The probabilistic tensor decomposition toolbox

Hinrich, Jesper Love; Madsen, Kristoffer Hougaard; Mørup, Morten

Published in: Machine Learning: Science and Technology

Link to article, DOI: 10.1088/2632-2153/ab8241

Publication date: 2020

Document Version Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA): Hinrich, J. L., Madsen, K. H., & Mørup, M. (2020). The probabilistic tensor decomposition toolbox. Machine Learning: Science and Technology, 1(2), [025011]. https://doi.org/10.1088/2632-2153/ab8241
The probabilistic tensor decomposition toolbox

To cite this article: Jesper L Hinrich et al 2020 Mach. Learn.: Sci. Technol. 1 025011

View the article online for updates and enhancements.
The probabilistic tensor decomposition toolbox

Jesper L. Hinrich1,2, Kristoffer H Madsen1,2,3 and Morten Mørup1

1 Department of Applied Mathematics and Computer Science, Technical University of Denmark, Kongens Lyngby, Denmark
2 Danish Research Centre for Magnetic Resonance, Copenhagen University Hospital Hvidovre, Hvidovre, Denmark
3 Author to whom any correspondence should be addressed.

E-mail: jehi@dtu.dk

Keywords: tensor decomposition, Bayesian inference, multi-way modelling, candecomp/PARAFAC, Tucker, tensor train

Abstract

This article introduces the probabilistic tensor decomposition toolbox - a MATLAB toolbox for tensor decomposition using Variational Bayesian inference and Gibbs sampling. An introduction and overview of probabilistic tensor decomposition and its connection with classical tensor decomposition methods based on maximum likelihood is provided. We subsequently describe the probabilistic tensor decomposition toolbox which encompasses the Canonical Polyadic, Tucker, and Tensor Train decomposition models. Currently, unconstrained, non-negative, orthogonal, and sparse factors are supported. Bayesian inference forms a principled way of incorporating prior knowledge, prediction of held-out data, and estimating posterior probabilities. Furthermore, it facilitates automatic model order determination, automatic regularization on factors (e.g. sparsity), and inherently penalizes model complexity which is beneficial when inferring hierarchical models, such as heteroscedastic noise modelling. The toolbox allows researchers to easily apply Bayesian tensor decomposition methods without the need to derive or implement these methods themselves. Furthermore, it serves as a reference implementation for comparing existing and new tensor decomposition methods. The software is available from https://github.com/JesperLH/prob-tensor-toolbox/.

1. Introduction

Tensors, higher-order or n-way arrays are increasingly encountered in all areas of science. While standard two-way matrix analysis methods can be applied by restructuring these higher order arrays into a matrix, such approaches fail to properly exploit the inherent multi-way structure. Instead, using multi-way or tensor methods, such as the Canonical Polyadic (CP) decomposition (also known as PARAFAC and CandeComp) [1–3], Tucker model [4, 5] or variations thereof are able to account for the intrinsic structure of the data. Tensors are prevalent in both research and industry (for reviews see [6–9]) and has a large heterogeneous research community and several tensor analysis toolboxes are publicly available. The most prominent being the Matlab based N-way Toolbox [10], Tensorlab [11], and Tensor Toolbox [12] enabling researchers to apply multi-way modeling across domains.

However, these existing prominent multi-way toolboxes are based solely on maximum likelihood (ML) estimation which only provides a point estimate of the underlying parameters and does not account for parameter uncertainty. Instead estimating uncertainty requires repeated model fitting using for instance jackknifing [13] or bootstrapping [14] which are increasingly expensive as the size of the data grows.

In contrast, Bayesian inference results in an approximation of the true posterior distribution either via sampling [15–17] or a direct approximation [18–20]. Apart from uncertainty quantification, benefits of Bayesian inference includes automatic penalty for increased model complexity, inference on model order, determining sparsity level, incorporation of more realistic noise assumptions, and a principled way to include prior information and performing cross-validation [16, 21] on left out slices or blocks of data.

© 2020 The Author(s). Published by IOP Publishing Ltd
1.1. Bayesian tensor decomposition

For comprehensive reviews of maximum likelihood based tensor decomposition methods the reader is referred to existing tensor decomposition reviews [6–9, 22, 23]. We presently provide a short overview of Bayesian tensor decomposition. For brevity, this section will drop the prefix Bayesian and unless otherwise stated all models are based on fully Bayesian inference.

Currently, probabilistic models are primarily based on the Tucker and CP decomposition. The earliest works using Tucker decomposition are found in [24, 25] where both the core array and factors follow a normal distribution. The latter also considered enforcing sparsity on the core array to automatically learn the most prominent multi-linear interactions. However, neither of these methods were fully Bayesian as they relied on maximum-a-posteriori estimation which provides a point estimate of the posterior distribution. The first fully Bayesian Tucker model with normal factors using Gibbs sampling was proposed in [26] and the indeterminacy and structure of the core array was explored in [27]. The Tucker decomposition was extended to handle missing values and sparse noise based on variational Bayesian (VB) inference [28] whereas the Tucker model with an infinite core size was explored in [29–31] with factors specified as Gaussian processes and inference based on VB. Extensions of the Tucker model to count data was explored in [32, 33] using the Poisson likelihood and either Gamma or Dirichlet factors. For categorical data, a Tucker model with multinomial likelihood and factors was considered in [34]. The connection between contingency tables, log-linear models, and the Tucker decomposition was further explored in [35], who proposed a collapsed Tucker core for higher-order data.

As in ML estimation, the Bayesian CP model has received more attention than Bayesian Tucker model. The earliest work appears to be a VB based 3-way CP with normal factors with the assumption of independent [36] or dependent [37] components. Its N-way extension was later proposed in [38, 39]. For time series data, temporal lag-1 dependence in a mode was explored in [40]. Encouraging sparse factors was investigated by placing a sparsity prior on normal distributed factors [41, 42]. An orthogonal CP was proposed in [43], but it relied on MAP estimates of the von Mises-Fisher matrix distribution.

Modelling non-negative data has been explored with factors following a rectified normal [44], truncated normal or exponential [45], Poisson [46], or Dirichlet distribution [47] for continuous data. Decomposition of count or discrete data has been explored using either the Poisson or multinomial likelihood [48–50], with factors following a Dirichlet distribution [48] or Poisson distribution [49, 50]. Finally, several recent works exist on improved scaling to high-dimensional or streaming data [51–53].

Beyond the CP and Tucker model, tensor regression and classification has been investigated [34, 54, 55] and the connection between tensor regression and Gaussian processes was explored in [56]. Analysing multiple matrices with one varying mode is possible using the VB based PARAFAC2 model [57, 58] whereas analysing multiple 3-way tensors with one or two varying modes was proposed in [59, 60]. Finally, a VB Tensor Train model has been proposed in [61].

1.2. Summary of contribution

This paper introduces the probabilistic tensor decomposition toolbox which gathers many of the existing tools for Bayesian tensor decomposition in one place, see https://github.com/JesperLH/prob-tensor-toolbox/. The toolbox interconnects different constraints proposed in the literature, provides easy access to probabilistic decomposition for researchers, and serves as a reference implementation for comparing existing and future tensor decomposition models. In particular, the toolbox provides implementations of Bayesian CP, Tucker, and Tensor Train decomposition and supports model inference using either variational Bayesian inference, Gibbs sampling, or a combination of the two. At the time of writing, Bayesian CP is the most versatile model, as it allows modelling factors following a normal, truncated normal or exponential, uniform, or von Mises-Fisher matrix distribution. These distributions facilitate factors which are real, non-negative, box constrained, or orthogonal, respectively. Furthermore, column-wise and element-wise sparsity is modelled through hyper-prior distributions. The CP implementation can account for both homoscedastic as well as mode-specific heteroscedastic noise. Currently, inference in the presence of missing values is only available for the CP model using either marginalization or Bayesian imputation.

2. Methods

Scalars will be indicated by lowercase letters $x$, vectors by bold lowercase letters $\mathbf{x}$, matrices by bold upper case letters $\mathbf{X}$, and $N$th order ($N > 2$) tensors by bold calligraphic letters $\mathbf{X}$. Generally, roman letters will be used to represent data and random variables while Greek letters represent parameters of the probability distribution. A notable exception is $\theta$ which will be used to indicate both the set of all parameters (random variables and distributional parameters) and the set of parameters for a specific distribution, i.e. $P(\theta)$ for the
distribution of $\theta$. Furthermore, let $\times_n$ denote the $n$-mode matrix multiplication and $\otimes_n$, $\circ$, and $\odot$ respectively denote the Kronecker product, Khatri-Rao product, and element-wise product.

2.1. Bayesian tensor decomposition

Bayesian methods for tensor modelling presently includes the CP and Tucker model with factors (and core array) following different distributions, as presented in section 1.1. Additionally, the Tensor Train [61], PARAFAC2 [57], and multi-tensor factorization [59, 60] model were all recently developed using Bayesian inference.

Tucker decomposition is one of the core tensor models and is here used for illustrate of the differences between maximum likelihood (ML) and Bayesian estimation. Consider an $N^\text{th}$ order data tensor with $I_n$ observations in the $n^{\text{th}}$ mode, e.g. $\mathbf{X}^{(1)} \times_1 \mathbf{A}^{(2)} \times_3 \ldots \times_N \mathbf{A}^{(N)} + \mathcal{E}$, the Tucker model can be written as

$$\mathbf{X} = \mathbf{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \ldots \times_N \mathbf{A}^{(N)} + \mathcal{E},$$

where $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times D_n}, n = 1, 2, \ldots, N$ are the $N$ factor matrices, $\mathbf{G} \in \mathbb{R}^{D_1 \times D_2 \times \ldots \times D_N}$ is the core array which contains the coefficients for all multi-linear interactions of the columns of the factor matrices, and $\mathcal{E}$ is the noise. For brevity, let $\mathcal{M}$ denote the model or data reconstruction where

$$\mathcal{M} \equiv \mathbf{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \ldots \times_N \mathbf{A}^{(N)}.$$  

This provides the framing for stating the ML and Bayesian inference schemes below. Note, that the inference is stated as a least squares and Gaussian likelihood for ML and Bayesian inference, respectively. While other choices are possible this is the most common choice.

