\tilde{\boldsymbol{\xi}}}}{\int p(\tilde{\tilde{\boldsymbol{\xi}}} \mid \boldsymbol{\theta}^{(t-1)}) p(\mathbb{Y} \mid \tilde{\tilde{\boldsymbol{\xi}}}, \boldsymbol{\theta}^{(t-1)}) d\tilde{\tilde{\boldsymbol{\xi}}}}.$$

We note that

$$(**) = \frac{\overbrace{\int p(\tilde{\boldsymbol{\xi}} \mid \tilde{\boldsymbol{\theta}}) p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^b}^{(***)} p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^{(1-b)} d\tilde{\boldsymbol{\xi}}}{\int p(\tilde{\boldsymbol{\xi}} \mid \tilde{\boldsymbol{\theta}}) p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^b d\tilde{\boldsymbol{\xi}}}.$$

By similar techniques in O'Hagan (1995), we can rewrite $p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^b$ as $p(\mathbb{Y}' \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})$ that is the likelihood based on $\mathbb{Y}'$ which is the training proportion b of data $\mathbb{Y}$. And rewrite (***) as

$$\begin{aligned}(***) &= p(\tilde{\boldsymbol{\xi}} \mid \tilde{\boldsymbol{\theta}}) p(\mathbb{Y}' \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}}) \\ &= p(\tilde{\boldsymbol{\xi}} \mid \mathbb{Y}', \tilde{\boldsymbol{\theta}}) \int p(\tilde{\boldsymbol{\xi}} \mid \tilde{\boldsymbol{\theta}}) p(\mathbb{Y}' \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}}) d\tilde{\boldsymbol{\xi}} \\ &= p(\tilde{\boldsymbol{\xi}} \mid \mathbb{Y}', \tilde{\boldsymbol{\theta}}) \int p(\tilde{\boldsymbol{\xi}} \mid \tilde{\boldsymbol{\theta}}) p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^b d\tilde{\boldsymbol{\xi}}.\end{aligned}$$

Then $B_b(\mathbb{Y})$ becomes

$$B_b(\mathbb{Y}) = \frac{\int p(\tilde{\boldsymbol{\xi}} \mid \mathbb{Y}', \tilde{\boldsymbol{\theta}}) p(\mathbb{Y} \mid \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\theta}})^{(1-b)} d\tilde{\boldsymbol{\xi}}}{\int p(\tilde{\tilde{\boldsymbol{\xi}}} \mid \mathbb{Y}', \boldsymbol{\theta}^{(t-1)}) p(\mathbb{Y} \mid \tilde{\tilde{\boldsymbol{\xi}}}, \boldsymbol{\theta}^{(t-1)})^{(1-b)} d\tilde{\tilde{\boldsymbol{\xi}}}}. \tag{31}$$

And (*) in (15) is evaluating the Fractional Bayes Factor $B_b(\mathbb{Y})$ at one sample of $\tilde{\boldsymbol{\xi}}$ and $\tilde{\tilde{\boldsymbol{\xi}}}$ instead of integrating out as in (31).

A.6 More general covariance structures for prior distributions and noise

In the main manuscript, we assign normal priors with diagonal covariance matrices for $\{\mathbf{U}_{(l)}\}_{l=1}^{L}$, $\{\mathbf{V}_{(m)}\}_{m=1}^{M}$, $\mathbb{G}$, and show that the full conditional posterior distributions can be derived in closed-forms. When the covariance matrices, $\boldsymbol{\Sigma}_U$, $\boldsymbol{\Sigma}_V$, and $\boldsymbol{\Sigma}_G$ have general structures, we can still derive the full conditional posteriors in closed-forms by rearranging the elements of covariance matrices according to the way we unfold matrices $\mathbf{U}, \mathbf{V}$ and tensor $\mathbb{G}$. For example, given that the prior of $vec\mathbf{V}_{(1)}$ is a normal distribution with arbitrary covariance matrix $\boldsymbol{\Sigma}_V$, to update $vec\mathbf{V}_{(1)}^T$, we need the covariance matrix of $vec\mathbf{V}_{(1)}^T$ by rearranging the elements of $\boldsymbol{\Sigma}_V$, and then the posterior distribution of $vec\mathbf{V}_{(1)}^T$ is given in the same formula as shown in equations (9) and (10). Similarly, to update $\mathbb{G}$, we need the covariance matrix of $vec\mathbb{G}_{\mathcal{R}\times\mathcal{C}}$, denoted as $\tilde{\boldsymbol{\Sigma}}_G$, which can be accessed by rearranging the elements of $\boldsymbol{\Sigma}_G$, the covariance matrix of $vec\mathbb{G}$.

