TENSOR-BASED ALGORITHMS FOR LEARNING MULTIDIMENSIONAL SEPARABLE DICTIONARIES

Florian Roemer, Giovanni Del Galdo, and Martin Haardt

Ilmenau University of Technology, Institute for Information Technology
{florian.roemer,giovanni.delgaldo,martin.haardt}@tu-ilmenau.de

ABSTRACT

Compressive Sensing (CS) allows to acquire signals at sampling rates significantly lower than the Nyquist rate, provided that the signals possess a sparse representation in an appropriate basis. However, in some applications of CS, the dictionary providing the sparse description is partially or entirely unknown. It has been shown that dictionary learning algorithms are able to estimate the basis vectors from a set of training samples. In some applications the dictionary is multidimensional, e.g., when estimating jointly azimuth and elevation in a 2-D direction of arrival (DOA) estimation context. In this paper we show that existing dictionary learning algorithms can be extended to exploit this structure, thereby providing a more accurate estimate of the dictionary. As examples we choose two prominent dictionary learning algorithms, the method of optimal directions (MOD) and the K-SVD algorithm. We propose tensor-based multidimensional extensions for both algorithms and show their improved performances numerically.

1. INTRODUCTION

The field of Compressive Sensing (CS) has led to a new understanding in the way we sample signals. It has been shown that sampling rates below the Nyquist rate can be used for signal acquisition without any loss of information if the signal is "sparse" in a suitable domain. This domain does not need to be known for sampling the signal but only for its reconstruction, which bears the potential to significantly lower the complexity of the sampling devices. The reconstruction process known as sparse recovery is a non-convex combinatorial problem. However, it has been shown that under mild assumptions, convex relaxations (such as the Basis Pursuit [1] or the LASSO algorithm [2]) or even simpler approximations (such as greedy [3] or thresholding-based algorithms [4]) perform very well.

Note that all recovery algorithms require the exact knowledge of the sensing matrix (which we refer to as "dictionary" in this paper) which provides the overcomplete basis for the sparse representation of our observations. In some applications, this matrix may be partially or completely unknown. For instance, in image and video coding it is common to estimate a sparsifying dictionary from training samples (thus, learning "typical patterns" from the observations). In other applications the knowledge of the dictionary may be erroneous, for instance due to calibration errors, jitter, or due to the gridding problem for continuous manifolds [5].

It has been shown that dictionary learning algorithms can estimate the dictionary vectors from a finite set of observations ("training samples"). A large manifold of dictionary learning algorithms have been proposed in recent years. There are two main families of algorithms: Maximum-Likelihood (or Maximum A Posterori) based schemes [6, 7, 8] and Least-Squares (LS) type schemes [9, 10, 11, 12, 13]. Due to their simplicity and popularity, we focus on LS-type algorithms. In fact, many of the recently developed LS-type dictionary learning algorithms can be seen as an extension of the Method of Optimal Directions (MOD) [9] or the K-SVD algorithm [10]. Therefore, we consider the MOD and the K-SVD algorithm as a baseline.

In this paper, we focus on learning multidimensional separable dictionaries. These appear in some multidimensional parameter estimation problems, e.g., 2-D DOA estimation on separable array manifolds [14], joint DOA-DOD (direction of departure) estimation in a co-located MIMO-Radar setting [15], biomedical signal analysis [16, 17], or communications [18, 19]. We show that the multidimensional dictionary estimation problem can be efficiently formulated in terms of tensors. This formulation guides to efficient tensor-based dictionary estimation algorithms which outperform existing schemes by exploiting the multilinear structure of the problem. In particular, we propose tensor extensions of the MOD and the K-SVD algorithm and show their improved performance numerically.

Throughout the paper, vectors, matrices, and tensors are denoted by lower-case bold-faced letters ($\boldsymbol{a}, \boldsymbol{b}, ...$), upper-case bold-faced letters ($\boldsymbol{A}, \boldsymbol{B}, ...$), and bold-faced calligraphic letters ($\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}, ...$), respectively. The transpose, Hermitian transpose, and pseudo-inverse of a matrix $\boldsymbol{A}$ are written as $\boldsymbol{A}^{\mathrm{T}}$, $\boldsymbol{A}^{\mathrm{H}}$, and $\boldsymbol{A}^{+}$, respectively. For an R-D tensor $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{M_1 \times M_2 \times \ldots \times M_R}$, the r-mode unfolding denoted as $[\boldsymbol{\mathcal{A}}]_{(r)} \in \mathbb{C}^{M_r \times (M_1 \cdot \ldots \cdot M_{r-1} \cdot M_{r+1} \cdot M_R)}$ represents a rearrangement of $\boldsymbol{\mathcal{A}}$ in a matrix, where the r-th index is used as a row

