Bayesian Nonparametric Tensor Completion

Amir Forouzandeh
Department of Computing Science

Jude Kong
Department of Mathematics

Alex Lewandowski
Department of Statistics

Mehran Mahmoudi
Department of Computing Science

Gian Matharu
Department of Physics

Abstract

In this paper, we propose a Bayesian nonparametric method to estimate missing data in tensors. The proposed method uses a Tucker-1 factorization to learn a smaller core tensor and a factor matrix via Gibbs sampling. Unlike most existing Tucker factorization algorithms, the tensor rank is estimated automatically. This is done by employing an Indian Buffet Process prior over the latent matrix, where the estimated number of non-zero columns corresponds to the tensor rank. The missing data is then estimated based on the latent structure learned from the factorization. As a result, the method provides a form of Bayesian model averaging over estimated ranks, allowing it to avoid overfitting. As an illustration, a number of datasets are artificially corrupted at various levels. Relative error is used to assess completion performance and our result is compared to parallel matrix factorization as a baseline. The empirical results show that our method does not beat the current state of the art, but provides pertinent information that other methods do not. That is, statistical quantities, like mean standard error, are readily available due to the Bayesian estimation framework of our method. Lastly, we identify some future avenues for research that may improve performance.

1 Introduction

Missing data is a problem that frequently arises during empirical analyses. In particular, domain specific phenomena may cause data to be missing for various reasons. Consider the following three cases: First, in the field of medical science, datasets may have missing data because of patients that do not want to disclose certain information about themselves. It is also possible that certain medical procedures are not performed on selected patients. In this case, one may need to estimate the missing data to perform further data analysis. Second, in the context of collaborative filtering, datasets that involve consumer ratings may have missing values because not every product has been consumed. Hence, estimates for missing values act as recommendations. Third, in the field of image processing it is common to have a corrupted image. Such missing data can have a negative impact on data analysis including biased parameter estimates and inflated standard errors [1, 2].

Many datasets can be organized as a two-dimensional array (matrix). A tensor (multi-dimensional array) provides an effective way of representing the structural properties of higher dimensional data. Colour images are simple examples of third order tensors since they consist of different intensities of red, green, and blue for each pixel. Tensor completion is a procedure that facilitates the imputation of missing data by using only the available entries and structural properties of the tensor. This can only be possible if there is a relationship between the missing entries and available entries. Techniques for tensor completion are an active area of research, due their potential applications and flexibility [3, 4].

Most work on tensor completion is based on Tucker [5] and CANDECOMP/PARAFAC (CP) [6] tensor factorizations which are introduced at a later stage. Both of these factorizations require an estimate of the tensor rank as a parameter. However, finding the true tensor rank is NP-hard [7].

In this project, we attempt to address these shortcomings by taking a probabilistic approach to the Tucker factorization. Our algorithm combines a Gibbs sampler and a Tucker-1 factorization to determine the ranks automatically. In particular, our model employs an Indian Buffet process prior over the latent factor matrix, which provides sparsity needed to estimate the tensor rank. The designed algorithm takes in an incomplete data set in tensor form, and produces estimates of the missing data values based on the latent structure. To evaluate our work, we use a set of artificially corrupted datasets and measure the relative error. Our results are compared to parallel matrix factorization (PMF) as a baseline [8]. However, since the algorithm produces a sample, we are also able to obtain various statistical quantities that other methods cannot.

The remainder of this paper is organized as follows: in section 2 we discuss background on tensors, including notation, definitions and common factorizations. In section 3, we review related work on tensor completion. In section 4, we discuss tensors in a Bayesian nonparametric framework and outline a Gibbs sampling method for inference. In section 5, we discuss MRI data as an illustrative example and outline our results on other datasets. Lastly, in section 6 we summarize what we did and outline avenues for further research.

2 Background

In this section we briefly review tensor notation, definitions and common factorizations. To maintain consistency with previous tensor literature, we use the notation conventions and definitions of [3]. We only include material pertinent to this study. For a more complete discussion of tensor theory, the reader is referred to [3].

2.1 Tensors

A tensor is a multidimensional array. The number of dimensions of a tensor is known as the order (used interchangeably with "modes" or "ways"). Vectors and matrices represent first and second order tensors, respectively. Vectors are denoted by bold lower-case letters e.g. $\mathbf{x}$, whereas matrices are denoted by bold upper-case letters e.g. $\mathbf{X}$. For this paper we restrict the usage of the term tensor to higher order tensors ($N \geq 3$) and represent them using Euler script (e.g. $\mathcal{X}$). Using index notation, a single element of a vector, matrix or tensor can be given as x_i, x_{ij} and x_{ijk} respectively.