Maximum Likelihood

$$\arg\min_{\mathbf{G}, \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}} ||\mathbf{X} - \mathcal{M}||^2$$

s.t. constraint on $\mathbf{G}, \mathbf{A}^{(n)} \forall n \in \mathbb{N}$

Bayesian

$$P(\theta), \quad \theta = \{\mathbf{G}, \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}, \tau\}$$

$$\mathcal{L}(\mathbf{X} | \theta) = \mathcal{N}(\text{vec}(\mathcal{M}), \tau^{-1}I)$$

$$P(\theta | \mathbf{X}) = \frac{\mathcal{L}(\mathbf{X} | \theta)P(\theta)}{\int_\theta \mathcal{L}(\mathbf{X} | \theta)P(\theta)d\theta}$$

The ML approach, as mentioned, is widely used for existing tensor methods and can be solved using alternating optimization [2, 62], all-at-once optimization [63], or non-linear least squares [64, 65]. Constraints such as non-negativity can be imposed using either active set procedures [66], or alternating direction method of multipliers [67, 68]. Historically, multiplicative updates [69] have also been used.

For Bayesian inference, the first step is to specify the prior distributions, $P(\theta)$, which describe the a priori knowledge about the parameters. The next step is to specify the likelihood function, $\mathcal{L}(\mathbf{X} | \theta)$, which is a probability distribution specifying how the data is generated based on the reconstructed array by the imposed model $\mathcal{M}$ and assumptions regarding noise. The posterior distribution, $P(\theta | \mathbf{X})$, is then calculated via Bayes rule as specified in equation (4).

The key challenge in Bayesian inference is that calculating the posterior is generally intractable, as calculating the marginal distribution or evidence, $P(\mathbf{X}) = \int_\theta \mathcal{L}(\mathbf{X} | \theta)P(\theta)d\theta$, is intractable. Therefore, most applications of Bayesian inference turn to approximate methods, which are primarily; 1) Markov Chain Monte Carlo (MCMC) sampling and its variants [15, 17, 70]. 2) Laplace approximation and its variants [20, 71]. 3) Variational approximation [18, 21].

This article and the presented toolbox focuses on variational Bayesian (VB) inference and MCMC Gibbs sampling due to their interconnection, ease of use, and wide application within the existing literature on Bayesian tensor modeling.

2.2. Gibbs sampling and VB approximation

Given the prior distribution, $P(\theta)$, and the likelihood, $\mathcal{L}(\mathbf{X} | \theta)$, the posterior distribution, $P(\theta | \mathbf{X})$ is given by Bayes rule,

$$P(\theta | \mathbf{X}) = \frac{\mathcal{L}(\mathbf{X} | \theta)P(\theta)}{\int_\theta \mathcal{L}(\mathbf{X} | \theta)P(\theta)d\theta} = \frac{\mathcal{L}(\mathbf{X} | \theta)P(\theta)}{P(\mathbf{X})}.$$  

The posterior is generally intractable as the marginal likelihood or evidence is intractable, $P(\mathbf{X})$. Gibbs sampling is useful when it is not possible or too expensive to directly sample from the joint posterior,
\( P(\theta|\mathbf{X}) \), but the conditional distribution \( P(\theta|\theta_{\text{adj}},\mathbf{X}) \) has a closed form or is easy to sample from. Gibbs sampling then draws a sample from the conditional distribution,

\[
\theta_i \sim P(\theta|\theta_{\text{adj}},\mathbf{X}),
\]

where \( \theta_i \) is the set of random variable which are sampled and \( \theta_{\text{adj}} \) are the random variables which are conditioned upon. A full sample \( \theta \) is obtained by cycling through all conditional distributions. When the sampler has converged, then these samples approximate the true posterior distribution [16].

For variational Bayesian inference, the main idea is to approximate the posterior, \( P(\theta|\mathbf{X}) \), using a set of variational distributions, \( Q(\theta) \). The Q distributions provides a lower-bound on the log evidence \( \log P(\mathbf{X}) \) which is called the evidence lower bound (ELBO). Using Jensen’s inequality, the ELBO is,

\[
\log P(\mathbf{X}) = \log \int_{\theta} P(\mathbf{X}, \theta) \frac{Q(\theta)}{Q(\theta)} d\theta \geq \int_{\theta} Q(\theta) \log \frac{P(\mathbf{X}, \theta)}{Q(\theta)} d\theta = \text{ELBO} (Q).
\]

The variational distributions are chosen such that the ELBO is tractable [18, 21]. This paper uses a mean-field approximation where the probability distributions are assumed to be independent between subsets of random variables, i.e. \( Q(\theta) = \prod_i Q(\theta_i) \). The optimal variational distribution for each subset \( \theta_i \) is then,

\[
Q(\theta) \propto \exp \{ \langle \ln P(\mathbf{X}, \theta) \rangle \setminus Q(\theta_i) \}
\]

where \( \langle \cdot \rangle \setminus Q(\theta) \) is the expected value with respect to all variational distributions except \( Q(\theta_i) \), see [72]. The optimal parameters of the variational distribution are then identified via moment matching in equation (8).

### 2.3. Probabilistic tensor decomposition toolbox

The probabilistic tensor decomposition toolbox is available at https://github.com/JesperLH/prob-tensor-toolbox/. Presently, it only considers the Gaussian likelihood which is analogous to the least squares error. Given a data tensor \( \mathbf{X}^{h \times I_2 \times \cdots \times I_N} \), the likelihood is,

\[
\mathcal{L}(\mathbf{X}|\theta) = \mathcal{N}_d (\mathbf{X}|\mathcal{M}, \Sigma^{(1)}_\epsilon, \Sigma^{(2)}_\epsilon, \ldots, \Sigma^{(N)}_\epsilon)
\]

where \( \mathcal{M} \) is the mean array and \( m \equiv \text{vec}(\mathcal{M}) \) its vectorization, \( \mathcal{N}_d \) and \( \mathcal{N} \) is the array and multivariate normal distribution, and \( \Sigma^\epsilon \equiv \Sigma^{(N)}_\epsilon \otimes \Sigma^{(N-1)}_\epsilon \otimes \cdots \otimes \Sigma^{(1)}_\epsilon \) is the residual error following a Kronecker structured covariance matrix [26, 27, 73].

A simpler heteroscedastic noise model where the mode specific covariance is diagonal, \( \Sigma^{(n)}_\epsilon \equiv \text{diag} (\sigma^{(n)}_\epsilon^2) \), is implemented for Bayesian CP. This noise structure allows quantifying mode-specific noise, such as noisy samples or unreliable features. For the Tucker and the Tensor Train decomposition only homoscedastic noise is considered, which is the same as specifying \( \Sigma^{(n)}_\epsilon \equiv \sigma^2 I \) and \( \Sigma^{(m)}_\epsilon \equiv I \forall m \neq n \) for an arbitrarily chosen \( n \).

The CP, Tucker and Tensor Train (TT) models are all specified by the structure of the mean array \( \mathcal{M} \). For the Tucker decomposition the mean array is defined as in equation (2). Similarly, CP is described by equation (2) but constraining the core array to the unit hyper-cube \( \mathcal{G} = \mathcal{I} \). The unit hyper-cube has value one along the hyper-diagonal and off-diagonal elements are zero. For tensor train decomposition, the mean array is,

\[
\mathcal{M} = \mathcal{U}^{(1)} \times [2,1] \mathcal{U}^{(2)} \times [3,1] \mathcal{U}^{(3)} \times [4,1] \cdots \times [N-1,1] \mathcal{U}^{(N)}
\]

where \( \times [a,b] \) is tensor contraction along mode \( a \) and \( b \) and \( \mathcal{U}^{(n)} \in \mathbb{R}^{D_{n-1} \times I_n \times D_n} \) is the latent factor or train cart for mode \( n = 1, \ldots, N \). Details on tensor train and its probabilistic extension are found in [61, 74].

Handling mode-specific heteroscedastic noise is feasible both for the Tucker decomposition and the TT decomposition. However, when imposing orthogonality constraints on a latent factor heteroscedastic noise is not permitted as the prior on the factor (von Mises-Fisher Matrix distributed) and likelihood (heteroscedastic Gaussian distributed) are not conjugate, i.e. the posterior distribution of each factor no longer follows a von Mises-Fisher Matrix distribution.
2.3.1. Partially observed data

For partially observed data, the missing elements can either be imputed or marginalized. For the latter, the likelihood is specified only on the observed elements of the tensor,

$$\mathcal{L}(\mathbf{X}|\theta) = \prod_{i \in \mathcal{O}} \mathcal{N}(x_i|m_i, \Sigma_i)$$

(12)

where \(\mathcal{O}\) is the set of observed elements and \(i = (i_1, i_2, \ldots, i_N)\) is the vector index of a single element. Marginalization is preferred over imputation as it does not introduce any bias in the estimated posterior distribution and preserves any convergence guarantees of the chosen inference method. Imputing missing values is generally faster and works well in many practical applications. Unfortunately, imputation can fail unexpectedly as the number of missing values increase as the inference is highly influenced by past (poor) imputations.

For Bayesian inference, imputation is more principled than its maximum likelihood counterpart [16]. In Bayesian inference, the likelihood is just the probability of data \(\mathbf{X}\) under some model \(\theta\), i.e.

$$\mathcal{L}(\mathbf{X}|\theta) \equiv P(\mathbf{X}|\theta).$$

In the presence of missing data, this can be rewritten as \(P(\mathbf{X}_{\text{obs}}, \mathbf{X}_{\text{miss}}|\theta)\) and the probability of a missing element \(x_i\) is then,

$$P(x_i|\mathbf{X}_{\text{obs}}) = \int_{\theta} P(x_i|\theta)P(\theta|\mathbf{X}_{\text{obs}}) \, d\theta, \quad \forall i \notin \mathcal{O}$$

(13)

Determining this distribution is intractable, but it can be approximated using VB [21, 28] or sampling [16]. The toolbox currently use a coarse approximation and assumes \(P(x_i|\mathbf{X}_{\text{obs}}) = g(\mathcal{M})_i\) for all missing elements \(i \notin \mathcal{O}\). For VB, \(g(\mathcal{M}) = \langle \mathcal{M} \rangle\) is the reconstructed element using the current expected first moments. For sampling, \(g(\mathcal{M}) = \mathcal{M}\) which is the reconstruction using the most recent sample of the random variables. Due to this coarse approximation, imputation should always be followed by at least one marginalization update to get valid posterior moments.

