Overview

Multidimensional compressed sensing and their applications

Cesar F. Caiafa\textsuperscript{1}\textsuperscript{*} and Andrzej Cichocki\textsuperscript{2}

Compressed sensing (CS) comprises a set of relatively new techniques that exploit the underlying structure of data sets allowing their reconstruction from compressed versions or incomplete information. CS reconstruction algorithms are essentially nonlinear, demanding heavy computation overhead and large storage memory, especially in the case of multidimensional signals. Excellent review papers discussing CS state-of-the-art theory and algorithms already exist in the literature, which mostly consider data sets in vector forms. In this paper, we give an overview of existing techniques with special focus on the treatment of multidimensional signals (tensors). We discuss recent trends that exploit the natural multidimensional structure of signals (tensors) achieving simple and efficient CS algorithms. The Kronecker structure of dictionaries is emphasized and its equivalence to the Tucker tensor decomposition is exploited allowing us to use tensor tools and models for CS. Several examples based on real world multidimensional signals are presented, illustrating common problems in signal processing such as the recovery of signals from compressed measurements for magnetic resonance imaging (MRI) signals or for hyper-spectral imaging, and the tensor completion problem (multidimensional inpainting).

\textsuperscript{*}Correspondence to: ccaiafa@iar.unlp.edu.ar

\textsuperscript{1}Instituto Argentino de Radioastronomía (IAR), CCT La Plata—CONICET and Facultad de Ingeniería, Universidad de Buenos Aires, Buenos Aires, Argentina

\textsuperscript{2}Laboratory for Advanced Brain Signal Processing (LABSP), RIKEN BSI and Research Systems Institute, PAS, Warsaw, Poland

Conflict of interest: The authors have declared no conflicts of interest for this article.

INTRODUCTION

One of the most remarkable properties of real world signals is that they are compressible. In fact, modern compression protocols such as JPEG, JPEG2000, and MPEG have exhibited a great success, providing high ratios of compression at a marginal loss of information. This fact shows that data sets have an internal structure which is revealed by coding the signals using appropriate bases, also known as ‘dictionaries’. Most popular dictionaries are, for example, those obtained through the discrete cosine transform (DCT), the wavelet transform (WT), and others.\textsuperscript{1} Traditional signal acquisition systems are based on sampling analog signals at a high Nyquist rate followed by a compression coding stage keeping in memory only the essential information about signals, i.e. storing only the most significant coefficients. Compressed sensing (CS) theory suggests the compelling idea that the sampling process can be greatly simplified by taking only few informative measurements from which full data sets can be reconstructed almost perfectly.\textsuperscript{2,3} CS theory has revolutionized signal processing, for instance, new imaging sensor paradigms were developed on the basis of CS,\textsuperscript{4–8} and some classical image processing problems were approached using results of CS theory as in the case of denoising,\textsuperscript{9–11} inpainting,\textsuperscript{12–16} super-resolution,\textsuperscript{15,17–19} deblurring,\textsuperscript{20–23} and others.

Originally, CS theory was developed for digital signals, in particular for signals that are mapped to a one-dimensional (1D) array (vector). Powerful reconstruction techniques were developed for relatively small sized vectors which are measured by using random matrices. There are excellent review papers in the literature that cover the state-of-the-art...
of CS theory and algorithms for data sets in vector form (see for example 24-27). However, today’s
technology faces the challenge of solving CS problems
where multidimensional signals are involved and the
size of data sets increases exponentially with the
number of dimensions. In this case, the computation
time and storage memory requirements turn out of
control easily. For example, a relatively small
two-dimensional (2D) image with 512 x 512 pixels
represents a vector with 262144 entries which
would take too much time to process with the
state-of-the-arts CS vector algorithms. Models for
multidimensional data sets, also known as tensors
(multiway arrays), have a long history in mathematics
and applied sciences. 28,29 While these models have
recently been applied to multidimensional signal
processing, they were developed independently of
the theory of sparse representations and CS. In
this paper, we give an overview of recent results
revealing connections among tensor decompositions
models and CS involving multidimensional signals.
We show that, by using multilinear representation
models, we are able to obtain a convenient
representation of signals that allows us to develop
efficient algorithms for multidimensional CS. By
keeping the multidimensional structure of signals
captured by tensor decomposition models, we are
able to exploit the signal structure in each mode
(dimension) simultaneously. For example, a 2D digital
image has two modes, each one composed by
vectors, i.e., columns and rows (mode-1 and mode-2
vectors, respectively). The Tucker model (defined
below) 29,30 takes into account the linear structure of
vectors in each mode allowing algorithms to
operate at mode-n level. This advantage is more
convenient when the number of dimensions increases
(N ≥ 3). Additionally, we illustrate through computer
simulations using real-world data sets, how some
CS-related problems such as: three-dimensional (3D)
magnetic resonance imaging (MRI) reconstruction,
hyper-spectral imaging, and tensor completion are
greatly benefited by using underlying tensor models.

NOMENCLATURE AND NOTATION

In this paper, tensors are denoted by underlined
boldface capital letters, e.g. $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is a 3D
tensor of real numbers. For some applications, such
as in the case of MRI signals, we need to work
with tensors of complex numbers. It is noted that,
in order to simplify the notation, we present some
of the results only for the 2D or 3D cases but the
reader should notice that these results are also valid
for higher order tensors, i.e. with the a number of
dimensions $N \geq 3$. Matrices (2D arrays) are denoted
by bold uppercase letters and vectors are denoted
by boldface lower-case letters, e.g. $D \in \mathbb{R}^{M \times I}$ and $y \in \mathbb{R}^I$
are examples of a matrix and a vector, respectively.
The $j$-th entry of a vector $y$ is denoted by $y_j$, the
element $(i, j)$ of a matrix $Y$ is denoted by either of
the following ways $Y(i, j) = y_{i,j}$, and the $j$-th column
of matrix $Y$ is denoted by $y_j$. Similar notation is used
for tensors by referring to the element $(i_1, i_2, i_3)$ as
$Y(i_1, i_2, i_3) = y_{i_1i_2i_3}$. The Frobenius norm of a tensor
is defined by $\|Y\|_F = \sqrt{\sum_{i} \sum_{j} \sum_{k} y_{ijk}^2}$.

Sub-tensors (blocks) are formed when indices
are restricted to certain subsets of values. Particularly,
a mode-n fiber is defined as a vector obtained
by fixing every index to a single value except
in mode-n, e.g. in MATLAB notation, a mode-
2 fiber is obtained as $Y(i_1; i_2, i_3)$ (i.e. $i_1$ and $i_3$ are
fixed). More generally, by defining sets of restricted
indices $I_n = \{i_n^{(1)}, i_n^{(2)}, \ldots, i_n^{(K_n)}\}$, the corresponding 3D
dblock is denoted by $Y(I_1, I_2, I_3)$. For example,
if $I_1 = [1, 5, 7, 8], I_2 = [10, 11, 20]$ and $I_3 = [2, 3]$, then
$Y(I_1, I_2, I_3)$ determines a 4 × 3 × 2 block.

Mode-n unfolding (called also mode-n matricization) of a tensor
$Y \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ yields a matrix $Y(n)$
whose columns are the corresponding mode-n fibers
arranged in a specific order, 29 i.e. $Y(1) \in \mathbb{R}^{I_2 \times I_3}$,
$Y(2) \in \mathbb{R}^{I_1 \times I_3}$, and $Y(3) \in \mathbb{R}^{I_1 \times I_2}$. Given a multi-
dimensional signal (tensor) $Y \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ and a matrix
$A \in \mathbb{R}^{I_1 \times I_2}$, the mode-n tensor by matrix product
$Z = Y \times_n A \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_{n-1} \times I_{n+1} \ldots \times I_N}$ is defined by

$$z_{i_1i_2\ldots i_{n-1}i_{n+1}\ldots i_N} = \sum_{i_n=1}^{I_n} y_{i_1i_2\ldots i_{n-1}i_ni_{n+1}\ldots i_N},$$

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

The Kronecker product of matrices is defined as follows: given two matrices $A \in \mathbb{R}^{I \times M}$ and $B \in \mathbb{R}^{J \times N}$, their Kronecker product $A \otimes B \in \mathbb{R}^{IJ \times MN}$ is defined by

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1M}B \\ a_{21}B & a_{22}B & \cdots & a_{2M}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}B & a_{I2}B & \cdots & a_{IM}B \end{pmatrix}$$

(2)

Also, for given two matrices $A \in \mathbb{R}^{I_1 \times M}$ and $B \in \mathbb{R}^{I_2 \times M}$, their Kratri-Rao product $A \circ B \in \mathbb{R}^{I_1 \times I_2 \times M}$ is defined by applying the column-wise Kronecker product, i.e.

$$A \circ B = (a_1 \otimes b_1, a_2 \otimes b_2, \ldots, a_M \otimes b_M)$$

(3)

The outer product of $N$ vectors $v^{(n)} \in \mathbb{R}^{I_n}$
($n = 1, 2, \ldots, N$) is denoted by $Y = v^{(1)} \cdots v^{(N)}$.
where \( Y \in \mathbb{R}^{I_1 \times \cdots \times I_N} \) and is obtained by
\[
y_{1\cdots N} = v_{1_1}^{(1)} \cdots v_{N_1}^{(N)}.
\]

**COMRESSED SENSING (CS): DISCOVERING UNIQUE SPARSE STRUCTURES FROM MINIMAL INFORMATION**

Let us consider for the moment the space of ID signals (vectors) \( y \in \mathbb{R}^I \). Real world signals do not cover the space uniformly; instead, they are concentrated around the union of subspaces determined by linear combination of few dictionary elements.\(^{25,31}\) To be more specific, given a dictionary \( D \in \mathbb{R}^{I \times I} \), CS assumes the following signal model
\[
y = Dx + e_y, \quad \text{with } ||x||_0 \leq K,
\]
where \( K < I \), the \( l_0 \)-pseudo-norm \( ||x||_0 \) accounts for the number of nonzero entries of vector \( x \in \mathbb{R}^I \) and \( e_y \in \mathbb{R}^I \) is an error vector with a small norm, i.e. \( y \approx \hat{y} = Dx \). This remarkable property of real world signals made possible the development of compression techniques based on the transformation of a data set to some specific domain, typically a wavelet domain, where most of the coefficients can be set to zero allowing a reconstruction with a small loss of information. In this paper, we consider only complete dictionaries, i.e. where the number of atoms is equal to the signal size \( I \), which combined with Eq. 5, gives us the following system of constrained linear equations:
\[
z = Bx + e_z, \quad \text{with } ||x||_0 \leq K,
\]
where \( B = \Phi D \in \mathbb{R}^{M \times I} \) and the error term is determined by \( e_z = \Phi e_y \). Thus, two main questions need to be answered:

1. Under which conditions on matrices \( \Phi, D \), and the number of nonzero coefficients \( K \), a unique solution \( x \) exists?
2. How can we correctly estimate \( \hat{y} \) from \( z \)?

