BLOCK SPARSE REPRESENTATIONS OF TENSORS USING KRONECKER BASES

Cesar F. Caiafa[*]

Instituto Argentino de Radioastronomía
(CCT La Plata, CONICET)
CC5 (1894) V. Elisa, ARGENTINA

Andrzej Cichocki[†]

Lab. for Advanced Brain Signal Processing
Brain Science Institute, RIKEN
2-1 Hirosawa, Wako, 351-0198, JAPAN

ABSTRACT

In this paper, we consider sparse representations of multidimensional signals (tensors) by generalizing the one-dimensional case (vectors). A new greedy algorithm, namely the Tensor-OMP algorithm, is proposed to compute a block-sparse representation of a tensor with respect to a Kronecker basis where the non-zero coefficients are restricted to be located within a sub-tensor (block). It is demonstrated, through simulation examples, the advantage of considering the Kronecker structure together with the block-sparsity property obtaining faster and more precise sparse representations of tensors compared to the case of applying the classical OMP (Orthogonal Matching Pursuit).

Index Terms— Kronecker Bases, Orthogonal Matching Pursuit (OMP), Sparse Representations, Tensors.

1. INTRODUCTION

A concept that underlies the recent developments in the area of *Compressed Sensing* [1] is *sparsity*. It was discovered that most signals of interest do not cover the entire vector space and can be well approximated by sparse representations in a known dictionary. Formally, a signal $\mathbf{y} \in \mathbb{R}^I$ is *s-sparse* with respect to the dictionary $\mathbf{D} \in \mathbb{R}^{I \times M}$ if $\mathbf{y} = \mathbf{D}\mathbf{x}$, with $||\mathbf{x}||_0 \leq s$, where typically $M \geq I$, $s << M$ and $||\mathbf{x}||_0$ is the ℓ_0 quasi-norm of a vector obtained by counting the number of nonzero entries.

The implications of the *sparsity* assumption have recently driven the development of many exciting applications in modern signal processing including: new and optimized sensors using the *Compressed Sensing* theory [1]; Blind Source Separation (BSS) algorithms based on the *sparsity* of morphological distinguishable source signals [2]; de-noising and inpainting of images using sparse representation [3], etc.

Another characteristic of real world signals is that they often have a multidimensional structure where each dimension (mode) has a particular physical meaning such as time, frequency, space, etc. Typical examples of tensor structured datasets are: video streams (rows, columns, time), images (rows, columns, channel), EEG records in neuroscience (channels, frequency, time, trials), etc. Multidimensional signals can be converted to one-dimensional vectors and represented in a Kronecker basis, i.e. the product of the dictionaries associated to each one of the modes as proposed in [4] motivated by real applications (see for example [5]).

For example, most popular transforms applied to a 2-dimensional signal (2D-image) $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2}$ are based on the application of a transformation of rows followed by a transformation of columns which means that the signal can be obtained as

$$\mathbf{Y} = \mathbf{D}_1 \mathbf{X} \mathbf{D}_2^T, \tag{1.1}$$

where $\mathbf{D}_1 \in \mathbb{R}^{I_1 \times M_1}$ and $\mathbf{D}_2 \in \mathbb{R}^{I_2 \times M_2}$ are dictionaries associated to mode-1 (columns) and mode-2 (rows) vectors, and $\mathbf{X}$ is the matrix of coefficients which is typically sparse[1]. A basic result of the matrix analysis allows us to write equation (1.1) in a vectorized form as follows:

$$\mathbf{y} = vec(\mathbf{D}_1 \mathbf{X} \mathbf{D}_2^T) = (\mathbf{D}_2 \otimes \mathbf{D}_1)\mathbf{x}, \tag{1.2}$$

where $\otimes$ stands for the Kronecker product and the operation $\mathbf{x} = vec(\mathbf{X})$ converts the matrix $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$ into a vector $\mathbf{x} \in \mathbb{R}^I$ ($I = I_1 I_2$). In other words, the signal $\mathbf{y} = vec(\mathbf{Y})$ can be explicitly expressed as a linear combination of elements of a dictionary with Kronecker structure $\mathbf{D} = \mathbf{D}_2 \otimes \mathbf{D}_1$.