2.2 Tensor norm

The norm of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined as

$$\|\mathcal{X}\|_F = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1,i_2,\ldots,i_N}}.$$

This definition provides a tensor equivalent to the Frobenius norm used for matrices.

2.3 Tensor rank and the CP decomposition

An N-th order tensor is defined to be rank one if it can be represented as the outer product of N vectors $\mathbf{a}^{(i)}$ for $i = 1, 2, \ldots, N$.

$$\mathcal{X} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(N)}.$$

The rank of a tensor, rank($\mathcal{X}$), is the smallest number of rank-one tensors that can be summed to form $\mathcal{X}$. For a third order tensor this can be expressed as

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$$

for vectors $\mathbf{a}_r \in \mathbb{R}^I, \mathbf{b}_r \in \mathbb{R}^J, \mathbf{c}_r \in \mathbb{R}^K$ and an integer $R > 0$. $\circ$ denotes an outer product. R represents the tensor rank. The above summation is known as an exact CP decomposition. An approximate factorization can be achieved by a truncated summation of rank one tensors (i.e. fewer than rank($\mathcal{X}$) summations). Determining the rank of a tensor is NP-hard [3]. A more prevalent definition of rank for tensors is the n-rank which relates to the ranks of matricizations of a tensor.

2.4 Matricization of a tensor and n-rank

The matricization (or unfolding) of a tensor involves fixing all but one index of a tensor and placing the remaining elements into columns of a matrix. The mathematical definition of the unfolding process is somewhat inelegant, therefore we provide an example to provide intuition.

An n^{th} order tensor has n mode unfoldings denoted by $\mathbf{X}_{(n)}$. Consider a 3rd-order tensor $\mathcal{X} \in \mathbb{R}^{2\times3\times2}$. The two layers of the tensor are

$$\mathcal{X}(:,:,1) = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}, \ \mathcal{X}(:,:,2) = \begin{bmatrix} 7 & 9 & 11 \\ 8 & 10 & 12 \end{bmatrix}$$

and the corresponding mode unfoldings are given by

$$\mathbf{X}_{(1)} = \begin{bmatrix} 1 & 3 & 5 & 7 & 9 & 11 \\ 2 & 4 & 6 & 8 & 10 & 12 \end{bmatrix},$$

$$\mathbf{X}_{(2)} = \begin{bmatrix} 1 & 2 & 7 & 8 \\ 3 & 4 & 9 & 10 \\ 5 & 6 & 11 & 12 \end{bmatrix},$$

$$\mathbf{X}_{(3)} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 8 & 9 & 10 & 11 & 12 \end{bmatrix}.$$

Let $\mathcal{X} \in \mathbb{R}^{I\times J\times K}, \mathbf{A} \in \mathbb{R}^{M\times I}, \mathcal{Y} \in \mathbb{R}^{M\times J\times K}$. Then a common way to express the product of a matrix and a tensor is in the form of a indexed-product. Below, we show an example where the index is 1, which we use in our work.

$$\mathcal{Y} = \mathcal{X} \times_1 \mathbf{A} \Leftrightarrow \mathcal{Y}_{mjk} = \sum_i \mathcal{X}_{ijk} B_{mi}$$

Having defined the n-mode unfoldings we now define the n-rank. The n-rank is a set of ranks $R_1, R_2, \ldots, R_n$ that provide the ranks for each of the n-mode unfoldings (R_i represents the rank of $\mathbf{X}_{(i)}$).

2.5 Tucker factorization

The tucker factorization represents a generalization of singular value decomposition (SVD) and principal component analysis applied to higher order tensors. For a third order tensor, the tucker factorization can be presented as

$$\mathcal{X}_{ijk} \approx \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} \mathcal{G}_{pqr} \cdot a_{ip} \cdot b_{jq} \cdot c_{kr}$$

with factor matrices $\mathbf{A} \in \mathbb{R}^{I\times P}, \mathbf{B} \in \mathbb{R}^{J\times Q}, \mathbf{C} \in \mathbb{R}^{K\times R}$. The factor matrices represent the left singular vectors/principal components obtained by applying SVD to the n-mode unfoldings ($\mathcal{X}_{(n)}$) of the original tensor. The core tensor performs a role comparable to the matrix of singular values in SVD i.e. it carries weights for the various principal components. Two variations on the Tucker decomposition are the Tucker-2 and Tucker-1 factorizations. Tucker 1 replaces factor matrices $\mathbf{B}$ and $\mathbf{C}$ with identities. Tucker 2 replaces only $\mathbf{C}$ with an identity.

