Scalable Tensor Factorizations with Missing Data*

Evrim Acar† Daniel M. Dunlavy‡ Tamara G. Kolda† Morten Mørup§

Abstract

The problem of missing data is ubiquitous in domains such as biomedical signal processing, network traffic analysis, bibliometrics, social network analysis, chemometrics, computer vision, and communication networks—all domains in which data collection is subject to occasional errors. Moreover, these data sets can be quite large and have more than two axes of variation, e.g., sender, receiver, time. Many applications in those domains aim to capture the underlying latent structure of the data; in other words, they need to factorize data sets with missing entries. If we cannot address the problem of missing data, many important data sets will be discarded or improperly analyzed. Therefore, we need a robust and scalable approach for factorizing multi-way arrays (i.e., tensors) in the presence of missing data. We focus on one of the most well-known tensor factorizations, CANDECOMP/PARAFAC (CP), and formulate the CP model as a weighted least squares problem that models *only* the known entries. We develop an algorithm called CP-WOPT (CP Weighted OPTimization) using a first-order optimization approach to solve the weighted least squares problem. Based on extensive numerical experiments, our algorithm is shown to successfully factor tensors with noise and up to 70% missing data. Moreover, our approach is significantly faster than the leading alternative and scales to larger problems. To show the real-world usefulness of CP-WOPT, we illustrate its applicability on a novel EEG (electroencephalogram) application where missing data is frequently encountered due to disconnections of electrodes.

Keywords- missing data, tensor factorization, CANDECOMP, PARAFAC, optimization

1 Introduction

Missing data can arise in a variety of settings due to loss of information, errors in the data collection process, or costly experiments. For instance, in biomedical signal processing, missing data can be encountered during EEG analysis, where multiple electrodes are used to collect the electrical activity along the scalp. If one of the electrodes becomes loose or disconnected, the signal is either lost or discarded due to contamination with high amounts of mechanical noise. We also encounter the missing data problem in other areas of data mining, such as packet losses in network traffic analysis [30] and occlusions in images in computer vision [8]. Many real-world data with missing entries are ignored because they are deemed unsuitable for analysis, but this work contributes to the growing evidence that such data can be analyzed.

Unlike most previous studies which have only considered matrices, we focus here on the problem of missing data in *tensors* because it has been shown increasingly that data often have more than two modes of variation and are therefore best represented as multi-way arrays (i.e., tensors) [3, 17]. For instance, in EEG data each signal from an electrode can be represented as a time-frequency matrix; thus, data from multiple channels is three-dimensional (temporal, spectral, and spatial) and forms a three-way array [19]. Social network data, network traffic data, and bibliometric data are of interest to many applications such as community detection, link mining, and more; these data can have multiple dimensions/modalities, are often massively large, and generally have at least some missing data. These are just a few of the many data analysis applications where one needs to deal with large multi-way arrays with missing entries. Other examples of multi-way arrays with missing entries from different disciplines have also been studied in the literature [28, 23, 14]. For instance, [28] shows that, in spectroscopy, intermittent machine failures or different sampling frequencies may result in tensors with missing fibers (i.e., a fiber is the higher-order analogue of a matrix row or column, see Figure 1). Similarly, missing fibers are encountered in multidimensional NMR (Nuclear Magnetic Resonance) analysis, where sparse sampling is used in order to reduce the experimental time [23].

Our goal is to capture the latent structure of the data via a higher-order factorization, even in the presence of missing data. Handling missing data in the context of matrix factorizations, e.g., the widely-used principal component analysis, has long been studied [25, 13] (see [8] for a review). It is also closely related to the matrix completion problem, where the goal is

*This work was supported by the Laboratory Directed Research and Development program at Sandia National Laboratories. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.

†Sandia National Laboratories, Livermore, CA 94551-9159. Email: {eacarat,tgkolda}@sandia.gov.

‡Sandia National Laboratories, Albuquerque, NM 87123-1318. Email: dmdunla@sandia.gov.

§Technical University of Denmark, 2800 Kgs. Lyngby, Denmark. Email: mm@imm.dtu.dk

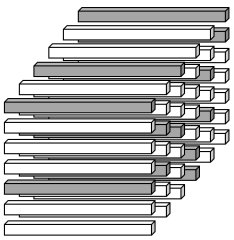


Figure 1: Tensor with missing row fibers (in gray).

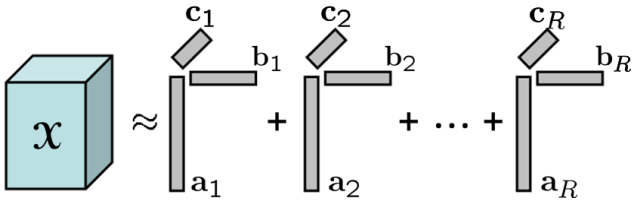


Figure 2: Illustration of an R-component CP model for a third-order tensor $\mathcal{X}$.

to recover the missing entries [10, 9] (see §3 for more discussion). Higher-order factorizations, i.e., tensor factorizations, have emerged as an important method for information analysis [3, 17]. Instead of flattening (unfolding) multi-way arrays as matrices and using matrix factorization techniques, tensor models preserve multi-way nature of the data and extract the underlying factors in each mode (dimension) of a higher-order array.

We focus here on the CANDECOMP/PARAFAC (CP) tensor decomposition [11, 15], which is a commonly-used tensor model in various applications [19, 18, 7, 1, 21]. Let $\mathcal{X}$ be a three-way tensor of size $I \times J \times K$, and assume its rank is R (see [17] for a detailed discussion on tensor rank). With perfect data, the CP decomposition is defined by *factor matrices* $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ of sizes $I \times R$, $J \times R$, and $K \times R$, respectively, such that

$$x_{ijk} = \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr},$$

for all $i = 1, \ldots, I$, $j = 1, \ldots, J$, and $k = 1, \ldots, K$.

In the presence of noise, the true $\mathcal{X}$ is not observable and we cannot expect equality. Instead, the CP decomposition should minimize the error function

$$f(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} \left(x_{ijk} - \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} \right)^2 . \tag{1.1}$$

An illustration of CP for third-order tensors is given in Figure 2. The CP decomposition is extensible to N-way tensors for $N \geq 3$, and there are numerous methods for computing it [2].

In the case of missing data, a standard practice is to impute (i.e., fill in missing entries using various choices of estimates such as the mean) the missing values in some fashion and then apply a standard factorization technique, perhaps using the factorization to re-impute the missing values and repeating the procedure iteratively. Another technique, which we use here, is to use a weighted version of the error function to ignore missing data and model only the known entries. Imputation can be useful as long as the amount of missing data is small but its performance degrades for large amounts of missing data [25] (also see §5). Alternating methods, which compute the factor matrices one at a time, combined with iterative imputation can also be quite effective and often preferred since they are simple and fast. Nevertheless, as the amount of missing data increases, the performance of the algorithm may suffer since the initialization and the intermediate models used to impute the missing values will increase the risk of converging to a wrong solution [28]. Also, the poor convergence of alternating methods due to vulnerability to flatlining is noted in [8].