During last years much effort was devoted to the development of algorithms for the computation of sparse representations of vectors $\mathbf{y} \in \mathbb{R}^I$ for a given dictionary $\mathbf{D} \in \mathbb{R}^{I \times M}$. Greedy algorithms [6] search for nonzero coefficients sequentially. They are very fast (linear complexity on the signal size $\mathcal{O}(I)$) compared to more sophisticated algorithms such as those based on solving a linear constrained optimization problem (minimizing ℓ_1-norm) [7] which have higher complexity (at least cubic complexity $\mathcal{O}(I^3)$).

The application of existing vector algorithms to multidimensional signals involves very heavy computations requir-

[*] Also from Facultad de Ingeniería - UBA, Av. Paseo Colón 850. 4to. piso, Ala sur (1063), Buenos Aires, ARGENTINA and visiting scientist at LABSP-BSI, RIKEN, JAPAN

[†] Also from Warsow University of Technology and System Research Institute, Warsaw, POLAND.

[1] Examples of separable transforms used for image processing are: the Discrete Fourier Transform (DFT), Cosine Transform (DCT), Wavelet Transform (DWT) and others.

ing also huge amount of memory storage making them not practical. In this paper, we demonstrate that we can exploit the Kronecker structure of dictionaries together with *block sparsity* to reduce dramatically the complexity and memory requirements.

In this work we generalize sparse representations to higher order tensors and introduce the concept of *block sparsity*. We develop a greedy algorithm for tensors with a block sparse representation. A selected set of simulation results are presented demonstrating the benefits of our approach in terms of computational complexity and accuracy of reconstructions.

2. SPARSE REPRESENTATIONS OF MULTIDIMENSIONAL SIGNALS (TENSORS)

Given a multidimensional signal (tensor) $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_N}$ its n-*mode* vectors are obtained by fixing every index but the one in the mode n. The *n-mode unfolding matrix* $\mathbf{Y}_{(n)} \in \mathbb{R}^{I_n \times I_1 I_2 \ldots I_{n-1} I_{n+1} \ldots I_N}$ is defined by arranging all the *n-mode vectors* as columns of a matrix. Note that for the 2D case, *1-mode vectors* and *2-mode vectors* are columns and rows respectively and the *2-mode unfolding matrix* is the transposed matrix. Given a multidimensional signal (tensor) $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_N}$ and a matrix $\mathbf{A} \in \mathbb{R}^{J \times I_n}$ the *n-mode tensor by matrix product* $\underline{\mathbf{Z}} = \underline{\mathbf{Y}} \times_n \mathbf{A} \in \mathbb{R}^{I_1 \times I_2 \ldots I_{n-1} \times J \times I_{n+1} \ldots I_N}$ is defined by:

$$z_{i_1 i_2 \ldots i_{n-1} j i_{n+1} \ldots i_N} = \sum_{i_n=1}^{I_n} y_{i_1 i_2 \ldots i_N} a_{j i_n}, \qquad (2.1)$$

with $i_k = 1, 2, ..., I_k$ $(k \neq n)$ and $j = 1, 2, ..., J$.

The Tucker decomposition [8] is a powerful compressed format that exploits the linear structure of the *unfolding matrices* of a tensor, more specifically, when ranks of these matrices are bounded by $rank(\mathbf{Y}_{(n)}) \leq R_n \leq I_n$ $(n = 1, 2, ..., N)$, then the following *multilinear* expression holds:

$$\underline{\mathbf{Y}} = \underline{\mathbf{G}} \times_1 \mathbf{A}_1 \times_2 \mathbf{A}_2 ... \times_N \mathbf{A}_N, \qquad (2.2)$$

with $\underline{\mathbf{G}} \in \mathbb{R}^{R_1 \times R_2 \times \ldots \times R_n}$ and $\mathbf{A}_n \in \mathbb{R}^{I_n \times R_n}$. It is easy to see that *n-mode vectors* of a tensor with a Tucker representation belongs to the span of columns of matrix $\mathbf{A}_n$. In fact, it can be shown that equation (2.2) implies [9]

$$\mathbf{Y}_{(n)} = \mathbf{A}_n \mathbf{G}_{(n)} (\mathbf{A}_N \otimes \mathbf{A}_{N-1} ... \mathbf{A}_{n+1} \otimes \mathbf{A}_{n-1} ... \mathbf{A}_1)^T. \qquad (2.3)$$

Let us define the vectorization operator on tensors as $vec(\underline{\mathbf{Y}}) \equiv vec(\mathbf{Y}_{(1)}) \in \mathbb{R}^I$ $(I = \prod_{n=1}^N I_n)$, i.e. by stacking all the *1-mode vectors*. Now, we are ready to state the relationship between the Tucker model and a Kronecker representation for multidimensional signals.