3 Related work

The work on tensor completion takes two main approaches. First, there are deterministic methods which tend to be optimization problems with the trace norm as the objective function. These

methods are usually convex and are quite fast. However, the current literature is not very focused on scalability and the optimization may become difficult for massive tensors. General approaches to the problem can be found in [9], where a CP factorization was used to impute missing entries. The problem becomes a weighted least-squares optimization, which can be solved quite quickly. In addition, [10] formulated an expectation-maximization type of algorithm that uses singular value decomposition to characterize subspaces as components of the Tucker factorization. One result is that these algorithms tend to overfit the data, and provide poor estimates of the missing entries, when only sparse observations are available. For our baseline, we use parallel matrix factorization for tensor completion [8]. The reason for this is that the method has state of the art performance and relatively straight forward implementation. One problem with all of these approaches is that the tensor rank must be manually specified. However, the determination of tensor rank remains a challenging problem [7] and most approaches treat the rank as a tuning parameter.

The second approach, which is the one that we take, is probabilistic. In particular, it combines a Bayesian nonparametric approach in [11] with automatic rank selection in [12]. The former paper places a nonparametric prior (Gaussian Process) on the core tensor, while we place a nonparametric prior over the factor matrices. The latter paper does not take a Bayesian nonparametric approach, and instead uses conventional Bayesian methods. These approaches, Bayesian nonparametric in particular, are computationally expensive. However, scalable methods are an active research area [13]. Indeed, while probabilistic methods are slower than deterministic ones, they are preferred due to the extra information you obtain, such as confidence intervals [14].

4 Probabilistic approach

4.1 Factorization

We consider a probabilistic approach to the Tucker factorization. In particular, we look at Tucker-1 which is a special case of the more general Tucker scheme. That is, we have two factor matrices chosen to be identity matrices of appropriate size.

$$\mathcal{X}_{ijk} = \sum_{r=1}^{R}\sum_{s=1}^{J}\sum_{t=1}^{K} \mathcal{G}_{rst} \cdot \mathbf{A}_{ir} \cdot \mathbb{1}_{(s=J)} \cdot \mathbb{1}_{(t=K)} + \epsilon_{ijk}$$

where $\epsilon_{ijk} \sim \mathcal{N}(0, \sigma^2_{\mathcal{X}})$

Consider a simple example, where $\mathcal{X}$ is an order 3 tensor of dimensions $3 \times 3 \times 3$. Then Tucker-1 attempts to find a matrix $\mathbf{A}$ of dimension $3 \times R$ and a tensor $\mathcal{G}$ of dimension $R \times 3 \times 3$, such that the product is close to $\mathcal{X}$. As mentioned earlier, traditional Tucker methods require specifying R as a parameter. This can be thought of a type of rank, but finding the value of R is an NP-hard problem and intractable with missing values. Hence, R is treated as a tuning parameter to minimize a measure of error. This approach is unsatisfactory for two reasons. First, training a model for multiple R values can be infeasible for large tensors. Second, the value of R may have importance in the context of the problem, as a proxy estimate for the rank.

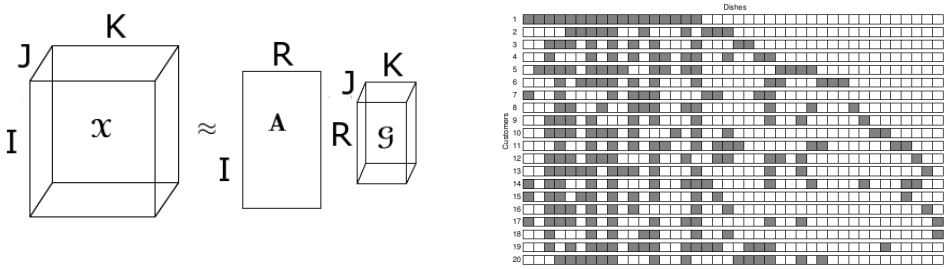


Figure 1: Left: Tucker-1 Decomposition, Right: A sample from an Indian Buffet Process

4.2 Bayesian nonparametric prior

Our approach to estimating R is to employ a nonparametric prior over the matrix $\mathbf{A}$. Consider the case where $R = \infty$, resulting in an infinite matrix $\mathbf{A}$ and infinite tensor $\mathcal{G}$. To ensure that the sum converges, we impose a sparsity inducing prior on $\mathbf{A}$. Specifically, we need the columns of $\mathbf{A}$ to be $\mathbf{0}$ vectors past some index R. Then, the problem is estimating the index R such that the $A(:, R + a) = \mathbf{0}$ for all $a > 0$.