For the noise term $vec\mathbb{E}$, we assume that $vec\mathbb{E}$ follows a normal distribution with a diagonal covariance matrix $\sigma^2\mathbf{I}_{NQ}$ in the main manuscript. Here we consider an alternative construction: $vec\mathbb{E} \sim N(0, \boldsymbol{\Sigma}_E)$ with $\boldsymbol{\Sigma}_E = \boldsymbol{\Sigma} \otimes \mathbf{I}_N$ in the form of Kronecker product, and an Inverse-Wishart prior to $\boldsymbol{\Sigma}$, i.e., $\boldsymbol{\Sigma} \sim IW(\boldsymbol{\Psi}, \nu)$. Under such a prior setup, the posterior updates of $vec\mathbf{U}_{(l)}$, $vec\mathbf{V}_{(s)}$, $vec\mathbb{G}$, and $\boldsymbol{\Sigma}$ are still in closed forms.

Specifically, the full conditional posterior distribution of $vec\mathbf{U}_{(1)}$ is normally distributed:

$$p(vec\mathbf{U}_{(1)} \mid vec\mathbb{Y}, \mathbb{X}, \boldsymbol{\Sigma}_E, \mathbf{V}_{(m)}, \mathbf{U}_{(l)}\ l \neq 1) \sim N(\boldsymbol{\mu}_U', \boldsymbol{\Sigma}_U'),$$

where

$$\begin{aligned}\boldsymbol{\Sigma}_U' &= \left(\mathbb{C}_{(\mathcal{R}\times\mathcal{C})}^T \boldsymbol{\Sigma}_E^{-1} \mathbb{C}_{(\mathcal{R}\times\mathcal{C})} + \boldsymbol{\Sigma}_U^{-1}\right)^{-1}, \\ \boldsymbol{\mu}_U' &= \boldsymbol{\Sigma}_U' \left(\mathbb{C}_{(\mathcal{R}\times\mathcal{C})}^T \boldsymbol{\Sigma}_E^{-1} vec\mathbb{Y} + \boldsymbol{\Sigma}_U^{-1} \boldsymbol{\mu}_U\right).\end{aligned}$$

And the full conditional posterior distribution of $vec\mathbf{V}_{(1)}{}^T$ is normally distributed:

$$p(vec\mathbf{V}_{(1)}^T \mid vec\tilde{\mathbf{Y}}, \mathbb{X}, \boldsymbol{\Sigma}_E, \mathbf{U}_{(l)}, \mathbf{V}_{(m)}\ m \neq 1) \sim N(\tilde{\boldsymbol{\mu}}_V', \tilde{\boldsymbol{\Sigma}}_V'),$$

where

$$\begin{aligned}\tilde{\boldsymbol{\Sigma}}_V' &= \left(\left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})}\right)^T \tilde{\boldsymbol{\Sigma}}^{-1} \left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})}\right) + \tilde{\boldsymbol{\Sigma}}_V^{-1}\right)^{-1}, \\ \tilde{\boldsymbol{\mu}}_V' &= \tilde{\boldsymbol{\Sigma}}_V' \left(\left(\mathbf{I}_{Q_1} \otimes \mathbb{D}_{(\mathcal{R}\times\mathcal{C})}\right)^T \tilde{\boldsymbol{\Sigma}}^{-1} vec\tilde{\mathbf{Y}} + \tilde{\boldsymbol{\Sigma}}_V^{-1} \tilde{\boldsymbol{\mu}}_V\right),\end{aligned}$$

and $\tilde{\boldsymbol{\Sigma}}$ is the covariance matrix of $vec\left(\mathbb{E}_{(2)}^T\right)$ by rearranging elements of $\boldsymbol{\Sigma}_E$.

