Sparse Representation of 3D Images for Piecewise Dimensionality Reduction with High Quality Reconstruction

Laura Rebollo-Neira and Daniel Whitehouse
Mathematics Department
Aston University
B3 7ET, Birmingham, UK

April 17, 2019

Abstract

Sparse representation of 3D images is considered within the context of data reduction. The goal is to produce high quality approximations of 3D images using fewer elementary components than the number of intensity points in the 3D array. This is achieved by means of a highly redundant dictionary and a dedicated pursuit strategy especially designed for low memory requirements. The benefit of the proposed framework is illustrated in the first instance by demonstrating the gain in dimensionality reduction obtained when approximating true color images as very thin 3D arrays, instead of performing an independent channel by channel approximation. The full power of the approach is further exemplified by producing high quality approximations of hyper-spectral images with a reduction of up to 371 times the number of data points in the representation.

Keywords: Image representation with Dictionaries; Greedy Pursuit Algorithms.

1   Introduction

Sparse representation of 2D images has been a subject of extensive research in the last fifteen years [1–3]. Applications which benefit from sparsity range from image restoration [4, 5] and classification [6–8] to feature extraction [9, 10] and super resolution reconstructions [11, 12].

While sparse representation of 3D arrays has received less attention, the advantage of modeling these arrays as a superposition of 3D elementary components is recognized in previous publications [13–16].

At present, the most widely used multichannel images in every day life are true color images. The simplest way of sparsely representing these images is channel by channel, or adding constraints of correlation across colors [4, 17]. However, as demonstrated in this work, sparsity in the representation of true color images can increase substantially if the approximation is realized by means of 3D elements taken from a highly redundant dictionary. The effect is of course more pronounced for arrays involving more channels, such as hyper-spectral images.

From a practical view point, the current drawbacks of 3D sparse modeling using a large dictionary are (i) storage requirements and (ii) the complexity of the concomitant calculations. In this paper we propose a method which, by addressing (i) leaves room for possible high performance implementations using Graphics Processing Unit (GPU) programming. While the approach is illustrated using Central Processing Unit (CPU) programming, the storage requirements are shown to fit within 48Kb's of fast access shared memory of a GPU when the approximation of a 3D image is realized with a partition block size of $8 \times 8 \times 8$ and with a separable dictionary of redundancy 125.

The main contributions of the paper are listed below.

- The low memory implementation of the Orthogonal Matching Pursuit (OMP) strategy, called Self Projected Matching Pursuit (SPMP) [18] is dedicated to operating in 3D (SPMP3D) with separable dictionaries. This technique delivers an iterative solution to the 3D least squares problem which requires much less storage than direct linear algebra methods. It could therefore be also applied with any other of the pursuit strategies that include a least squares step [14, 19–22].

- The C++ MEX file for the SPMP3D method has been made available on a dedicated website [23]. All the scripts for reproducing the results of the paper in the MATLAB environment have also been placed on that website.

- Remarkable reduction in the dimensionality of the representation of true color images and hyper-spectral images, with high quality reconstruction, is demonstrated using highly redundant and highly coherent separable dictionaries.

The results suggest that the method may be of assistance to image processing applications which rely on a transformation for data reduction as a first step of further processing. For examples of relevant applications we refer to [24–28].

2 Notational Convention

$\mathbb{R}$ represents the set of real numbers. Boldface letters are used to indicate Euclidean vectors, 2D and 3D arrays. Standard mathematical fonts indicate components, e.g., $\mathbf{d} \in \mathbb{R}^N$ is a vector of components $d(i) \in \mathbb{R}$, $i = 1, \ldots, N$. The elements of a 3D array $\mathbf{I} \in \mathbb{R}^{N_x \times N_y \times N_z}$ are indicated as $I(i, j, m)$, $i = 1, \ldots, N_x$, $j = 1, \ldots, N_y$, $m = 1, \ldots, N_z$. Moreover, for each m-value $\mathbf{I}_m \in \mathbb{R}^{N_x \times N_y}$ stands for the 2D array of elements $I_m(i, j) = I(i, j, m)$, $i = 1, \ldots, N_x$, $j = 1, \ldots, N_y$, which, when not leaving room for ambiguity will also be represented as $I(:, :, m)$. The transpose of a matrix, $\mathbf{G}$ say, is indicated as $\mathbf{G}^\top$.