Lemma 2.1 (Relationship between the Tucker model and a Kronecker representation)**.** *Given* $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_N}$, $\underline{\mathbf{X}} \in \mathbb{R}^{M_1 \times M_2 \ldots \times M_N}$, $\mathbf{D}_n \in \mathbb{R}^{I_n \times M_n}$ *(*$n = 1, 2, ..., N$*),* $\mathbf{x} = vec(\underline{\mathbf{X}})$ *and* $\mathbf{y} = vec(\underline{\mathbf{Y}})$*, the following two representations are equivalent:*

$$\underline{\mathbf{Y}} = \underline{\mathbf{X}} \times_1 \mathbf{D}_1 \times_2 \mathbf{D}_2 ... \times_N \mathbf{D}_N, \qquad (2.4)$$

$$\mathbf{y} = (\mathbf{D}_N \otimes \mathbf{D}_{N-1} \otimes ... \otimes \mathbf{D}_1)\mathbf{x}, \qquad (2.5)$$

Proof. Using equation (2.3) for mode-1 in equation (2.4) we have $\mathbf{Y}_{(1)} = \mathbf{D}_1 \mathbf{X}_{(1)} (\mathbf{D}_N \otimes \mathbf{D}_{N-1} \otimes ... \otimes \mathbf{D}_2)^T$, and by applying the property of equation (1.2) we finally obtain the desired result (2.5). □

Based on this equivalence, we say that a multidimensional signal (tensor) $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_N}$ has a sparse representation with respect to the n-mode dictionaries $\mathbf{D}_n$ $(n = 1, 2, ..., N)$ if its vectorized version admits a s-sparse representation over the Kronecker dictionary $\mathbf{D} = \mathbf{D}_N \otimes \mathbf{D}_{N-1} \otimes ... \otimes \mathbf{D}_1$. In other words, it has an equivalent Tucker representation (equation 2.4) with a sparse core tensor $\underline{\mathbf{X}}$, i.e. with only s nonzero entries.

In the standard Tucker model, a core tensor $\underline{\mathbf{G}}$ has usually much smaller size than data tensor $\underline{\mathbf{Y}}$ with $R_n << I_n$ and the main objective is to find such decomposition, that is to compute $\underline{\mathbf{G}}$ and factor matrices $\mathbf{A}_n$, usually with additional constraints. In contrast, in our approach the data tensor (measurements) $\underline{\mathbf{Y}}$ and dictionaries $\mathbf{D}_n$ are assumed to be known and our objective is to approximately recover the core tensor $\underline{\mathbf{X}}$ which is assumed to be very sparse and its size is larger than the size of $\underline{\mathbf{Y}}$ $(M_n \geq I_n)$. It is noted that, in the compressed sensing setting, the n-mode factors $\mathbf{D}_n$ correspond to the products of a sensing matrix by a sparsifying basis in each mode [4].

In this work, we consider that nonzero entries are located within a subtensor or block of the core tensor. More specifically, we introduce the following generalized definition of sparsity where each mode is characterized by having a specific sparsity profile:

Definition 2.2 (Multidimensional block-sparsity)**.** A multidimensional signal (tensor) $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_N}$ is $(s_1, s_2, ..., s_N)$-*block sparse* with respect to the factors $\mathbf{D}_n \in \mathbb{R}^{I_n \times M_n}$ $(n = 1, 2, ..., N)$ if it admits a Tucker representation based only on few s_n selected columns of each factor (typically $s_n << M_n$), i.e. if $\mathcal{I}_n = [i_n^1, i_n^2, ..., i_n^{s_n}]$ denote a subset of indices for mode n $(n = 1, 2, ..., N)$, then

$$\underline{\mathbf{Y}} = \underline{\mathbf{X}} \times_1 \mathbf{D}_1 \times_2 \mathbf{D}_2 \times_3 ... \times_N \mathbf{D}_N, \qquad (2.6)$$

with $x_{i_1 i_2 \ldots i_N} = 0 \; \forall (i_1, i_2, ..., i_N) \notin \mathcal{I}_1 \times \mathcal{I}_2 \times ... \times \mathcal{I}_N$.