2.3.2. Factor matrices

The Bayesian Tensor Train model only supports orthogonal train carts and details are provided in the original reference [61]. Therefore, this section only concerns the specification of factor matrices for Bayesian CP and Tucker decomposition, i.e. \(\mathbf{A}^{(n)}\) for mode \(n\) as defined in equation (2). Presently, the following priors are implemented in the toolbox,

$$P\left(\mathbf{A}^{(n)}|\mu^{(n)}, \Lambda^{(n)}\right) = \prod_{i=1}^{I_n} \mathcal{N}\left(a_{id}^{(n)}|\mu_{id}^{(n)}, \Lambda_{id}^{(n)-1}\right)$$

(14)

$$P\left(\mathbf{A}^{(n)}|\mu^{(n)}, \Lambda^{(n)}\right) = \prod_{i=1}^{I_n} \prod_{d=1}^{D_n} \mathcal{N}_{[0,\infty]}\left(\epsilon_{id}^{(n)}|\mu_{id}^{(n)}, \lambda_{id}^{(n)-1}\right)$$

(15)

$$P\left(\mathbf{A}^{(n)}|\Lambda^{(n)}\right) = \prod_{i=1}^{I_n} \prod_{d=1}^{D_n} \text{Exponential}\left(a_{id}^{(n)}|\lambda_{id}^{(n)}\right)$$

(16)

$$P\left(\mathbf{A}^{(n)}|l, u\right) = \prod_{i=1}^{I_n} \prod_{d=1}^{D_n} \text{Uniform}\left(a_{id}^{(n)}|l, u\right)$$

(17)

$$P\left(\mathbf{A}^{(n)}|\mathbf{K}\right) = \nu \mathcal{MF}\left(\mathbf{A}^{(n)}|\mathbf{K}\right), \quad \mathbf{K} \equiv 0$$

(18)

where each \(n\)-mode factor matrix \(\mathbf{A}^{(n)}\) has observations \(i = 1, 2, \ldots, I_n\) and latent components \(d = 1, 2, \ldots, D_n\). Presently, both Tucker and CP support von Mises-Fisher Matrix (\(\nu \mathcal{MF}\)) distributed factor matrices which represent an orthogonal assumption on the factor matrix. For CP, the multivariate normal distribution is used to model unconstrained factors, while the truncated normal and exponential distribution are used to model non-negativity. Furthermore, the uniform distribution can be applied to achieve an uninformative box constraint.
2.3.3. The core array

For the Tucker decomposition an explicit core array \( \mathcal{G}^{D_1 \times D_2 \times \cdots \times D_N} \) is inferred which models multi-linear interactions between the components. Here, the core array is specified using its vectorized version \( \text{vec}(\mathcal{G}) \) and follows a normal distribution,

\[
P(\text{vec}(\mathcal{G})) = \mathcal{N}(0, \Psi^{-1}),
\]

where \( \Psi \) is the precision matrix and the precision of a single element in \( \mathcal{G} \) is \( \psi_{d_1, d_2, \ldots, d_N} \).

The toolbox considers learning a shared scale equation \( \psi \equiv \mathbf{I} \), an automatic relevance determination (ARD) prior for slice activation \( \text{diag}(\Psi) \equiv \psi^{(N)}(\mathbf{1}) \cdots \psi^{(1)}(\mathbf{1}) \) [28], and element-wise sparsity \( \text{diag}(\Psi) \equiv \psi \) [25]. For these priors off-diagonal elements of \( \Psi \) are zero and the prior on diagonal elements is assumed independent and from a Gamma distribution, \( P(\psi) \sim \mathcal{G}(\alpha_\psi, \beta_\psi) \) with shape \( \alpha_\psi \) and rate \( \beta_\psi \).

In the context of Bayesian analysis, a Kronecker structured precision matrix \( \Psi \equiv \psi^{(N)}(\mathbf{1}) \cdots \psi^{(1)}(\mathbf{1}) \) was explored by [27] and for count data a core array of CP structured sub-cores [33]. The latter is related to the block term decomposition [75] and can be viewed as a special case for count data using Bayesian inference.

2.3.4. Component and Core Precision Matrix

The probabilistic toolbox allows specifying a prior \( P(A^{(n)}) \) on the precision of each observation \( i \) and latent components \( D_n \) for factor matrix \( A^{(n)} \). Similarly, for each element of the core array \( g_{d_1, d_2, \ldots, d_N} \) it is possible to specify its precision \( P(\psi_{d_1, d_2, \ldots, d_N}) \). At present time, only precisions following a Gamma distribution are considered. For the core array, the priors are

\[
P(\psi) = \mathcal{G}(\psi|\alpha_\psi, \beta_\psi), \quad P(\psi^{(n)}) = \prod_{d=1}^{D_n} \mathcal{G}(\psi_{d}^{(n)}|\alpha_\psi, \beta_\psi)
\]

for learning a shared scale \( \psi \equiv \mathbf{I} \), element-wise sparsity \( \Psi \equiv \text{diag}(\psi) \), and prune irrelevant slices \( \Psi \equiv \text{diag}(\psi^{(1)}, \ldots, \psi^{(N)}) \), respectively. Note \( \alpha_\psi \) and \( \beta_\psi \) is the rate and shape of the Gamma distributions which are specified as broad priors.

For the latent factors, a full precision matrix for each observation \( i \) and mode \( n \) is rarely needed. Instead, it is modelled as

\[
A^{(n)}_i \equiv \mathbf{I}, \quad A^{(n)}_i \equiv \text{diag}(\lambda), \quad A^{(n)}_i \equiv \text{diag}(\Lambda^{(n)}), \quad A^{(n)}_i \equiv \text{diag}(\Lambda^{(n)}), \quad \text{or} \quad A^{(n)}_i \equiv \Lambda^{(n)}.
\]

These specifications model a shared scale, ARD shared over modes, mode-specific ARD, mode specific sparsity, and a mode specific interaction between the latent components. The latter \( P(\Lambda^{(n)}|\cdot) \) follow a Wishart distribution while the elements in each of the former specifications follow a Gamma distribution.

The Gamma distribution is used as it is the conjugate prior for the precision of a normal or truncated normal distribution, as well as for the rate of an exponential distribution. The Wishart prior is the conjugate prior of the precision matrix from a normal distribution and is only applicable to normal factor priors equation (14).

Gamma based ARD priors work well in many applications [25, 72, 76], but they do not necessarily provide the desired shrinkage [77, 78].

2.3.5. Inferring the Posterior Distribution

For a given decomposition model, its parameters or all the random variables of interest are denoted as \( \theta \). As an example, consider Bayesian CP with three factors \( A^{(1)}, A^{(2)}, A^{(3)} \), shared component precision \( \lambda \), and homoscedastic noise \( \tau \) for which \( \theta = \{A^{(1)}, A^{(2)}, A^{(3)}, \lambda, \tau\} \) is the set of random variables.

The probabilistic tensor decomposition toolbox seeks to characterize these random variables via their posterior distribution \( P(\theta|\mathcal{X}) \). As discussed in section 2.2, exact inference is not feasible and the posterior is approximated using either variational Bayesian (VB) inference or Gibbs sampling, see also [16, 18] for details.
2.4. Prediction on Held-out Slices or Samples

Bayesian inference facilitates a principled way of predicting on held-out slices or samples from a tensor $\mathbf{X}$. This allows splitting the tensor into a disjoint training and test set, $\mathbf{X} = [\mathbf{X}_{\text{train}}, \mathbf{X}_{\text{test}}]$. This approach facilitate greater independence between training and test set when compared with leaving out elements or fibers of the tensor. Test set performance is assessed using the predictive posterior distribution \[ \log P(\mathbf{X}_{\text{test}}|\mathbf{X}_{\text{train}}) = \log \int_\theta \int_{\theta^*} P(\mathbf{X}_{\text{test}}, \theta, \theta^*|\mathbf{X}_{\text{train}}) d\theta d\theta^*. \quad (23) \]

Here $\theta$ and $\theta^*$ are the random variables for the training and test set, respectively. Generally, this integral is intractable and the probabilistic tensor decomposition toolbox considers two approximation approaches inspired by [21].

The first approach relies on the assumption that the estimated variational posterior distribution is a good approximation of the true posterior distribution, i.e. $P(\theta|\mathbf{X}_{\text{train}}) \approx Q(\theta|\mathbf{X}_{\text{train}})$. This provides the following lower-bound,

\[ \log P(\mathbf{X}_{\text{test}}|\mathbf{X}_{\text{train}}) \approx \log \int_\theta \int_{\theta^*} P(\mathbf{X}_{\text{test}}|\theta, \theta^*) Q(\theta|\mathbf{X}_{\text{train}}) P(\theta^*) d\theta d\theta^* \]

\[ \geq \int_\theta \int_{\theta^*} Q(\theta^*) Q(\theta|\mathbf{X}_{\text{train}}) \log \left( \frac{P(\mathbf{X}_{\text{test}}|\theta, \theta^*)}{Q(\theta^*)} \right) d\theta d\theta^* \quad (24) \]

where $Q(\theta^*)$ is a variational approximation of the test set parameters.