The derivation of the full conditional posterior of $vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}$ is relatively more complex compared to the posteriors of $\mathbf{U}$ and $\mathbf{V}$. Starting from

$$vec\tilde{\tilde{\mathbf{Y}}} = (\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U}))vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})} + vec\tilde{\tilde{\mathbf{E}}},$$

where $\tilde{\tilde{\mathbf{E}}} = \mathbb{E}_{(1)}\mathbf{V}(\mathbf{V}^T\mathbf{V})^{-1}$, we can write $vec\tilde{\tilde{\mathbf{E}}} = \left(\left((\mathbf{V}^T\mathbf{V})^{-1}\mathbf{V}^T\right) \otimes \mathbf{I}_N\right) vec\mathbb{E}$. Then the covariance matrix of $vec\tilde{\tilde{\mathbf{E}}}$ becomes

$$\tilde{\tilde{\boldsymbol{\Sigma}}} = \tilde{\mathbf{V}}\boldsymbol{\Sigma}_E\tilde{\mathbf{V}}^T,$$

Table A.1. Mean RPE with SD on uncorrelated-t data

RPE (SD)	BayTensor MCMC	BayTensor Fast	CP Method	OLS Method	SAS Method	HS Method
$\theta^* = (4,4,2,2)$, SNR=2	**0.353 (0.017)**	0.374 (0.097)	0.407 (0.037)	1.019 (0.029)	0.968 (0.024)	1.058 (0.137)
$\theta^* = (3,3,3,3)$, SNR=2	**0.344 (0.004)**	0.350 (0.016)	0.415 (0.032)	1.014 (0.032)	0.958 (0.025)	1.109 (0.281)
$\theta^* = (4,4,2,2)$, SNR=5	**0.173 (0.002)**	0.208 (0.125)	0.235 (0.035)	0.747 (0.028)	0.949 (0.030)	1.104 (0.227)
$\theta^* = (3,3,3,3)$, SNR=5	**0.172 (0.001)**	0.182 (0.038)	0.256 (0.038)	0.747 (0.024)	0.931 (0.035)	1.173 (0.388)

Table A.2. Core tensor dimension recovery and number of model parameters on uncorrelated-t data

	BayTensor MCMC		BayTensor Fast	
	Dimension Recovery	# Parameters (SD)	Dimension Recovery	# Parameters (SD)
$\theta^* = (4,4,2,2)$, SNR=2	72%	205 (19)	42%	188 (29)
$\theta^* = (3,3,3,3)$, SNR=2	96%	218 (6)	58%	205 (26)
$\theta^* = (4,4,2,2)$, SNR=5	90%	213 (10)	68%	195 (31)
$\theta^* = (3,3,3,3)$, SNR=5	100%	219 (0)	82%	213 (24)

where $\tilde{\mathbf{V}} = \left(\left((\mathbf{V}^T\mathbf{V})^{-1}\mathbf{V}^T\right) \otimes \mathbf{I}_N\right)$. The full conditional posterior distribution of $vec\mathbb{G}_{(\mathcal{R}\times\mathcal{C})}$ is a normal distribution with

$$\tilde{\boldsymbol{\mu}}_G' = \tilde{\boldsymbol{\Sigma}}_G' \left((\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U}))^T \tilde{\tilde{\boldsymbol{\Sigma}}}^{-1} vec\tilde{\tilde{\mathbf{Y}}} + \tilde{\boldsymbol{\Sigma}}_G^{-1}\tilde{\boldsymbol{\mu}}_G \right),$$

$$\tilde{\boldsymbol{\Sigma}}_G' = \left((\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U}))^T \tilde{\tilde{\boldsymbol{\Sigma}}}^{-1} (\mathbf{I}_S \otimes (\mathbb{X}_{(1)}\mathbf{U})) + (\tilde{\boldsymbol{\Sigma}}_G)^{-1} \right)^{-1}.$$

Lastly, the full conditional posterior of $\boldsymbol{\Sigma}$ is an Inverse-Wishart distribution with $\tilde{\nu} = N + \nu$, and $\tilde{\boldsymbol{\Psi}} = \mathbf{S}^T\mathbf{S} + \boldsymbol{\Psi}$ where $\mathbf{S} = \mathbb{Y}_{(1)} - \mathbb{X}_{(1)}\mathbb{B}_{(\mathcal{P}\times\mathcal{Q})}$.

A.7 Simulation results: uncorrelated-t setup

We present the detailed results under the proposed BayTensor MCMC, BayTensor Fast, and four alternative methods (including the CP method, the OLS method, the SAS method, and the HS method) to datasets under the uncorrelated-t setup.

The prediction RPE results are presented in Table A.1. The RPE results under Baytensor MCMC and BayTensor Fast are comparable in all cases. And both of them have better RPEs than the CP method followed by the SAS method, and the OLS method. The HS performs the worst in terms of yielding the highest RPEs in all cases. And when the signal to noise ratio increases from 2 to 5, the prediction accuracies are improved under all methods, except for the HS method.