We highlight that the nonzero entries of the core tensor $\underline{\mathbf{X}}$ belongs to a subtensor (block) defined by $\underline{\mathbf{X}}(\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_N)$. It is easy to see that $(s_1, s_2, ..., s_N)$-*block sparsity* of tensor $\underline{\mathbf{Y}}$ implies that its vectorized version $\mathbf{y} = vec(\underline{\mathbf{Y}})$ is s-*sparse* $(s = s_1 s_2 ... s_N$ and $I = I_1 I_2 ... I_N)$ with respect to the corresponding Kronecker basis. Additionally, it can be proven that

$(s_1, s_2, ..., s_N)$-*block sparsity* implies that its n-*mode* vectors are s_n-*sparse* for each $n = 1, 2, ..., N$ with respect to the corresponding n-mode dictionary.

3. TENSOR-OMP: A GREEDY ALGORITHM FOR COMPUTING BLOCK-SPARSE REPRESENTATIONS

Here we develop an efficient greedy type algorithm to find a $(s_1, s_2, ..., s_N)$-block sparse representation of a tensor (Definition 2.2) with respect to the factors $\mathbf{D}_n \in \mathbb{R}^{I_n \times M_n}$ $(n = 1, 2, ..., N)$. We show that, since the nonzero entries are restricted to be located within a subtensor (block) of size $s_1 \times s_2 \times ... \times s_N$, they can be identified very quickly and in much fewer iterations compared to a classical greedy algorithm: the Orthogonal Matching Pursuit (OMP) algorithm.

Let us first recall the basic OMP algorithm (see Algorithm 1) which was adapted and studied in [6, 10]. The objective of this algorithm is to find a sparse representation of a vector $\mathbf{y} \in \mathbb{R}^I$ over a known Dictionary $\mathbf{D} \in \mathbb{R}^{I \times M}$. OMP iteratively refines a sparse solution by successively identifying one component that yield the greatest improvement in quality. An optimized implementation of this algorithm was recently proposed in [11] where the Cholesky factorization is used to implement the step 5 efficiently.

Algorithm 1 : Vector-OMP [6]

Require: Dictionary $\mathbf{D} \in \mathbb{R}^{I \times M}$, signal $\mathbf{y} \in \mathbb{R}^I$, sparsity s, tolerance ϵ
Ensure: Sparse representation $\mathbf{y} = \mathbf{D}\mathbf{x}$ with $||\mathbf{x}||_0 \leq s$ ($\mathbf{x}(\mathcal{I}) = \mathbf{a}$)
1: $\mathcal{I} = [\emptyset]$, $\mathbf{r} = \mathbf{y}$, $\mathbf{x} = \mathbf{0}$, $k = 1$;
2: **while** $k \leq s$ and $||\mathbf{r}||_2 > \epsilon$ **do**
3: $i^k = \arg\max_i |\mathbf{d}_i^T \mathbf{r}|$;
4: $\mathcal{I} = [\mathcal{I}, i^k]$;
5: $\mathbf{x}(\mathcal{I}) = \arg\min_u ||\mathbf{D}(:, \mathcal{I})\mathbf{u} - \mathbf{y}||_2^2$;
6: $\mathbf{r} = \mathbf{y} - \mathbf{D}\mathbf{x}$;
7: $k = k + 1$;
8: **end while**
9: **return** $\mathcal{I}$,$\mathbf{a}$;