The second approach assumes a point estimate of the training parameters $\theta$ is a good approximation of the posterior distribution, e.g. $P(\theta|\mathbf{X}_{\text{train}}) \approx \langle \theta \rangle$ where the mean value of $\theta$ is used. This approximation then only requires evaluating the integral over test set parameters $\theta^*$,

\[ P(\mathbf{X}_{\text{test}}|\mathbf{X}_{\text{train}}) \approx \int_{\theta^*} P(\mathbf{X}_{\text{test}}|\theta^*, \langle \theta \rangle) P(\theta^*) d\theta^* \quad (25) \]

where $P(\mathbf{X}_{\text{test}}|\cdot)$ is the likelihood of the test data and $P(\theta^*)$ the prior on the test parameters. This integral is sometimes tractable, but otherwise it is approximated via variational inference or sampling.

3. Results and discussion

This section contains a few experiments to illustrate the use of the probabilistic tensor decomposition toolbox. These experiments demonstrate the benefits of probabilistic modelling, but also serve to highlight potential limitations.

These illustrations are primarily based on two common fluorescence spectroscopy datasets. The Amino Acid dataset [79–81] contains five samples where fluorescence was measured as different emission levels (250–450 mm) and excitation levels (250–300 mm). The data is represented in the tensor $\mathbf{X}^{5 \times 201 \times 61}$ where modes are samples, emission levels, and excitation levels. The samples are mixtures of three pure components and the concentrations are known.

The Sugar Process dataset [82, 83] consists of 268 equally spaced samples and for each sample fluorescence was measured at 571 emission levels and 7 excitation levels, the data is represented in the tensor $\mathbf{X}^{268 \times 571 \times 7}$. The samples were collected every 8 hours from a sugar plant and the dataset contains chemical shifts and is more noisy than the Amino Acid dataset.

For illustrating tensor completion, section 3.3, two additional datasets are considered. The Wakeman dataset [84, 85] which contains electroencephalogram (EEG) data from a visual experiment. Before analysis, the data was preprocessed as described in Chapter 42.3 of the SPM12 Manual (https://www.fil.ion.ucl.ac.uk/spm/) except for the addition of an initial filtering step for removing high frequency noise. This step used a band pass filter (FIR) [1,35]Hz with filter order 1815 and applied it to each channel and in both directions. This is further detailed in the script acq_wakeman_eeg_data.m. In this paper, only data from subject 7 and condition Famous is used. The data is represented as a tensor $\mathbf{X}^{701 \times 181 \times 296}$ where the modes are channels, time points, and trials, respectively.

The Human Allen Brain dataset [86] consists of gene expression data from different brain tissues in six subjects, represented by $\mathbf{X}^{26892 \times 414 \times 6}$ where the modes are genes, areas, and subjects, respectively. The data is available from http://human.brain-map.org/. For some brain areas, multiple tissue samples where gathered, but the present analysis ignores this aspect and averages over all samples within an area and subject. Previously, this preprocessing scheme was applied when analysing the data using a non-negative CP model.
Figure 1. Component Extraction: Estimation of Bayesian CP with either $D_{\text{init}} = 10$ or $D_{\text{init}} = 100$ initial components. For each scenario, the mode specific (samples, emission, and excitation) loadings are shown. The components are sorted by their estimated standard deviation ($\langle \lambda_d \rangle^{-1/2}$) which is to the right.

Based on either maximum likelihood [62] or Bayesian inference [45]. Further details are found in the script acq_allen_geneexpr_data.m. The toolbox is made to be extensible and future changes might affect how the developed functions are called. Therefore, specific function calls are not included. Instead, the name of the relevant demonstration file is given, i.e. demo_*.m.

3.1. Component extraction: amino acid and sugar process dataset

Tensor decomposition methods seek to factorize data, such that the extracted components represent the underlying structure of the data. Commonly, the components of this underlying structure are of interest as they provide a more compact and possibly interpretable representation of the data. The toolbox, presently, includes Bayesian Canonical Polyadic/PARAFAC (CP), Tucker, and Tensor Train (TT) decomposition. Calling these methods are demonstrated in the script demo_component_extraction.m which also visualizes the results obtained by each method.

This section only illustrates the CP decomposition, as neither Tucker nor TT decomposition have simple intuitive visualizations. The CP decomposition is applied to the Amino Acid and Sugar Process datasets, as described in demo_fluorescence.m which also generates figures 1(a)–(d).

To investigate if irrelevant components are pruned appropriately, the model is initialized with either 10 or 100 components, i.e. $D_{\text{init}} = 10$ or $D_{\text{init}} = 100$. The maximum number of iterations was 100 and 500 for the Amino Acid and Sugar Process datasets, respectively. Results are shown in figure 1.

For the Amino Acid dataset, the model correctly estimates the latent subspace and correctly identifies the components and concentrations of each sample, see figures 1(a) and (b). Note, that the ARD for $D_{\text{init}} = 10$
estimates 4 components, but the fourth component is a replicate of the yellow component and has near zero loadings in the sample mode.

For the Sugar Process dataset, seven components are estimated for \( D_{\text{init}} = 10 \). However, this is not the true chemical spectra, as samples contain chemical shifts that violate the tri-linear assumption of the CP model. Therefore, additional components are necessary to adequately represent the data. The effect of this is most visible in the emission mode, as seen in both figures 1(c) and (d).

Figure 1(d), the initial number of components are increased \( D_{\text{init}} = 100 \) and while most are pruned, it does not estimate the same number of components. This illustrates, that if the initial number of components are greatly over-specified, then the correct subspace might not be identified and it can be beneficial to refit the model with a lower number of initial components.

Generally, determining the true number of components for any dataset remains difficult, but the Bayesian-CP model has proven to provide more robust estimation when the number of components are over-specified \([39, 45]\). This added robustness of using a Bayesian formulation has also been observed for other tensor decomposition methods, for instance \([27, 28, 57, 59–61]\).

3.1.1. Heteroscedastic noise estimation

Modelling mode-specific heteroscedastic noise allows quantification of data quality of the samples, as well as investigation of which part of the spectra the model has an uncertain representation of. This example shows an analysis of the Amino Acid data using a Bayesian CP with truncated normal factors, column-wise precision (ARD) prior on the sample mode, and heteroscedastic noise on all modes. The set of random variables is now \( \theta = \{ A^{(1)}, A^{(2)}, A^{(3)}, \lambda, \tau^{(1)}, \tau^{(2)}, \tau^{(3)} \} \). The model is initialized with 10 components and results are shown in figure 2. The analysis is included in script demo_tb_heteroscedasticCP.m.

The estimated noise, presently, suffers from scale indeterminacy as \( \tau^{(1)} \odot \tau^{(2)} = (\tau^{(1)} \alpha^{-1}) \odot (\tau^{(2)} \alpha) \) for an arbitrary \( \alpha \in \mathbb{R}_{\geq 0} \). Therefore, the estimates are interpretable within each mode but not in absolute magnitude across modes. For comparison across modes, the estimate has to be invariant to scale changes, c.f. \([27]\). The heteroscedastic noise estimation is useful for down-weighting specific samples, emission levels, or excitation levels. However, it does not enforce a fixed threshold where an observation \( i_n \) for some mode \( n \) should be left out of the estimation.

3.2. Data denoising

Tensor decomposition methods can be used for data denoising where the goal is to remove unstructured and/or structured noise from the data. This is illustrated on the Amino Acid dataset using either a CP, Tucker or Tensor Train decomposition. This reproduces an experiment from the Bayesian TT article \([61]\), but presently also including the Bayesian Tucker decomposition with orthogonal factors.

Similar to \([61]\), the number of components are set to \( D = 3 \) in the CP model and \( D = (1, 6, 5, 1) \) in the TT model. For the Tucker decomposition, the core size is determined by exhaustive evaluation using maximally \( D_1 = 5, D_2 = D_3 = 10 \) resulting in \( 10^5 \) models for which the model achieving the highest evidence lowerbound had core size \( D = [5, 9, 9] \). Each of the decomposition is fit until convergence or 100 iterations. The toolbox includes the script demo_denoising.m which reproduces this analysis.

The original data, reconstructed data, and the residual error for each sample using each decomposition are shown in figure 3. The differences between the decomposition methods are most apparent for sample 3 and 5 where Rayleigh scattering (structured noise) is also most pronounced.
Figure 3. Data Denoising: For the Amino Acid dataset, the raw data from each sample (top row) and the reconstructed sample based on either CP, Tucker, and TT decomposition are shown. The residual error under each decomposition is also shown.

The CP model is the closest approximation to the ground truth model [79, 81] and provides the best denoised reconstruction. Structured noise is retained in both the Tucker and TT decomposition which is attributed to these models being more flexible than the CP model.

3.3. Tensor completion

The last example is missing value prediction or data completion for which a variety of tensor methods has been proposed, for a review see [87]. Presently, the developed toolbox only allows missing values for CP decomposition which limits this demonstration to the Bayesian CP model. For the Bayesian Tucker model, marginalizing over missing values was presented in [28].

The CP model is applied to the Sugar Process data, subject 7 from the Wakeman EEG data, and the Allen Human Brain data. Missing values are generated by holding out a subset of the observed data, the remaining data is used for training. Model performance is assessed on the held out data, i.e. test data, using the root mean square error (RMSE). Two missing value scenarios are displayed, first elements missing at random and then entire fibers missing. The missing fibers are emission spectras (mode-2 fibers), trials (mode-1 fibers), and samples (mode-1 fibers) for the Sugar Process, Wakeman, and Allen Brain data, respectively. In contrast to the Sugar Process and Wakeman data, the Allen Brain data is not fully observed and has 43.44% missing values due to missing samples (mode-1 fibers).

For each dataset, the true number of components is unknown and a CP with normal factors and ARD is fit using $D = 10$ initial components. For both VB and Gibbs samplings 250 iterations are run. To reconstruct the data, VB uses the expected value in the final iteration whereas for Gibbs sampling the average reconstruction error of the last 125 samples is used. To mitigate the effect of local minima, each model is fit 10 times and the median error is displayed. The analysis is reproduced by the script demo_tensor_completion.m and results are given in figure 4.