During last years, many advances have been made in order to answer these theoretical questions and understand the applicability of the CS theory to real world problems. Let us focus first on the ideal noiseless case, i.e. \( e_y = 0 \). The first question has to do with the uniqueness of the solution of Eq. 6. Note that, without the sparsity constraint, there exists an infinite number of vectors \( x \) for which \( z = Dx \) holds as there are more unknowns than equations. But, if there are two different coefficient vectors \( x_1, x_2 \) for which the same measure vector is obtained, i.e. \( z = Bx_1 = Bx_2 \), then those related signals cannot be identified uniquely. Fortunately, if the vector of coefficients \( x \) is sparse enough (small \( K \) compared to the size of the signal \( I \)) and if matrices \( \Phi, D \) are ‘good’ enough, then the solution is unique. There are several ways of characterizing a ‘good’ matrix \( B \), for example, \( B \) is considered ‘good’ if at least one of the following conditions are met:

- **Large spark**: \( \text{spark}(B) > 2K \),\(^{35}\) where the spark of a given matrix is the smallest number of columns that are linearly dependent.
- **Low coherence**: \( \mu(B) < 1/(2K - 1) \),\(^{35}\) where the coherence is defined as the largest normalized absolute inner product between any two columns, i.e. \( \mu(B) = \max_{i \neq j} \| b_i^T b_j \| / (\| b_i \|_2 \| b_j \|_2) \).

These results give us absolute guarantees about the uniqueness of the solution for the case of signals having exact sparse representations. It is important to note that, on one hand, dictionaries \( D \) are generally determined by the class of signals of interest and, on the other hand, usually we have some freedom to chose a proper sensing matrix \( \Phi \) such that, multiplied by matrix \( D \) would produce a good matrix \( B \). One remarkable result from this theory is that a random sensing matrix \( \Phi \) usually provides a sufficiently incoherent matrix \( B \).\(^{36}\)

The second question is related to the existence of algorithms able to recover the signal \( y \) from the measurement vector \( z \). The idea is to solve Eq. 6 for \( x \) and then to compute the approximate signal by using \( \hat{y} = Dx \). The problem of finding the sparsest solution of an under-determined system of linear equations can be formally formulated as follows:\(^{37}\)
\[
\arg\min_x ||x||_0 \text{ subject to } ||z - Bx||_2 \leq \epsilon,
\]
where \( \epsilon \) is a small constant determining the accuracy of the approximation.

Equation 7 is a combinatorial problem and would require an exhaustive search over all possible sparse supports of \( x \) which is not practical. Instead, several methods have been proposed to approximate this problem by another, a more tractable one. These methods can be basically divided into two main groups: basis pursuit (BP) and matching pursuit (MP).
• **Basis pursuit (BP) algorithms:** In BP, the combinatorial problem is transformed to a convex optimization problem by replacing the $\ell_0$-norm $\|x\|_0$ with its convex approximation, basically, the $\ell_1$-norm $\|x\|_1$.\textsuperscript{35,38} Then, the new optimization problem is formulated as follows:

$$\arg\min_{x} \|x\|_1 \text{ subject to } \|z - Bx\|_2 \leq \epsilon,$$  \hspace*{1cm} (8)

which is more tractable and can be solved, for example, by using linear programming (LP)\textsuperscript{39} or by using second-order cone programs (SOCPs).\textsuperscript{40} Both techniques are computationally expensive and turn out to be impractical for large sized signals, i.e., for signals with tens of thousands entries, as in the case of multidimensional signals. A more practical method is the spectral projected gradient $\ell_1$ minimization (SPGL1)\textsuperscript{41} algorithm whose optimized implementation in MATLAB is available in Ref 42. SPGL1 relies only on matrix–vector operations $Bx$ and $B^Ty$ and accepts both, explicit matrices or functions that evaluate these products and, therefore, becomes very useful for CS when matrices $B$ and $B^T$ have some kind of structure, as for example Kronecker structure.\textsuperscript{43} A general description of SPGL1 algorithm is included in Algorithm 1 and illustrated by an example in Figure 1(a).\textsuperscript{41}

**Algorithm 1: Spectral Projected Gradient $\ell_1$ (SPGL1)\textsuperscript{41}**

**Require:** Compressed signal $z \in \mathbb{R}^n$, matrix $B \in \mathbb{R}^{K \times I}$ and tolerance $\epsilon$

**Ensure:** Sparse vector of coefficients $x \in \mathbb{R}^n$

1: $r = z$; initial residual and $\ell_1$-norm bound
2: while $\|r\|_2 > \epsilon$ do
3: \hspace*{0.5cm} $x = \arg\min_{x} \|Bx - z\|_2$ subject to $\|x\|_1 \leq \tau$;
4: \hspace*{0.5cm} $r = z - Bx$; residual update
5: \hspace*{0.5cm} $\tau \leftarrow \text{Newton}\_\text{Update}(B, z, r)$; see details in\textsuperscript{41}
6: end while
7: return $x$;

• **Matching pursuit (MP) algorithms:** MP methods, which are also known as greedy algorithms,\textsuperscript{38,44} are faster than BP, especially with very sparse signals (low $K$).\textsuperscript{45} A powerful standard greedy algorithm is the orthogonal matching pursuit (OMP) which was optimized and studied in Refs 38 and 46. OMP iteratively refines a sparse solution by successively identifying the dictionary elements which are more correlated to the current residual and incorporating it to the support. This process is repeated until a desired sparsity level $K$ is reached or the approximation error is below some predetermined threshold level. A general description of OMP algorithm, for the case of a normalized matrix $B$, i.e. with unit-norm columns ($\|b\|_2 = 1$ for $j = 1, 2, ..., I$), is shown in Algorithm 2. An efficient MATLAB implementation of OMP can be found in Ref 47. In Figure 1(b) the recovery of a 1D signal having a sparse representation on a WT basis is shown by applying the OMP algorithm.

**Algorithm 2: Orthogonal Matching Pursuit (OMP)\textsuperscript{38,44,47}**

**Require:** Compressed signal $z \in \mathbb{R}^n$ and matrix $B \in \mathbb{R}^{K \times I}$

**Ensure:** set of $K$ non-zero coefficients (sparse representation)

1: $r = z$; initial residual
2: for $k = 1$ to $K$ do
3: \hspace*{0.5cm} $i_k = \arg\max_j |b_j^T r|$; select max. correlated column
4: \hspace*{0.5cm} $\hat{a} = \sum_{n=1}^{K} a_n b_{i_n}$; $a_n$ are such that $\|\hat{z} - z\|_2$ is minimized (see details in\textsuperscript{47})
5: \hspace*{0.5cm} $r = r - \hat{a} b_{i_k}$; residual update
6: end for
7: return $\{i_1, i_2, \ldots, i_K\}$

\textbf{Theoretical Aspects of Algorithms}

It is known that, in the ideal noiseless case, the condition $\mu(B) < 1/(2K - 1)$ is enough to guarantee that BP and MP algorithms will converge to the unique solution.\textsuperscript{38} However, in practical applications, signals are not truly sparse and there is noise involved in the sensing process. Thus, not only sparse signals should be uniquely determined in the noiseless case, but also they should be stably determined in the presence of noise. Formally, an algorithm is said to be stable if the error in the estimated vector of coefficients $\hat{x}$ is comparable to the model error, i.e. $\|\hat{x} - x\|_p \sim \|e_x\|_p$ for some $p$-norm. One way to guarantee stability is to ask the matrix $B$ to meet the restricted isometry property (RIP) which was introduced by Candès and Tao in Ref 48. A matrix $B$ satisfies the RIP of order $K$ if there exists a number $\delta_K \in (0, 1)$ such that

$$(1 - \delta_K) \|x\|_2^2 \leq \|Bx\|_2^2 \leq (1 + \delta_K) \|x\|_2^2,$$  \hspace*{1cm} (9)
Multidimensional compressed sensing

(a) Basis Pursuit (BP) based sparsity recovery (SPGL1 algorithm)

(b) Matching Pursuit (MP) based sparsity recovery (OMP algorithm)

FIGURE 1 | One-dimensional Compressed sensing (CS) example illustrating the application of the basis pursuit (BP) and matching pursuit (MP) sparsity recovery methods. A 1D signal $y \in \mathbb{R}^I$ is modeled by combining only $K = 5$ atoms (out of 256) of the Daubechies wavelet transform (WT) basis, i.e. $y = Dx + e$ with $|x|_0 = 5$. Nonzero coefficients and errors are generated by using a Gaussian distribution with standard deviation $\sigma = 1$ and $\sigma = 0.01$, respectively. The compressive measurements are obtained by $z = \Phi y$, with a sensing matrix $\Phi \in \mathbb{R}^{64 \times 256}$ being a random (Gaussian) matrix determining a sampling ratio = 25%. In (a), the sparsity patterns recovered by the SPGL1 algorithm in iteration $k = 1, 2, \ldots, 79$ are shown. In (b), the correlations between the residual and basic elements for iterations $k = 1, 2, \ldots, 5$ of the OMP algorithm are displayed.

for every $K$-sparse vector. RIP is directly related with the coherence of the matrix by $\delta_K \leq \mu(B)(K-1)^{35}$ thus small coherence imposes a small value of $\delta_K$. There are several available theoretical results about stability of algorithms with explicit error bounds. See for example Refs 46 and 49 for error bounds based on RIP in MP algorithms and$^{36,30}$ for the case of BP algorithms.

TENSOR DECOMPOSITIONS AND MULTI-WAY ANALYSIS

Many real world problems involve multidimensional signals, for instance, a 3D image produced by a computed tomography (CT) system or an MRI system, corresponds to a sampled version of a 3D function $f(x_1, x_2, x_3)$. In this case, the multidimensional image is stored in memory as a tensor $Y \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ whose elements are samples taken on a grid, i.e. $y_{i_1i_2i_3} = f(i_1h_1, i_2h_2, i_3h_3)$ ($i_n = 1, 2, \ldots, I_n, n = 1, 2, 3$) with $h$ being the discretization step for all dimensions. Models for tensors have a long history in mathematics and applied sciences.$^{28,29}$ Tensor decompositions allow us to approximate tensor data sets by models depending on few parameters, i.e. less parameters than the total number of entries of the tensor. This reduction of degree of freedom allows us to capture the essential structures in multidimensional data sets.