Table A.2 shows the empirical probabilities of dimension recovery and the average numbers of parameters required by BayTensor MCMC and BayTensor Fast in 50 replicated experiments. In all 4 cases, BayTensor MCMC and BayTensor Fast both require a smaller number of parameters than the CP method. And in terms of recovering the dimension of the core tensor, BayTensor MCMC has higher empirical probabilities of recovering the true dimension than BayTensor Fast in all cases. And the recovering probabilities in the cases where $\boldsymbol{\theta}^* = (3,3,3,3)$ are higher than those in cases where $\boldsymbol{\theta}^* = (4,4,2,2)$ for both methods.

We also observe that the RPEs under uncorrelated-t setups are larger than those under uncorrelated-normal setups for the BayTensor methods and the CP method. This is expected because the fitting model is different from the data generation model. However, the magnitudes of RPE increase under BayTensor algorithms are small. Moreover, the dimension recovery rates and number of parameters are also similar to those under uncorrelated-normal setups. These results indicate that the proposed BayTensor algorithms have reasonably good performance when the fitting model is misspecified.

A.8 Traceplots of the estimated number of parameters associated with θ in simulation study

We plot the estimated number of parameters associated with $\theta^{(t)}$ versus iteration t in the first 100 iterations of BayTensor MCMC algorithm for some randomly-selected experiments under the uncorrelated-normal simulation setup in Figure A.1, and under the correlated setup in Figure A.2.

A.9 Full simulation study results of CP method with different ranks

For each case of the simulation study, we tried the CP method with CP-rank R values from 1 to 6. Note that when $R = 5, 6$, the total number of parameters in CP method are 230 and 276 respectively which are larger than the true number of parameters (212 for $\boldsymbol{\theta}^* = (4,4,2,2)$ cases and 219 for $\boldsymbol{\theta}^* = (3,3,3,3)$ cases). We calculate the RPE results with different R values for all 50 repeated experiments under all simulation setups. And means and standard deviations of RPEs averaging over the 50 repeated experiments for all 8 simulation setups are shown in Table A.3.