When elements are missing at random, figure 4(a)-(c), the results are similar and performance is mostly unchanged from the initial 10% missing values until around 80% missing values. The test set performance then degrades for both VB and Gibbs sampling.

When entire fibers are missing at random, figure 4(d)-(f), the performance visibly differ between the datasets. Figure 4(d) and (f) show performance degrades earlier when entire fibers are missing, both for VB and Gibbs sampling. The transition for the Allen data is less abrupt than in the Sugar Process data. This is attributed to missing scheme matching the actual reason for missing data in the Allen data. In contrast, for the Sugar Process data, leaving emission fibers out rapidly removes all essential information and does not reflect a valid missing data mechanism.

For the EEG data, the missing fibers (trials) have little effect on the predictive performance. The reason for this is that the time-lock signal is never fully removed due to event related EEG data being highly correlated across channels and trials.

Consider leaving out all timepoints for channel $i$ and trial $j$, i.e. a mode-2 fiber $X_{i,j}$, then the left out channel is highly correlated with all other channels during trial $j$ due to spatial correlation of the signal.
Figure 4. Tensor Completion: Data entries are missing at random (top row) or fibers are missing at random (bottom row). The percentage of missing values is varied and for each setting the root mean square error (RMSE) is given for the training and test set. The training-test split was generated such that there where no missing slices in the training data.

Furthermore, the time locked nature of the experiment ensures that even if the maximum possible fibers are removed, such that each trial contains only one observed channel, the magnitude of the underlying signal at each time point can still be recovered as each channel has on average 4.2 observed trials. This demonstration illustrates how the amount of missing data and its structure affects tensor completion. Furthermore, it highlights the importance selecting a test set which reflect the actual missing data mechanism. This was illustrated on the Allen data, were leaving out individual elements leads to over-confidence in the results.

4. Conclusion

In this paper, we introduced the probabilistic tensor decomposition toolbox which allows for Bayesian inference in the Canonical Polyadic/PARAFAC (CP), Tucker, and Tensor Train decomposition models. The toolbox offers flexibility in the choice of priors/constraints on factors, both homo- and heteroscedastic noise modelling and efficient inference via the variational approximation. The results indicate that Bayesian inference successfully prunes factors facing over-specified models. However, in practice violations of the model structure still leads to ambiguity with respect to the number of factors to be included. The results show that heteroscedastic noise modelling is useful for quantification of noise and provides additional useful information for model interpretation, which can help researchers identify noisy samples and indicate potential issues with the data acquisition process. In conclusion, the proposed Bayesian toolbox for probabilistic tensor decomposition naturally extends existing methods based on maximum likelihood estimation, and offers more comprehensive modelling accounting for uncertainty readily applicable in the many disparate fields in which tensor decomposition is already widely applied.

5. Data availability

Data sharing is not applicable to this article as no new data were created or analysed in this study. The analyzed data are available online.

The Amino Acid dataset [79–81] and Sugar Process dataset [82, 83] are available at http://www.models.life.ku.dk/datasets. The Human Allen Brain dataset [86] is available at http://human.brain-map.org/static/download. The Wakeman dataset [84, 85] is available at ftp://ftp.mrc-cbu.cam.ac.uk/personal/rik.henson/wakemandg_hensonrn and details on acquiring and processing it are given in Chapter 42 of the SPM12 Manual (https://www.fil.ion.ucl.ac.uk/spm/).

The scripts for reproducing the analysis are provided as part of the Probabilistic Tensor Toolbox, see https://github.com/JesperLH/prob-tensor-toolbox/.
6. VB and Gibbs based inference

This appendix describes how to infer each of the random variables using either Gibbs sampling [15, 16] or variational Bayesian inference [18, 21]. The former relies on determining the conditional distributions and the latter on finding a set of distribution that approximate the true posterior distribution. These two inference methods result in similar updates schemes. To avoid duplicating the appendix, the notation is abused and \( \langle \cdot \rangle \) denote either a sample form the relevant conditional distribution (when using Gibbs sampling) or the expected value under the Q-distribution (when using VB).

Updating of the CP factor matrices under homoscedastic noise, heteroscedastic noise, and in the presence of missing values is presented in section 6.1. Section 6.2 concerns inferring the precision prior on factor matrices under the different prior choices. Section 6.3 concerns inferring Tucker factor matrices while the core array and precision prior are presented in section 6.4. Finally, section 6.5 gives updates for inferring homoscedastic or mode-specific heteroscedastic noise precision.

Inference of the Probabilistic Tensor Train model are stated in [61]. These are not restated as this paper does not expand on the model.

Most of the update rules presented in this appendix are not novel, but restating them in a common notation provides an overview and more easy comparison of their differences and similarities.

To monitor the progress of the inference procedure, variational inferences uses the evidence lowerbound \( \text{ELBO} \), which increases monotonically. The expressions are calculated as,

\[
\text{ELBO}(\mathbf{X}, \mathbf{\theta}) = \langle \log P(\mathbf{X}(\mathbf{\theta})) \rangle_Q + \langle \log P(\mathbf{\theta}) \rangle_Q - \langle \log Q(\mathbf{\theta}) \rangle_Q
\]

where \( \langle \cdot \rangle_Q \) is expectation under the Q-distribution for variational inference. For sampling \( \mathbf{\theta}^t \) is the sample of random variables at iteration \( t \).

Notation
In addition to the previous notation, the sequential product of \( N \) matrices is defined as,

\[
\bigotimes_{n=1}^{N} \mathbf{A}^{(n)} = \mathbf{A}^{(N)} \otimes \mathbf{A}^{(N-1)} \otimes \cdots \otimes \mathbf{A}^{(1)}, \quad \mathbf{A}^{(n)} \in \mathbb{R}^{l_n \times l_n} \forall \mathbf{n},
\]

\[
\bigodot_{n=1}^{N} \mathbf{A}^{(n)} = \mathbf{A}^{(N)} \mathbf{A}^{(N-1)} \otimes \cdots \otimes \mathbf{A}^{(1)}, \quad \mathbf{A}^{(n)} \in \mathbb{R}^{l_n \times l_n} \forall \mathbf{n},
\]

\[
\bigodot_{n=1}^{N} \mathbf{A}^{(n)} = \mathbf{A}^{(N)} \mathbf{A}^{(N-1)} \odot \cdots \odot \mathbf{A}^{(1)}, \quad \mathbf{A}^{(n)} \in \mathbb{R}^{l_n \times l_n} \forall \mathbf{n},
\]

which is the sequential Kronecker, Khatri-Rao, and element-wise product, respectively. For the Kronecker and Khatri-Rao products, the order of multiplication matters and here it is assumed to be numerically descending. Additionally, \( \bigodot_{n=1}^{N} \mathbf{A}^{(n)} \) will be used for the sequential product of the \( n = N, N-1, \ldots, m+1, m-1, \ldots, 1 \) matrices.

6.1. Updating factor priors
6.1.1. Prior follows a normal distribution
Let the n-mode factor matrix, \( \mathbf{A}^{(n)} \), follow a normal distribution \( P(\mathbf{A}^{(n)}|\mathbf{\theta}) = \prod_{i=1}^{l_n} \mathcal{N} \left( \mathbf{a}^{(n)} | \mu_i^{(n)} , \Sigma_i^{(n)} \right) \), then the Q-distribution follows

\[
Q(\mathbf{A}^{(n)}) = \prod_{i=1}^{l_n} \mathcal{N} \left( \mathbf{m}_i^{(n)} | \mu_i^{(n)} , \Sigma_i^{(n)} \right),
\]

where \( \mathbf{m}_i^{(n)} \) and \( \Sigma_i^{(n)} \) are the estimated mean and covariance for observation \( i \) in mode \( n \). Generally, the covariance is not observation specific, but this depends on the prior on precision matrix, missing values, and noise modelling.

Under a homoscedastic noise assumption, the factor is updated as

\[
\Sigma_i^{(n)} = \left( \left( \mathbf{A}_i^{(n)} \right) + \langle \mathbf{\tau} \rangle \bigodot_{m \neq n} \left( \mathbf{A}^{(m)} \mathbf{A}^{(m)} \right) \right)^{-1}
\]
\[ m^{(n)}_i = \Sigma_i^{(n)} \left( \tau \right) \left\langle \text{MTTKRP}(\mathcal{X}; \{ \mathbf{A}^{(m)} \}_{m \neq n}) \right\rangle_i + \left\langle \Lambda_i^{(n)} \right\rangle \left\langle \mu_i^{(n)^\top} \right\rangle. \] (30)

If heteroscedastic noise is modelled on all modes, then the factor is updated as

\[ \Sigma_i^{(n)} = \left( \left\langle \mathbf{A}_i^{(n)} \right\rangle + \left\langle \Lambda_i^{(n)} \right\rangle \right) \circ \mathbb{O} \mathbb{R} \left( \mathbf{A}^{(m)} \right) \left( \mathbf{A}^{(m)} \right)^\top \mathbf{diag} \left( \mathbf{\tau}^{(m)} \right) \mathbf{A}^{(m)} )^{-1} \] (31)

\[ m^{(n)^\top}_i = \Sigma_i^{(n)} \left( \left\langle \tau_i^{(n)} \right\rangle \left\langle \text{MTTKRP}(\mathcal{X}; \{ \mathbf{A}^{(m)} \}_{m \neq n}) \right\rangle_i + \left\langle \Lambda_i^{(n)} \right\rangle \left\langle \mu_i^{(n)^\top} \right\rangle \right) \] (32)

Marginalizing over missing values leads to

\[ \Sigma_i^{(n)} = \left( \left\langle \mathbf{A}_i^{(n)} \right\rangle + \left\langle \tau \right\rangle \sum_{i \in \mathcal{O}} \circ \mathbb{O} \mathbb{R} \left( \right) \right)^{-1} \] (33)

\[ m^{(n)^\top}_i = \Sigma_i^{(n)} \left( \left\langle \mathbf{A}_i^{(n)} \right\rangle + \left\langle \tau \right\rangle \sum_{i \in \mathcal{O}} \circ \mathbb{O} \mathbb{R} \left( \right) \right)_i + \left\langle \Lambda_i^{(n)} \right\rangle \left\langle \mu_i^{(n)^\top} \right\rangle \] (34)

where \( i = (i_1, \ldots, i_{n-1}, i_n, i_{n+1}, \ldots, i_N) \) and indexes an observed element in \( \mathcal{O} \). Here \( \mathcal{O} \) is both the set of observed elements and a binary indicator. The update rules are similar to the fully observed case, but here only the expectation when reconstructing present elements is included.