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

index and all other indices are aligned along the columns (aligned in reverse cyclical ordering, cf. [20]). The r-mode product between a tensor $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{M_1 \times M_2 \times \ldots \times M_R}$ and a matrix $\boldsymbol{U} \in \mathbb{C}^{N_r \times M_r}$ is written as $\boldsymbol{\mathcal{A}} \times_r \boldsymbol{U}$ and defined via $[\boldsymbol{\mathcal{A}} \times_r \boldsymbol{U}]_{(r)} = \boldsymbol{U} \cdot [\boldsymbol{\mathcal{A}}]_{(r)}$. The Kronecker product between two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ is written as $\boldsymbol{A} \otimes \boldsymbol{B}$. The two-norm of a vector is denoted by $\|\boldsymbol{a}\|_2$ and the Frobenius norm of a matrix or a tensor as $\|\boldsymbol{A}\|_{\mathrm{F}}$.

2. DATA MODEL

Consider a generic sparse recovery problem of the following form

$$\boldsymbol{X} = \boldsymbol{A} \cdot \boldsymbol{S} + \boldsymbol{W}, \tag{1}$$

where $\boldsymbol{X} \in \mathbb{C}^{M \times T}$ represents T consecutive observations from M sensors, $\boldsymbol{A} \in \mathbb{C}^{M \times N}$ is the overcomplete dictionary, $\boldsymbol{S} \in \mathbb{C}^{N \times T}$ represents the sparse coefficient matrix, $\boldsymbol{W} \in \mathbb{C}^{M \times T}$ is the additive noise, and we have $M < N < T$. If $\boldsymbol{A}$ is known, we can recover $\boldsymbol{S}$ from $\boldsymbol{X}$ by using an appropriate sparse recovery algorithm [1, 2, 3, 4].

In our setting, $\boldsymbol{A}$ is partially or entirely unknown. Therefore, $\boldsymbol{A}$ and $\boldsymbol{S}$ must be estimated jointly from $\boldsymbol{X}$, using the fact that $\boldsymbol{S}$ is known to be sparse. In LS-type algorithms the objective is given by [9]

$$\begin{aligned} &\min_{\boldsymbol{A},\boldsymbol{S}} \|\boldsymbol{X} - \boldsymbol{A} \cdot \boldsymbol{S}\|_{\mathrm{F}} \\ &\text{s.t. } \|\boldsymbol{S}\|_0 \leq K_{\max}, \|\boldsymbol{a}_n\|_2 = 1, n = 1, 2, \ldots, N, \end{aligned} \tag{2}$$

where $K_{\max}$ is the maximum number of non-zero elements[1] in $\boldsymbol{S}$, and $\boldsymbol{a}_n$ is the n-th column of $\boldsymbol{A}$. The norm constraint on $\boldsymbol{A}$ is introduced to remove scaling ambiguities between dictionary elements and the corresponding amplitudes in $\boldsymbol{S}$.

We now turn to the R-dimensional setting. For brevity, we discuss the special case of $R = 2$ only. However, the extension to $R > 2$ is straightforward. Consider a sparse recovery problem for a separable 2-D manifold, i.e., a manifold that can be written as the product of two 1-D manifolds [14]. To apply CS, we need to discretize the 2-D manifold. If we choose a separable 2-D sampling grid, we can write the dictionary $\boldsymbol{A}$ as

$$\boldsymbol{A} = \boldsymbol{A}^{(1)} \otimes \boldsymbol{A}^{(2)}, \tag{3}$$

where $\boldsymbol{A}^{(1)} \in \mathbb{C}^{M_1 \times N_1}$, $\boldsymbol{A}^{(2)} \in \mathbb{C}^{M_2 \times N_2}$, and we have $M = M_1 \times M_2$, $N = N_1 \times N_2$. Therefore, (1) becomes

$$\boldsymbol{X} = (\boldsymbol{A}^{(1)} \otimes \boldsymbol{A}^{(2)}) \cdot \boldsymbol{S} + \boldsymbol{W}. \tag{4}$$

The Kronecker model (4) can be rewritten in an equivalent tensor form. Applying the algebraic rules for unfoldings of n-mode products [21, 20] we can rewrite (4) into

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{S}} \times_1 \boldsymbol{A}^{(1)} \times_2 \boldsymbol{A}^{(2)} + \boldsymbol{\mathcal{W}}, \tag{5}$$

[1] This constraint is only used for clarity of presentation. We do not need to know $K_{\max}$ in advance for the tensor-based dictionary learning algorithms.