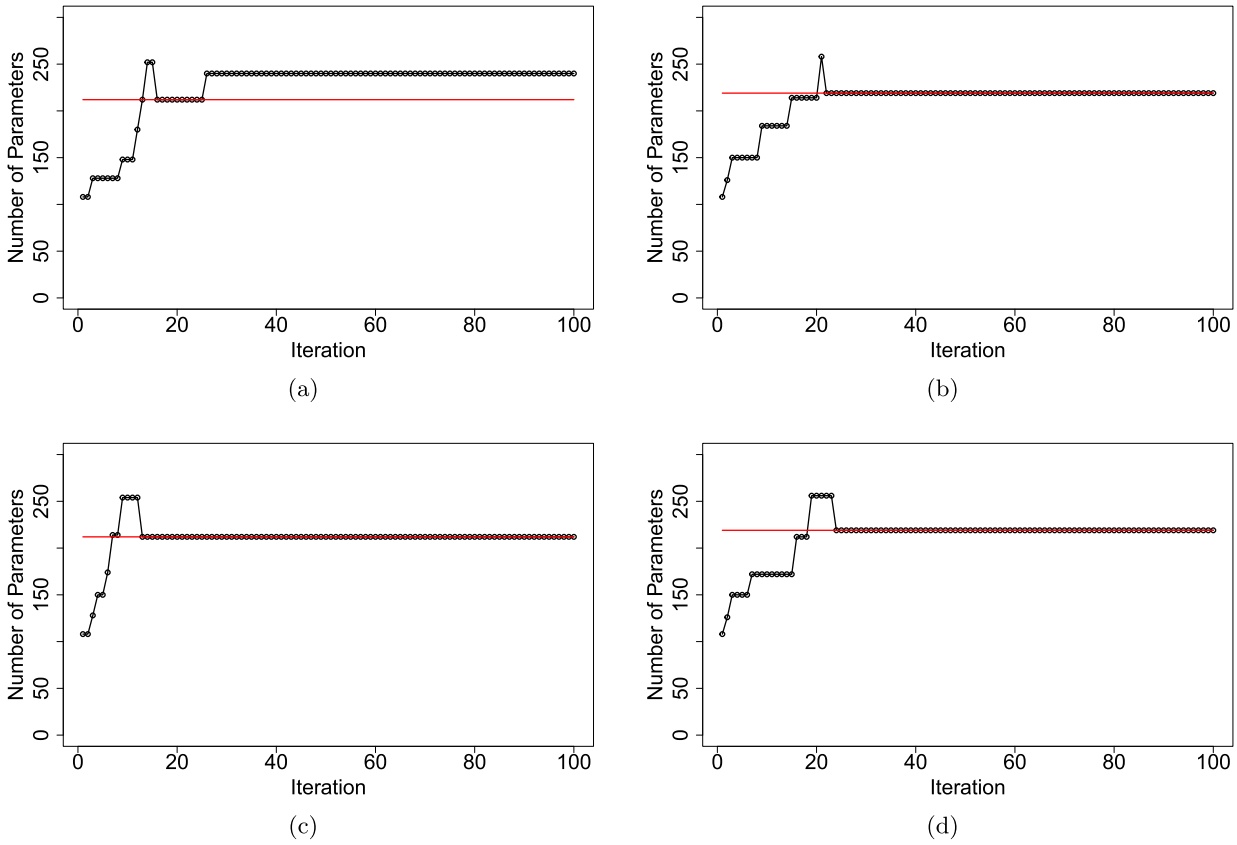


Figure A.1. The estimated number of parameters associated with $\theta^{(t)}$ versus iteration t in the first 100 iterations of BayTensor MCMC algorithm for some randomly-selected experiments under the uncorrelated-normal simulation setup. The true number of parameters is shown by the red line. (a): SNR = 2, $\theta^* = (4, 4, 2, 2)$. (b): SNR = 2, $\theta^* = (3, 3, 3, 3)$. (c): SNR = 5, $\theta^* = (4, 4, 2, 2)$. (d): SNR = 5, $\theta^* = (3, 3, 3, 3)$.

Table A.3. Mean RPE with SD for the CP method with different R values

RPE (SD)	$R = 1$	$R = 2$	$R = 3$	$R = 4$	$R = 5$	$R = 6$
Uncorrelated Data						
$\theta^* = (4, 4, 2, 2)$, SNR=2	0.771(0.102)	0.627(0.087)	0.529(0.095)	0.463(0.061)	0.409(0.036)	0.377(0.014)
$\theta^* = (3, 3, 3, 3)$, SNR=2	0.780(0.081)	0.623(0.061)	0.535(0.048)	0.466(0.038)	0.422(0.024)	0.391(0.016)
$\theta^* = (4, 4, 2, 2)$, SNR=5	0.718(0.137)	0.530(0.109)	0.392(0.072)	0.308(0.052)	0.241(0.039)	0.209(0.046)
$\theta^* = (3, 3, 3, 3)$, SNR=5	0.718(0.095)	0.527(0.074)	0.414(0.060)	0.320(0.044)	0.266(0.032)	0.222(0.016)
Correlated Data						
$\theta^* = (4, 4, 2, 2)$, SNR=2	0.533(0.152)	0.423(0.095)	0.382(0.043)	0.371(0.021)	0.370(0.011)	0.377(0.014)
$\theta^* = (3, 3, 3, 3)$, SNR=2	0.514(0.120)	0.429(0.070)	0.391(0.037)	0.374(0.019)	0.370(0.010)	0.369(0.007)
$\theta^* = (4, 4, 2, 2)$, SNR=5	0.407(0.170)	0.266(0.104)	0.219(0.056)	0.198(0.034)	0.190(0.014)	0.188(0.006)
$\theta^* = (3, 3, 3, 3)$, SNR=5	0.413(0.187)	0.280(0.085)	0.230(0.047)	0.205(0.026)	0.194(0.015)	0.187(0.006)

ACKNOWLEDGMENTS

This work was supported by NSF 1940107, NSF 1918854, and NIH R01MH128085 to Dr. Xu.

Received 6 October 2022

REFERENCES

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 37(2):697–725. MR2502648

Billio, M., Casarin, R., Kaufmann, S., and Iacopini, M. (2018). Bayesian dynamic tensor regression. *University Ca'Foscari of Venice, Dept. of Economics Research Paper Series No*, 13.

Carvalho, C. M., Polson, N. G., and Scott, J. G. (2010). The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480.