The inner product between 3D arrays, say $\mathbf{I} \in \mathbb{R}^{N_x \times N_y \times N_z}$ and $\mathbf{G} \in \mathbb{R}^{N_x \times N_y \times N_z}$, is given as:

$$\langle \mathbf{G}, \mathbf{I} \rangle_{3D} = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \sum_{m=1}^{N_z} G(i, j, m) I(i, j, m).$$

For $\mathbf{G} \in \mathbb{R}^{N_x \times N_y \times N_z}$ with tensor product structure, i.e. for $\mathbf{G} = \mathbf{g}^x \otimes \mathbf{g}^y \otimes \mathbf{g}^z$, with $\mathbf{g}^x \in \mathbb{R}^{N_x}, \mathbf{g}^y \in \mathbb{R}^{N_y}$ and $\mathbf{g}^z \in \mathbb{R}^{N_z}$, we further have

$$\langle \mathbf{G}, \mathbf{I} \rangle_{3D} = \sum_{m=1}^{N_z} \langle \mathbf{g}^x, \mathbf{I}_m \mathbf{g}^y \rangle g^z(m) = \langle \mathbf{p}, \mathbf{g}^z \rangle, \tag{1}$$

where for each value of m the vector $\mathbf{I}_m \mathbf{g}^y$ in $\mathbb{R}^{N_x}$ arises by the standard matrix-vector multiplication rule and $\mathbf{p} \in \mathbb{R}^{N_z}$ is given by its components $p(m) = \langle \mathbf{g}^x, \mathbf{I}_m \mathbf{g}^y \rangle$, $m = 1, \ldots, N_z$. Note that $\langle \mathbf{p}, \mathbf{g}^z \rangle$ indicates the Euclidean inner product in 1D, i.e.

$$\langle \mathbf{p}, \mathbf{g}^z \rangle = \sum_{m=1}^{N_z} p(m) g^z(m).$$

The definition (1) induces the norm $\|\mathbf{I}\|_{3\mathrm{D}} = \sqrt{\langle \mathbf{I}, \mathbf{I} \rangle_{3\mathrm{D}}}$.

3   Sparse Representation of Multi-channel Images

Suppose that a 3D image, given as an array $\mathbf{I} \in \mathbb{R}^{N_x \times N_y \times N_z}$ of intensity pixels, is to be approximated by the linear decomposition

$$\mathbf{I}^k = \sum_{n=1}^{k} c(n) \mathbf{D}_{\ell_n}, \tag{2}$$

where each $c(n)$ is a scalar and each $\mathbf{D}_{\ell_n}$ is an element of $\mathbb{R}^{N_x \times N_y \times N_z}$ to be selected from a set, $\mathcal{D} = \{\mathbf{D}_n\}_{n=1}^{M}$, called a 'dictionary'.

A sparse approximation of $\mathbf{I} \in \mathbb{R}^{N_x \times N_y \times N_z}$ is an approximation of the form (2) such that the number k of elements in the decomposition is significantly smaller than $N = N_x N_y N_z$. The terms in the decomposition (2) are taken from a large redundant dictionary, from where the elements $\mathbf{D}_{\ell_n}$ in (2), called 'atoms', are chosen according to an optimality criterion.

Within the redundant dictionary framework for approximation, the problem of finding the sparsest decomposition of a given multi-channel image can be formulated as follows: *Given an image and a dictionary, approximate the image by the 'atomic decomposition' (2) such that the number k of atoms is minimum.* Unfortunately, the numerical minimization of the number of terms to produce an approximation up to a desired error, involves a combinatorial problem for exhaustive search. Hence, the solution is intractable. Consequently, instead of looking for the sparsest solution, one looks for a 'satisfactory solution', i.e., a solution such that the number of k-terms in (2) is considerably smaller than the image dimension. For 2D images this can be effectively achieved by greedy pursuit strategies in the line of the Matching Pursuit (MP) [29] and OMP [30] methods, if dedicated to 2D separable dictionaries [14,18,32,33]. Within a tensor product framework the consideration of OMP in 3D is natural.

Let's assume that a 3D dictionary is obtained as the tensor product $\mathcal{D} = \mathcal{D}^x \otimes \mathcal{D}^y \otimes \mathcal{D}^z$ of three 1D dictionaries $\mathcal{D}^x = \{\mathbf{d}_m^x \in \mathbb{R}^{N_x}\}_{m=1}^{M_x}$, $\mathcal{D}^y = \{\mathbf{d}_m^y \in \mathbb{R}^{N_y}\}_{m=1}^{M_y}$, and $\mathcal{D}^z = \{\mathbf{d}_m^z \in \mathbb{R}^{N_z}\}_{m=1}^{M_z}$, with $M_x M_y M_z = M$. For computational purposes the 1D dictionaries are stored as three

matrices $\mathbf{D}^x \in \mathbb{R}^{N_x \times M_x}$, $\mathbf{D}^y \in \mathbb{R}^{N_y \times M_y}$ and $\mathbf{D}^z \in \mathbb{R}^{N_z \times M_z}$. Suppose now that a 3D array $\mathbf{I} \in \mathbb{R}^{N_x \times N_y \times N_z}$ is to be approximated by an atomic decomposition of the form

$$\mathbf{I}^k = \sum_{n=1}^{k} c(n) \mathbf{d}^x_{\ell_n^x} \otimes \mathbf{d}^y_{\ell_n^y} \otimes \mathbf{d}^z_{\ell_n^z}, \tag{3}$$

where for $n = 1, \ldots, k$ the atoms $\mathbf{d}^x_{\ell_n^x}$, $\mathbf{d}^y_{\ell_n^y}$ and $\mathbf{d}^z_{\ell_n^z}$ are selected from the given 1D dictionaries. The common step of the techniques we consider for constructing approximations of the form (3) is the stepwise selection of the atoms in the atomic decomposition. On setting $k = 1$ and $\mathbf{I}^0 = 0$ at iteration k the algorithm selects the indices ℓ_k^x, ℓ_k^y and ℓ_k^z as follows