In this paper, in order to overcome the problems of imputation and alternating methods, we use direct nonlinear optimization to solve the weighted least squares problem for the CP model. The weighted version of (1.1) is

$$f_{\mathcal{W}}(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} \left\{ w_{ijk} \left(x_{ijk} - \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} \right) \right\}^2 , \tag{1.2}$$

where $\mathcal{W}$, which is of the same size as $\mathcal{X}$, is a nonnegative weight tensor defined as

$$w_{ijk} = \begin{cases} 1 & \text{if } x_{ijk} \text{ is known,} \\ 0 & \text{if } x_{ijk} \text{ is missing,} \end{cases}$$

for all $i = 1, \ldots, I$, $j = 1, \ldots, J$, and $k = 1, \ldots, K$.

Our contributions in this paper are summarized as follows:

- We develop a scalable algorithm called CP-WOPT (CP Weighted OPTimization) for tensor factorizations in the presence of missing data. CP-WOPT uses first-order optimization to solve the weighted least squares objective function.
- Using extensive numerical experiments on simulated data sets, we show that CP-WOPT can successfully factor tensors with noise and up to 70%

missing data. Moreover, CP-WOPT is significantly faster than the best published method in the literature [28].

- We demonstrate the applicability of the proposed algorithm on a real data set in a novel EEG application where data is incomplete due to failures of particular electrodes. This is a common occurance in practice, and our experiments show that even if signals from almost half of the channels are missing, underlying brain activities can still be captured using the CP-WOPT algorithm, illustrating the usefulness of our proposed method.

The paper is organized as follows. We introduce the notation in §2. In §3, we discuss related work in matrix and tensor factorizations. The computation of the function and gradient values for the general N-way weighted version of the error function in (1.2) and the presentation of the CP-WOPT method are given in §4. Numerical results on both simulated and real data are given in §5. Conclusions and future work are discussed in §6.

2 Notation

Tensors of order $N \geq 3$ are denoted by Euler script letters ($\mathcal{X}, \mathcal{Y}, \mathcal{Z}$), matrices are denoted by boldface capital letters ($\mathbf{A}, \mathbf{B}, \mathbf{C}$), vectors are denoted by boldface lowercase letters ($\mathbf{a}, \mathbf{b}, \mathbf{c}$), and scalars are denoted by lowercase letters (a, b, c). Columns of a matrix are denoted by boldface lower letters with a subscript ($\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ are first three columns of $\mathbf{A}$). Entries of a matrix or a tensor are denoted by lowercase letters with subscripts, i.e., the $(i_1, i_2, \ldots, i_N)$ entry of an N-way tensor $\mathcal{X}$ is denoted by $x_{i_1 i_2 \cdots i_N}$.

An N-way tensor can be rearranged as a matrix; this is called *matricization*, also known as *unfolding* or *flattening*. The mode-n matricization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoted by $\mathbf{X}_{(n)}$ and arranges the mode-n one-dimensional "fibers" to be the columns of the resulting matrix. Specifically, tensor element $(i_1, i_2, \ldots, i_N)$ maps to matrix element (i_n, j) where

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

$$J_k = \begin{cases} 1, & \text{if } k = 1 \text{ or if } k = 2 \text{ and } n = 1, \\ \prod_{\substack{m=1 \\ m \neq n}}^{k-1} I_m, & \text{otherwise.} \end{cases}$$

Given two tensors $\mathcal{X}$ and $\mathcal{Y}$ of equal size $I_1 \times I_2 \times \cdots \times I_N$, their Hadamard (elementwise) product is denoted by $\mathcal{X} * \mathcal{Y}$ and defined as

$$(\mathcal{X} * \mathcal{Y})_{i_1 i_2 \cdots i_N} = x_{i_1 i_2 \cdots i_N} y_{i_1 i_2 \cdots i_N}$$

for all $1 \leq i_n \leq I_N$.

The *inner product* of two same-sized tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the sum of the products of their entries, i.e.,

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \cdots i_N} y_{i_1 i_2 \cdots i_N}.$$

For a tensor $\mathcal{X}$ of size $I_1 \times I_2 \times \cdots \times I_N$, its norm is

$$\| \mathcal{X} \| = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \cdots i_N}^2}.$$

For matrices (i.e., second-order tensors), $\|\cdot\|$ refers to the analogous Frobenius norm, and for vectors (i.e., first-order tensors), $\|\cdot\|$ refers to the analogous two-norm.

Given a sequence of matrices $\mathbf{A}^{(n)}$ of size $I_n \times R$ for $n = 1, \ldots, N$, $[\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]$ defines an $I_1 \times I_2 \times \cdots \times I_N$ tensor whose elements are given by

$$\left([\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!] \right)_{i_1 i_2 \cdots i_N} = \sum_{r=1}^{R} \prod_{n=1}^{N} a_{i_n r}^{(n)},$$

for all $i_n \in \{1, \ldots, I_n\}$ and $n \in \{1, \ldots, N\}$.

For just two matrices, this reduces to familiar expressions: $[\![\mathbf{A}, \mathbf{B}]\!] = \mathbf{A}\mathbf{B}^{\mathsf{T}}$. Using the notation defined here, (1.2) can be rewritten as

$$f_{\mathcal{W}}(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]) \|^2.$$

3 Related Work in Factorizations with Missing Data

In this section, we first review the approaches for handling missing data in matrix factorizations and then discuss how these techniques have been extended to tensor factorizations.

3.1 Matrix Factorizations Matrix factorization in the presence of missing entries is a problem that has been studied for several decades; see, e.g., [25, 13]. The problem is typically formulated analogously to (1.2) as

$$f_{\mathbf{W}}(\mathbf{A}, \mathbf{B}) = \left\| \mathbf{W} * \left(\mathbf{X} - \mathbf{A}\mathbf{B}^{\mathsf{T}} \right) \right\|^2. \tag{3.3}$$

A common procedure is to use an alternating approach, that combines imputation and alternation and is also known as expectation maximization (EM) [28, 26]. In this approach, the missing values of $\mathbf{X}$ are imputed using the current model, $\hat{\mathbf{X}} = \mathbf{A}\mathbf{B}^{\mathsf{T}}$ as follows:

$$\bar{\mathbf{X}} = \mathbf{W} * \mathbf{X} + (\mathbf{1} - \mathbf{W}) * \hat{\mathbf{X}},$$

where $\mathbf{1}$ is the matrix of all ones. Once $\bar{\mathbf{X}}$ is generated, the matrices $\mathbf{A}$ and/or $\mathbf{B}$ can then be updated according to the error function $\|\bar{\mathbf{X}} - \mathbf{A}\mathbf{B}^{\mathsf{T}}\|^2$. See [26, 16] for further discussion in the missing data and general weighted case.

Recently, a direct nonlinear optimization approach was proposed for matrix factorization with missing data [8]. In this case, (3.3) is solved directly using a 2nd-order damped Newton method. This new method is compared to other standard techniques based on some form of alternation and/or imputation as well as hybrid techniques that combine both approaches. Overall, the conclusion is that nonlinear optimization strategies are key to successful matrix factorization. Moreover, the authors observe that the alternating methods tend to take much longer to converge to the solution even though they make faster progress initially. This work is theoretically the closest to what we propose—the differences are that it focuses on matrices rather than tensors and uses a second-order optimization method rather than first-order (in fact, the paper mentions first-order as future work).