- Initialize $\hat{\boldsymbol{A}}$
- Repeat for N_{It} iterations
  - $\hat{\boldsymbol{S}} \leftarrow \mathrm{SparseRec}\{\boldsymbol{X}, \hat{\boldsymbol{A}}\}$
  - $\hat{\boldsymbol{A}} \leftarrow \boldsymbol{X} \cdot \hat{\boldsymbol{S}}^{+}$
  - Normalize columns of $\hat{\boldsymbol{A}}$

Table 1. Summary of the MOD algorithm [9].

- Initialize $\hat{\boldsymbol{A}}^{(1)}, \hat{\boldsymbol{A}}^{(2)}$
- Repeat for N_{It} iterations
  - $\hat{\boldsymbol{S}} \leftarrow \mathrm{SparseRec}\{\boldsymbol{X}, \hat{\boldsymbol{A}}^{(1)} \otimes \hat{\boldsymbol{A}}^{(2)}\}$
  - $\hat{\boldsymbol{A}}^{(1)} \leftarrow [\boldsymbol{\mathcal{X}}]_{(1)} \cdot \left(\left[\hat{\boldsymbol{\mathcal{S}}}\right]_{(1)} \cdot (\hat{\boldsymbol{A}}^{(2)} \otimes \boldsymbol{I}_T)^{\mathrm{T}} \right)^{+}$
  - $\hat{\boldsymbol{A}}^{(2)} \leftarrow [\boldsymbol{\mathcal{X}}]_{(2)} \cdot \left(\left[\hat{\boldsymbol{\mathcal{S}}}\right]_{(2)} \cdot (\boldsymbol{I}_T \otimes \hat{\boldsymbol{A}}^{(1)})^{\mathrm{T}} \right)^{+}$
  - Normalize columns of $\hat{\boldsymbol{A}}^{(1)}$, $\hat{\boldsymbol{A}}^{(2)}$

Table 2. Summary of the T-MOD algorithm.

where $\boldsymbol{\mathcal{X}} \in \mathbb{C}^{M_1 \times M_2 \times T}$, $\boldsymbol{\mathcal{S}} \in \mathbb{C}^{N_1 \times N_2 \times T}$, and $\boldsymbol{\mathcal{W}} \in \mathbb{C}^{M_1 \times M_2 \times T}$ are rearranged versions of the matrices $\boldsymbol{X}$, $\boldsymbol{S}$, and $\boldsymbol{W}$ such that $\boldsymbol{X} = [\boldsymbol{\mathcal{X}}]_{(3)}^{\mathrm{T}}$, $\boldsymbol{S} = [\boldsymbol{\mathcal{S}}]_{(3)}^{\mathrm{T}}$, and $\boldsymbol{W} = [\boldsymbol{\mathcal{W}}]_{(3)}^{\mathrm{T}}$, respectively. Note that (5) is known as a "Tucker-2" decomposition [22]. The optimization problem (2) can be written in tensor form as

$$\begin{aligned} &\min_{\boldsymbol{A}^{(1)},\boldsymbol{A}^{(2)},\boldsymbol{\mathcal{S}}} \left\| \boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{S}} \times_1 \boldsymbol{A}^{(1)} \times_2 \boldsymbol{A}^{(2)} \right\|_{\mathrm{F}} \\ &\text{s.t. } \|\boldsymbol{\mathcal{S}}\|_0 \leq K_{\max}, \left\| \boldsymbol{a}_{n_r}^{(r)} \right\|_2 = 1, n_r = 1, 2, \ldots, N_r, r = 1, 2, \end{aligned} \tag{6}$$

where $\boldsymbol{a}_{n_r}^{(r)}$ is the n_r-th column of $\boldsymbol{A}^{(r)}$ for $r = 1, 2$. The advantage of the tensor-based optimization problem (6) compared to the matrix-based optimization problem (2) is that it captures the multilinear nature, i.e., the way the matrices $\boldsymbol{A}^{(1)}$ and $\boldsymbol{A}^{(2)}$ act on $\boldsymbol{S}$ to construct $\boldsymbol{X}$ in a more natural form. As we show in the next section, this guides us to find efficient algorithms that exploit this multilinear structure.