Algorithm 2 : Tensor-OMP

Require: n-mode dictionaries $\{\mathbf{D}_1, \mathbf{D}_2, ..., \mathbf{D}_N\}$ with $\mathbf{D}_n \in \mathbb{R}^{I_n \times M_n}$, signal $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$, Max. number of coefficients k_{max}, tolerance ϵ
Ensure: Sparse representation $\underline{\mathbf{Y}} = \underline{\mathbf{X}} \times_1 \mathbf{D}_1 \times_2 \mathbf{D}_2 \times_3 ... \times_N \mathbf{D}_N$ with $x_{i_1 i_2 ... i_N} = 0 \ \forall (i_1, i_2, ..., i_N) \notin \mathcal{I}_1 \times \mathcal{I}_2 \times ... \times \mathcal{I}_N$ ($\underline{\mathbf{X}}(\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_N) = \underline{\mathbf{A}}$).
1: $\mathcal{I}_n = [\emptyset]$ $(n = 1, 2, ..., N)$, $\underline{\mathbf{R}} = \underline{\mathbf{Y}}$, $\underline{\mathbf{X}} = \underline{\mathbf{0}}$, $k = 1$;
2: **while** $|\mathcal{I}_1||\mathcal{I}_2|...|\mathcal{I}_N| < k_{max}$ and $||\underline{\mathbf{R}}||_F > \epsilon$ **do**
3: $[i_1^k i_2^k ... i_N^k] = \arg\max_{[i_1 i_2 ... i_N]} |\underline{\mathbf{R}} \times_1 \mathbf{D}_1^T(:, i_1) \times_2 ... \times_N \mathbf{D}_N^T(:, i_N)|$;
4: $\mathcal{I}_n = \mathcal{I}_n \cup [i_n^k]$ $(n = 1, 2, ..., N)$, $\mathbf{B}_n = \mathbf{D}_n(:, \mathcal{I}_n)$;
5: $\mathbf{a} = \arg\min_{\mathbf{u}} ||(\mathbf{B}_N \otimes \mathbf{B}_{N-1} \otimes ... \otimes \mathbf{B}_1)\mathbf{u} - \mathbf{y})||_2^2$;
6: $\underline{\mathbf{R}} = \underline{\mathbf{Y}} - \underline{\mathbf{A}} \times_1 \mathbf{B}_1 \times_2 \mathbf{B}_2 ... \times_N \mathbf{B}_N$;
7: $k = k + 1$;
8: **end while**
9: **return** $\{\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_N\}, \underline{\mathbf{A}}$;

We can use the same steps of Algorithm 1 for tensors as it is shown in Algorithm 2. In this case, the step 5 corresponds to the following general minimization problem:

$$\mathbf{a} = \arg\min_{\mathbf{u}} ||(\mathbf{B}_N \otimes \mathbf{B}_{N-1} \otimes ... \otimes \mathbf{B}_1)\mathbf{u} - \mathbf{y})||_2^2, \quad (3.1)$$

where $\mathbf{y} \in \mathbb{R}^I$ $(I = \prod_{n=1}^N I_n)$ is the vectorized version of tensor $\underline{\mathbf{Y}}$ and $\mathbf{B}_n \in \mathbb{R}^{I_n \times s_n}$ correspond to the sub-matrices obtained by restricting the n-mode dictionaries to the columns indicated by indices $\mathcal{I}_n$, i.e. $\mathbf{B}_n = \mathbf{D}_n(:, \mathcal{I}_n)$. Then, the approximation of the signal is given by a Tucker model using these matrices as factors which, written in vector form is:

$$\hat{\mathbf{y}} = (\mathbf{B}_N \otimes \mathbf{B}_{N-1} \otimes ... \otimes \mathbf{B}_1)\mathbf{a}, \quad (3.2)$$

with $\mathbf{a} = vec(\underline{\mathbf{A}}) \in \mathbb{R}^t$ $(t = \prod_{n=1}^N |\mathcal{I}_n|)$ being the vectorized tensor of nonzero entries ($|\mathcal{I}_n|$ stands for the number of indices selected in the mode n at iteration k). By defining $\mathbf{B} = \mathbf{B}_N \otimes \mathbf{B}_{N-1} \otimes ... \otimes \mathbf{B}_1$, we see that the least squares solution of this problem is given by $\mathbf{a} = [\mathbf{B}^T\mathbf{B}]^{-1}\mathbf{B}\mathbf{y}$ which means that $[\mathbf{B}^T\mathbf{B}]\mathbf{a} = \mathbf{B}\mathbf{y}$. This allows us to write

$$\mathbf{B}_1^T\mathbf{B}_1\mathbf{A}_{(1)}(\mathbf{B}_N^T\mathbf{B}_N \otimes ... \otimes \mathbf{B}_2^T\mathbf{B}_2) = \mathbf{B}_1^T\mathbf{Y}_{(1)}(\mathbf{B}_N^T \otimes ... \otimes \mathbf{B}_2^T). \quad (3.3)$$