A major difference between matrix and tensor factorizations is worth noting here. In [26, 8], the lack of uniqueness in matrix decompositions is discussed. Given any invertible matrix $\mathbf{G}$, $[\![\mathbf{A}, \mathbf{B}]\!] = [\![\mathbf{A}\mathbf{G}, \mathbf{B}\mathbf{G}^{-\mathsf{T}}]\!]$. This means that there is an infinite family of equivalent solutions. In [8], regularization is recommended as a partial solution, but this can only control scaling and not rotational freedom. In the case of the CP model, there is often only one solution (excepting trivial indeterminacies of scaling and column permutation) that can be recovered exactly; see, e.g., [17] for further discussion on uniqueness of the CP decomposition.

Factorization of matrices with missing entries is also closely related to the matrix completion problem. In matrix completion, one tries to recover the missing matrix entries using the low-rank structure of the matrix. Recent work on this area [9, 10] shows that even if a small amount of matrix entries are available and those are corrupted with noise, it is still possible to recover the missing entries up to noise. In [9], it is also discussed how this problem relates to the field of compressive sensing, which exploits structures of the data. Practically speaking, the difference between completion and factorization is how they measure success. Factorization methods seek accuracy in the factors and this is the measure used in §5. Completion methods, on the other hand, seek accuracy in filling in the missing data. Obviously, once a factorization has been computed, it can be used to reconstruct the missing entries. In fact, most completion methods use this procedure.

3.2 Tensor Factorizations The EM procedure discussed for matrices has also been widely employed for tensor factorizations with missing data. If the current model is $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, then we fill in the missing entries of $\mathcal{X}$ to produce a complete tensor according to

$$\bar{\mathcal{X}} = \mathcal{W} * \mathcal{X} + (\mathbf{1} - \mathcal{W}) * [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!],$$

where $\mathbf{1}$ is the tensor of all ones. The factor matrices are then updated using alternating least squares (ALS) as those that best fit $\bar{\mathcal{X}}$. See, e.g., [6, 29] for further details.

Paatero [24] and Tomasi and Bro [28] have investigated direct nonlinear approaches based on Gauss-Newton (GN). The code from [24] is not widely available; therefore, we focus on [28] and its `INDAFAC` (INcomplete DAta paraFAC) procedure which specifically uses the Levenberg-Marquardt version of GN for fitting the CP model to data with missing entries. The primary application in [28] is missing data in chemometrics experiments. This approach is compared to EM-ALS with the result being that INDAFAC and EM-ALS perform almost equally well in general with the exception that INDAFAC is more accurate for difficult problems, i.e., higher collinearity and systematically missing patterns of data. In terms of computational efficiency, EM-ALS is usually faster but becomes slower than INDAFAC as the percentage of missing entries increases and also depending on the missing entry patterns.

Both INDAFAC and CP-WOPT address the problem of fitting the CP model to incomplete data sets by solving (1.2). The difference is that INDAFAC is based on second-order optimization while CP-WOPT is first-order with a goal of scaling to larger problem sizes.

4 CP-WOPT Algorithm

We consider the general N-way CP factorization problem for tensors with missing entries. Let $\mathcal{X}$ be a real-valued tensor of size $I_1 \times I_2 \times \cdots \times I_N$ and assume its rank is known to be R.[1] Define a nonnegative weight tensor $\mathcal{W}$ of the same size as $\mathcal{X}$ such that

$$w_{i_1 i_2 \cdots i_N} = \begin{cases} 1 & \text{if } x_{i_1 i_2 \cdots i_N} \text{ is known,} \\ 0 & \text{if } x_{i_1 i_2 \cdots i_N} \text{ is missing,} \end{cases} \tag{4.4}$$

for all $i_n \in \{1, \ldots, I_n\}$ and $n \in \{1, \ldots, N\}$.

Our goal is to find matrices $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R}$ for $n = 1, \ldots, N$ that minimize the weighted objective

[1] In practice, the rank is generally not known and is not easily determined. Understanding the performance of the methods under consideration in that scenario is a topic of future work. Results in [2] indicate that direct optimization methods have an advantage when the rank is overestimated.

function (defined below in §4.1), i.e., we want to solve $\min_{\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}} f_{\mathcal{W}}(\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)})$. This objective function can be considered as a mapping from the cross product of N two-dimensional vector spaces to $\mathbb{R}$, i.e.,

$$f_{\mathcal{W}} : \mathbb{R}^{I_1 \times R} \otimes \mathbb{R}^{I_2 \times R} \otimes \cdots \otimes \mathbb{R}^{I_N \times R} \mapsto \mathbb{R}.$$

Although $f_{\mathcal{W}}$ is written as a function of matrices, it can be thought of as a vector function where the parameter vector contains the vectorized and stacked matrices $\mathbf{A}^{(1)}$ through $\mathbf{A}^{(N)}$, i.e.,

$$\left[\mathbf{a}_1^{(1)\mathsf{T}} \quad \cdots \quad \mathbf{a}_R^{(1)\mathsf{T}} \quad \cdots \quad \mathbf{a}_1^{(N)\mathsf{T}} \quad \cdots \quad \mathbf{a}_R^{(N)\mathsf{T}}\right]^{\mathsf{T}}.$$

In this view, $f_{\mathcal{W}} : \mathbb{R}^P \mapsto \mathbb{R}$, where $P = R\sum_{n=1}^{N} I_n$. We derive the weighted objective function in §4.1 and its gradient in §4.2. Once the gradient is known, any gradient-based optimization method [22] can be used to solve the optimization problem.

4.1 Function The N-way objective function is defined by

$$\begin{aligned}(4.5)\quad & f_{\mathcal{W}}(\mathbf{A}^{(1)},\mathbf{A}^{(2)},\ldots,\mathbf{A}^{(N)}) \\ &= \left\| \mathcal{W} * \left(\mathcal{X} - [\![\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}]\!]\right) \right\|^2 \\ &= \sum_{i_1=1}^{I_1}\sum_{i_2=1}^{I_2}\cdots\sum_{i_N=1}^{I_N} w_{i_1 i_2 \cdots i_N}^2 \left\{ x_{i_1 i_2 \cdots i_N}^2 \right. \\ &\left. - 2x_{i_1 i_2 \cdots i_N} \sum_{r=1}^{R}\prod_{n=1}^{N} a_{i_n r}^{(n)} + \left(\sum_{r=1}^{R}\prod_{n=1}^{N} a_{i_n r}^{(n)}\right)^2 \right\}.\end{aligned}$$

This does not need to be computed element-wise, but rather can be computed efficiently using tensor operations. If we pre-compute $\mathcal{Y} = \mathcal{W} * \mathcal{X}$ and $\mathcal{Z} = \mathcal{W} * [\![\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}]\!]$, then

$$(4.6)\qquad f_{\mathcal{W}} = \|\mathcal{Y} - \mathcal{Z}\|^2.$$

Due to the well-known indeterminacies of the CP model, it may also be desirable to add regularization to the objective function as in [2], but this has not been necessary thus far in our experiments.