3. TENSOR-BASED DICTIONARY LEARNING ALGORITHMS

In this section we review the MOD [9] and the K-SVD [10] algorithm that estimate $\boldsymbol{A}$ and $\boldsymbol{S}$ from observations $\boldsymbol{X}$ that obey (1). We provide tensor extensions to estimate $\boldsymbol{A}^{(1)}$, $\boldsymbol{A}^{(2)}$ and $\boldsymbol{\mathcal{S}}$ from observations $\boldsymbol{\mathcal{X}}$ that follow (5).

Let us begin with the MOD. It is based on the following observation: if we know $\boldsymbol{S}$ in (1), $\boldsymbol{A}$ can be estimated solving a linear least squares (LS) problem. Conversely, knowing $\boldsymbol{A}$ we obtain $\boldsymbol{S}$ by solving a sparse recovery problem. MOD iterates between the two steps, starting with an initial guess of $\boldsymbol{A}$ (e.g., by using training samples or our partial knowledge of $\boldsymbol{A}$). The MOD algorithm is summarized in Table 1, where $\mathrm{SparseRec}\{\boldsymbol{X}, \boldsymbol{A}\}$ denotes an arbitrary sparse recovery algorithm such as BP, LASSO, OMP, etc.

- Initialize $\hat{\boldsymbol{A}}$
- Repeat for N_{It} iterations
  - $\hat{\boldsymbol{S}} \leftarrow \mathrm{SparseRec}\{\boldsymbol{X}, \hat{\boldsymbol{A}}\}$
  - for $n = 1, 2, ..., N$
    - Compute $\tilde{\boldsymbol{X}}_n$ according to (7), (8)
    - Determine truncated SVD of $\tilde{\boldsymbol{X}}_n \approx \boldsymbol{u} \cdot \sigma \cdot \boldsymbol{v}^{\mathrm{H}}$
    - Update $\boldsymbol{a}_n \leftarrow \boldsymbol{u}$ and $\tilde{\boldsymbol{s}}_n^{\mathrm{T}} \leftarrow \sigma \cdot \boldsymbol{v}^{\mathrm{H}}$

Table 3. Summary of the K-SVD algorithm [10].

- Initialize $\hat{\boldsymbol{A}}^{(1)}$, $\hat{\boldsymbol{A}}^{(2)}$
- Repeat for N_{It} iterations
  - $\hat{\boldsymbol{S}} \leftarrow \mathrm{SparseRec}\{\boldsymbol{X}, \hat{\boldsymbol{A}}^{(1)} \otimes \hat{\boldsymbol{A}}^{(2)}\}$
  - for $n_1 = 1, 2, ..., N_1$, $n_2 = 1, 2, ..., N_2$
    - Compute $\tilde{\boldsymbol{\mathcal{X}}}_{n_1,n_2}$ according to (10), (11)
    - Determine truncated HOSVD of $\tilde{\boldsymbol{\mathcal{X}}}_{n_1,n_2} \approx \sigma \cdot \boldsymbol{u} \circ \boldsymbol{v} \circ \boldsymbol{w}$
    - Update $\boldsymbol{a}_{n_1}^{(1)} \leftarrow \boldsymbol{u}$, $\boldsymbol{a}_{n_2}^{(2)} \leftarrow \boldsymbol{v}$, and $\tilde{\boldsymbol{s}}_n \leftarrow \sigma \cdot \boldsymbol{w}$

Table 4. Summary of the K-HOSVD algorithm.

The MOD algorithm to solve (2) can easily be extended to solve (6). Instead of iterating between updating $\boldsymbol{S}$ and $\boldsymbol{A}$, we iterate between updating $\boldsymbol{\mathcal{S}}$, $\boldsymbol{A}^{(1)}$, and $\boldsymbol{A}^{(2)}$. The LS-optimal solutions for $\boldsymbol{A}^{(1)}$ and $\boldsymbol{A}^{(2)}$ are found by solving LS problems in the 1-mode and the 2-mode unfolding of $\boldsymbol{\mathcal{X}}$. For finding $\boldsymbol{\mathcal{S}}$ we apply the same solutions as in the matrix case[2]. The resulting algorithm called T-MOD ("Tensor-MOD") is summarized in Table 2.