By denoting $\underline{\mathbf{Z}}^{(1)} = \underline{\mathbf{A}} \times_1 \mathbf{I} \times_2 \mathbf{B}_2^T\mathbf{B}_2 ... \times_N \mathbf{B}_N^T\mathbf{B}_N$ and $\underline{\mathbf{P}} = \underline{\mathbf{Y}} \times_1 \mathbf{B}_1^T \times_2 ... \times_N \mathbf{B}_N^T$ we have

$$\mathbf{B}_1^T\mathbf{B}_1(\mathbf{Z}^{(1)})_{(1)} = \mathbf{P}_{(1)}, \quad (3.4)$$

which can be solved for $(\mathbf{Z}^{(1)})_{(1)}$ in a very efficient way by using a Cholesky factorization of the Hermitian matrix $\mathbf{B}_1^T\mathbf{B}_1$. Note that this is a very small problem because size of matrix $\mathbf{B}_1^T\mathbf{B}_1$ is $(|\mathcal{I}_1| \times |\mathcal{I}_1|)$. Now we can use the solution $\underline{\mathbf{Z}}^{(1)}$ of the subproblem (3.4) and write its mode-2 unfolded version as follows:

$$\mathbf{B}_2^T\mathbf{B}_2\mathbf{A}_{(2)}(\mathbf{B}_N^T\mathbf{B}_N \otimes ... \otimes \mathbf{B}_3^T\mathbf{B}_3 \otimes \mathbf{I}) = (\mathbf{Z}^{(1)})_{(2)}, \quad (3.5)$$

where, by defining $\underline{\mathbf{Z}}^{(2)} = \underline{\mathbf{A}} \times_1 \mathbf{I} \times_2 \mathbf{I} \times_3 \mathbf{B}_3^T\mathbf{B}_3 ... \times_N \mathbf{B}_N^T\mathbf{B}_N$ leads us to the following simple subproblem also solved efficiently by using the Cholesky factorization of the Hermitian matrix $\mathbf{B}_2^T\mathbf{B}_2$:

$$\mathbf{B}_2^T\mathbf{B}_2(\mathbf{Z}^{(2)})_{(2)} = (\mathbf{Z}^{(1)})_{(2)}. \quad (3.6)$$

By subsequently applying this procedure, after N steps we finally arrive to the desired matrix $\mathbf{A}_{(N)}$ which corresponds to the coefficients for selected indices in the current iteration.

We highlight that Tensor-OMP algorithm not only optimizes the memory storage but also requires far fewer iterations compared to the classical OMP algorithm because the maximum number of iterations is $k_{max} << s = s_1 s_2 ... s_N$, with s being the number of nonzero entries within $\underline{\mathbf{X}}$.

4. SELECTED EXPERIMENTAL RESULTS

In the first experiment we analyzed the computation complexity by running the vector-OMP and our Tensor-OMP algorithms for synthetically generated tensors with block sparse

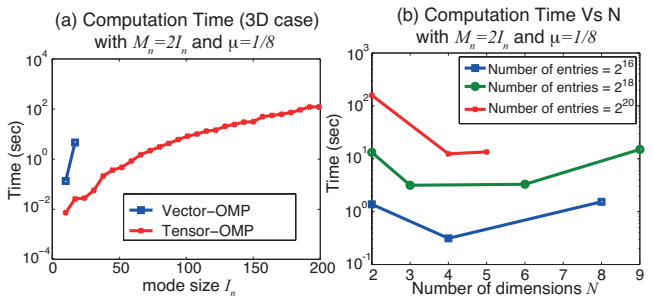

Fig. 1. Tensor-OMP Computational time

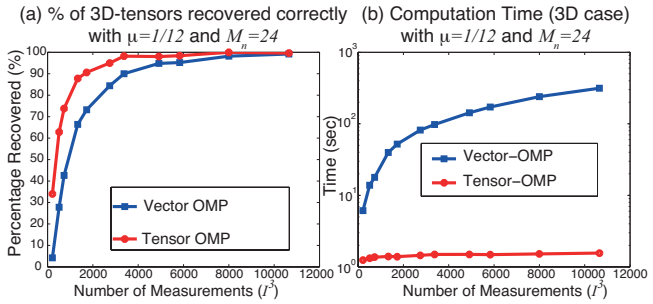

Fig. 2. Comparison of Tensor-OMP and Vector-OMP algorithms

representations in Kronecker basis with Gaussian mode dictionaries. We used a fixed mode sparsity $\mu = s_n/M_n = 1/8$. In Fig. 1, the computation time versus the mode size I_n (a); and versus the number of dimensions $N = 2, 3, ..., 9$ (b), are shown. In a second experiment we analyzed the capability of the algorithm to recover the correct set of non-zero coefficients on an ensemble of 500 simulations. In Fig. 2 (a) we compare the percentage of correctly recovered representations (same non-zero coefficients) by applying Vector-OMP and Tensor-OMP. It is interesting to note that Tensor-OMP outperforms the classical OMP by recovering significantly more coefficients and much faster (Fig. 2). Finally, in Fig 3 we present an application to Compressed Sensing of a