The Tucker decomposition$^{30,51}$ provides a powerful compressed format that exploits the linear structure of the unfolding matrices of a tensor simultaneously. More specifically, it is defined by the following multilinear expression:

$$ Y = X \times_1 D_1 \times_2 D_2 \cdots \times_N D_N, \quad (10) $$

with a core tensor $X \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_n}$ and factor matrices $D_n \in \mathbb{R}^{I_n \times R_n}$. It is easy to see that mode-$n$ vectors of a tensor with a Tucker representation
belongs to the span of the columns of matrix $D_{\alpha}$. A tensor $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is said to have multilinear rank-$(R_1, R_2, \ldots, R_N)$ if $R_1, R_2, \ldots, R_N$ is the set of minimal values for which Eq. 10 holds exactly.

As a particular case, when matrices $D_{\alpha} \in \mathbb{R}^{I_1 \times R_{\alpha}}$ are defined as the left singular vectors of matrices $Y_{(\alpha)}$, this model is called the high-order singular value decomposition (HOSVD). Additionally, by truncating the number of left singular vectors included in the factors we obtain the truncated HOSVD which is known to provide a suboptimal rank-$(R_1, R_2, \ldots, R_N)$ approximation $\hat{\mathbf{Y}}$ of the original tensor. In contrast to the matrix truncated SVD, which yields the best low-rank approximation, the tensor approximation resulting from the truncated HOSVD is usually not optimal. However, we have the following bound $||\mathbf{Y} - \hat{\mathbf{Y}}||_F \leq \sqrt{N}||\mathbf{Y} - \mathbf{Y}_{opt}||_F$.

This quasi-optimality condition is usually sufficient in most applications. The HOSVD has been thoroughly investigated in the literature and has found many applications in signal processing and data mining.

Another widely used tensor decomposition method is the canonical decomposition (CANDDECOMP) also known as parallel factor analysis (PARAFAC) jointly abbreviated CPD, which can be considered as a particular case of the Tucker model when the core tensor $\mathbf{X}$ is diagonal, i.e. $x_{i_1 \cdots i_N} \neq 0$ if and only if $i_1 = \cdots = i_N$. An important property of CPD model is that the restriction imposed on the Tucker core leads to uniqueness of the representation under mild assumptions. CPD model has been applied to a variety of problems, including telecommunications and sensor networks applications, biomedical applications, text mining analysis of social network and web links data sets, and others.

There are only a few recent works that link CS theory with tensor decompositions concepts. Lim et al. have shown in Ref 60 that the uniqueness property of the CPD model can be expressed in terms of CS concepts such as restricted isometry and incoherence applied to loading matrices (factors). In Refs 61 and 62, methods to recover low-rank tensors have been proposed based on convex optimization techniques adopted from CS concepts developed for matrices. Sidirooulos et al. have recently shown in Ref 63 that the identifiability properties of the CPD model could be used to recover a low-rank tensor from Kronecker measurements, i.e. by compressing each mode separately. More recently, in Refs 64 and 65 it was shown that multidimensional signals can be efficiently represented by means of the Tucker model with large but sparse core tensors with block structure, thus, they can be recovered from Kronecker compressive samples by using greedy and fast algorithms. In this paper, we consider Tucker model and its application to solve CS-related problems.

**PRACTICAL ALGORITHMS FOR MULTIDIMENSIONAL CS**

A direct application of the CS theory to multidimensional data sets consists of vectorizing tensors and use a large dictionary for their representation. To be more precise, given a tensor $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ we may convert it to a vector $\mathbf{y} = \text{vec}(\mathbf{Y}) \in \mathbb{R}^I$, with $I = I_1 I_2 I_3$, and assume that it admits a sparse representation over a dictionary $D \in \mathbb{R}^{I \times M}$, where $M < I$. Then, from a set of linear measurements given by $z = \Phi \mathbf{y}$ with $\Phi \in \mathbb{R}^{I \times M}$ we can apply any of the traditional algorithms such as MP or BP in order to recover the sparse structure and reconstruct the original multidimensional signal. This is quite impractical because even using the most optimized algorithms available today such as OMP or SPGL1, the size of the resulting explicit matrix $B = \Phi D$ scales exponentially with the number of dimensions $N$. Memory usage is not the only problem, also the number of operations involved in the matrix by vector products are very expensive when the size of the signal increases becoming prohibitive for $N \geq 2$. In real life applications, sometimes fast operators are available, which allows one to avoid using explicit versions of the dictionary $D \in \mathbb{R}^{I \times I}$, for instance, this is the case for the DCT and WT which cost $\Theta(I \log(I))$ instead of $\Theta(I^2)$. On the other side, most of existing theoretical results related to CS performance are based on random sensing matrices $\Phi \in \mathbb{R}^{M \times I}$ such as Gaussian or Bernoulli ones, but in practice, fast sensing operators with structured matrices need to be developed in order to avoid explicit matrices. While dictionaries $D$ are determined by the class of signals, sensing matrices $\Phi$ are determined by the specific application. For example, in the case of MRI signals, tomographic imaging, and optical microscopy, measurements are available in the Fourier transform domain, and the sensing operator usually consists on subsampling in this space. Thus the sensing matrix consists of a deterministic matrix given by a selection of a subset of rows in the Fourier transform matrix (see the application to CS MRI section) and the fast Fourier transform (FFT) algorithm can be used to implement it in a fast way. In the case of the channel estimation problem that arises in wireless communication applications, the sensing matrix is random Toeplitz.
or subsampled Toeplitz/circulant matrices for which recoverability guarantees results have been recently reported in Refs 70 and 71. As another example, CS single pixel cameras have been developed by designing special-purpose acquisition hardware that performs the projection of the scene against a binary array which correspond to use as sensing matrix with Walsh–Hadamard structure.4

The Sparse Tucker Tensor Representation

A natural way to exploit the multidimensional structure in signals is to avoid the vectorization step and take advantage of the structure contained in all different modes (dimensions) simultaneously. To this end, we can use the Tucker model of Eq. 10, where each of the factors captures the linear structure of the data set in the corresponding mode. In order to introduce this concept, a sparse Tucker representation of a 3D brain image is shown in Figure 2(a). The original data set $\mathbf{Y} \in \mathbb{R}^{256 \times 256 \times 64}$ is decomposed by using matrices $\mathbf{D}_1, \mathbf{D}_2 \in \mathbb{R}^{256 \times 256}$ and $\mathbf{D}_3 \in \mathbb{R}^{64 \times 64}$ as Tucker factors, i.e.

$$\mathbf{Y} = \mathbf{X} \times_1 \mathbf{D}_1 \times_2 \mathbf{D}_2 \times_3 \mathbf{D}_3. \quad (11)$$

When factor matrices are assumed to be orthogonal ($\mathbf{D}_n^T \mathbf{D}_n = \mathbf{I}$), the optimal core tensor, in the least squares sense, is obtained by

$$\mathbf{X} = \mathbf{Y} \times_1 \mathbf{D}_1^T \times_2 \mathbf{D}_2^T \times_3 \mathbf{D}_3^T. \quad (12)$$

Thus, the idea is to choose proper matrices $\mathbf{D}_n^T (n = 1, 2, 3)$ such that the resulting core tensor $\mathbf{X}$ is as sparse as possible. This is the case when we use sparsifying transforms in each mode, e.g. WT, DCT, or other specific operators according to the characteristics of the mode- $n$ vectors of the signal under analysis. In Figure 2(b), the corresponding sparse Tucker approximation is shown, which is obtained by keeping the largest absolute coefficients in the core tensor and setting the rest of them to zero. It is remarkable that, for example, by keeping only 5% of the coefficients we are able to approximate the original signal almost perfectly (with peak signal to noise ratio (PSNR) = 41 dB).6

It is straightforward to show that the Tucker representation of Eq. 11 can be equivalently expressed in terms of the vectorized tensors $\mathbf{x} = \text{vec} (\mathbf{X})$ and $\mathbf{y} = \text{vec} (\mathbf{Y})$, with $\text{vec} (\mathbf{Y}) = \text{vec} (\mathbf{Y}_{(1)})$, as follows65:

$$\mathbf{y} = (\mathbf{D}_3 \otimes \mathbf{D}_2 \otimes \mathbf{D}_1) \mathbf{x}. \quad (13)$$

Additionally, as it is discussed in section Selected Applications, in some cases the measurements can be taken in a multilinear way, i.e. by using linear operators in each mode separately as follows:

$$\mathbf{Z} = \mathbf{Y} \times_1 \mathbf{F}_1 \times_2 \mathbf{F}_2 \times_3 \mathbf{F}_3, \quad (14)$$

$$\mathbf{Z} = \mathbf{X} \times_1 \mathbf{D}_1 \times_2 \mathbf{D}_2 \times_3 \mathbf{D}_3, \quad (15)$$

which can be also written in terms of vectorized tensors as follows:

$$\mathbf{z} = (\mathbf{B}_3 \otimes \mathbf{B}_2 \otimes \mathbf{B}_1) \mathbf{x}, \text{ with } ||\mathbf{x}||_0 \leq K, \quad (16)$$

where $\mathbf{z} = \text{vec} (\mathbf{Z})$ and $\mathbf{B}_n = \mathbf{F}_n \mathbf{D}_n (n = 1, 2, 3)$ and $K$ is the maximum number of nonzero coefficients. In this case, to solve the CS problem with tensor structure data means to find the solution of a large underdetermined system of equations with Kronecker structure which can be solved by using traditional algorithms such as MP or BP applied to the corresponding vectorized tensors. The formulation of Eq. 16 is called as Kronecker CS72 and was recently exploited in the literature for the case of 2D signals (images) in Refs 43 and 73 and extended to multidimensional signals in Ref 72. In Ref 65 the Kronecker-OMP algorithm (Kron-OMP) was introduced, which allows one to perform the steps of the classical OMP exploiting the Kronecker structure of matrix $\mathbf{B} = (\mathbf{B}_3 \otimes \mathbf{B}_2 \otimes \mathbf{B}_1) \in \mathbb{R}^{M \times I}$, i.e. by working directly on the relatively small matrices $\mathbf{B}_n \in \mathbb{R}^{M_n \times I_n}$ (see Algorithm 3). In Figure 3(a) and (b), a 2D example is presented, showing how a signal having a Kronecker sparse representation can be recovered by the Kron-OMP algorithm. It is noted that, Kron-OMP needs $K$ iterations to complete the recovery ($K$ is equal to the number of nonzero entries in the sparse core matrix $\mathbf{X} \in \mathbb{R}^{I \times I}$).