4.2 Gradient We derive the gradient of (4.5) by computing the partial derivatives of $f_{\mathcal{W}}$ with respect to each element of the factor matrices, i.e., $a_{i_n r}^{(n)}$ for all $i_n = 1,\ldots,I_n$, $n = 1,\ldots,N$, and $r = 1,\ldots,R$. The partial derivatives of the objective function $f_{\mathcal{W}}$ in (4.5) are given by

$$\frac{\partial f_{\mathcal{W}}}{\partial a_{i_n r}^{(n)}} = 2\sum_{i_1=1}^{I_1}\cdots\sum_{i_{n-1}=1}^{I_{n-1}}\sum_{i_{n+1}=1}^{I_{n+1}}\cdots\sum_{i_N=1}^{I_N} w_{i_1 i_2 \cdots i_N}^2 \left(-x_{i_1 i_2 \cdots i_N} + \sum_{l=1}^{R}\prod_{m=1}^{N} a_{i_m l}^{(m)}\right)\prod_{\substack{m=1\\ m\neq n}}^{N} a_{i_m r}^{(m)}$$

for all $i_n = 1,\ldots,I_n$, $n = 1,\ldots,N$, and $r = 1,\ldots,R$.

Once again, the gradient does not need to be computed element-wise. In matrix notation, we can rewrite the gradient equation as

$$(4.7)\qquad \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{A}^{(n)}} = 2\left(\mathbf{Z}_{(n)} - \mathbf{Y}_{(n)}\right)\mathbf{A}^{(-n)},$$

where

$$\mathbf{A}^{(-n)} = \mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)}$$

for $n = 1,\ldots,N$. The symbol $\odot$ denotes the Khatri-Rao product and is defined as follows for two matrices $\mathbf{A}$ and $\mathbf{B}$ of sizes $I \times K$ and $J \times K$ (both have the same number of columns):

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix}\mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_2 \otimes \mathbf{b}_2 & \cdots & \mathbf{a}_K \otimes \mathbf{b}_K\end{bmatrix}$$

where $\otimes$ denotes the vector Kronecker product.

The computations in (4.7) exploit the fact that $\mathcal{W}$ is binary, see (4.4), such that $\mathcal{W}^2 * \mathcal{X} = \mathcal{W} * \mathcal{X} = \mathcal{Y}$ and $\mathcal{W}^2 * [\![\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}]\!] = \mathcal{W} * [\![\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}]\!] = \mathcal{Z}$. The primary computation in (4.7) is called a "matricized tensor times Khatri-Rao product" and can be computed efficiently [4]. The algorithm is summarized in Figure 3.

Now that we have the gradient, we can use any first-order optimization method such as nonlinear conjugate gradient (NCG) and limited-memory BFGS [22].

```
% assume Y = W * X is precomputed
Z = W * [[A^(1), ..., A^(N)]]

% function computation
f_W = ||Y||^2 - 2 <Y, Z> + ||Z||^2

% gradient computation
FOR n = 1 TO N
  G^(n) = -2 Y_(n) A^(-n) + 2 Z_(n) A^(-n)
END
```

Figure 3: CP-WOPT computation of function value ($f_{\mathcal{W}}$) and gradient ($\mathbf{G}^{(n)} \equiv \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{A}^{(n)}}$ for $n \in \{1,\ldots,N\}$). It is possible to make this implementation more efficient by computing $\mathbf{G}^{(n)} = -2\,(\mathbf{Y}_{(n)} - \mathbf{Z}_{(n)})\mathbf{A}^{(-n)}$.

5 Experiments

On both real and simulated three-way data, we assess the performance of the CP-WOPT method in terms of its ability to recover the underlying factors in the presence of missing data. We demonstrate that even if a significant percentage, e.g., 70%, of the tensor entries are missing, the CP factor matrices can still be recovered successfully. Furthermore, we compare our method in terms of efficiency and scalability with the best known method in the literature.

CP-WOPT is implemented using the Tensor Toolbox [5], based on the gradient and function computations shown in Figure 3. For optimization, we implemented the nonlinear conjugate gradient (NCG) method with Hestenes-Stiefel updates [22], globalized via the Moré-Thuente line search [20]. As mentioned in §3, a Gauss-Newton approach has been evaluated previously by Tomasi and Bro [28] and shown to be effective on the missing data problem. We compare CP-WOPT against their implementation of the Gauss-Newton approach called INDAFAC [27].

Both CP-WOPT and INDAFAC are iterative methods. Starting points are generated using the left singular vectors of $\mathbf{X}_{(n)}$ ($\mathcal{X}$ unfolded in mode n) with missing entries replaced by zero. The stopping conditions are set as follows. Both algorithms use the relative change in the function value $f_{\mathcal{W}}$ in (1.2) as a stopping condition (set to 10^{-6}). In INDAFAC, the tolerance on the infinity norm of the gradient is set to 10^{-8} and the maximum number of iterations is set to 10^3. These choices are based on the values used in [28]. In CP-WOPT, the tolerance on the two-norm of the gradient divided by the number of entries in the gradient is set to 10^{-8}, the maximum number of iterations is set to 10^3, and the maximum number of function evaluations is set to 10^4. All experiments were performed using Matlab 7.6 on a Linux Workstation (RedHat 5.2) with 2 Quad-Core Intel Xeon 3.0GHz processors and 32GB RAM.

5.1 Simulated Data We randomly generate three-way tensors with different ranks ($R = 5, 10$), different sizes ($50 \times 50 \times 50$ and $150 \times 150 \times 150$), and varying percentages (10%, 40%, and 70%) and patterns of missing entries (single entries and fibers, see Figure 1). Factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ of appropriate sizes are generated randomly so that the *collinearity* of the columns of the factor matrices in each mode is set to a particular value, C. This means that

$$\frac{\mathbf{a}_r^{\mathsf{T}}\mathbf{a}_s}{\|\mathbf{a}_r\|\|\mathbf{a}_s\|} = \frac{\mathbf{b}_r^{\mathsf{T}}\mathbf{b}_s}{\|\mathbf{b}_r\|\|\mathbf{b}_s\|} = \frac{\mathbf{c}_r^{\mathsf{T}}\mathbf{c}_s}{\|\mathbf{c}_r\|\|\mathbf{c}_s\|} = C,$$

for all $r \neq s$ and $r, s = 1, \ldots, R$. We use $C = 0.5$ in all our experiments as in [28]. The goal is to recover these underlying factor matrices.

From each set of factor matrices, a third-order tensor, $\mathcal{T} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, is generated. Another tensor, $\mathcal{N} \in \mathbb{R}^{I \times J \times K}$, the same size as $\mathcal{T}$ with entries randomly chosen from a standard normal distribution, is then used to add noise to $\mathcal{T}$ as follows:

$$\mathcal{X} = \mathcal{T} + (100/\eta - 1)^{-1/2} \frac{\|\mathcal{T}\|}{\|\mathcal{N}\|}\mathcal{N},$$

where $\eta\%$ denotes the noise percentage. The value $\eta = 2$ is used in our experiments to be comparable with the results in [28].

Finally, we set some entries of each generated tensor to missing. We use two different patterns of missing entries: randomly missing entries and randomly missing fibers. We consider 10%, 40%, and 70% missing data. In the case of randomly missing fibers, we ignore the cases when a complete slice of a tensor turns out to be missing because if we miss a whole slice of a tensor, we cannot recover the CP factors. This is similar to the problem of *coherence* in the matrix completion problem. For instance, if we miss an entire row (or a column) of a matrix, we can never recover its true factorization because we do not have enough information.