Despite its simplicity, the MOD algorithm has been criticized for its slow convergence, which is due to the fact that $\boldsymbol{A}$ and $\boldsymbol{S}$ are updated separately, which may lead to a slow convergence [10]. This has motivated the authors in [10] to come up with an algorithm that updates the elements of $\boldsymbol{A}$ and $\boldsymbol{S}$ jointly. It is based on the idea that the matrix product $\boldsymbol{A} \cdot \boldsymbol{S}$ can be written as a sum of N rank-one matrices via $\boldsymbol{A} \cdot \boldsymbol{S} = \sum_{n=1}^{N} \boldsymbol{a}_n \cdot \boldsymbol{s}_n^{\mathrm{T}}$, where $\boldsymbol{s}_n^{\mathrm{T}}$ denotes the n-th row of $\boldsymbol{S}$. Therefore, instead of iterating between updating $\boldsymbol{A}$ and $\boldsymbol{S}$ they propose to iterate between the N atoms in the sum, for each atom updating the corresponding elements $\boldsymbol{a}_n$ and $\boldsymbol{s}_n^{\mathrm{T}}$ jointly. The optimization problem (2) for the n-th atom becomes

$$\min_{\boldsymbol{a}_n, \boldsymbol{s}_n^{\mathrm{T}}} \left\| \boldsymbol{X} - \sum_{p=1}^{N} \boldsymbol{a}_p \cdot \boldsymbol{s}_p^{\mathrm{T}} \right\|_{\mathrm{F}} = \min_{\boldsymbol{a}_n, \boldsymbol{s}_n^{\mathrm{T}}} \left\| \boldsymbol{X}_n - \boldsymbol{a}_n \cdot \boldsymbol{s}_n^{\mathrm{T}} \right\|_{\mathrm{F}},$$

$$\text{where} \quad \boldsymbol{X}_n = \boldsymbol{X} - \sum_{p=1 p \neq n}^{N} \boldsymbol{a}_p \cdot \boldsymbol{s}_p^{\mathrm{T}} \tag{7}$$

Using the estimate of the support set of $\boldsymbol{S}$ from the sparse recovery stage, the columns of $\boldsymbol{X}_n$ and $\boldsymbol{s}_n^{\mathrm{T}}$ corresponding to the zero elements in $\boldsymbol{s}_n^{\mathrm{T}}$ can be deleted. We obtain the reduced-size problem

$$\min_{\boldsymbol{a}_n, \tilde{\boldsymbol{s}}_n^{\mathrm{T}}} \left\| \tilde{\boldsymbol{X}}_n - \boldsymbol{a}_n \cdot \tilde{\boldsymbol{s}}_n^{\mathrm{T}} \right\|_{\mathrm{F}}, \tag{8}$$

where $\tilde{\boldsymbol{X}}_n \in \mathbb{C}^{M \times K_n}$ and $\tilde{\boldsymbol{s}}_n^{\mathrm{T}} \in \mathbb{C}^{1 \times K_n}$ are the reduced-size versions of $\boldsymbol{X}_n$ and $\boldsymbol{s}_n^{\mathrm{T}}$ and K_n is the number of non-zero elements in $\boldsymbol{s}_n^{\mathrm{T}}$. The resulting problem (7) is a matrix rank-one approximation problem which is easily solved via a truncated SVD. LS-optimality is then guaranteed by the Eckart-Young theorem [23]. The K-SVD algorithm is shown in Table 3.

We can follow the same train of thought in the tensor case. In particular, the noise-free $\boldsymbol{\mathcal{X}}$ can be expanded into $N = N_1 \cdot N_2$ rank-one tensors in the following manner

$$\boldsymbol{\mathcal{S}} \times_1 \boldsymbol{A}^{(1)} \times_2 \boldsymbol{A}^{(2)} = \sum_{n_1=1}^{N_1} \sum_{n_2=1}^{N_2} \boldsymbol{a}_{n_1}^{(1)} \circ \boldsymbol{a}_{n_2}^{(2)} \circ \boldsymbol{s}_n \tag{9}$$

where $n = (n_1 - 1) \cdot N_2 + n_2$ and $\circ$ denotes the outer product. Consequently, the optimization problem (6) for one atom becomes