6.1.2. Prior follows a truncated normal, exponential, or uniform distribution

Let the n-mode factor matrix, \( \mathbf{A}^{(n)} \), follow \( P(\mathbf{A}^{(n)} ; \mathbf{0}) = \prod_{i=1}^{L} \prod_{d=1}^{D} g(a^{(n)}_{id}) \) where \( g(a^{(n)}_{id}) \) is either \( \mathcal{N}[0,\infty] \left( a^{(n)}_{id} | \mu_{id}^{(n)}, \lambda_{id}^{(n)} \right) \) Truncated \( \left( a^{(n)}_{id} | \lambda_{id}^{(n)} \right) \) with rate \( \lambda_{id}^{(n)} \), or Uniform \( [\alpha,\beta] \left( a^{(n)}_{id} | \alpha, \beta \right) \) with lower and upper-bounds \( \alpha \) and \( \beta \). Under a normal likelihood, then each of these prior distributions has the following Q-distribution,

\[ Q(\mathbf{A}^{(n)}) = \prod_{i=1}^{L} \prod_{d=1}^{D} \mathcal{N}[0,\infty] \left( m^{(n)}_{id}, \sigma_{id}^{(n)^2} \right) \] (35)

Under a homoscedastic assumption, the factor is updated via,

\[ \sigma_{id}^{(n)^2} = \left( \sigma_C + \left\langle \tau \right\rangle \prod_{m \neq n} \left\langle a^m_{id} \right\rangle \left\langle a^m_{id} \right\rangle^\top \right)^{-1} \] (36)

\[ m^{(n)}_{id} = \sigma_{id}^{(n)^2} \left( \mu_C + \left\langle \tau \right\rangle \left( \left\langle \bigcirc a^m_{id} \right\rangle \right)^\top x^{(n)}_{(n),i} - \sum_{d' \neq d} \left\langle \sigma_{id'} \right\rangle \left\langle a^m_{id'} \right\rangle \left\langle a^m_{id'} \right\rangle^\top \right) \] (37)

where \( \sigma_C \) and \( \mu_C \) depend on the prior distribution, \( P(\mathbf{a}^{(n)}_{id}) \), which slightly changes the resulting update. These values are, \( \left( \mu_C, \sigma_C \right) = \left( \left\langle \mu^{(n)}_{id} \right\rangle, \left\langle \lambda^{(n)}_{id} \right\rangle, \left\langle \lambda^{(n)}_{id} \right\rangle \right) \) for the truncated normal distribution, \( \left( \mu_C, \sigma_C \right) = \left( - \left\langle \lambda^{(n)}_{id} \right\rangle, 0 \right) \) for the exponential distribution, and \( \left( \mu_C, \sigma_C \right) = (0, 0) \) for the uniform distribution.

Similarly, under a heteroscedastic noise assumption, the factor is updated via,

\[ \sigma_{id}^{(n)^2} = \left( \sigma_C + \left\langle \tau_i^{(n)} \right\rangle \prod_{m \neq n} \left\langle a^m_{id} \right\rangle \left\langle \mathbf{diag} \left( \mathbf{\tau}^{(m)} \right) \right\rangle \right)^{-1} \] (38)

\[ m^{(n)}_{id} = \sigma_{id}^{(n)^2} \left( \mu_C + \left\langle \tau_i^{(n)} \right\rangle \left( \left\langle \bigcirc \mathbf{diag} \left( \mathbf{\tau}^{(m)} \right) \right\rangle a^m_{id} \right)^\top x^{(n)}_{(n),i} \right) \] (39)
Marginalizing over missing values is the same as only accounting for the contribution of present elements and leads to,

\[ \sigma_{i,d}^{2} = \left( \sigma_C + \langle \tau \rangle \sum_{i \in \mathcal{O}} \prod_{m \neq n} \left\langle a_{i,m}^{(n)} \right\rangle \right)^{-1} \]  

(40)

\[ m_i^{(n)} = \sigma_{i,d}^{2} \left( \mu_C + \langle \tau \rangle \sum_{i \in \mathcal{O}} \left( \prod_{m \neq n} \left\langle a_{i,m}^{(m)} \right\rangle \right) ^{T} x_i \right. 

\left. - \sum_{d' \neq d} \left\langle a_{i,d}^{(n)} \right\rangle \prod_{m \neq n} \left\langle a_{i,m}^{(m)} \right\rangle \left( a_{i,d}^{(m)} a_{i,d'}^{(m)} \right) \right) \]  

(41)

where \( i = (i_1, \ldots, i_{n-1}, i_n, i_{n+1}, \ldots, i_N) \) and provides the index of each present element in \( \mathcal{O} \). The expression for log-prior and entropy contributions are easily derived following standard textbook definitions. However, evaluating tail probabilities in the truncated normal distribution can be numerically unstable and the toolbox implements the approach described in [88].

6.1.3. Prior follows a von Mises-Fisher matrix distribution

Let the n-mode factor matrix, \( \mathbf{A}^{(n)} \), follow a von Mises-Fisher distribution \( P(\mathbf{A}^{(n)}|\mathbf{F}_0) = \mathcal{MF} \left( \mathbf{A}^{(n)}|\mathbf{F}_0 \right) \), then the Q-distribution follows,

\[ Q(\mathbf{A}^{(n)}) = \mathcal{MF} \left( \mathbf{F}^{(n)} \right) \]  

(42)

Under a homoscedastic noise assumption, the concentration matrix is,

\[ \mathbf{F}^{(n)} = \langle \tau \rangle \text{MTTKRP} \left( \mathbf{X}; \left\{ \mathbf{A}^{(m)} \right\}_{m \neq n} \right) + \mathbf{F}_0 \]  

(43)

If any factor follow a \( \mathcal{MF} \)-distribution, then marginalization of missing values is not supported. Instead, it is possible to impute missing values and perform updates as in the fully observed case. The usual caveats for imputation of missing values then apply.

Heteroscedastic noise is not supported on any mode following the \( \mathcal{MF} \)-distribution. It is, however, still possible to model heteroscedastic noise on non-orthogonal modes. For heteroscedastic noise and non-orthogonal factors on all but mode \( n \), the factor is updated as,

\[ \mathbf{F}^{(n)} = \langle \tau \rangle \text{MTTKRP} \left( \mathbf{X}; \left\{ \text{diag}(\tau^{(m)})\mathbf{A}^{(m)} \right\}_{m \neq n} \right) + \mathbf{F}_0 \]  

(44)

For variational inference, the expected value of the von Mises-Fisher matrix distribution is determined as in [89]. Determining the log prior and entropy requires evaluating the hyper-geometric function \( g_i \) with a matrix argument for which efficient approaches are presented in [89, 90]. Presently, sampling is not included, but can be implemented following the work in [91].

6.2. Updating factor precision or rate prior

For factor matrices following a normal or truncated normal distribution it is possible to infer the precision matrix \( \mathbf{A}^{(n)} \) or a restricted form of it as described in section 2.3.4. Similarly, for factors following an exponential distribution it is possible to infer the rate parameter. For the truncated normal and exponential distribution, it is possible to infer either the scale, component relevance (ARD), or element-wise sparsity. Similarly, the scale and component relevance can be inferred for the multivariate normal distribution, as well as the full precision matrix following a Wishart distribution. However, the toolbox presently does not allow multivariate factors with element-wise sparsity.

Note, uniform factors has no inferable parameters and for the von Mises-Fisher matrix distribution inferring the concentration matrix \( \mathbf{F}_0 \) is not supported.

The toolbox allow sharing precision/rate priors across several modes, where the shared set is defined as \( n \in \Omega_{\text{shared}} \). For each distribution, a constant \( \eta_n \) affects how much the \( n \)th factor contribute to the prior update. For the normal and truncated normal distribution \( \eta_n = 0.5 \) and for the exponential distribution \( \eta_n = 1 \). Dependent on the choice of prior \( P(\cdot) \), the updates are as follows.
For learning the scale of \( n \in \Omega_{\text{shared}} \) let \( \Lambda^{(n)}_i \equiv \Lambda_D \) and the distribution and updates are,

\[
P(\lambda) = \mathcal{G}(\alpha_\lambda, \beta_\lambda), \quad Q(\lambda) = \mathcal{G} \left( \tilde{\alpha}_\lambda, \tilde{\beta}_\lambda \right)
\]

\[
\tilde{\alpha}_\lambda = \alpha_\lambda + D \sum_{n \in \Omega} I_n \eta_n, \quad \tilde{\beta}_\lambda = \beta_\lambda + \sum_{n \in \Omega} \text{trace} \left( \left\langle A^{(n)} A^{(n)} \right\rangle \right) \eta_n
\]

(45)

For automatic relevance determination of the components, let \( \Lambda^{(n)}_i \equiv \text{diag}(\lambda) \) and the distributions and their updates are,

\[
P(\lambda_i) = \prod_{d=1}^D \mathcal{G}(\lambda_{di}|\alpha_{\lambda i}, \beta_{\lambda i}), \quad Q(\lambda_i) = \prod_{d=1}^D \mathcal{G} \left( \tilde{\lambda}_{di}|\tilde{\alpha}_{\lambda i}, \tilde{\beta}_{\lambda i} \right)
\]

\[
\tilde{\alpha}_{\lambda i} = \alpha_{\lambda i} + \sum_{n \in \Omega} I_n \eta_n, \quad \tilde{\beta}_{\lambda i} = \beta_{\lambda i} + \sum_{n \in \Omega} \text{diag} \left( \left\langle A^{(n)} A^{(n)} \right\rangle \right) \eta_n
\]