We say that the factor matrices have been successfully recovered if the following holds. Let $\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}$ be the recovered factor matrices. We require, for all $r \in \{1, \ldots, R\}$:

$$\text{sim}(r) = \frac{|\mathbf{a}_r^T \bar{\mathbf{a}}_r|}{\|\mathbf{a}_r\|\|\bar{\mathbf{a}}_r\|} \times \frac{|\mathbf{b}_r^T \bar{\mathbf{b}}_r|}{\|\mathbf{b}_r\|\|\bar{\mathbf{b}}_r\|} \times \frac{|\mathbf{c}_r^T \bar{\mathbf{c}}_r|}{\|\mathbf{c}_r\|\|\bar{\mathbf{c}}_r\|} > 0.97 \approx (0.99)^3. \tag{5.8}$$

The uniqueness of the CP model enables the direct comparison of the recovered factor matrices with the factor matrices used to generate the data. However, the CP model has a permutation ambiguity, i.e., there is an ambiguity in the column orderings, so we have to try all possible permutations of the columns of $\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}$. The accuracy of a method is defined as the percentage of times it successfully recovers the factor matrices.

Table 1 reports the accuracy of CP-WOPT and INDAFAC. Thirty sets of factor matrices were generated for each value of R ($R = 5, 10$). Each entry in the table is the percentage of correctly recovered set of factor matrices out of the thirty corresponding CP models. In the case of randomly missing entries, CP-WOPT can perfectly recover the underlying factors with up to 70% missing data. In the case of randomly missing fibers, CP-WOPT only has trouble with the smaller tensor at 70% missing data. It is worth noting that the smaller problems are more difficult for the following reason. Suppose we have a tensor of size $I \times I \times I$ with proportion

Table 1: Accuracy in terms of recovering the underlying factor matrices.

Accuracy for Randomly Missing Entries

Missing Data:	**10%**				**40%**				**70%**			
	INDAFAC		**CP-WOPT**		**INDAFAC**		**CP-WOPT**		**INDAFAC**		**CP-WOPT**	
Rank:	**R=5**	**R=10**	**R=5**	**R=10**	**R=5**	**R=10**	**R=5**	**R=10**	**R=5**	**R=10**	**R=5**	**R=10**
$50 \times 50 \times 50$	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	83.3	96.7	100.0	100.0
$150 \times 150 \times 150$	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	93.3	100.0	100.0

Accuracy for Randomly Missing Fibers

Missing Data:	**10%**				**40%**				**70%**			
	INDAFAC		**CP-WOPT**		**INDAFAC**		**CP-WOPT**		**INDAFAC**		**CP-WOPT**	
Rank:	**R=5**	**R=10**	**R=5**	**R=10**	**R=5**	**R=10**	**R=5**	**R=10**	**R=5**	**R=10**	**R=5**	**R=10**
$50 \times 50 \times 50$	100.0	100.0	100.0	100.0	93.3	100.0	100.0	100.0	26.7	6.7	86.7	76.7
$150 \times 150 \times 150$	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	83.3	80.0	100.0	100.0

M of missing data. Let D be defined as

$$D = \frac{\text{Number of known tensor entries}}{\text{Number of variables}} = \frac{(1-M)I^3}{3RI}.$$

The difficulty of the problem is inversely proportional to D, so that problem becomes easier as I increases as long as M and R are constant. Figure 4 demonstrates how accuracy changes with respect to D. For instance, for $M = 70\%$ missing data (randomly missing fibers) and $R = 5$, D is 50 for tensors of size $50 \times 50 \times 50$ and accuracy of CP-WOPT is 86.7%; for tensors of size $150 \times 150 \times 150$, on the other hand, D is 450 and accuracy of CP-WOPT goes up to 100%. Overall, CP-WOPT does as well or better than INDAFAC in all cases. We point out, however, that we do not use the initialization suggested in [28] in the results presented here.

The experiments show that the underlying factors can be captured even if the CP model is fit to a tensor with significant amount of missing data. This is because the low-rank structure of the tensor is being exploited. A rank-R tensor of size $I \times J \times K$ has $R(I+J+K)$ degrees of freedom. The reason that the factors can be recovered even with 70% missing data is that there is still a lot more data than variables, i.e., the size of the data is equal to $0.3IJK$ which is much greater than the $R(I+J+K)$ variables. Because it is a nonlinear problem, we do not know exactly how many data entries are needed in order to recover a CP model of a low-rank tensor. However, the lower bound for the number of entries needed to recover a low-rank matrix has been derived in [10].

Figure 5 and Figure 6 show average computation times for varying levels of missing values M, i.e., $M = 10\%, 40\%, 70\%$, rank R, i.e., $R = 5, 10$ and data set sizes, i.e., $50 \times 50 \times 50$ and $150 \times 150 \times 150$. For each value of R and data set size, 30 sets of factor matrices were generated. Each bar in Figure 5

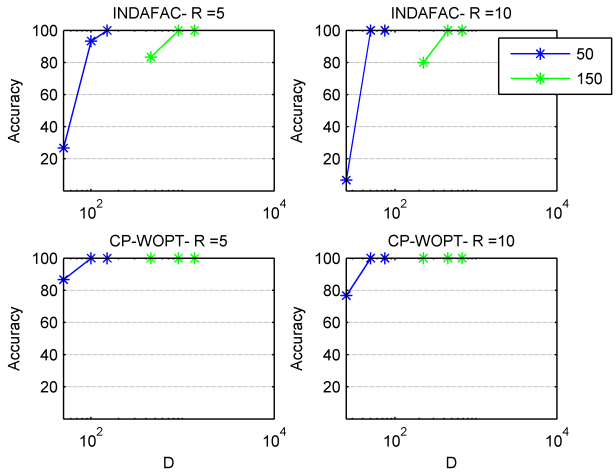


Figure 4: Accuracy versus D, i.e., the ratio of the number of known tensor entries to the number of variables, for INDAFAC and CP-WOPT algorithms for tensors of size $50 \times 50 \times 50$ and $150 \times 150 \times 150$. The plots are for the randomly missing fiber case in Table 1.

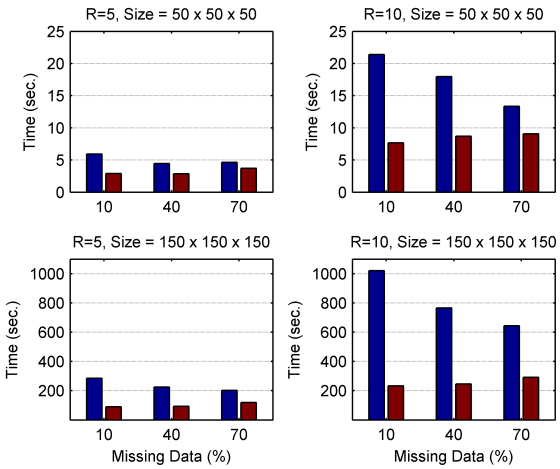


Figure 5: *Randomly Missing Entries:* Average computation time of INDAFAC (blue) and CP-WOPT (red) for varying levels of missing values, ranks and data set sizes.