Algorithm 3: Kron-OMP Algorithm65

Require: Compressed signal $\mathbf{Z} \in \mathbb{R}^{M_1 \times \cdots \times M_N}$ and normalized matrices $\mathbf{B}_n \in \mathbb{R}^{M_n \times I_n}$ ($n = 1, 2, \ldots, N$) Ensure: set of $K$ non-zero coefficients (sparse representation)

1: $\mathbf{R} = \mathbf{Z}$; initial tensor residual
2: for $k = 1$ to $K$ do
3: \begin{align*}
   &\left( i_1^{(k)}, i_2^{(k)}, \ldots, i_N^{(k)} \right) = \arg \max_{i_1, i_2, \ldots, i_N} |\mathbf{R} \times \mathbf{N}_{i_1, i_2, \ldots, i_N}|; \\
   &\mathbf{B}_1 (i_1), \mathbf{B}_2 (i_2), \ldots, \mathbf{B}_N (i_N) \text{; select max. entry of correlation tensor} \\
   &\hat{\mathbf{Z}} = \sum_{n=1}^{N} \alpha_n \mathbf{B}_n (i_1^{(n)}, i_2^{(n)}, \ldots, i_N^{(n)}); \quad \alpha_n \end{align*}
4: are such that $||\hat{\mathbf{Z}} - \mathbf{Z}||_F$ is minimized (see details in65)
5: $\mathbf{R} = \mathbf{Z} - \hat{\mathbf{Z}}$; residual update
6: end for
7: return $\left( i_1^{(k)}, i_2^{(k)}, \ldots, i_N^{(k)} \right)$, for $k = 1, 2, \ldots, K$ and $\{\alpha_1, \alpha_2, \ldots, \alpha_K\}$.
A Tucker representation of a tensor is obtained by applying orthogonal sparsifying transforms in each mode, e.g. Wavelet Transform (WT), Discrete Cosine Transform (DCT), etc.

(b) A sparse Tucker approximation is obtained by keeping the largest absolute coefficients (core tensor entries)

(c) Types of tensor sparsity

Unstructured multidimensional sparsity

Structured multidimensional sparsity (tensor block-sparsity)

Exploiting Multidimensional Sparsity Structure

For multidimensional data sets, the Kronecker structure of the dictionary and the sensing operator allows one to implement every iteration of the OMP strategy in an efficient way as it involves operations with smaller matrices. However, for large-scale data sets and high number of dimensions, the number of required iterations $K$ increases as a power of the number of dimensions (curse of dimensionality) making Kron-OMP impractical. In order to reduce complexity we can make additional assumptions about the location of nonzero coefficients based on some a priori knowledge about the signals of interest. By doing so we will be able to simplify the sparse pattern discovery algorithm and, at the same time, to improve the quality of reconstructions or reduce the number of compressive measurements. This brings us with the concept of ‘structured sparsity’ for vectors $y = Dx \in \mathbb{R}^I$ (as discussed in Ref 74 for the vector case and references therein), in which a subset of the...
(a) 2D signal kronecker sparse modelling

\[ Y \in \mathbb{R}^{32 \times 32} \]

(2D signal)

\[ D_1 \in \mathbb{R}^{32 \times 32} \]

(mode-1 dictionary)

\[ X \in \mathbb{R}^{32 \times 32} \]

(sparse core matrix)

\[ D_2^T \in \mathbb{R}^{32 \times 32} \]

(mode-2 dictionary)

\[ E \in \mathbb{R}^{32 \times 32} \]

(noise)

\[ Z \in \mathbb{R}^{16 \times 16} \]

(compressed measurement)

\[ \Phi_1 \in \mathbb{R}^{16 \times 32} \]

(mode-1 sensing matrix)

\[ \Phi_2^T \in \mathbb{R}^{32 \times 16} \]

(mode-2 sensing matrix)

(b) KRON-OMP ALGORITHM BASED SPARSITY RECOVERY

Absolute correlation between the residual and components in Kron-OMP iterations \(|b_j^T R b_j|\)

\( k = 1 \)

(14,20) detected

\( k = 2 \)

(1,1) detected

\( k = 3 \)

(1,2) detected

\( k = 4 \)

(25,25) detected

\( k = 5 \)

(10,15) detected

\( k = 6 \)

(10,20) detected

\( k = 7 \)

(5,3) detected

\( k = 8 \)

(6,4) detected

\( k = 9 \)

(20,20) detected

**FIGURE 3** | Illustration of a two-dimensional (2D) signal having a Kronecker sparse representation and its recovery by the Kron-OMP algorithm. In (a), a 32 × 32 patch image is modeled as a linear combination of only \( K = 9 \) atoms (out of 1024) of the separable discrete cosine transform (DCT) basis, i.e. \( Y = D_1 X D_2^T + E \) with \( ||X||_0 = 9 \). Nonzero coefficients and errors are generated by using a Gaussian distribution with standard deviation \( \sigma = 1 \) and \( \sigma = 0.01 \), respectively. Compressive measurements are taken by multiplying each mode by a random (Gaussian) sensing matrices, i.e. \( Z = \Phi_1 Y \Phi_2^T \) with \( \Phi_{1,2} \in \mathbb{R}^{16 \times 32} \) (sampling ratio = 25%). In (b), the successive correlations of the residual with the Kronecker basis \( B_2 \otimes B_1 \) are shown for each iteration of the Kron-OMP algorithm. The sparsity pattern is correctly recovered after exactly \( K = 9 \) iterations.

Sparse solutions \( x \in \mathbb{R}^I \) are allowed and others are discarded. For example, it is known that piecewise smooth signals and images tend to live on a rooted, connected ‘tree structure’.\(^{75}\) Moreover, usually large coefficients are clustered together into blocks in the 1D vector \( x \in \mathbb{R}^I \).\(^{76,77}\)

For the case of tensors having a Kronecker sparse representation, i.e. \( Y = X \times_1 D_1 \times_2 D_2 \times_3 \)
$D_3 \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ ($X \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, $D_n \in \mathbb{R}^{I_n \times I_n}$) a simple and natural sparsity structure is to assume that most significant coefficients are concentrated on a sub-tensor of $X$\textsuperscript{65}. More specifically, ‘tensor block-sparsity’ assumes that $x_{i_1i_2i_3} = 0$ if $(i_1,i_2,i_3) \notin (I_1, I_2, I_3)$ where $I_n (n = 1, 2, 3)$ are subsets of indices in each mode determining sub-tensor. Figure 2(c) illustrates the concepts of unstructured and block-structured sparsity of core tensors.

When the ‘tensor block-sparsity’ assumption is approximately valid, a super fast greedy algorithm, namely the N-way block OMP (NBOMP algorithm), can be used requiring substantially fewer iterations than Kron-OMP (see comparison in the following section). In Figure 4, the NBOMP algorithm is illustrated through an example. A $32 \times 32$ section. In Figure 4, the NBOMP algorithm is illustrated through an example. A $32 \times 32$ patch having a block sparse representation with $K = 9$ atoms contained in a $3 \times 3$ block is recovered from compressive measurements taken in the same way as in the example of Figure 3. It is highlighted that NBOMP recovers the correct sparse pattern after exactly 3 iterations instead of 9 iterations that would be required the OMP (unstructured) algorithm.

Algorithm 4: NBOMP Algorithm\textsuperscript{65}

**Require**: Compressed signal $Z \in \mathbb{R}^{M_1 \times \ldots \times M_N}$, normalized matrices $B_n \in \mathbb{R}^{M_n \times I_n}$ ($n = 1, 2, \ldots, N$) and threshold $\epsilon$

**Ensure**: set of non-zero coefficients within a sub-tensor of size $S_1 \times S_2 \times \ldots \times S_N$ (block-sparse representation)

1: $R = Z$; initial tensor residual
2: $I_n = [], n = 1, 2, \ldots, N$; initial set of indices in each mode
3: $k = 1$;
4: while $\|R\|_F > \epsilon$ do
5: \hspace{1em} $(i_1^{(k)}, i_2^{(k)}, \ldots, i_N^{(k)}) = \arg\max_{i_1, i_2, \ldots, i_N} |R|_{X_N} B_1(:, i_1^T) x_{2} \cdots x_{N} B_N(:, i_N^T)$; select max. entry of correlation tensor
6: \hspace{1em} $I_n = I_n \cup i_n^{(k)}, n = 1, 2, \ldots, N$; increase subtensor support
7: \hspace{1em} $\hat{Z} = G x_{1} B_1(:, I_1) x_{2} \cdots x_{N} B_N(:, I_N)$; $G \in \mathbb{R}^{K_1 \times K_2 \cdots \times K_N}$ is such that $\|\hat{Z} - Z\|_F$ is minimized (see details in Ref 65)
8: \hspace{1em} $R = Z - \hat{Z}$; residual update
9: \hspace{1em} $k = k + 1$
10: end while
11: $-I_1, I_2, \ldots, I_N$ and $G \in \mathbb{R}^{K_1 \times K_2 \cdots \times K_N}$

MULTIDIMENSIONAL CS ALGORITHMS COMPARISON

In this section we demonstrate the advantage of using the Tucker model based tensor representations, i.e. by using Kronecker bases, and block sparsity compared to the case of using classical vectorized CS algorithms for tensor data sets regarding memory usage, complexity, and quality of reconstructions.

Memory Usage

In Figure 5(a) the memory requirements to store the resulting explicit matrix $B$ for 1D, 2D, and 3D signals (tensors) for the case of having a typical $M_n = I_n/4$ are shown. Note that, with 16 GB of available RAM memory, the dictionary for a 2D signal with a size of only $420 \times 420$ can be stored and for the 3D case it corresponds to a tensor with size $70 \times 70 \times 70$.