(47)

For element-wise sparsity, a precision prior is placed on each observation, i.e. \( \Lambda^{(n)}_i \equiv \text{diag}(\lambda_i) \). This gives the following update,

\[
P(\lambda_i) = \prod_{d=1}^D \mathcal{G}(\lambda_{di}|\alpha_{\lambda i}, \beta_{\lambda i}), \quad Q(\lambda_i) = \prod_{d=1}^D \mathcal{G} \left( \tilde{\lambda}_{di}|\tilde{\alpha}_{\lambda i}, \tilde{\beta}_{\lambda i} \right)
\]

\[
\tilde{\alpha}_{\lambda i} = \alpha_{\lambda i} + \sum_{n \in \Omega} I_n \eta_n, \quad \tilde{\beta}_{\lambda i} = \beta_{\lambda i} + \sum_{n \in \Omega} \text{diag} \left( \left\langle a^{(n)} a^{(n)} \right\rangle \right) \eta_n
\]

(49)

For inferring the full precision matrix under multivariate normal factors, let \( \Lambda^{(n)}_i \equiv \Lambda \) then,

\[
P(\Lambda) = \text{Wishart}(w, V), \quad Q(\Lambda) = \text{Wishart}(\tilde{w}, \tilde{V})
\]

\[
\tilde{w} = w + \sum_{n \in \Omega} I_n \eta_n, \quad \tilde{V} = V + \sum_{n \in \Omega} \left\langle A^{(n)} A^{(n)} \right\rangle \eta_n
\]

(51)

Note, the updates presented in this section are not affected by whether the noise is homoscedastic or heteroscedastic.

6.3. Updating tucker factor matrix

Specifying the prior on the factor matrices is done identically for the CP and Tucker model. However, updating factors in the Tucker model requires accounting for the core array which is not done in the updates presented in section 6.1.

Presently, the toolbox only implements factors following the von Mises-Fisher matrix distribution, i.e. orthogonal factors, where the prior distribution is \( P(A^{(n)}) = \nu MF(A^{(n)}|F_0) \). The Q-distribution then is then,

\[
Q(A^{(n)}) = \nu MF \left( A^{(n)}|\tilde{F}^{(n)} \right)
\]

(53)

and the estimated concentration matrix is

\[
\tilde{F}^{(n)} = (\tau) X^{(n)} \left( \bigotimes_{m \neq n} \left\langle A^{(m)} \right\rangle \right)^\top \left\langle G^{(n)} \right\rangle + F_0.
\]

(54)

The toolbox assumes \( F_0 = 0 \) which is the uniform prior on the \( D_n \)-sphere.
6.4. Updating the core prior and its precision prior

Let the core array $G$ be represented by its vectorization $g^{D_1 D_2 \ldots D_n} = \text{vec}(G)$ whose prior follows a normal distribution $P(g|\mu_G, \Psi) = N(g|\mu_G, \Psi^{-1})$. Then, the Q-distribution has the following form,

$$Q(g) = N(g|\mu_g, \Sigma_g)$$  \hspace{1cm} (55)

Under homoscedastic noise, the core is then updated as,

$$\Sigma_g = \left( (\Psi) + (\tau) \right)^{-1} \left( \bigotimes_{n=1}^{N} \frac{A^{(n)^T} A^{(n)}}{\Psi^{(n)}} \right),$$ \hspace{1cm} (56)

$$\mu_g = \Sigma_g \left( \tau \right) \left( \bigotimes_{n=1}^{N} \frac{A^{(n)}}{\Psi^{(n)}} \right)^T \text{vec}(X) + \langle \Psi \rangle \mu_G \right),$$ \hspace{1cm} (57)

Presently, the toolbox only considers $\mu_G = 0$ and $A^{(n)} \sim \nu M F(F_0)$. The latter makes updating equation (56) exceedingly simple as $A^{(n)^T} A^{(n)} = I$ result in $\Sigma_g$ being a diagonal matrix.

As presented in section 2.3.4, how $P(\Psi)$ is specified determines the functionality of the core. The possibilities are inferring the scale, determining element-wise sparsity, or slice-wise pruning.

For inferring the scale, then $\Psi \equiv \psi I_D$ and the distributions and updates are,

$$P(\psi) = G(\psi|\alpha_\psi, \beta_\psi), \hspace{1cm} Q(\psi) = G(\tilde{\alpha}_\psi, \tilde{\beta}_\psi)$$  \hspace{1cm} (58)

$$\tilde{\alpha}_\psi = \alpha_\psi + \frac{1}{2} \prod_{m=1}^{N} D_m, \hspace{1cm} \tilde{\beta}_\psi = \beta_\psi + \frac{1}{2} \langle g^T g \rangle$$  \hspace{1cm} (59)

For inferring element-wise sparisty, the distributions and updates are,

$$P(\psi) = \prod_d G(\psi_d|\alpha_\psi, \beta_\psi), \hspace{1cm} Q(\psi) = \prod_d G(\psi_d|\tilde{\alpha}_\psi, \tilde{\beta}_\psi)$$  \hspace{1cm} (60)

$$\tilde{\alpha}_\psi = \alpha_\psi + \frac{1}{2}, \hspace{1cm} \tilde{\beta}_{\psi_d} = \beta_\psi + \frac{1}{2} \langle \psi_d^2 \rangle$$  \hspace{1cm} (61)

For slice-wise automatic relevance determination $\Psi \equiv \bigotimes_{n=1}^{N} \text{diag}(\psi^{(n)})$ and the distribution and updates for each mode $n$ are,

$$P(\psi^{(n)}) = \prod_{d=1}^{D_n} G(\psi^{(n)}_d|\alpha_\psi, \beta_\psi), \hspace{1cm} Q(\psi^{(n)}) = \prod_{d=1}^{D_n} G(\psi^{(n)}_d|\tilde{\alpha}_\psi, \tilde{\beta}_{\psi^{(n)}})$$  \hspace{1cm} (62)

$$\tilde{\alpha}_\psi = \alpha_\psi + \frac{1}{2} \prod_{m \neq n} D_m, \hspace{1cm} \tilde{\beta}_{\psi_d^{(n)}} = \beta_\psi + \frac{1}{2} \text{diag} \left( \frac{G^{(n)}}{\bigotimes_{m \neq n} \Psi^{(m)}} G^{(n)} \right)$$  \hspace{1cm} (63)

An alternating update scheme is then used, such each mode $n = 1, 2, \ldots, N$ is updated conditioned on all other random variables.

6.5. Updating noise precision prior

The toolbox assumes a normal likelihood with either homoscedastic or heteroscedastic noise variance, the latter is only support CP decomposition. For homoscedastic noise, the noise precision $\tau$ follows a Gamma distribution, i.e. $P(\tau) = G(\alpha_\tau, \beta_\tau)$ with shape $\alpha_\tau$ and rate $\beta_\tau$. The resulting Q-distribution is,

$$Q(\tau) = G\left(\tilde{\alpha}_\tau, \tilde{\beta}_\tau\right)$$  \hspace{1cm} (64)

For fully observed data, the shape parameter is $\tilde{\alpha}_\tau = \alpha_\tau + \frac{1}{2} \prod_{n=1}^{N} I_n$. For CP decomposition the rate is
\[ \bar{\beta}_r = \beta_r + \frac{1}{2} \left( \text{vec}(\mathbf{X})^\top \text{vec}(\mathbf{X}) + \bigodot_{n=1}^N \left\langle A^{(n)} \right\rangle \bigodot \left\langle A^{(n)} \right\rangle \right) - 2 \text{vec}(\mathbf{X})^\top \text{vec} \left( \bigodot_{n=1}^N \left\langle A^{(n)} \right\rangle \right) \]

(65)

Tucker decomposition has a similar update, but differs as it has to account for the core array \( \mathcal{G} \) which changes the estimated rate,

\[ \bar{\beta}_r = \beta_r + \frac{1}{2} \left( \text{vec}(\mathbf{X})^\top \text{vec}(\mathbf{X}) + \bigodot_{n=1}^N \left\langle A^{(n)} \right\rangle \bigodot \left\langle A^{(n)} \right\rangle \right) - 2 \text{vec}(\mathbf{X})^\top \text{vec} \left( \bigodot_{n=1}^N \left\langle A^{(n)} \right\rangle \right) \text{vec}(\mathcal{G}) \]

For the CP model with partially observed data, the estimated shape changes to \( \tilde{\alpha}_r = \alpha_r + \frac{1}{2} |\mathcal{O}| \) while rate changes to,

\[ \bar{\beta}_r = \beta_r + \frac{1}{2} \sum_{(i_1, i_2, \ldots, i_N) \in \mathcal{O}} \left( \sum_{d=1}^D \prod_{n=1}^N a_{i_n, d} + \text{trace} \left( \bigodot_{n=1}^N \left\langle a_{i_n}^{(n)} \right\rangle \bigodot \left\langle a_{i_n}^{(n)} \right\rangle \right) \right) \]

(66)

where \( \mathcal{O} \) is the set of present values, \(|\mathcal{O}|\) the number of present values, and \((i_1, i_2, \ldots, i_N)\) indexes a single element in \( \mathcal{O} \).