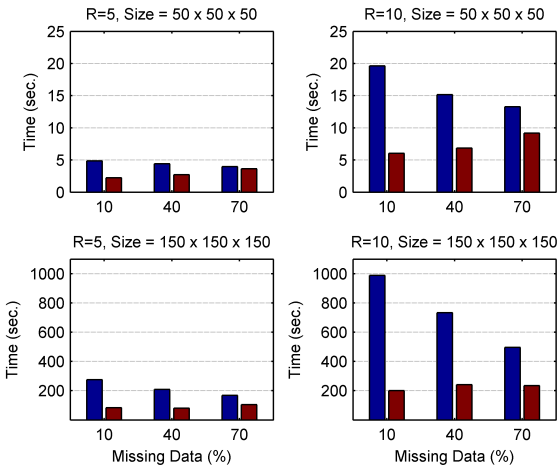


Figure 6: *Randomly Missing Fibers:* Average computation time of INDAFAC (blue) and CP-WOPT (red) for varying levels of missing values, ranks and data set sizes.

and Figure 6 demonstrates the average computation times of the CP models, which successfully recover the underlying factor matrices out of those 30 CP models. In all but two cases, the mean computation times for CP-WOPT were significantly lower than those for INDAFAC using a two-sample t-test at a 95% confidence level. For the two cases where no significant difference in mean computation time was observed (70% missing data for size $50 \times 50 \times 50$ tensors using ranks $R = 5, 10$), the results may be biased by too few samples resulting in two few degrees of freedom (df) in the tests. For those cases, INDAFAC successfully recovered the factors in only 8 of 30 (p-value=0.2117, df=10.04, 95% CI=$[-0.35, \infty)$) and 2 of 30 (p-value=0.2127, df=1.03, 95% CI=$[-15.7, \infty)$) models, respectively. See Table 4 and Table 5 in Appendix A for the detailed timing information, including results from the t-tests.

As R increases, we also see that the increasing computational cost of INDAFAC grows faster than that of CP-WOPT. That is mainly because for an Nth-order tensor $\mathcal{X}$ of size $I_1 \times I_2 \times \cdots \times I_N$, the cost per function/gradient evaluation for CP-WOPT is $O(NQR)$, where $Q = \prod_{n=1}^{N} I_n$ while each iteration of INDAFAC costs $O(P^3)$, where $P = R\sum_{n=1}^{N} I_n$. Note that INDAFAC gets faster as the amount of missing data increases and data gets sparse. On the other hand, CP-WOPT implementation does not yet exploit the data sparsity; therefore, it does not behave in the same way. We plan to address this issue in near future by extending our implementation to sparse data sets.

We also test whether CP-WOPT scales even to larger data sets and observe that it recovers the underlying factor matrices for a data set of size $500 \times 500 \times 500$ (with $M = 10\%, 40\%, 70\%$ randomly missing entries) successfully in approximately 80 minutes on average for $R = 5$. On the other hand, INDAFAC cannot be used to fit the CP model to data sets of that size due to memory problems.

5.2 EEG Data In this section, we use CP to model an EEG data set in order to observe the gamma activation during proprioceptive stimuli of left and right hand. The data set contains multi-channel signals (64 channels) recorded from 14 subjects during stimulation of left and right hand (i.e., 28 measurements in total). For each measurement, the signal from each channel is represented in both time and frequency domains using continuous wavelet transform and vectorized (forming a vector of length 4392); in other words, each measurement can be represented by a *channel* by *time-frequency* matrix. The data for all measurements can then be arranged as a *channels* by *time-frequency* by *measurements* tensor of size $64 \times 4392 \times 28$. For details about the data, see [21].

We model the data using a CP model with $R = 3$, denoting $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ as the extracted factor matrices corresponding to the channels, time-frequency and measurements modes, respectively. We demonstrate the columns of the factor matrices in each mode in Figure 7-A. The 3-D head plots correspond to the columns

of $\mathbf{A}$, i.e., coefficients corresponding to the channels ranging from low values in blue to high values in red. The time-frequency domain representations correspond to the columns of $\mathbf{B}$ rearranged as a matrix and again ranging from low values in blue to high values in red. The bar plots represent the columns of $\mathbf{C}$. Note that 3 rows of images in Figure 7-A (3-D head plot, matrix plot, bar plot) correspond to columns $r = 1, 2, 3$ of the factor matrices ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$), respectively. Observe that the first row of images highlights the differences between left and right hand stimulation while the second and third rows of images pertain to frontal and parietal activations that are shared by the stimuli. Unlike [21], we do not use nonnegativity constraints and upon converting the data from complex to real by the absolute value, we center the data across the channels mode before the analysis.

It is not uncommon in EEG analysis that the signals from some channels are ignored due to malfunctioning of the electrodes. This will create missing fibers in a tensor when we arrange the data as described above (as in Figure 1). To reflect such cases of missing data, we randomly set data for one or more of the 64 channels for each measurement to be missing, center the tensor across the channels mode ignoring the missing entries and then fit a CP model with $R = 3$ to the resulting data using the CP-WOPT algorithm. Let $\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}$ be the factor matrices extracted from a tensor with missing entries using the CP-WOPT algorithm. Table 2 illustrates how the number of missing channels per measurement affects the similarity between the columns of factor matrices extracted from missing data, i.e., $\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}$, and the columns of factor matrices extracted from the original data with no missing entries, i.e., $\mathbf{A}, \mathbf{B}, \mathbf{C}$. The similarity is defined in terms of the measure given in (5.8). For each number of missing channels, we generate 50 tensors with randomly missing channels and extract the corresponding 50 sets of $\bar{\mathbf{A}}, \bar{\mathbf{B}}$ and $\bar{\mathbf{C}}$. The values given in Table 2 are the average similarities between $\mathbf{A}, \mathbf{B}, \mathbf{C}$ and those 50 sets of $\bar{\mathbf{A}}, \bar{\mathbf{B}}$ and $\bar{\mathbf{C}}$. We observe that as the number of missing channels increases, the similarities decrease as expected. However, even up to 30 missing channels per measurement, or about 47% of the data, the extracted factor matrices match with the original factor matrices extremely well, with similarity measures still above 0.90. Furthermore, Figure 7(b), images for $\bar{\mathbf{A}}, \bar{\mathbf{B}}$ and $\bar{\mathbf{C}}$ analogous to those for $\mathbf{A}, \mathbf{B}, \mathbf{C}$ in Figure 7(a), illustrates that the underlying brain dynamics are still captured even when 30 channels per measurement are missing. Note that only slight local distortions can be observed with respect to the corresponding images for the original factor matrices in Figure 7(a).

It can be argued that the activations of the electrodes are highly correlated and even if some of the electrodes are removed, the underlying brain dynamics can still be captured. However, in these experiments we do not set the same channels to missing for each measurement; the channels are randomly missing from each measurement. On the other hand, CP decompositions may not be able to recover factors as well for data with certain patterns of missing values, e.g., missing the signals from the same side of the brain for all measurements. However, it is not very likely to have such data. We still note that the success of the proposed approach depends on the patterns of missing entries, which is the case for any factorization approach proposed for handling missing data.