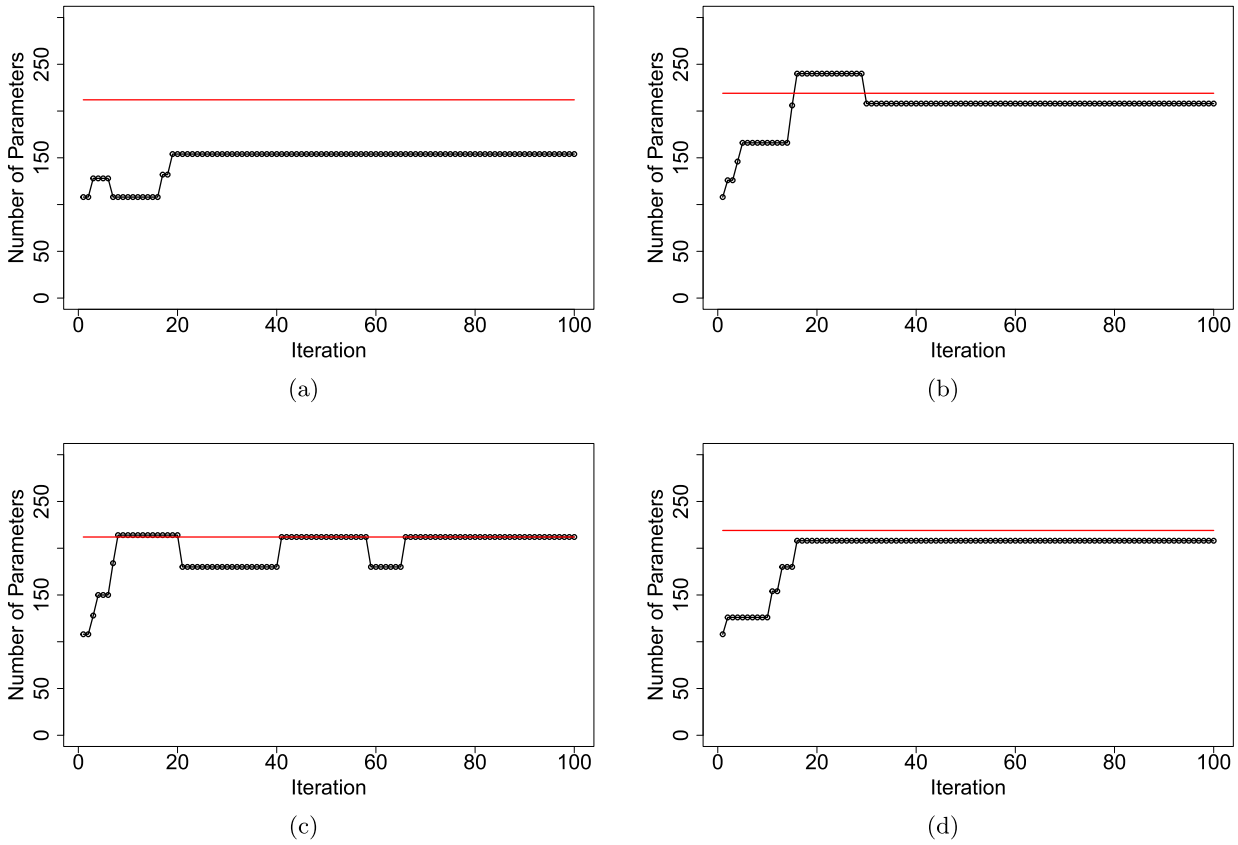


Figure A.2. The estimated number of parameters associated with $\theta^{(t)}$ versus iteration t in the first 100 iterations of BayTensor MCMC algorithm for some randomly-selected experiments under the correlated simulation setup. The true number of parameters is shown by the red line. (a): SNR $= 2$, $\theta^* = (4, 4, 2, 2)$. (b): SNR $= 2$, $\theta^* = (3, 3, 3, 3)$. (c): SNR $= 5$, $\theta^* = (4, 4, 2, 2)$. (d): SNR $= 5$, $\theta^* = (3, 3, 3, 3)$.

MR2650751

Dosso, S. and Oldenburg, D. (1991). Magnetotelluric appraisal using simulated annealing. *Geophysical Journal International*, 106(2):379–385.

Gahrooei, M. R., Yan, H., Paynabar, K., and Shi, J. (2021). Multiple tensor-on-tensor regression: An approach for modeling processes with heterogeneous sources of data. *Technometrics*, 63(2):147–159. MR4251490

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):721–741.