One possibility is an Indian Buffet Process (IBP), which provides a prior over sparse infinite binary arrays. The process is commonly described as a culinary metaphor: The first customer enters an Indian restaurant and samples $poisson(\alpha)$ dishes. Then, customer i enters and adds dishes to their plate in proportion to the dishes popularity. Afterwards, they sample poisson($\frac{\alpha}{i}$) new dishes [15]. For example, say the first customer samples $poisson(4) = 4$ dishes. The second customer will go to each of these dishes and flip a coin. if the coin lands on heads they sample that dish, they skip it otherwise. They then sample $poisson(2)$ new dishes. This goes on for every customer and since there are only finite customers at each step, there will be finite non-zero columns.

This process is described inductively, but an important result is that the distribution is actually exchangeable. That is, the distribution does not depend on the order that the customers enter the restaurant. Another important detail is that there are an infinite amount of dishes, however the number of rows is fixed or known beforehand. In our simple example, we have that the number of customers corresponds to $I = 3$. The matrix generated from an IBP can be thought of as infinite, with $A(:, R + a) = \mathbf{0}$ for some $a > 0$, which is exactly the criterion that we need.

It can be easily seen that if $R_1^{(i)}$ is the number of dishes sampled by customer i, then $R_1^{(i)} \sim poisson(\frac{\alpha}{i})$. Then, we have that $R = \sum_{I=1}^{I} R_1^{(i)} = \sum_{i=1}^{I} poisson(\frac{\alpha}{i})$. Since the sum of poisson distributed variable is also poisson, we have that $R \sim poisson\left(\alpha \sum_{i=1}^{I} (\frac{1}{i})\right)$.

The exact form of the probability density function of an IBP is a technical derivation, depending on the notion of a left ordered equivalence class of matrices. The process can be constructed by taking the infinite limit of a Beta-Bernoulli hierarchical model, or by a stick-breaking construction [16]. The distribution can be written as follows, where H_I denotes the I-th harmonic number.

$$P(\mathbf{A}) = \frac{\alpha^R}{(N!)^R \prod_{i=1}^{I} R_1^{(i)}!} exp(-\alpha H_I) \prod_{n=1}^{R} (I - m_k)!(m_k - 1)!$$

Since we are taking a Bayesian approach, we describe the conditional distribution $P(A_{ir} = 1 | A_{-ir})$ where A_{-ir} denotes the elements of A other than A_{ir}. There are two cases to consider, either A_{ir} is a previously sampled dish or A_{ir} is a newly sampled dish. In the first case, A_{ir} is sampled with proportional probability $\frac{m_{-ik}}{N}$ where m_{-ik} denotes the number of ones in column r, not counting A_{ir}. In the second case, A_{ir} is sampled from a $poisson(\frac{\alpha}{N})$ [17]. In the next section, we will develop a Gibbs sampler for our model.

4.3 Completion

Missing data is typically dealt with imputation approaches, in which suitable estimates are substituted based on a "good guess". A common example is mean substitution or K-Nearest Neighbours. More complicated approaches can use the EM Algorithm, which finds the values that maximize the expectation of seeing the observed data. In a Bayesian setting, we can use Gibbs sampling to sample values for the missing values. If we are able to write the conditional distributions for the missing data and the unknown parameters, then the sequence of samples from the full conditionals will converge to samples from the posterior [15].