Computational Cost

A distinctive characteristic of OMP algorithm is that it requires a number of iterations equal to the sparsity $K$ of the signal to converge to the desire solution, as one coefficient is detected per iteration. On the other hand, BP algorithms usually require much more iterations because all coefficients are updated in every iteration and the solution converges, sometimes slowly, to coefficients close to zero for those which should be exactly zero. For this reason, sometimes OMP is preferred over BP specially when the number of nonzero coefficients is small.\textsuperscript{38} Let us consider the case of the recovery of $N$-dimensional tensors from the measurements given by $Z \in \mathbb{R}^{M \times M \times \ldots \times M}$ having a block sparse representation $Z = X \times_1 B_1 \times_2 B_2 \cdots \times_N B_N$ with factors matrices $B_n \in \mathbb{R}^{M \times I_n}$ ($M < I_n$) and a $(S_1, S_2, \ldots, S_N)$ block-sparse core tensor $X \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$, i.e. with nonzero coefficients concentrated in a $S \times S \times \cdots \times S$ subtensor. The asymptotic complexities of the vectorized OMP, Kronecker-OMP, and NBOMP algorithms for large $I$, $M$, and small $S$ ($S \ll M < I$) are given in Table 1. The main reason for the fast convergence of NBOMP is that it requires a number of iterations bounded by $S \leq K \leq NS$, i.e linear in $S$ which is significantly lower than other OMP algorithms that require $SN$ iterations (polynomial of order $N$). In order to illustrate the complexity of classical BP, MP algorithms and their tensor versions, which takes into account the Kronecker structure (Kronecker-SPGL1 and Kronecker-OMP) and the block-sparsity of signals (NBOMP), we have simulated measurements $Z \in \mathbb{R}^{M \times M \times M}$ by using
(a) 2D signal kronecker block-sparse modelling

\[ Y \in \mathbb{R}^{32 \times 32} \quad \text{(2D signal)} \]
\[ D_1 \in \mathbb{R}^{32 \times 32} \quad \text{(mode-1 dictionary)} \]
\[ X \in \mathbb{R}^{32 \times 32} \quad \text{(block-sparse)} \]
\[ D_2^T \in \mathbb{R}^{32 \times 32} \quad \text{(mode-2 dictionary)} \]

\[ Z \in \mathbb{R}^{16 \times 16} \quad \text{(compressed measurement)} \]
\[ \Phi_1 \in \mathbb{R}^{16 \times 32} \quad \text{(mode-1 sensing matrix)} \]
\[ Y \in \mathbb{R}^{32 \times 32} \quad \text{(2D signal)} \]
\[ \Phi_2^T \in \mathbb{R}^{32 \times 16} \quad \text{(mode-2 sensing matrix)} \]

(b) Block-sparsity recovery by using the nbomp algorithm

Absolute correlation between the residual and components in NBOMP iterations (\(|b_i^T R b_j|\))

\[ k = 1 \]
\[ (5,12) \text{ detected: } I_1 = [5] \]
\[ I_2 = [12] \]

\[ k = 2 \]
\[ (2,1) \text{ detected: } I_1 = [5,2] \]
\[ I_2 = [12,1] \]

\[ k = 3 \]
\[ (15,20) \text{ detected: } I_1 = [5,2,15] \]
\[ I_2 = [12,1,20] \]

**FIGURE 4** | Illustration of a two-dimensional (2D) signal having a Kronecker block-sparse representation and their recovery by the NBOMP algorithm. In (a), a $32 \times 32$ patch image is modeled as a linear combination of only $K = 9$ atoms (out of 1024) of the separable DCT basis, i.e. $Y = D_1 X D_2^T + E$ with the nonzero entries coefficients concentrated in a $3 \times 3$ block. Nonzero coefficients and errors are generated by using a Gaussian distribution with standard deviation $\sigma = 1$ and $\sigma = 0.01$, respectively. Compressive measurements are taken by multiplying each mode by a random (Gaussian) sensing matrices, i.e. $Z = \Phi_1 Y \Phi_2^T$ with $\Phi_{1,2} \in \mathbb{R}^{16 \times 32}$ (sampling ratio = 25%). In (b), the successive correlations of the residual with the Kronecker basis $B_1 \otimes B_1$ are shown for each iteration of the N-way block orthogonal matching pursuit (NBOMP) algorithm and detected indices are incorporated into the block-support of the signal. The sparsity pattern is correctly recovered after exactly $K = 3$ iterations.

block-sparse signals with the 3D-DCT dictionary and sampling them through Kronecker random measurement matrices. In Figure 5(b) the computation time versus the sampling ratio $M^3/I^3$ is shown where the advantage of NBOMP over the rest of the methods is clear.

**Quality of Reconstructions**

Another important characteristic of NBOMP is that it is able to recover correctly block-sparse signals with higher probability compared with the other methods as Figure 5(c) shows. In fact, in Ref 65 it was demonstrated that a sufficient condition for
FIGURE 5 | (a) Memory requirements in order to storage a explicit dictionary for 1D, 2D, and 3D signals (tensors) for the case of having $M_n = l_n/4$ ($N = 1, 2, 3$). (b) Comparison of computation times versus sampling ratio $M^3/I^3$ for vectorized and tensor compressed sensing (CS) algorithms with $(32 \times 32 \times 32)$ signals having a $(4 \times 4 \times 4)$-block sparse representation with a Kronecker discrete cosine transform (DCT) dictionary. (c) Percentage of correctly recovered tensors over a total of 100 simulations with $(32 \times 32 \times 32)$ signals having a $(4 \times 4 \times 4)$-block sparse representation with a Kronecker DCT dictionary.

NBOMP to recover correctly a block sparse signals is given by

$$S^N < 2 - (1 + (S - 1) \mu_T)^N,$$  \hspace{1cm} (17)

where $\mu_T = \max\{\mu(B_1), \mu(B_2), ..., \mu(B_N)\}$. It is noted that this performance guarantee is better than the available result for OMP and BP algorithms which is$^{65,72,78}$

$$S^N < \frac{1}{2} \left( 1 + \frac{1}{\mu_T} \right).$$ \hspace{1cm} (18)

In Figure 5(c) the percentage of correctly recovered tensors over a total of 100 simulations with $(32 \times 32 \times 32)$ signals having a $(4 \times 4 \times 4)$-block sparse representation with a Kronecker DCT dictionary is shown where the advantage of NBOMP over the rest of the methods is clear.

ON THE PROPER CHOICE OF A DICTIONARY

An important task in the field of CS is to choose a proper dictionary $D$ on which signals have as sparse as possible representations and, at the same time, it is incoherent with respect of the sampling scheme given by the matrix $\Phi$. To choose a proper
Mathematical Model-Based Dictionaries

The Fourier dictionary allows to describe a signal in terms of its global frequency content. This is efficient for uniformly smooth signals but signal discontinuities generate large coefficients over all frequencies. A close related dictionary is the DCT which has found close related dictionary is the DCT which has found to efficiently compress small patches, typically $8 \times 8$ size, in 2D images. But, DCT still has the problem of nonlocalization and cannot represent discontinuities efficiently. Motivated by this problem, the Gabor transform, also known as short time Fourier transform (STFT) became more popular for representation of real-world signals as it gives the optimal time and frequency localization which is limited by the celebrated uncertainty principle (Gabor limit). This principle states that one cannot simultaneously localize a signal in both the time domain and frequency domain (Fourier transform).

Modern mathematical models for signal representations are based on dictionaries whose atoms are translated and dilated versions of a single elementary waveform known as "wavelet". WT have revolutionized signal processing, and they have shown to provide effective signal representations. Very fast algorithms were developed on the basis of the idea of multi-resolution which states that a signal can be represented as a series of difference signals where each one correspond to a different scale. WT was incorporated into the JPEG2000 compression standard replacing the former DCT dictionary. There are also extensions of WT to higher number of dimensions that efficiently represent (sparsely) piecewise smooth images such as the case of 'wavelet' introduced by Donoho in Ref 79 for 2D signals and 'ridgelets' proposed by Candès et al. in Ref 80 for higher number of dimensions. Other mathematical models for WT of 2D images where developed by various authors as for example: 'curvelets', 'contourlets', 'bandelets', and others.

## Dictionary Learning

A more recent approach to choose a dictionary for sparse representations is to train it based on a large set of data samples which is known as 'dictionary learning'.

### Mathematical Model-Based Dictionaries

#### Table 1: Complexity of OMP Algorithms Applied to an N-Dimensional Tensor $\mathbf{Z} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with Factor Matrices $\mathbf{B}_i \in \mathbb{R}^{I_i \times M}$ and the Block Sparsity Parameter $S (S \ll I < M)$

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Asymptotical cost per iteration</th>
<th>Number of Iteration</th>
<th>Total cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vectorized-OMP</td>
<td>$(IM)^N$</td>
<td>$S^N$</td>
<td>$(SIM)^N$</td>
</tr>
<tr>
<td>Kronecker-OMP</td>
<td>$I(M)^N$</td>
<td>$S^N$</td>
<td>$I(SM)^N$</td>
</tr>
<tr>
<td>NBOMP</td>
<td>$I(M)^N$</td>
<td>$\leq NS$</td>
<td>$INS(M)^N$</td>
</tr>
</tbody>
</table>

This problem can be formulated as follows:

$$\text{arg min}_{\mathbf{D}, \mathbf{X}} \| \mathbf{DX} - \mathbf{Y} \|_F \quad \text{s.t.} \quad \| \mathbf{x}^{(t)} \|_0 \leq K;$$

(19)

where matrix $\mathbf{X}$ contains in its columns the vector coefficients for the $t$-th signal sample, i.e. $\mathbf{y}^{(t)} = \mathbf{Dx}^{(t)}$ and $\mathbf{Y} = [\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(T)}]$. This optimization problem is combinatorial and highly non-convex and thus existing algorithms search for local minima by alternating the minimization with respect to $\mathbf{D}$ and $\mathbf{X}$. This setting was first proposed as the method of optimal directions (MOD) in Ref 86 and the same problem was also approached and solved by the k-means singular value decomposition (KSVD) algorithm in a much efficient way.33

KSVD has been successfully applied to model small patches on images, but it turns out computationally expensive as the size of the patches becomes larger and its application to higher number of dimensions ($N \geq 3$) becomes prohibitive. To reduce the computation complexity, it is necessary to use some structure on dictionaries in order to alleviate the dictionary learning task. For example, in Ref 87 the ‘double sparsity’ strategy was proposed where the dictionary is assumed to have itself a sparse Tucker representations of the tensor data set.