Green, P. J. (1995). Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732. MR1380810

Guhaniyogi, R., Qamar, S., and Dunson, D. B. (2017). Bayesian tensor regression. *The Journal of Machine Learning Research*, 18(1):2733–2763. MR3714242

Guhaniyogi, R. and Spencer, D. (2021). Bayesian tensor response regression with an application to brain activation studies. *Bayesian Analysis*, 16(4):1221–1249. MR4381133

Guo, W., Kotsia, I., and Patras, I. (2012). Tensor learning for regression. *IEEE Transactions on Image Processing*, 21(2):816–827. MR2932176

Harshman, R. A. (1970). Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis.

Hasan, K. M., Walimuni, I. S., Abid, H., and Hahn, K. R. (2011). A review of diffusion tensor magnetic resonance imaging computational methods and software tools. *Computers in Biology and Medicine*, 41(12):1062–1072.

Hassner, T., Harel, S., Paz, E., and Enbar, R. (2015). Effective face frontalization in unconstrained images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4295–4304.

Hoff, P. D. (2015). Multilinear tensor regression for longitudinal relational data. *The annals of applied statistics*, 9(3):1169. MR3418719

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*. MR3081926

Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E. (2007). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst.

Ishwaran, H. and Rao, J. S. (2005). Spike and slab variable selection: frequentist and bayesian strategies. *The Annals of Statistics*, 33(2):730–773. MR2163158

Kirkpatrick, S., Gelatt Jr, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680. MR0702485

Kolda, T. G. (2006). *Multilinear operators for higher-order decompositions*, volume 2. United States. Department of Energy.

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM Review*, 51(3):455–500. MR2535056

Kumar, N., Berg, A., Belhumeur, P. N., and Nayar, S. (2011). Describable visual attributes for face verification and image search.

IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(10):1962–1977.

Lankao, P. R., Nychka, D., and Tribbia, J. L. (2008). Development and greenhouse gas emissions deviate from the 'modernization'theory and 'convergence'hypothesis. *Climate Research*, 38(1):17–29.

Lee, J., Müller, P., Sengupta, S., Gulukota, K., and Ji, Y. (2016). Bayesian feature allocation models for tumor heterogeneity. In *Statistical Analysis for High-Dimensional Data*, pages 211–232. Springer. MR3616270

Li, L. and Zhang, X. (2017). Parsimonious tensor response regression. *Journal of the American Statistical Association*, 112(519):1131–1146. MR3735365

Li, X., Xu, D., Zhou, H., and Li, L. (2013). Tucker tensor regression and neuroimaging analysis. *Statistics in Biosciences*, pages 1–26.

Lock, E. F. (2018). Tensor-on-tensor regression. *Journal of Computational and Graphical Statistics*, pages 1–10. MR3863764

Miranda, M. F., Zhu, H., Ibrahim, J. G., et al. (2018). Tprm: Tensor partition regression models with applications in imaging biomarker detection. *The Annals of Applied Statistics*, 12(3):1422–1450. MR3852683

O'Hagan, A. (1995). Fractional bayes factors for model comparison. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):99–118. MR1325379

Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311. MR0205395

Van der Aa, N., Luo, X., Giezeman, G.-J., Tan, R. T., and Veltkamp, R. C. (2011). Umpm benchmark: A multi-person dataset with synchronized video and motion capture data for evaluation of articulated human motion and interaction. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1264–1269. IEEE.

Vasilescu, M. A. O. and Terzopoulos, D. (2002). Multilinear analysis of image ensembles: Tensorfaces. In *European Conference on Computer Vision*, pages 447–460. Springer.

Wang, M., Fischer, J., and Song, Y. S. (2019). Three-way clustering of multi-tissue multi-individual gene expression data using semi-nonnegative tensor decomposition. *The Annals of Applied Statistics*, 13(2):1103. MR3963564

Zhang, L., Guindani, M., and Vannucci, M. (2015). Bayesian models for functional magnetic resonance imaging data analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(1):21–41. MR3348719

Zhou, H., Li, L., and Zhu, H. (2013). Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association*, 108(502):540–552. MR3174640

Kunbo Wang
3400 N. Charles Street
Baltimore, MD 21218
E-mail address: kunbo.wang@jhu.edu

Yanxun Xu
3400 N. Charles Street
Baltimore, MD 21218
E-mail address: yanxun.xu@jhu.edu