We make this idea more concrete by denoting the index set of missing values as $\mathcal{M}$. Then, we begin by writing the full joint probability as follows

$$P(\mathcal{X}, \mathcal{G}, \mathbf{A}, \sigma_{\mathcal{G}}^2, \sigma_{\mathcal{X}}^2, \alpha) \propto P(\mathcal{X} | \mathcal{G}, \mathbf{A}, \sigma_{\mathcal{X}}^2) P(\mathcal{G} | \sigma_{\mathcal{G}}^2) P(\mathbf{A} | \alpha) P(\sigma_{\mathcal{G}}^2) P(\sigma_{\mathcal{X}}^2) P(\alpha)$$

The first term in this expression is the likelihood, which is normally distributed because of the gaussian noise assumption. The second term, the prior on the core tensor, is also normally distributed by assumption. The third term is the IBP prior term that we specified to employ automatic rank estimation. The last three terms are hyper-priors, where $\alpha \sim Gamma(a_\alpha, b_\alpha)$ and

$\sigma^2_{\mathcal{X}} \sim invGamma(a_{\mathcal{X}}, b_{\mathcal{X}})$, $\sigma^2_{\mathcal{G}} \sim invGamma(a_{\mathcal{G}}, b_{\mathcal{G}})$. In practice, the hyper parameters are set to $a = b = 1$.

The assumptions placed on the various terms are for ease of computation and conjugacy. In particular, observe that the prior on $\mathcal{G}$ is conjugate to the likelihood term. Although the samples from $\mathcal{G}$ may give insight on the latent structure of $\mathcal{X}$, we are mostly interested in the samples of the missing values $\mathcal{X}_{\mathcal{M}}$. Since we are interested in sampling $\mathcal{X}_{\mathcal{M}}$ for tensor completion, we can integrate $\mathcal{G}$ out of the joint distribution for a faster Gibbs sampler [18]. The resulting joint distribution can be written as

$$P(\mathcal{X}, \mathbf{A}, \sigma^2_{\mathcal{G}}, \sigma^2_{\mathcal{X}}, \alpha) \propto P(\mathcal{X}|\sigma^2_{\mathcal{G}}, \mathbf{A}, \sigma^2_{\mathcal{X}})P(\mathbf{A}|\alpha)P(\sigma^2_{\mathcal{G}})P(\sigma^2_{\mathcal{X}})P(\alpha)$$

Now, we write the the full conditional for updating the matrix $\mathbf{A}$ as follows

$$P(A_{i,j} = 1|\mathcal{X}, \mathbf{A}_{-(i,j)}, \sigma^2_{\mathcal{G}}, \sigma^2_{\mathcal{X}}) \propto P(\mathcal{X}|\mathbf{A}, \sigma^2_{\mathcal{G}}, \sigma^2_{\mathcal{X}})P(A_{i,j} = 1|\mathbf{A}_{-(i,j)})$$

The first term is simply the likelihood, which is distributed as $\mathcal{N}(\mathbf{A} \times_1 \mathcal{G}, \sigma^2_{\mathcal{G}})$. The second term is the prior based on the IBP which is outlined in the previous section.

$$P(\sigma^2_{\mathcal{G}} \,|\, \mathcal{X}, \mathbf{A}, \sigma^2_{\mathcal{X}}, \alpha) \propto P(\mathcal{X}|\sigma^2_{\mathcal{G}}, \mathbf{A}, \sigma^2_{\mathcal{X}})P(\sigma^2_{\mathcal{G}})$$

$$P(\sigma^2_{\mathcal{X}} \,|\, \mathcal{X}, \mathbf{A}, \sigma^2_{\mathcal{G}}, \alpha) \propto P(\mathcal{X}|\sigma^2_{\mathcal{G}}, \mathbf{A}, \sigma^2_{\mathcal{X}})P(\sigma^2_{\mathcal{X}})$$

$$P(\alpha \,|\, \mathcal{X}, \mathbf{A}, \sigma^2_{\mathcal{G}}, \sigma^2_{\mathcal{X}}) \propto P(\mathbf{A}|\alpha)P(\alpha)$$

This is the tensorized version of the linear-Gaussian binary latent feature model [15], and the exact derivation of the posterior estimates can be found in [18], where source code for a Gibbs sampler was used. Then, given that every parameter is sampled, we can retrieve estimates for the missing values $\mathcal{X}_{\mathcal{M}}$ as below. Note that only missing values are sampled, since sampling the full tensor may be more expensive and introduce additional noise.

$$\hat{\mathcal{X}}_{\mathcal{M}} \sim P(\mathcal{X}|\mathcal{G}, \mathbf{A}, \sigma^2_{\mathcal{X}}) = \mathcal{N}((\mathcal{G} \times_1 \mathbf{A})_{\mathcal{M}}, \sigma^2_{\mathcal{X}})$$