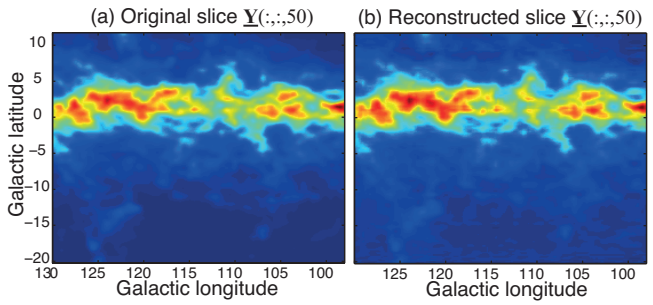

Fig. 3. Using Tensor-OMP for Compressed Sensing of an astrophysical 3D image $\underline{\mathbf{Y}} \in \mathbb{R}^{256\times256\times128}$. Original and reconstructed 50^{th} slice corresponding to a velocity of $v = -74Km/s$. A global PSNR=$34.4dB$ is obtained in only 109 sec.

3D astrophysical signal[2] $\underline{\mathbf{Y}} \in \mathbb{R}^{256\times256\times128}$ corresponding to the observation of the neutral hydrogen (HI) 21 cm in a patch of the sky. This data cube has a sparse representation using an Orthogonal Daubechies Wavelet Kronecker basis in $N = 3$ dimensions. We take measurements by multiplying each mode with a Gaussian sensing matrix, i.e. $\underline{\mathbf{Y}} \times_1 \mathbf{S}_1 \times_2 \mathbf{S}_2 \times_3 \mathbf{S}_3$. We obtained a very good reconstruction (global PSNR = $34.4dB$) in only 109 sec by using $\epsilon = 0.01$ and only 10^6 measurements (12% of the total number of samples).

5. CONCLUSIONS

We introduced the concept of block sparse representation of tensors by using the equivalence between the Tucker model and the representation of vectorized tensors in Kronecker bases. A fast greedy algorithm was proposed to compute sparse representations tensors. Selected experimental results on synthetic as well as on real world signals were presented demonstrating the advantage of Tensor-OMP algorithm in terms of computation time and accuracy of reconstructions.

6. REFERENCES

[1] E. Candes, M. Wakin "An Introduction To Compressive Sampling," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 21–30, 2008.

[2] J. Bobin, J. L. Starck, J. Fadili, and Y. Moudden, "Sparsity and morphological diversity in blind source separation," *Image Proc., IEEE Trans. on*, vol. 16, no. 11, pp. 2662–2674, 2007.

[3] M. Elad, M. A. T. Figueiredo, and Y. Ma, "On the role of sparse and redundant representations in image processing," *Proc. of the IEEE*, vol. 98, no. 6, pp. 972–982, 2010.

[4] M. F. Duarte and R. G. Baraniuk, "Kronecker compressive sensing," *Image Proc., IEEE Trans. on*, to appear, 2011.

[5] Y. Rivenson, A. Stern, "Compressed imaging with a separable sensing operator," *IEEE Signal Processing Letters*, vol. 16, no. 6, pp. 449–452, 2009.

[6] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *Inf. Theory, IEEE Trans. on*, vol. 50, no. 10, pp. 2231–2242, 2004.

[7] S. Chen and D. L. Donoho, "Atomic decomposition by basis pursuit," *SIAM Review*, Jan 2001.

[8] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, pp. 1253–1278, 2000.

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

[10] D. Needell, "Topics in Compressed Sensing," *PhD Dissertation, Univ. of California Davis*, 2009.

[11] R. Rubinstein, M. Zibulevsky, and M. Elad Elad, "Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit," *CS Technion*, 2008.

[2]We thank Laura Suad from IAR/IAFE-CONICET (Argentina) for providing us the astrophysical dataset (Leiden-Argentine-Bonn HI survey).