$$\min_{\boldsymbol{a}_{n_1}^{(1)}, \boldsymbol{a}_{n_2}^{(2)}, \boldsymbol{s}_n} \left\| \boldsymbol{\mathcal{X}}_{n_1,n_2} - \boldsymbol{a}_{n_1}^{(1)} \circ \boldsymbol{a}_{n_2}^{(2)} \circ \boldsymbol{s}_n \right\|_{\mathrm{F}}, \quad \text{where} \tag{10}$$

$$\boldsymbol{\mathcal{X}}_{n_1,n_2} = \boldsymbol{\mathcal{X}} - \sum_{\substack{p_1=1 \\ (p_1,p_2) \neq (n_1,n_2)}}^{N_1} \sum_{p_2=1}^{N_2} \boldsymbol{a}_{p_1}^{(1)} \circ \boldsymbol{a}_{p_2}^{(2)} \circ \boldsymbol{s}_{(p_1-1) \cdot N_2 + p_2}$$

As before, we reduce the sparse vector $\boldsymbol{s}_n \in \mathbb{C}^{T \times 1}$ to its non-zero entries $\tilde{\boldsymbol{s}}_n \in \mathbb{C}^{K_n \times 1}$. Similarly, $\boldsymbol{\mathcal{X}}_{n_1,n_2} \in \mathbb{C}^{M_1 \times M_2 \times T}$ is reduced to $\tilde{\boldsymbol{\mathcal{X}}}_{n_1,n_2} \in \mathbb{C}^{M_1 \times M_2 \times K_n}$. This results in

$$\min_{\boldsymbol{a}_{n_1}^{(1)}, \boldsymbol{a}_{n_2}^{(2)}, \tilde{\boldsymbol{s}}_n} \left\| \tilde{\boldsymbol{\mathcal{X}}}_{n_1,n_2} - \boldsymbol{a}_{n_1}^{(1)} \circ \boldsymbol{a}_{n_2}^{(2)} \circ \tilde{\boldsymbol{s}}_n \right\|_{\mathrm{F}}. \tag{11}$$

Clearly, (11) represents a rank-one tensor approximation problem. Such problems have been intensively studied in the literature. In general, the LS-optimal solution cannot be computed in closed-form but it requires iterative algorithms [24, 25]. However, a truncated version of the multilinear extension of the SVD known as Higher-Order SVD (HOSVD) [20] provides a rank-one approximation that is very close to the LS-optimal one (in particular, for medium to high SNRs). Since its complexity is much lower than the iterative LS-optimal schemes, we propose to solve (10) via a truncated HOSVD. Therefore, the K-SVD algorithm is extended to tensors by replacing the sequential rank-one matrix approximation step obtained via truncated SVDs by a sequential rank-one tensor approximation step obtained via truncated HOSVDs. The resulting algorithm is called K-HOSVD. We summarize it in Table 4.

Concerning the computational cost in floating point operations (FLOPS) per iteration, if we only count the dictionary update stage (since the sparse recovery stage is the same

[2] So far we have not found a simple way how the sparse recovery algorithms could benefit from the multilinear structure of the dictionary. We leave this as an issue of future research.