Where, $\mathcal{G}$ can be retrieved from the estimates of $\mathbf{A}, \sigma^2_{\mathcal{G}}, \sigma^2_{\mathcal{X}}$ as follows:

$$\mathcal{G} = \hat{\mathcal{X}} \times_1 (\mathbf{A}^T\mathbf{A} + \frac{\sigma^2_{\mathcal{G}}}{\sigma^2_{\mathcal{X}}}\mathbb{I})^{-1}\mathbf{A}^T$$

Algorithm 1 Input: Tensor, MissingValueIndexSet; Output: CompletedTensor

```
procedure IBPTUCKER
    burnIn ← 500
    numSamples ← 1000
    Gibbs.Chain = zeros(size(Tensor), numSamples-burnIn)
for i = 1:numSamples:
    Tensor.GibbsSample(MissingValueIndexSet)
    if i ≥ burnIn then
        Gibbs.Chain ← Tensor.newX.
    CompletedTensor = mean(Gibbs.Chain)
```

Then our evaluation criteria is relative error for prediction, where $\|\mathcal{T}\|_F = \sum_{\mathcal{I}} \mathcal{T}^2_{\mathcal{I}}$ is the Frobenius norm, *i.e.* the sum of squares of every element. For this measure, lower values of $\mathcal{E}$ are better.

$$\mathcal{E} = \frac{\|\mathcal{T}_{\mathcal{M}} - \hat{\mathcal{T}}_{\mathcal{M}}\|_F}{\|\mathcal{T}_{\mathcal{M}}\|_F}$$

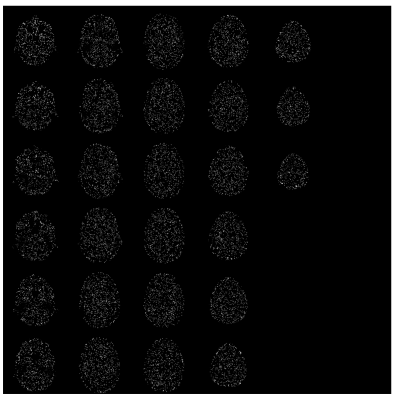
Returning to our simple example from section 4.3, where $\mathcal{X}$ is an order 3 tensor of dimensions $3 \times 3 \times 3$. Say one value, $\mathcal{X}_{222}$, is missing. Then $\mathcal{M} = \{(222)\}$ and we need to find a matrix $\mathbf{A}$ of dimension $3 \times R$ as well as $\mathcal{G}$ of dimension $R \times 3 \times 3$. All hyperparameters are initialized to 1 and $\mathbf{A}$ is generated from $IBP(\alpha) = IBP(1)$. Then, the estimate for R is the number of columns of $\mathbf{A}$. This process is repeated until the number of iterations is greater than the burn in time, then $\hat{\mathcal{X}}_{222} \sim \mathcal{N}((\mathcal{G} \times_1 \mathbf{A})_{222}, \sigma^2_{\mathcal{X}})$ is sampled and saved to $\hat{\mathcal{X}}_{\mathcal{M}}$. This is continued until convergence, where $mean(\hat{\mathcal{X}}_{\mathcal{M}})$ is returned as the estimate for the missing value $\mathcal{X}_{222}$.

	50% IBPT	50% PFA	90% IBPT	90% PFA
MRI	0.2560	0.1870	0.4600	0.3300
AA	0.0495	0.0214	0.1208	0.3910
FIA	0.1804	0.0083	0.3850	0.0360
KG	0.2208	0.1300	0.3717	0.3840

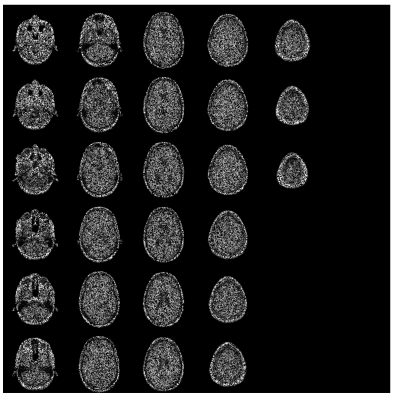
Table 1: Relative Errors for IBP-Tucker (IBPT) and Parallel Matrix Factorization (PFA)