Here we propose a simple algorithm that allows us to adapt Kronecker dictionaries to a given set of tensor data samples. Suppose that we have at our disposal a set of $T$ tensors $\mathbf{Y}^{(t)} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$ ($t = 1, 2, \ldots, T$) so we would like to build a set of matrices $\mathbf{D}_i \in \mathbb{R}^{I_i \times I_n}$ such that they provide good sparse Tucker representations of the tensor data set. Thus, the objective is to minimize the global error, i.e.

$$\text{arg min}_{\mathbf{D}_i, \mathbf{G}^{(t)}} \left( \sum_{t=1}^{T} \| \mathbf{Y}^{(t)} - \mathbf{G}^{(t)} \times_1 \mathbf{D}_1 \cdots \times_N \mathbf{D}_N \|_F^2 \right)$$

(20)

s.t. $\| \mathbf{G}^{(t)} \|_0 \leq K,$
for a predefined sparsity level $K$. We propose here an alternate least squares method by updating one factor matrix $D_n$ at a time and considering the rest of the variables fixed. It is noted that the global error of Eq. 20 can be written in terms of the concatenation of tensors in a particular mode-$n$ as follows:

$$
\sum_{i=1}^{T} \| \mathbf{Y}^{(i)} - \mathbf{G}^{(i)} \times_1 \mathbf{D}_1 \cdots \times_N \mathbf{D}_N \|_F^2 = (21)
$$

$$
\| \left[ \mathbf{Y}^{(1)}(n) \cdots \mathbf{Y}^{(n)}(n) \right] - \left[ \mathbf{D}_n \mathbf{G}^{(1)}(n) \mathbf{C}_n \cdots \mathbf{D}_n \mathbf{G}^{(T)}(n) \mathbf{C}_n(1) \right] \|_F^2 = (22)
$$

$$
= \| \left[ \mathbf{Y}^{(1)}(n) \cdots \mathbf{Y}^{(n)}(n) \right] - \mathbf{D}_n \left[ \mathbf{G}^{(1)}(n) \mathbf{C}_n \cdots \mathbf{G}^{(T)}(n) \mathbf{C}_n(1) \right] \|_F^2 = (23)
$$

where $\mathbf{C}_n = (\mathbf{D}_N \otimes \cdots \otimes \mathbf{D}_{n+1} \mathbf{D}_{n-1} \cdots \otimes \mathbf{D}_1)^T$. Then the least squares solution for matrix $D_n$ is given by

$$
\mathbf{D}_n = \left[ \mathbf{Y}^{(1)}(n) \cdots \mathbf{Y}^{(n)}(n) \right] \left[ \mathbf{G}^{(1)}(n) \mathbf{C}_n \cdots \mathbf{G}^{(T)}(n) \mathbf{C}_n(1) \right]^\dagger \quad (24)
$$

with $\dagger$ being defined as the Moore-Penrose pseudo-inverse computed as $\mathbf{A}^\dagger = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T$. The proposed Kronecker dictionary algorithm is shown in Algorithm 5.

In order to illustrate the algorithm behavior we have applied it to a set of 10,000 (8 x 8)-patches randomly selected from two different images: ‘Barbara’ and ‘Text’ shown in Figure 6(a). A sparsity level of $K=6$ was used and initial factor matrices $\mathbf{D}_1$ and $\mathbf{D}_2$ were generated by using independent Gaussian numbers. The obtained dictionaries are shown together with the classical Kronecker DCT and WT (Daubechies) dictionaries for a visual comparison. In Figure 6(b), the evolution of the global error per pixel, defined by

$$
E_G = \frac{1}{Ns \times 8 \times 8} \sqrt{\sum_{i=1}^{T} \| \mathbf{Y}^{(i)} - \mathbf{G}^{(i)} \times_1 \mathbf{D}_1 \cdots \times_N \mathbf{D}_N \|_F^2}, \quad (25)
$$

is shown for both cases. Global errors per pixel for the case of DCT and WT (Daubechies) dictionaries are shown for reference. It is interesting to note that, the WT is better (lower error) than DCT for the case of ‘Text’ while for the case of ‘Barbara’ DCT is better than WT. It is highlighted that the obtained dictionaries improve considerably the global errors capturing common structures of the datasets.

### Algorithm 5: Kronecker Dictionary Learning

**Require**: Set of $T$ tensors $\mathbf{Y}^{(1)}, \mathbf{Y}^{(2)}, \ldots, \mathbf{Y}^{(T)}$, sparsity level $K$ and threshold $\epsilon$

**Ensure**: matrices $\mathbf{D}_n$ ($n=1, 2, \ldots, N$) (unit-norm columns)

1: Initialize factor matrices $\mathbf{D}_n \in \mathbb{R}^{K \times \cdots \times \mathbb{R}^{T \times \cdots \times \mathbb{R}^{N \times \cdots \times \mathbb{R}^{L \times \cdots \times \mathbb{R^{1}}}}}$ randomly;

2: $e = \sum_{i=1}^{T} \| \mathbf{Y}^{(i)} - \mathbf{G}^{(i)} \times_1 \mathbf{D}_1 \cdots \times_N \mathbf{D}_N \|_F^2$;

3: $k = 1$, $\Delta = \infty$;

4: while $\Delta > \epsilon$ do

5: for $n = 1$ to $N$ do

6: $\mathbf{D}_n = \left[ \mathbf{Y}^{(1)}(n) \cdots \mathbf{Y}^{(T)}(n) \right] \left[ \mathbf{G}^{(1)}(n) \mathbf{C}_n \cdots \mathbf{G}^{(T)}(n) \mathbf{C}_n(1) \right]^\dagger$;

7: Update matrix factor in mode-$n$

8: Normalize matrices factors $\mathbf{D}_n$

9: Update sparse core tensors $\mathbf{G}^{(i)} \in \mathbb{R}^{K \times \cdots \times \mathbb{R}^{T \times \cdots \times \mathbb{R}^{N \times \cdots \times \mathbb{R}^{L \times \cdots \times \mathbb{R^{1}}}}}$ using vectorized-OMP, Kronecker-OMP or NBOMP algorithm

10: $e^* = \sum_{i=1}^{T} \| \mathbf{Y}^{(i)} - \mathbf{G}^{(i)} \times_1 \mathbf{D}_1 \cdots \times_N \mathbf{D}_N \|_F^2$;

11: $\Delta = ||e-e^*||$; Compute error change

12: $k = k+1$; $e = e^*$;

13: end while

14: return $\mathbf{D}_n$ ($n=1, 2, \ldots, N$);

### SELECTED APPLICATIONS

In this section, we present multiway CS models for selected applications and illustrate through computer simulations using real world multidimensional data sets. All the results were performed using MATLAB software on a Mac Book Pro notebook, equipped with an Intel Core i7 processor (2.2 GHz) and 8 GB RAM. Selected MATLAB codes are provided through our personal website (see Further Reading section).

### CS Magnetic Resonance Imaging (MRI)

MRI and computer tomographic (CT) technologies have motivated the development of CS theory since Candès et al. have demonstrated that structured signals can be recovered from incomplete Fourier samples. CS MRI has matured during last years providing us today with a real application of CS...
FIGURE 6 | Analysis of the application of the Kronecker dictionary learning algorithm to (8 × 8)-patches taken from ’Barbara’ and ’Text’ images. A total number of $N_s = 10,000$ patches are selected randomly from each of these datasets. Kronecker dictionaries are obtained by applying Algorithm 5 using Gaussian initial matrices $D_1$ and $D_2$. The global error per pixel is defined as $E_G$. A sparsity level of $K = 6$ was used.

theory to an essential medical imaging tool allowing to reduce the acquisition data process (see Ref 69 for an excellent tutorial on this topic).

In MRI, the acquisition process consists on taking samples in the $k$-space, which is essentially the 3D Fourier space (complex domain). In classical MRI, the $k$-space sampling pattern is designed to meet the Nyquist criterion, which determines the resolution of the image. In this case, the reconstruction simply consists on the application of the inverse Fourier transform of the measurements which is called as the minimum energy (ME) reconstruction.
acquisition in 3D is usually performed by collecting samples of different 2D slices. This multi-slice acquisition usually takes considerable time, making patients to be uncomfortable and increasing the cost of health care. The idea of CS MRI is to considerably reduce the amount of samples required in the 2D slices of the k-space in order to decrease the scan time without losing quality of the image. There has been great advances in signal processing techniques applied to 3D MRI reconstructions by developing sophisticated trajectories in the Fourier domain (k-space) and taking into account physical constraints of the MRI acquisition system, see for example Refs 88–92. It is worth to mention that another close related application of CS is dynamic MRI, where the tensor data set consists of a sequence of 2D slices acquired at different times. In the latter case the reduction of k-space measurements is mandatory in order to increase time resolution.93–95 These techniques are usually computationally expensive and their application to large data sets are sometimes prohibitive. Here we show that CS MRI with multi-slice scanning perfectly fits the Kronecker sensing scheme allowing us to apply fast algorithms that exploit the Kronecker structure.

Here, we assume that the 3D image \( Y \in \mathbb{R}^{I_1 \times I_2 \times I_3} \) can be approximated by a sparse Tucker representation using, for example, the WT Daubechies dictionaries \( D_n \) \((n = 1, 2, 3)\) or any other transform that efficiently sparsifies the data set (Eq. 11). Then the MRI device will take measurements in the k-space given by the 3D-Fourier transform \( W \in \mathbb{C}^{I_1 \times I_2 \times I_3} \):

\[
W = Y \times_1 T_1 \times_2 T_2 \times_3 T_3,
\]

(26)

where \( T_n \in \mathbb{C}^{I_n \times I_n} \) \((n = 1, 2, 3)\) are the matrices associated to the Fourier transform (see Figure 7(a)). In the MRI multi-slice mode the samples are taken by selecting a unique 2D pattern for every slice. More specifically, the sampling process will correspond to the following:

\[
Z = W \times_1 H_1 \times_2 H_2 \times_3 H_3,
\]

(27)

where \( H_1 \in \mathbb{R}^{M_1 \times I_1} \) and \( H_2 \in \mathbb{R}^{M_2 \times I_2} \) are subsampling matrices obtained by keeping selected rows of the identity matrix and \( H_3 \) is just the identity matrix. Putting Eqs 11, 26, and 27 together we obtain:

\[
Z = X \times_1 \Phi_1 \times_2 \Phi_2 \times_3 \Phi_3,
\]

(28)

with \( \Phi_n = H_n T_n D_n \).

The objective of CS MRI is to compute the sparse core tensor \( X \) compatible with the measurements \( Z \) and this can be performed efficiently by using a BP or MP algorithm that exploits the Kronecker structure.