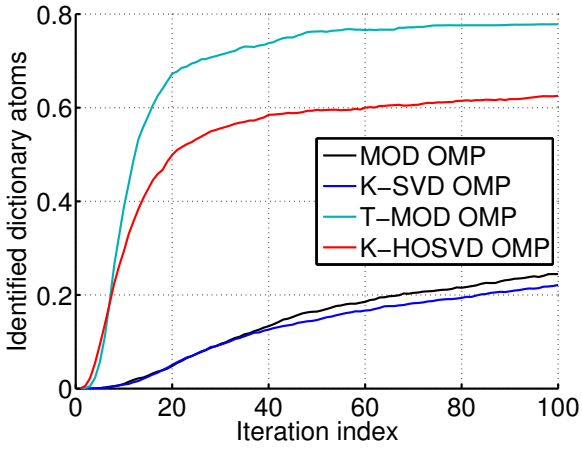

Fig. 1. Identified dictionary atoms vs. the iteration index

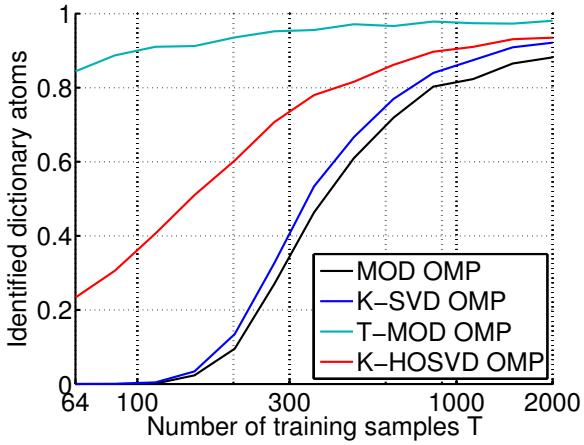

Fig. 2. Identified dictionary atoms vs. the number of training samples T

in all methods) we can conclude the following. Assuming $T \gg N > M$, a "symmetrical" scenario where $M_1 = M_2 = \sqrt{M}$ and $N_1 = N_2 = \sqrt{N}$, and reasonable assumptions on the complexity of LS and the SVD, we obtain approximately $2NT(N+M)$ FLOPS for MOD, $4T(N\sqrt{M} + M\sqrt{N})$ for T-MOD, $2M^2(KT+MN)$ for the K-SVD, and $2M^2(KT+MN) + 4M\sqrt{M}KT$ for the K-HOSVD. Therefore, the K-HOSVD is slightly more complex than the K-SVD (however, the asymptotic complexity order is the same) and T-MOD is less complex than MOD (by a factor of approximately $N/\sqrt{M}/2$).

4. NUMERICAL RESULTS

In order to demonstrate the performance of the tensor-based dictionary learning algorithms we present some numerical results obtained via Monte Carlo simulations in this section. We consider observations according to (4) for $M_1 = M_2 = 5$, $N_1 = N_2 = 8$. The dictionary elements in $\boldsymbol{A}^{(1)}$ and $\boldsymbol{A}^{(2)}$ are drawn from an i.i.d. zero mean Gaussian random process, normalized such that both matrices have unit-norm columns. The matrix $\boldsymbol{W}$ follows the same distribution where each element has a variance of $P_{\mathrm{N}} = 0.001$. For the matrix $\boldsymbol{S}$ we assume that each column has $K = 3$ non-zero entries. Their amplitudes are drawn from a zero mean variance one random Gaussian process. Due to its low computational complexity we employ the Orthogonal Matching Pursuit (OMP) algorithm [3] to for the sparse recovery stage in all the dictionary learning algorithms, assuming that K is known. To assess the quality of the trained dictionaries we adopt the same metric that was used in [10]: we count the relative number (fraction) of *identified dictionary atoms* where we declare $\hat{\boldsymbol{a}}_m$ a successful estimate of $\boldsymbol{a}_n$ when $1 - \left|\hat{\boldsymbol{a}}_m^{\mathrm{H}} \cdot \boldsymbol{a}_n\right| > 0.99$. We compare the MOD [9] and the K-SVD [10] algorithm with their tensor extensions T-MOD and K-HOSVD proposed in this paper.

In Figure 1 we use $T = 500$ training samples and show the performance vs. the number of iterations N_{It}. We initialize the algorithms with a randomly drawn dictionary, i.e., no prior knowledge of the dictionaries is assumed. The results are averaged over 200 Monte Carlo trials. For Figure 2 we vary T, fix the number of iterations to $N_{\mathrm{It}} = 80$, and average over 100 trials. Moreover, we initialize the dictionary estimates via

$$\hat{\boldsymbol{A}}^{(r)} = \boldsymbol{A}^{(r)} + \sqrt{P_{\mathrm{E}}} \cdot \boldsymbol{E}^{(r)},\ r = 1, 2 \qquad (12)$$

where $\boldsymbol{E}^{(r)}$ is drawn from an i.i.d. zero mean variance one Gaussian random process and $P_{\mathrm{E}} = 0.1$. This corresponds to a situation where coarse initial knowledge of the dictionary is available. The simulation results verify that the tensor-based dictionary learning algorithms are able to identify more dictionary atoms, they converge faster, and they require fewer training samples. Based on the examples we investigated it seems that the T-MOD algorithm outperforms the K-HOSVD. However, more studies are needed to confirm whether this is true in general.

5. CONCLUSIONS

In this paper, we have studied the problem of learning sparsifying overcomplete dictionaries for the case where these are multidimensional and separable. We have shown that this problem has a multidimensional structure that can be conveniently expressed using tensors. Based on this formulation, we have extended existing popular LS-based dictionary learning algorithms and derived two new tensor-based dictionary learning schemes, the T-MOD and the K-HOSVD algorithm. Based on numerical results, we have shown the superior performance of these algorithms which stems from the fact that they explicitly exploit the multidimensional structure.