5 Results

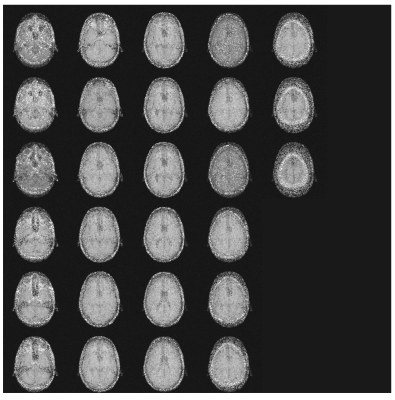
As an illustrative example, we use an MRI dataset found in MATLAB. The dataset provides a three dimensional image of the brain, with dimensions $127 \times 127 \times 27$. We proceed to randomly select 50% and 90% of the pixels and mark them as corrupted by substituting the value zero. As before, we let $\mathcal{M}$ be the index set of missing values. Referring to Figure 2, we see that the lower corruption rate of 50% results in an image with more detail. Referring to Table 2, we find some summary statistics and note that the standard errors for the estimates are quite small, and credible intervals for regions of interests would be precise.



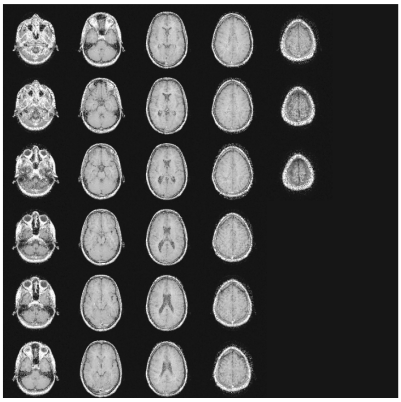
(a) 90% Missing Values



(b) 50% Missing Values



(c) 90% Missing Values, completed



(d) 50% Missing Values, completed

Figure 2: Horizontal Slices from MRI data with missing values

In addition to the MRI example, we also look at three other popular tensor datasets. First, we look at the Amino Acids (AA) dataset which has dimension $5 \times 201 \times 61$. The data measures the fluorescence by excitation and emission of 5 different laboratory-made samples of varying chemical

	50% $\sigma^2_{\mathcal{X}}$	90% $\sigma^2_{\mathcal{X}}$	50% $\sigma^2_{\mathcal{G}}$	90% $\sigma^2_{\mathcal{G}}$	50% R	90% R	50% $s.e.(\hat{\mathcal{X}}_{\mathcal{M}})$	90% $s.e.(\hat{\mathcal{X}}_{\mathcal{M}})$
MRI	0.03332	0.05149	0.07706	0.09134	37.605	29.000	0.04551	0.06200
AA	7.85217	17.9079	35.8488	40.0121	38.000	41.000	9.27719	22.6816
FIA	0.03156	0.04467	0.07167	0.08151	5.0000	3.4280	0.03912	0.05009
KG	0.02436	0.03098	0.06676	0.07716	12.000	6.9080	0.03685	0.04598

Table 2: Summary Statistics for IBPT Gibbs Sample, all values are mean estimates

concentrations. This data exhibits no problems and is used as a control. Second, we look at the Flow Injection Analysis (FIA) dataset which has dimension $12 \times 50 \times 45$. The data is similar to the Amino Acid data, but is known to have severe problems with local minima. Third, we look at the Kojima Girls (KG) dataset, which has dimension $153 \times 4 \times 20$. The data measures suffers from degeneracy in that there is no least squares solutions, and the covariates are highly correlated. Referring to Table 1 for relative errors, we see that our method performed better at the 90% level for the AA and KG dataset. However, when it came to the FIA dataset, our method performed very poorly. It seems that smaller datasets with local minima hamper the IBP-Tucker method in terms of relative error. Perhaps with more computing power, we could run the Gibbs sampler for longer and achieve better results.

Referring to Table 2 again, we find that the standard error for estimates ($s.e.(\hat{\mathcal{X}}_{\mathcal{M}})$) increases from 50% to 90%. This makes sense since there is less observed data, and therefore less information regarding latent structure. Moreover, the standard error for estimates is always higher than $\sigma^2_{\mathcal{X}}$. Intuitively, estimated missing values are predictions and standard statistical theory says that standard errors on predictions are higher than sample standard errors. Another interesting result is that there is no clear trend for the estimator tensor 1-rank (R). In general, it seems to decrease as there is more missing data, but this warrants further research.

6 Discussion

Throughout this paper, many simplifications were made for simplicity or ease of computation. These were not necessary, however they made the Gibbs sampler much faster. In future work, it would be interesting to explore sensitivity in the hyperprior specification. That is, if we change the hyperparameters or the hyperpriors over $\sigma^2_{\mathcal{G}}, \sigma^2_{\mathcal{X}}, \alpha$, are we able to estimate the missing values better? In addition, we could tailor distributional assumptions to a particular interesting dataset.