In Figure 7(b) we show the results of recovering a real 256 \( \times \) 256 \( \times \) 64 MRI brain image. In order to simulate the k-space measurements we first compute the 3D Fourier transform according to Eq. 26. Using \( H_3 \) equal to the identity matrix implies that entire mode-3 fibers are selected in the k-space and their positions are determined by the selected indices in the modes 1 and 2. In this example, we have selected the indices in order to cover a block in the central part (low frequency content) and have selected the rest of the indices randomly as illustrated in Figure 7(a). By applying the 3D tensor reconstruction based on the Kronecker-BP (SPGL1 algorithm), with a sampling ratio \( (M_1 M_2 M_3)/(I_1 I_2 I_3) = 70\% \) (\( M_1 = M_2 = 214 \) and \( M_3 = 64 \)), we obtained a PSNR = 42.7 dB (Figure 7(b)—4th column).

It is important to highlight that most of recent methods for 3D CS are based on the idea of processing of 2D slices one by one using classical CS algorithms as for example the technique presented in Ref 69. In Figure 7(b)—3rd column, we present the results of applying a slice by slice reconstruction using the 2D Kronecker-BP (SPGL1 algorithm) to every subsampled frontal slice of the k-space with a sampling ratio of 70\%. The obtained reconstruction has a PSNR = 40.8 dB which is lower than the case of using a full 3D Kronecker-BP because the latter takes into account the structure of the signal in mode-3 too. For reference, the ME reconstruction is also shown, which consists of applying the inverse 3D Fourier transform of the k-space with the unavailable samples set to zero, giving a PSNR = 37.5 dB (see Figure 7(b)—2nd column). It is highlighted that the CS approach allows us to avoid the aliasing effect that appears when the classical Fourier reconstruction (ME) is applied with a sampling scheme that violates the Nyquist principle.

### CS Hyper-Spectral Imaging

A direct application of CS theory is the development of new image acquisition devices such that, by exploiting the compressibility property of signals, they can considerably reduce the number of required samples allowing a cheaper and faster imaging technique called compressive sampling imaging (CSI). Real application examples are the single-pixel cameras developed in Refs 4 and 5, where a digital micro mirror array device (DMD) is used whose orientations are modulated randomly obeying to the Walsh-Hadamard or the Noiselet patterns.96 Other CSI proposed devices are the one-shot with random phase mask camera,6 the random lens technique,7 the linear sensors based method,97 and the coded aperture imaging.8 These imaging techniques allow one to implement the...
(a) Kronecker sampling of a 3D tensor: indices are randomly selected in mode-1 and mode-2 of the k-space (Fourier domain)

Subsampled k-space

\[ (M_1 \times M_2 \times M_3) \]
\[ M_1 < 256 \]
\[ M_2 < 256 \]
\[ M_3 = 64 \]

k-space (Fourier domain)

\[ (256 \times 256 \times 64) \]

\[ Y = \begin{bmatrix} T_1 & T_2 & T_3 \end{bmatrix} \]
\[ (256 \times 256) \]
\[ (64 \times 64) \]

(b) Recovery of a 3D MRI brain image from the subsampled k-space (sampling ratio = 70%). 3D tensors (top), (100 x 100)-zoom of slice 44 (middle) and absolute error (bottom) are shown.

Original

(256 x 256 x 64)

Minimum energy reconstruction

PSNR = 37.5dB

Slice by slice reconstruction

(2D kronecker-BP)

PSNR = 40.8 dB

Tensor reconstruction

(3D kronecker-BP)

PSNR = 42.7 dB

Absolute errors

FIGURE 7 | (a) Illustration of Kronecker-compressed sensing (CS) applied to a three-dimensional (3D) magnetic resonance image (MRI). (b) Reconstructions using the full 3D Kronecker-basis pursuit (BP) algorithm (PSNR = 42.7 dB), the slice by slice 2D Kronecker-BP algorithm (PSNR = 40.8 dB) and the minimum energy method (PSNR = 37.5 dB) using compressive measurements with a sampling ratio of \((M_1 M_2 M_3) / (I_1 I_2 I_3) = 70%\) and assuming sparse representations based on the separable 3D Daubechies wavelet transform (WT) basis.

Hardware that is capable to perform projections of the incident light field (corresponding to the desire image) against a class of sensing vectors.

However, the problem of CSI is that the random measurements as well as the reconstruction methods become impractical because of the large amount of data when medium to large images are considered. To alleviate this problem, the Kronecker structure of the sensing operators and dictionaries could help us significantly. Separable operators, i.e. with Kronecker structure, arise naturally in many optical implementations. In Ref 43 the authors proposed a practical CS imaging based on separable imaging operators and, in Ref 73 a CMOS-based hardware was proposed. In Refs 72 and 78, theoretical guarantees for CS with Kronecker structured matrices were provided,
and optimized greedy algorithms were developed in Refs 64 and 65 taking into account the Kronecker structure and block-sparsity.

A specific application of CS using the Kronecker structure is hyper-spectral compressive imaging (HCI) which consists of the acquisition of 2D images at several spectral bands providing a 3D signal where each slice corresponds to a different channel.72 Here, a 2D-separable operator is applied to the hyperspectral light field which means that each spectral band’s image is multiplexed by the same sensing operator simultaneously. The resulting multilinear measurement $\mathbf{Z} \in \mathbb{R}^{M_1 \times M_2 \times M_3}$ of the hyper-spectral data-cube $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is given by Eq. 14, where matrices $\Phi_1 \in \mathbb{R}^{M_1 \times I_1}$ and $\Phi_2 \in \mathbb{R}^{M_2 \times I_2}$ determine the separable sensing operator applied to each 2D slice and matrix $\Phi_3$ is the identity matrix $(I_3 \times I_3)$.

When dealing with large data sets in many dimensions ($N \geq 2$), OMP and SPGL1 (BP) algorithms become prohibitively expensive or intractable and a super computer must be used. However, if the data set can be represented by block-sparse representations then the reconstruction can be efficiently solved by applying the N-BOMP algorithm.64,65 Here we show an example of hyper-spectral image $(1024 \times 1024 \times 32)$ corresponding to Scene 4 in the Foster & Nascimento & Amano natural scenes database.98 No preprocessing steps were applied to this data set. This hyper-spectral image contains scene reflectances measured at 32 different frequency channels acquired by a low-noise Peltier-cooled digital camera in the wavelength range of 400 – 720 nm (see details in Ref 98). For each channel we apply a separable random sensing operator given by $\Phi \times \Phi$ where $\Phi \in \mathbb{R}^{M \times 1024}$ ($M < 1024$) is a Gaussian random matrix. We also assume that the data set has a multiway block-sparse representation using the separable Daubechies WT basis given by $\mathbf{W}_3 \otimes \mathbf{W}_2 \otimes \mathbf{W}_1$, with $\mathbf{W}_1 = \mathbf{W}_2 \in \mathbb{R}^{1024 \times 1024}$ and $\mathbf{W}_3 \in \mathbb{R}^{32 \times 32}$ (see Figure 8(a)). In Figure 8(b) the results for sampling ratios 33% and 50% are shown. Quantitatively, the reconstructions perform very well with PSNR = 39.44 dB (33%) and PSNR = 42.47 dB (50%) and they can be qualitatively verified by visual inspection of the images.

### Multidimensional Inpainting

A general problem found in many practical signal processing applications is how to estimate missing samples in multidimensional data sets. This problem was extensively studied for 2D signals (images), in such case it is commonly referred as ‘inpainting’ or ‘matrix completion’. In the recent years there have been many new approaches to this problem by exploiting the theory of CS since sparsity representations can be used as prior information for missing pixels inference.12–16 Although there are many available algorithms for the 2D case, there are only few very recent works about ‘tensor completion’. For example, in Refs 61 and 62 the approach of convex optimization proposed by Candès et al.16 for matrix completion was generalized to tensors. This method relies on the assumption that the matrix (tensor) to recover is low rank so the strategy consists on the minimization of the nuclear norm constrained to the available measurements. However, the computational complexity of these methods is high and its application to large-scale tensors is limited.

Missing data problems using tensor decompositions were already approached in other kind of applications too. For instance, in Ref 99 by assuming a multilinear structure given by the CPD model, an algorithm was proposed for optimization of the CPD model parameters allowing one to interpolate the missing values. The authors have applied this algorithm to an EEG (electroencephalogram) application, where missing data is frequently encountered, for example, owing to disconnections of electrodes. Recently, the same idea was extended and applied to the problem of estimate missing values in medical questionnaires100,101 and missing data imputation in road traffic networks.102 In Ref 103 Tan et al. have proposed a Tucker model-based method for missing traffic data completion where the Tucker factors and the core tensor are optimized to fit the available information.

In this section, we introduce a novel tensor completion method based on the Tucker model specially designed for multidimensional images where Kronecker dictionaries are already known or can be learned to provide good sparse representation of small overlapped tensor patches. Thus, for each tensor patch we solve a CS problem where the measurements are the available data samples. The Kronecker structure of the dictionary combined with a greedy technique similar to OMP allows us to successfully recover the missing entries achieving to very good results, better or similar to the state-of-the-art algorithms. This simple technique is similar to the inpainting and denoising technique described by Elad in Ref 12, chapter 15.3, with the main difference that here we consider the Kronecker structure explicitly which allows us to extend the method to larger image patches $(32 \times 32)$ and to the tensor case too $(3D$ signals).

We assume that only a set of $L$ entries of a tensor $\mathbf{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ are available and we denote by $\mathbf{Y}^{(t)} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ ($t = 1, 2, \ldots, T$) a set of overlapped...
Multidimensional compressed sensing

(a) Hyperspectral CS imaging model with a kronecker dictionary and block-sparsity

Multidimensional signal
\[ \mathbf{Y} (I_1 \times I_2 \times I_3) \]

CS measurements
\[ \mathbf{Z} (M_1 \times M_2 \times M_3) \]

Block-sparse tensor
\[ \mathbf{X} (I_1 \times I_2 \times I_3) \]

\[ \mathbf{Y} = \mathbf{X} \mathbf{D} \]

\[ \mathbf{B}_3 = \mathbf{D}_3 \mathbf{D}_2 \mathbf{D}_1 \]

(b) Hyperspectral CS imaging simulation results

Original (1024 x 1024 x 32)
RGB display

Reconstruction (RGB display)
sampling ratio = 33%
PSNR = 39.44dB

Reconstruction (RGB display)
sampling ratio = 50%
PSNR = 42.47dB

(150 x 150) Zoom
(150 x 150) Zoom
(150 x 150) Zoom

FIGURE 8 | Hyper-spectral compressed sensing (CS) imaging example. Block-sparsity is assumed on a separable (Kronecker) orthogonal basis given by the Daubechies wavelet transform (WT). A random (Gaussian) separable two-dimensional (2D) operator is used for sensing every slice of the hyper-spectral cube (1024 x 1024 x 32). The reconstruction is obtained by applying the N-way block orthogonal matching pursuit (NBOMP) algorithm on the three-dimensional 3D compressive signal \( \mathbf{Z} \).

tensor patches of sizes \( I_1 \times I_2 \times I_3 \), typically with \( I_1 = I_2 = I_3 = 8, 16 \) or \( 32 \), that cover the whole original tensor \( \mathbf{Y} \) (see Figure 9(a)).