6. REFERENCES

[1] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by Basis Pursuit," *SIAM J. Sci. Comput.*, vol. 20, pp. 33–61, 1999.

[2] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society*, vol. 58, no. 1, pp. 267–288, 1996.

[3] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *Journal of Constructive Approximation*, vol. 13, pp. 57–98, 1997.

[4] I. Daubechies, M. Defrise, and C. D. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Comm. Pure Appl. Math.*, vol. 57, pp. 1413–1457, 2004.

[5] O. Teke, A. C. Gurbuz, and O. Arikan, "Perturbed orthogonal matching pursuit," *IEEE Transactions on Signal Processing*, 2013, preprint, scheduled to appear.

[6] B. Olshausen and D. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vision Research*, vol. 37, pp. 3311–3325, 1997.

[7] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete respresentations," *Neural Computation*, vol. 12, no. 2, pp. 337–365, 2000.

[8] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representations," *Neural Computations*, vol. 15, no. 2, pp. 349–396, 2003.

[9] K. Engan, S. O. Aase, and J. H. Husy, "Multi-frame compression: Theory and design," *EURASIP Signal Process.*, vol. 90, no. 10, pp. 2121–2140, 2000.

[10] M. Aharon, M. Elad, and A. Bruckstein, "The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation," *IEEE Trans. on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[11] J. M. Duarte-Carvajalino and G. Sapiro, "Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization," *IEEE Trans. Image Proc.*, vol. 18, no. 7, pp. 1395–1408, 2009.

[12] K. Skretting and K. Engang, "Recursive least squares dictionary learning algorithm," *IEEE Transactions on Signal Processing*, vol. 58, pp. 2121–2130, Apr. 2010.

[13] W. Dai, T. Xu, and W. Wang, "Simultaneous codeword optimization (SimCO) for dictionary update and learning," *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6340–6353, Dec. 2012.

[14] M. Haardt, M. Pesavento, F. Roemer, and M. N. El Korso, "Subspace methods and exploitation of special array structures," in *Academic Press Library in Signal Processing: Volume 2*, M. Viberg, Ed. Elsevier, 2013, chapter 23, ISBN 978-0-12-396500-4.

[15] D. Nion and N. D. Sidiropoulos, "Tensor algebra and multidimensional harmonic retrieval in signal processing for MIMO radar," *IEEE Trans. on Signal Processing*, vol. 58, no. 11, pp. 5693–4705, Nov. 2010.

[16] H. Becker, P. Comon, L. Albera, M. Haardt, and I. Merlet, "Multi-way space-time-wave-vector analysis for EEG source separation," *EURASIP Signal Processing*, pp. 1021–1031, Apr. 2012.

[17] S. Aviyente, "Compressed sensing framework for EEG compression," in *IEEE/SP 14th Workshop on Statistical Signal Processing (SSP '07)*, Madison, WI, Aug. 2007, pp. 181–184.

[18] N. Sidiropoulos, G. Giannakis, and R. Bro, "Blind PARAFAC receivers for DS-CDMA systems," *IEEE Transactions on Signal Processing*, vol. 48, pp. 810–823, 2000.

[19] C. Bockelmann, H. Schepker, and A. Dekorsy, "Compressive sensing based multi-user detection for machine-to-machine communication," *Transactions on Emerging Telecommunications Technologies: Special Issue on Machine-to-Machine: An emerging communication paradigm*, vol. 24, no. 4, pp. 389–400, Jun. 2013.

[20] L. de Lathauwer, B. de Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 21, no. 4, pp. 1253–1278, 2000.

[21] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, pp. 279–311, 1966.

[22] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, Sep. 2009.

[23] C. Eckart and G. Young, "The approximation of one matrix by another one of lower rank," *Psychometrika*, vol. 1, pp. 211–218, 1936.

[24] L. de Lathauwer, B. de Moor, and J. Vandewalle, "On the best rank-1 and rank-$(r_1, r_2, \ldots, r_n)$ approximation of higher-order tensors," *SIAM J. Matrix Anal. Appl.*, vol. 21, p. 13241342, 2000.

[25] T. Zhang and G. H. Golub, "Rank-one approximation to high order tensors," *SIAM J. Matrix Anal. Appl.*, vol. 23, no. 2, pp. 534–550, 2001.