**CP with Heteroscedastic Noise**

Estimating mode-specific heteroscedastic noise assumes a special Kronecker structured covariance, see section 2.3, equation (9). Modelling heteroscedastic noise on mode \( n \) is done by defining

\[ P \left( \tau^{(n)} \right) = \prod_{i=1}^{I_n} \mathcal{G} \left( \tau_{i}^{(n)} | \alpha_r, \beta_r \right) \]

The Q-distribution is then,

\[ Q(\tau^{(n)}) = \prod_{i=1}^{I_n} \mathcal{G} \left( \tau_{i}^{(n)} | \tilde{\alpha}_{r(n)}, \tilde{\beta}_{r(n)} \right) \]

(67)

The noise precision is then estimated for each heteroscedastic mode using the following equations in an alternating update scheme.

\[ \tilde{\alpha}_{r(n)} = \alpha_r + \frac{1}{2} \prod_{m \neq n} I_m \]

(68)

\[ \tilde{\beta}_{r(n)} = \beta_r + \frac{1}{2} \left( \text{diag}(\mathbf{X}^{(n)}) \text{diag} \left( \bigodot_{m \neq n} \left\langle \tau^{(m)} \right\rangle \right) \mathbf{X}^{(n)} \right) + \text{diag} \left( \left\langle A^{(n)} \right\rangle \bigodot \left\langle A^{(n)} \right\rangle \right) \]

\[ - 2 \text{diag} \left( \mathbf{X}^{(n)} \bigodot \left\langle \text{diag}(\tau^{(m)}) \right\rangle \left\langle A^{(m)} \right\rangle \bigodot \left\langle A^{(n)} \right\rangle \right) \]

(69)

where \( \tilde{\beta}_{r(n)} \) is a vector \( \mathbb{R}_{\geq 0}^{I_n \times 1} \) of rates.

**Acknowledgments**

The Probabilistic Tensor Toolbox was developed as part of the PhD thesis of the first author and supervised by both co-authors. The PhD was funded by a scholarship from the Technical University of Denmark, Department of Applied Mathematics and Computer Science.
[Beliveau V, Papoutsakis G, Hinrich J Love and Mørup M 2016 Sparse probabilistic parallel factor analysis for the modeling of PET and task-fMRI data In Medical Computer Vision and Bayesian and Graphical Models for Biomedical Imaging (Berlin: Springer) pp 186–98]

[Hore V, Vi Nuela A, Buil A, Knight J, McCarthy M I, Small K and Marchini J 2016 Probabilistic tensor decomposition for multiple-tissue gene expression experiments Nat. Genet. 48 1094]

[Cheng L, Yik-Chung W and Poor H V 2016 Probabilistic tensor canonical polyadic decomposition with orthogonal factors IEEE Trans. Signal Process. 65 663–76]

[Schmidt M N and Mohamed S 2009 Probabilistic non-negative tensor factorization using Markov chain Monte Carlo 17th European Signal Conf. pp 1918–22 IEEE]

[Hinrich J L, Nielsen Søren F V, Madsen K H and Mørup M 2018 Variational Bayesian partially observed non-negative tensor factorization IEEE 28th Int. Workshop on Machine Learning for Signal Processing (MLSP) pp 1–6 IEEE]

[Changwei H, Rai P, and Carin L 2015 Zero-truncated Poisson tensor factorization for massive binary tensors Uncertainty in Artificial Intelligence - Proc. of the 31st Conf., UAI 2015 pp 375–84]

[Qian D, Wang B, Qing X, Zhang T, Zhang Y, Wang X and Nakamura M 2016 Bayesian nonnegative CP decomposition-based feature extraction algorithm for drowsiness detection IEEE Trans. on Neural Systems and Rehabilitation Engineering 25 2897–308]

[Porteous I, Bart E and Welling M 2008 Multi-hdp: A non parametric Bayesian model for tensor factorization AAAI 8 1487–90]

[Zhou J, Bhattacharya A, Herring A H and Dunson D B 2015 Bayesian factorizations of big sparse tensors J. Am. Stat. Assoc. 110 1562–76]

[Ermins B and Taylan Cemgil A 2014 A Bayesian tensor factorization model via variational inference for link prediction arXiv:1409.8276]

[Cheng J. and Yik-Chung W 2018 and H Vincent Poor. Scaling probabilistic tensor canonical polyadic decomposition to massive data IEEE Trans. Signal Process. 66 5554–48]

[Hawkins C and Zhang Z 2018 Robust factorization and completion of streaming tensor data via variational Bayesian inference arXiv:1809.01269]

[Yishuai D, Zheng Y, Lee K-Chih and Zhe S 2018 Probabilistic streaming tensor decomposition IEEE Int. Conf. on Data Mining (ICDM) pp 99–108 IEEE]

[Guhaniyogi R, Qamar S and Dunson D B 2017 Bayesian tensor regression J. Mach. Learn. Res. 18 2733–63]

[Hoff P D 2015 Multilinear tensor regression for longitudinal relational data Ann. Appl. Stat. 9 1169]

[Rose Y, Guanguo Li and Liu Y 2018 Tensor regression meets Gaussian processes Int. Conf. on Artificial Intelligence and Statistics, AISTATS 2018 84 482–90]

[Jørgensen P J H, Nielsen Søren F V, Hinrich J L, Schmidt M N, Madsen K H and Mørup M 2018 Probabilistic parafac2 arXiv:1806.08195]

[Jørgensen P J H, Nielsen Søren F V, Hinrich J Love, Schmidt M Nørgaard, Madsen K H and Mørup M 2019 Analysis of chromatographic data using the probabilistic parafac2 33rd Conf. on Neural Information Processing Systems]

[Khan S A and Kaski S 2014 Bayesian multi-view tensor factorization Joint Conf. on Machine Learning and Knowledge Discovery in Databases pp 656–71 Springer]

[Khan S A, Leppäaho E and Kaski S 2016 Bayesian multi-tensor factorization Mach. Learn. 105 233–53]

[Hinrich J L and Mørup M 2019 Probabilistic tensor train decomposition 27th European Signal Conf. (EUSIPCO)–IEEE 2019]

[Nielsen Søren F V, Madsen K H and Mørup M 2014 Non-negative tensor factorization with missing data for the modeling of gene expressions in the human brain 2014 IEEE Int. Workshop on Machine Learning for Signal Processing (MLSP) pp 1–6 IEEE]

[Ascar E, Kolba T G and Dunlavy D M 2011 All-at-once optimization for coupled matrix and tensor factorizations arXiv:1105.3422]

[Tomasi G and Bro R 2006 A comparison of algorithms for fitting the parafac model Comput. Stat. Data Anal. 50 1700–34]

[Vandecappelle M, Vervliet N and Lathauwer L D 2017 Nonlinear least squares updating of the canonical polyadic decomposition 2017 25th European Signal Conf. (EUSIPCO) pp 663–7 IEEE]

[Bro R and Sijmen D J 1997 A fast non-negativity-constrained least squares algorithm J. Chemometr. 1 393–401]

[Galab D and Mercier B 1976 A dual algorithm for the solution of nonlinear variational problems via finite element approximation Comput. Math. Appl. 2 17–40]

[Lavinas A P and Sidiropoulos N D 2015 Parallel algorithms for constrained tensor factorization via alternating direction method of multipliers IEEE Trans. Signal Process. 63 5450–63]

[Lee D D and Sebastian Seung H 2001 Algorithms for non-negative matrix factorization Advances in Neural Information Processing Systems pp 556–62]

[Yi-An M, Chen T and Fox E 2015 A complete recipe for stochastic gradient mcmc In Advances in Neural Information Processing Systems pp 2917–25]

[Tierney L and Kadane J B 1986 Accurate approximations for posterior moments and marginal densities J. Am. Stat. Assoc. 81 82–6]

[Bishop C M 1999 Variational principal components Ann. Appl. Stat. 9 1169]

[Bishop C M 1999 Variational principal components J. Mach. Learn. Res. 18 2733–63]

[Biswas A, Datta S, Leppäaho E and Kaski S 2016 Bayesian multi-tensor factorization Mach. Learn. 105 233–53]

[Ikeda K, Tanno M and Ito K 2016 Bayesian non-negative Tucker decomposition for learning the small sample size problem J. Mach. Learn. Res. 18 2733–63]

[Khan S A, Leppäaho E and Kaski S 2016 Bayesian multi-tensor factorization Mach. Learn. 105 233–53]

[Khan S A and Kaski S 2014 Bayesian multi-view tensor factorization Joint Conf. on Machine Learning and Knowledge Discovery in Databases pp 656–71 Springer]

[Khan S A, Leppäaho E and Kaski S 2016 Bayesian multi-tensor factorization Mach. Learn. 105 233–53]

[Khan S A and Kaski S 2014 Bayesian multi-view tensor factorization Joint Conf. on Machine Learning and Knowledge Discovery in Databases pp 656–71 Springer]

[Khan S A, Leppäaho E and Kaski S 2016 Bayesian multi-tensor factorization Mach. Learn. 105 233–53]
[82] Bro R 1999 Exploratory study of sugar production using fluorescence spectroscopy and multi-way analysis Chemometr. Intell. Lab. Syst. 46 133–47
[83] Munck L, nørgaard L, Balling Engelsen S, Bro R and Andersson C A 1998 Chemometrics in food science—a demonstration of the feasibility of a highly exploratory, inductive evaluation strategy of fundamental scientific significance Chemometr. Intell. Lab. Syst. 44 31–60
[84] Wakeman D G and Henson R N 2015 A multi-subject multi-modal human neuroimaging dataset Sci. Data 2 150001
[85] Henson R N, Wakeman D G, Litvak V and Friston K J 2011 A parametric empirical Bayesian framework for the EEG/MEG inverse problem: generative models for multi-subject and multi-modal integration Front. Hum. Neurosci. 5 76
[86] Hawrylycz M J et al 2012 An anatomically comprehensive atlas of the adult human brain transcriptome Nature 489 391
[87] Song Q, Hancheng G, Caverlee J and Xia H 2019 Tensor completion algorithms in big data analytics ACM Trans. Knowledge Dis. Data (TKDD) 13 1–48
[88] Cunningham J P, Hennig P and Lacoste-Julien S 2011 Gaussian probabilities and expectation propagation arXiv:1111.6832
[89] šmídl Vaclav and Quinn A 2007 On Bayesian principal component analysis Comput. Stat. Data Anal. 51 4101–23
[90] Khatri C G and Mardia K V 1977 The von Mises–Fisher matrix distribution in orientation statistics J. R. Stat. Soc. B 39 95–106
[91] Hoff P D 2009 Simulation of the matrix Bingham–von Mises–Fisher distribution, with applications to multivariate and relational data J. Computat. Graph. Stat. 18 438–56