In order to simplify the notation here we avoid the explicit reference to the current patch index, i.e. for each tensor patch \( \mathbf{P} \equiv \mathbf{Y}^{(t)} \) only a set of \( L_t \) entries are available whose indices are listed as \( \left( s_{1}^{(t)}, s_{2}^{(t)}, s_{3}^{(t)} \right) \), with \( t = 1, 2, \ldots, L_t \). If we arrange the available entries in a vector \( \mathbf{z} \in \mathbb{R}^{L_t} \) it can be obtained by the following operation:

\[ \mathbf{z} = \left( \mathbf{S}_1^T \odot \mathbf{S}_2^T \odot \mathbf{S}_3^T \right) \mathbf{p}, \]  

(29)

where ‘\( \odot \)’ stands for the Khatri–Rao product, \( \mathbf{z} \in \mathbb{R}^{L_t} \) is the vectorized version of the \( t \)-th tensor.
**FIGURE 9**  
(a) Illustration of the tensor completion algorithm: the original tensor is divided in a set of overlapped tensor patches $\mathbf{Y}^{(t)}$. For every tensor patch $\mathbf{P}$, its available entries (blue dots) are arranged in a vector $\mathbf{z} \in \mathbb{R}^{L_t}$ which is shown to verify a set of linear equations with Khatri–Rao structure and a sparsity constraint. (b) Results of applying the tensor completion algorithm with Kronecker dictionary (orthogonal discrete cosine transform (DCT)) to $32 \times 32$ patches of 'Barbara' data set ($512 \times 512$). A PSNR = 27.66 dB is obtained which is comparable to the state-of-the-arts methods. For example, in Ref 15 it was reported a PSNR = 27.4 dB for the same data set and the same sampling ratio, and in Ref 23 the authors have reported a PSNR = 27.65 dB, also for the same conditions.
patch and the ‘selection matrices’ $S_n \in \mathbb{R}^{L_n \times J_n}$ are defined such that their $l$-th rows contain all zeros except at entry $s_n^{(l)}$, which contains a 1, i.e.

$$S_n(l, m) = \begin{cases} 1, & \text{if } m = s_n^{(l)} \\ 0, & \text{elsewhere} \end{cases} \quad (30)$$

for $l = 1, 2, ..., L_n$ and $m = 1, 2, ..., J_n$.

Equivalently, we can arrange the nonselected entries in a vector $z^* \in \mathbb{R}^{L \times T}$ ($L^*_t = I_1I_2I_3 - L_t$) as follows:

$$z^* = \left( S^*_3 \odot S^*_2 \odot S^*_1 \right) p, \quad (31)$$

where the matrices $S^*_n \in \mathbb{R}^{L_n \times J_n}$ are the ‘complement selection matrices’ which allow us to extract the selected entries and $p \in \mathbb{R}^{L^*_t}$ is the vectorized version of tensor $p$.

By assuming that every tensor patch has a sparse representation on a Kronecker basis, i.e. $p = (D_3 \odot D_2 \odot D_1)x$ with $||x||_0 \leq K$ ($K << I_1I_2I_3$) and by using the following basic linear algebra result $(A \otimes B) \odot (C \odot D) = ABCD$ into Eq. 29, we arrive at:

$$z = \left( C^*_3 \odot C^*_2 \odot C^*_1 \right) x, \quad \text{with } ||x||_0 \leq K, \quad (32)$$

where $C_n = S_nD_n \in \mathbb{R}^{L_n \times J_n}$. Thus the objective is to find the sparsest vector $x \in \mathbb{R}^{I_1I_2I_3}$ that is compatible with the available samples (see Figure 9(a)). It is interesting to note that Eq. 32 is similar to Eq. 16 but with a Khatri–Rao structure instead of the Kronecker one and, therefore, the OMP algorithm can be implemented in an efficient way avoiding to work with large matrices.

Once the vectors of coefficients $x$’s are obtained for every tensor patch ($t = 1, 2, ..., T$) we are able to compute a set of estimations of the missing entries by using their sparse representation through Eq. 31. When using overlapped tensor patches we get several estimates for every single missing entry, i.e. one estimate corresponding to each tensor-patch containing that entry. Thus we may ask ourselves which estimates should be used as final (optimal) approximation. In Ref 12, it was shown that, for the 2D inpainting problem, the optimal estimate, in the least squares sense, is given as the average of all available estimates, and the same result remains valid for a higher number of dimensions.

Here we present some examples of multidimensional signal completion for 2D and 3D data sets. In Figure 9(b), an example of the application of our algorithm to the ‘Barbara’ (512 × 512) data set is shown. In this example, a Kronecker dictionary based on the complete DCT (orthogonal) was used for (32 × 32) patches and selected pixels were randomly chosen. We have obtained a PSNR = 27.66 dB for a sampling ratio of 20% which is comparable to the state-of-the-art methods, for instance, in Ref 15 it was reported a PSNR = 27.4 dB for the same data set and the same sampling ratio, and in Ref 23 the authors reported a PSNR = 27.65 dB using a method that operates on 12 × 12 patches for this data set. It is noted that in Ref 15 the authors used more sophisticated method based on a sparse representation that uses a combination of ‘curvelets’ and DCT in order to efficiently capture edges and smooth regions, respectively, applied to small 8 × 8 patches probably because larger patches would make the method very time consuming.

In a second example, we have applied the tensor completion technique to the 3D MRI brain data (256 × 256 × 100) using 8 × 8 × 8 tensor patches. In Figure 10, the results of applying this technique with Kronecker dictionary (orthogonal DCT) are shown for randomly chosen missing entries. The obtained quality of reconstructions were PSNR = 39 dB and PSNR = 36 dB for sampling ratios of 50% and 25%, respectively.

Other Applications

There are some other signal processing applications that involve multidimensional data sets that can be approached by applying the methods and tools discussed in this paper. For instance, methods based on sparse representation of images were already proposed in order to solve the problem of ‘super-resolution’ or ‘zooming’. Super-resolution intends to obtain a high-resolved output image from several or a single low-resolution image. These methods can be potentially benefited from adopting the Kronecker structure of dictionaries allowing one to alleviate the computational burden and making possible the processing of larger patches in data sets. In fact, it is interesting to point out that, in this case, the sensing operator is related to the convolution kernel which can be modeled by a separable kernel, i.e. obeying the Kronecker structure. Other image processing applications related to the methods here presented are ‘deblurring’ where the objective is to estimate an original signal from a filtered version of it, or ‘denoising’ of multidimensional signals, etc.

There are also other fields of research that could be benefit from the methods discussed in this paper, for example, by learning the appropriate Kronecker dictionaries, our tensor completion algorithm could be applied to missing data problems for EEG recordings, to estimate missing values in medical questionnaires, or to missing data imputation in road traffic networks.
CONCLUSION

In this paper, recent developments on sparse representation and compressed sensed applications for multidimensional signals have been reviewed. An emphasis was focused on recent trends and techniques that allow to exploit the structure of the multidimensional sensing operator as well of the signal representation through the Tucker decomposition model whose intrinsic Kronecker structure allows one to obtain a huge improvement in the algorithm efficiency (speed and saving of memory). We illustrated, through numerical simulations using real-world multidimensional data sets, how these approaches can be successfully applied to solve

FIGURE 10 | Results of applying the tensor completion algorithm with Kronecker dictionary (orthogonal DCT) to 8 × 8 × 8 tensor patches of the three-dimensional (3D) brain data set (256 × 256 × 100).
multidimensional signal processing problems as the case of the reconstruction of MRI images and hyperspectral images from compressive measurements. Also, a new tensor completion technique that exploits the Kronecker structure of dictionaries and the Khatri–Rao structure of the sensing operator was introduced and applied to real 2D and 3D data sets achieving excellent results comparable with the state-of-the-arts techniques. We highlight that exploiting structure in data sets is a key factor for the development of new CS-based technology and the Kronecker structure is a natural assumption for the case of multidimensional signals.

This paper provides a solid ground to develop new signal processing algorithms and models for multidimensional CS. Future directions of research on the topic includes, for example, the development of optimized learning techniques for structured dictionary, development and evaluation of CS algorithms for multidimensional signals with overcomplete dictionaries, exploiting Kronecker and Khatri–Rao structures in the Cosparse Analysis model,\(^{106}\) and many others.

NOTES

\(^{a}\)The MRI brain data set is included in the ‘University of North Carolina Volume Rendering Test Data Set’ archive which is available in public domain at http://www-graphics.stanford.edu/data/voldata/. In this data set, part of the skull was partially removed (synthetically) in order to reveal the brain structure (provided courtesy of Siemens Medical Systems, Inc., Iselin, NJ. Data edited (skull removed) by Dr. Julian Rosenman, North Carolina Memorial Hospital).

\(^{b}\)Peak signal-to noise ratio is defined as

\[
\text{PSNR} (dB) = 20 \log_{10} \left( \frac{\text{max}(\hat{Y})}{||Y - \hat{Y}||_F} \right).
\]

\(^{c}\)Available at http://personalpages.manchester.ac.uk/staff/david.foster/.

\(^{d}\)The nuclear norm of a matrix \(A \in \mathbb{R}^{I_1 \times I_2}\) is defined as

\[
||A||_* = \sum_{n=1}^{\min(I_1, I_2)} \sigma_n
\]

where \(\sigma_n\) are the singular values of matrix \(A\).

ACKNOWLEDGMENTS

This work has been partially supported by CONICET under the project PIP 2012–2014, number 11420110100021.

REFERENCES


14. Guleryuz OG. Nonlinear approximation based image recovery using adaptive sparse reconstructions and


**FURTHER READING**

A complete set of MATLAB codes is available at the author’s personal webpage http://web.fi.uba.ar/~ccaiafa/Cesar/Tensor-CS.html, which allows to reproduce the figures presented in this paper.