The biggest limitation of this paper is the Gibbs sampling procedure, which may become completely unfeasible for large datasets. Hence, another avenue for research lies in variational methods of inference, as opposed to Gibbs sampling. This would allow for an even more flexible model, with a more general Tucker factorization. Lastly, we may explore an application of our Bayesian nonparametric tensor factorization to compression. We can approximate elements of a known tensor since the factorization has less elements than the entire tensor. However, theoretical guarantees on error are needed, as well as an optimal compression rate.

In this paper, we proposed a Bayesian nonparametric method for tensor completion with automatic rank selection. Our method is a combination of two distinct works: one on bayesian nonparametric tensor completion and another on Bayesian automatic rank selection for tensor factorization. In particular, we use an Indian Buffet Process prior on the factor matrix of a Tucker-1 decomposition. The rank is then estimated by estimating the number of columns in the factor matrix. Although slow, we use Gibbs sampling to learn the latent structure of the tensor, which is encoded in the factorization. Specifically, we employ a collapsed Gibbs sampler that integrates out the core tensor in the Tucker-1 factorization.

Our empirical results show that there is some room for improvement. In particular, IBP-Tucker is beat consistently at the 50% missing data level. However, IBP-Tucker provides much more information about the dataset than point estimates. In particular, our method was able to estimate the tensor 1-rank as well as various measures of standard error. Therefore, despite suboptimal performance in some cases, IBP-Tucker should be used when there is a need for confidence or credibility.

References

[1] Hugo Peyre, Alain Leplège, and Joël Coste. Missing data methods for dealing with missing items in quality of life questionnaires. a comparison by simulation of personal mean score, full information maximum likelihood, multiple imputation, and hot deck techniques applied to the sf-36 in the french 2003 decennial health survey. *Quality of Life Research*, 20(2):287–300, 2011.

[2] Uwe Müller-Bühl, Bernhard Franke, Katja Hermann, and Peter Engeser. Lowering missing item values in quality-of-life questionnaires: an interventional study. *International journal of public health*, 56(1):63–69, 2011.

[3] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[4] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

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

[6] Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multi-modal factor analysis. 1970.

[7] Christopher J Hillar and Lek-Heng Lim. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):45, 2013.

[8] Yangyang Xu, Ruru Hao, Wotao Yin, and Zhixun Su. Parallel matrix factorization for low-rank tensor completion. *arXiv preprint arXiv:1312.1254*, 2013.

[9] Evrim Acar, Daniel M Dunlavy, Tamara G Kolda, and Morten Mørup. Scalable tensor factorizations for incomplete data. *Chemometrics and Intelligent Laboratory Systems*, 106(1):41–56, 2011.

[10] Xin Geng, Kate Smith-Miles, Zhi-Hua Zhou, and Liang Wang. Face image modeling by multilinear subspace analysis with missing values. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(3):881–892, 2011.

[11] Zenglin Xu, Feng Yan, and Yuan Qi. Bayesian nonparametric models for multiway data analysis. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):475–487, 2015.

[12] Qibin Zhao, Liqing Zhang, and Andrzej Cichocki. Bayesian cp factorization of incomplete tensors with automatic rank determination. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1751–1763, 2015.

[13] Shandian Zhe, Zenglin Xu, Xinqi Chu, Yuan (Alan) Qi, and Youngja Park. Scalable nonparametric multiway data analysis. In *AISTATS*, 2015.

[14] Wei Chu and Zoubin Ghahramani. Probabilistic models for incomplete multi-dimensional arrays.

[15] Thomas L Griffiths and Zoubin Ghahramani. The indian buffet process: An introduction and review. *Journal of Machine Learning Research*, 12(Apr):1185–1224, 2011.

[16] Samuel J Gershman and David M Blei. A tutorial on bayesian nonparametric models. *Journal of Mathematical Psychology*, 56(1):1–12, 2012.

[17] Peter Orbanz and Yee Whye Teh. Bayesian nonparametric models. In *Encyclopedia of Machine Learning*, pages 81–89. Springer, 2011.

[18] Finale Doshi-Velez and Zoubin Ghahramani. Accelerated sampling for the indian buffet process. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 273–280. ACM, 2009.