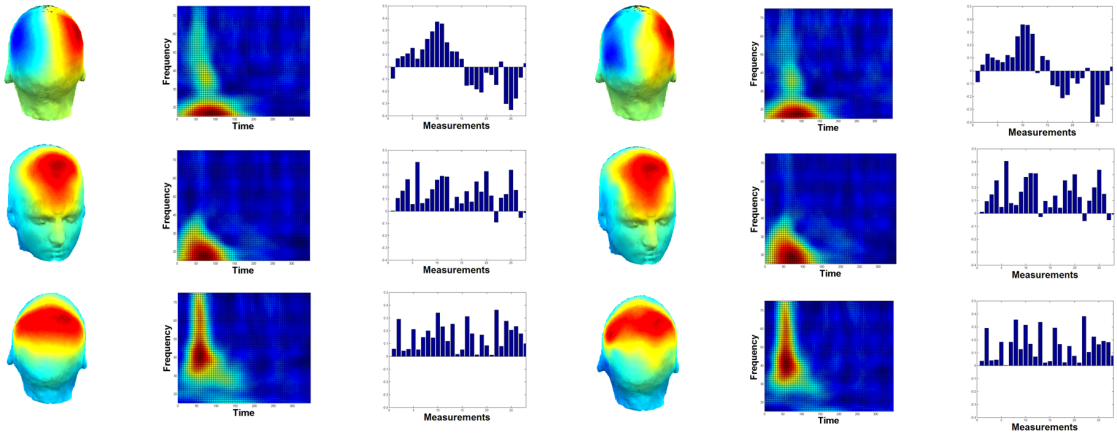
Finally, it may also look reasonable to impute missing entries simply with the mean rather than ignoring them. However, this is not a valid approach especially as the percentage of missing entries increases [25], which we also observe in Table 3. Let $\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}$ be the factor matrices extracted from data with missing entries when missing entries are replaced by the mean across the channel mode. Since the data is centered across the channels mode, missing entries are replaced with

Table 2: *CP-WOPT EEG Results:* The similarity between the columns of $\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}$ and $\mathbf{A}, \mathbf{B}, \mathbf{C}$ when missing entries are ignored and only the known entries are modeled. The similarity measure is defined in (5.8).

Missing Channels	sim($r = 1$)	sim($r = 2$)	sim($r = 3$)
1	0.9989	0.9995	0.9991
10	0.9869	0.9936	0.9894
20	0.9560	0.9826	0.9697
30	0.9046	0.9604	0.9312
40	0.6192	0.8673	0.7546

Table 3: *Imputation EEG Results:* The similarity between the columns of $\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}$ and $\mathbf{A}, \mathbf{B}, \mathbf{C}$ when missing entries are replaced with the mean across the channel mode. The similarity measure is defined in (5.8).

Missing Channels	sim($r = 1$)	sim($r = 2$)	sim($r = 3$)
1	0.9970	0.9984	0.9982
10	0.9481	0.9729	0.9752
20	0.9002	0.9475	0.9371
30	0.6435	0.8719	0.8100
40	0.3045	0.7126	0.5699



(a) No missing entries

(b) 30 channels missing per measurement

Figure 7: Columns of the CP factor matrices ($\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ with $R = 3$) extracted from the EEG data arranged as a *channels* by *time-frequency* by *measurements* tensor with. The 3-D head images were drawn using EEGLab [12].

zeros. Table 3 shows how the similarities (again defined in terms of (5.8)) between the columns of $\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}$ and the columns of the factor matrices extracted from the original data with no missing entries, i.e., $\mathbf{A}, \mathbf{B}, \mathbf{C}$, change as the amount of missing data increases. The same data sets used in Table 2 are again used here for comparison. We can see that when there is large amount of missing data, the structure in the original data can be captured better by ignoring the missing entries rather than replacing them with the means.

6 Conclusions

The closely related problems of matrix factorizations with missing data and matrix completion have recently been receiving a lot of attention. In this paper, we consider the more general problem of tensor factorization in the presence of missing data, formulating the canonical tensor decomposition for incomplete tensors as a weighted least squares problem. Unlike imputation-based techniques, this formulation ignores the missing entries and models only the known data entries. We develop a scalable algorithm called CP-WOPT using gradient-based optimization to solve the weighted least squares formulation of the CP problem.

Our numerical studies suggest that the proposed CP-WOPT approach is accurate and scalable. CP-WOPT recovered the underlying factors successfully even with 70% missing data and scaled to large dense tensors with more than 100 million entries, i.e., $500 \times 500 \times 500$. Moreover, CP-WOPT is faster than the best alternative approach which is based on second-order optimization. Most importantly, we have demonstrated that the factors extracted by the CP-WOPT algorithm can capture brain dynamics in EEG analysis even if signals from some channels are missing, suggesting that practitioners can now make better use of incomplete data in their analyses.

In future studies, we plan to extend our results in several directions. Because it is naturally amenable to this, we will extend our method to large-scale sparse tensors in the case where the amount of missing data is either very large or very small (i.e., $\mathcal{W}$ or $\mathbf{1} - \mathcal{W}$ should also be sparse). We will also include constraints such as non-negativity and penalties to encourage sparsity, which enable us to find more meaningful latent factors from large-scale sparse data. Finally, we will consider the problem of collective factorizations with missing data, where we are jointly factoring multiple tensors with shared factors.

A Detailed numerical results

In this section, we present the detailed timing results corresponding to the plots in Figure 5 and Figure 6. Table 4 and Table 5 contain the average computation times ($\pm$ sample standard deviations) for runs using INDAFAC and CP-WOPT. For each comparison of INDAFAC and CP-WOPT (i.e., each cell of the tables), 30 runs were performed. All statistics reported in the tables—averages, sample standard deviations, p-values, and degrees of freedom(df)—were computed using only those runs in which the factors were successfully recovered. Two-sample t tests using 95% confidence levels were performed for each comparison, where the null hypothesis tested was that the mean computation time

for runs using INDAFAC was greater than the mean time using CP-WOPT. The t statistics were computed assuming unequal variances for the INDAFAC and CP-WOPT runs (as confirmed by F tests on the sample standard deviations), and thus the effective degrees of freedom (df) reported in the tables are those computed using the standard Satterthwaite approximation.

References

[1] E. Acar, C. A. Bingol, H. Bingol, R. Bro, and B. Yener. Multiway analysis of epilepsy tensors. *Bioinformatics*, 23(13):i10–i18, 2007.

[2] E. Acar, T. Kolda, and D. Dunlavy. An optimization approach for fitting canonical tensor decompositions. Technical Report SAND2009-0857, Sandia National Laboratories, 2009.

[3] E. Acar and B. Yener. Unsupervised multiway data analysis: A literature survey. *IEEE T. Knowl. Data En.*, 21(1):6–20, Jan. 2009.

[4] B. W. Bader and T. G. Kolda. Efficient MATLAB computations with sparse and factored tensors. *SIAM J. Sci. Comput.*, 30(1):205–231, Dec. 2007.

[5] B. W. Bader and T. G. Kolda. Tensor toolbox for matlab, version 2.2, last accessed November, 2008. `http://csmr.ca.sandia.gov/~tgkolda/TensorToolbox/`.

[6] R. Bro. *Multi-way analysis in the food industry: Models, algorithms, and applications.* PhD thesis, University of Amsterdam, 1998.

[7] R. Bro. Review on multiway analysis in chemistry—2000–2005. *Crit. Rev. Anal. Chem.*, 36(3–4):279–293, 2006.

[8] A. M. Buchanan and A. W. Fitzgibbon. Damped Newton algorithms for matrix factorization with missing data. In *CVPR'05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 316–322. IEEE Computer Society, 2005.

[9] E. J. Candes and Y. Plan. Matrix completion with noise. arXiv:0903.3131v1, 2009.

[10] E. J. Candes and T. Tao. The power of convex relaxation: Near-optimal matrix completion. arXiv:0903.1476v1, 2009.

[11] J. D. Carroll and J. J. Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35:283–319, 1970.

[12] A. Delorme and S. Makeig. EEGLAB: An open source toolbox for analysis of single-trial EEG dynamics. *J. Neurosci. Meth.*, 134:9–21, 2004.

[13] K. R. Gabriel and S. Zamir. Lower rank approximation of matrices by least squares approximation with any choice of weights. *Technometrics*, 21(4):489–498, Nov. 1979.

[14] X. Geng, K. Smith-Milesa, Z.-H. Zhou, and L. Wang. Face image modeling by multilinear subspace analysis with missing values. In *MM'09: Proceedings of the 17th ACM International Conference on Multimedia*, pages 629–632, 2009.

[15] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. *UCLA working papers in phonetics*, 16:1–84, 1970.

[16] H. A. L. Kiers. Weighted least squares fitting using ordinary least squares algorithms. *Psychometrika*, 62(2):215–266, June 1997.

[17] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.

[18] T. G. Kolda, B. W. Bader, and J. P. Kenny. Higher-order web link analysis using multilinear algebra. In *ICDM 2005: Proceedings of the 5th IEEE International Conference on Data Mining*, pages 242–249, Nov. 2005.

[19] F. Miwakeichi, E. Martínez-Montes, P. A. Valdés-Sosa, N. Nishiyama, H. Mizuhara, and Y. Yamaguchi. Decomposing EEG data into space-time-frequency components using parallel factor analysis. *Neuroimage*, 22(3):1035–1045, July 2004.

[20] J. J. Moré and D. J. Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM T. Math. Software*, 20(3):286–307, Sept. 1994.

[21] M. Mørup, L. K. Hansen, and S. M. Arnfred. ERPWAVELAB a toolbox for multi-channel analysis of time-frequency transformed event related potentials. *J. Neurosci. Meth.*, 161(2):361–368, Apr. 2007.

[22] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, 1999.

[23] V. Y. Orekhov, I. Ibraghimov, and M. Billeter. Optimizing resolution in multidimensional nmr by three-way decomposition. *Journal of Biomolecular NMR*, 27:165–173, 2003.

[24] P. Paatero. A weighted non-negative least squares algorithm for three-way "PARAFAC" factor analysis. *Chemometr. Intell. Lab.*, 38(2):223–242, Oct. 1997.

[25] A. Ruhe. Numerical computation of principal components when several observations are missing. Technical report, University of Umea, Sweden, 1974.

[26] N. Srebro and T. Jaakkola. Weighted low-rank approximations. In *IMCL'03: Proceedings of the 20th International Conference on Machine Learning*, pages 720–727, 2003.

[27] G. Tomasi. Incomplete data PARAFAC (INDAFAC), last accessed May, 2009. `http://www.models.kvl.dk/source/indafac/index.asp`.

[28] G. Tomasi and R. Bro. PARAFAC and missing values. *Chemometr. Intell. Lab.*, 75(2):163–180, Feb. 2005.

[29] B. Walczak and D. L. Massart. Dealing with missing data: Part I. *Chemometr. Intell. Lab.*, 58(1):15–27, 2001.

[30] Y. Zhang, M. Roughan, W. Willinger, and L. Qiu. Spatio-temporal compressive sensing and internet traffic matrices. In *SIGCOMM '09: Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, pages 267–278, New York, NY, USA, 2009. ACM.

Table 4: *Randomly Missing Entries:* Average computation time (± standard deviation) to fit an R-component CP model to a tensor with randomly missing entries. The p-values and effective degrees of freedom (df) are those computed for the two-sample t-tests using 95% confidence levels.

Time for Randomly Missing Entries (sec)

Rank:	$\boldsymbol{R=5}$					
Missing Data :	**10%**		**40%**		**70%**	
	INDAFAC	**CP-WOPT**	**INDAFAC**	**CP-WOPT**	**INDAFAC**	**CP-WOPT**
$50 \times 50 \times 50$	5.9 ± 1.5	2.9 ± 0.6	4.5 ± 0.4	2.9 ± 0.6	4.6 ± 0.9	3.7 ± 0.6
	p-value: 0.0000, df: 37.45		p-value: 0.0000, df: 55.14		p-value: 0.0000, df: 42.23	
$150 \times 150 \times 150$	285.7 ± 30.8	89.4 ± 16.1	225.7 ± 27.0	93.0 ± 17.9	203.0 ± 63.6	118.8 ± 21.7
	p-value: 0.0000, df: 43.79		p-value: 0.0000, df: 50.34		p-value: 0.0000, df: 35.66	
Rank:	$\boldsymbol{R=10}$					
Missing Data :	**10%**	**40%**	**70%**	**10%**	**40%**	**70%**
	INDAFAC	**CP-WOPT**	**INDAFAC**	**CP-WOPT**	**INDAFAC**	**CP-WOPT**
$50 \times 50 \times 50$	21.4 ± 3.2	7.7 ± 0.9	18.0 ± 2.9	8.7 ± 1.7	13.3 ± 2.3	9.1 ± 1.5
	p-value: 0.0000, df: 33.51		p-value: 0.0000, df: 46.48		p-value: 0.0000, df: 46.84	
$150 \times 150 \times 150$	1020.5 ± 168.4	233.3 ± 43.1	765.4 ± 95.9	245.8 ± 35.5	643.6 ± 134.0	291.6 ± 47.0
	p-value: 0.0000, df: 32.79		p-value: 0.0000, df: 36.82		p-value: 0.0000, df: 33.14	

Table 5: *Randomly Missing Fibers:* Average computation time (± standard deviation) to fit an R-component CP model to a tensor with randomly missing fibers. The p-values and effective degrees of freedom (df) are those computed for the two-sample t-tests using 95% confidence levels.

Time for Randomly Missing Fibers (sec)

Rank:	$\boldsymbol{R=5}$					
Missing Data :	**10%**		**40%**		**70%**	
	INDAFAC	**CP-WOPT**	**INDAFAC**	**CP-WOPT**	**INDAFAC**	**CP-WOPT**
$50 \times 50 \times 50$	4.8 ± 0.6	2.2 ± 0.4	4.4 ± 0.7	2.7 ± 0.5	4.0 ± 0.9	3.7 ± 0.8
	p-value: 0.0000, df: 47.74		p-value: 0.0000, df: 49.03		p-value: 0.2117, df: 10.04	
$150 \times 150 \times 150$	275.7 ± 34.2	84.0 ± 26.9	209.5 ± 18.0	80.2 ± 13.4	167.8 ± 20.2	105.2 ± 29.0
	p-value: 0.0000, df: 54.99		p-value: 0.0000, df: 53.69		p-value: 0.0000, df: 51.52	
Rank:	$\boldsymbol{R=10}$					
Missing Data :	**10%**		**40%**		**70%**	
	INDAFAC	**CP-WOPT**	**INDAFAC**	**CP-WOPT**	**INDAFAC**	**CP-WOPT**
$50 \times 50 \times 50$	19.6 ± 3.8	6.0 ± 1.9	15.2 ± 1.7	6.8 ± 1.0	13.3 ± 4.6	9.2 ± 1.8
	p-value: 0.0000, df: 42.01		p-value: 0.0000, df: 45.76		p-value: 0.2127, df: 1.03	
$150 \times 150 \times 150$	989.4 ± 160.5	201.2 ± 40.3	733.6 ± 87.6	241.1 ± 26.7	496.3 ± 47.2	234.6 ± 38.8
	p-value: 0.0000, df: 32.64		p-value: 0.0000, df: 34.35		p-value: 0.0000, df: 44.29	