$$\ell_k^x, \ell_k^y, \ell_k^z = \underset{\substack{n=1,\ldots,M_x \\ i=1,\ldots,M_y \\ s=1,\ldots,M_z}}{\arg \max} \left| \langle \mathbf{d}^x_n \otimes \mathbf{d}^y_i \otimes \mathbf{d}^z_s, \mathbf{R}^{k-1} \rangle_{3D} \right|, \tag{4}$$

with $\mathbf{R}^{k-1} = \mathbf{I} - \mathbf{I}^{k-1}$. It is the determination of the coefficients $c(n)$, $n = 1, \ldots, k$ in (3) that gives rise to pursuit strategies which go with different names.

3.1   Matching Pursuit in 3D (MP3D)

The MP approach in 3D would simply calculate the coefficients in (3) as

$$c(n) = \langle \mathbf{d}^x_{\ell_n^x} \otimes \mathbf{d}^y_{\ell_n^y} \otimes \mathbf{d}^z_{\ell_n^z}, \mathbf{R}^{n-1} \rangle_{3D}, \quad n = 1, \ldots, k. \tag{5}$$

The main drawback of the MP method is that it may select linearly dependent atoms. Moreover, that approximation is not stepwise optimal because at iteration k the coefficients (5) do not minimize the norm of the residual error. The pursuit strategy that overcomes these limitations is the so called OMP [30].

3.2   Orthogonal Matching Pursuit in 3D

The implementation of OMP in 3D (OMP3D) we describe here is the 3D extension of the implementation of OMP in 2D given in [32]. An alternative algorithm called Kronecker-OMP, which is based on the Tucker representation of a tensor, is discussed in [14]. Our algorithm

is based on adaptive biorthogonalization and Gram-Schmidt orthogonalization procedures, as proposed in [31] for the one dimensional case.

In order to ensure the coefficients $c(n)$, $n = 1, \ldots, k$ involved in (3) are such that $\|\mathbf{R}^k\|_{3D}^2 = \langle \mathbf{R}^k, \mathbf{R}^k \rangle_{3D}$ is minimum, the decomposition (3) should fulfill that

$$\mathbf{I}^k = \sum_{n=1}^{k} c(n) \mathbf{d}_{\ell_n^x}^x \otimes \mathbf{d}_{\ell_n^y}^y \otimes \mathbf{d}_{\ell_n^z}^z = \hat{\mathrm{P}}_{\mathbb{V}_k} \mathbf{I}, \tag{6}$$

where $\hat{\mathrm{P}}_{\mathbb{V}_k}$ is the orthogonal projection operator onto $\mathbb{V}_k = \mathrm{span}\{\mathbf{d}_{\ell_n^x}^x \otimes \mathbf{d}_{\ell_n^y}^y \otimes \mathbf{d}_{\ell_n^z}^z\}_{n=1}^{k}$. This is ensured by requiring that $\mathbf{R}^k = \mathbf{I} - \hat{\mathrm{P}}_{\mathbb{V}_k} \mathbf{I}$, where $\hat{\mathrm{P}}_{\mathbb{V}_k}$ is the orthogonal projection operator onto $\mathbb{V}_k = \mathrm{span}\{\mathbf{d}_{\ell_n^x}^x \otimes \mathbf{d}_{\ell_n^y}^y \otimes \mathbf{d}_{\ell_n^z}^z\}_{n=1}^{k}$. The required representation of $\hat{\mathrm{P}}_{\mathbb{V}_k}$ is of the form $\hat{\mathrm{P}}_{\mathbb{V}_k} \mathbf{I} = \sum_{n=1}^{k} \mathbf{A}_n \langle \mathbf{B}_n^k, \mathbf{I} \rangle_{3D}$, where each $\mathbf{A}_n \in \mathbb{R}^{N_x \times N_y \times N_z}$ is an array with the selected atoms $\mathbf{A}_n = \mathbf{d}_{\ell_n^x}^x \otimes \mathbf{d}_{\ell_n^y}^y \otimes \mathbf{d}_{\ell_n^z}^x$. The concomitant biorthogonal reciprocal set $\mathbf{B}_n^k$, $n = 1, \ldots, k$ comprises the unique elements of $\mathbb{R}^{N_x \times N_y \times N_z}$ satisfying the conditions:

i) $\langle \mathbf{A}_n, \mathbf{B}_m^k \rangle_{3D} = \delta_{n,m} = \begin{cases} 1 & \text{if } n = m \\ 0 & \text{if } n \neq m. \end{cases}$

ii) $\mathbb{V}_k = \mathrm{span}\{\mathbf{B}_n^k\}_{n=1}^{k}$.

Thus, the coefficients $c(n)$, $n = 1, \ldots, N$ in (6) which guarantee minimum norm of the residual error are calculated as

$$c(n) = \langle \mathbf{B}_n^k, \mathbf{I} \rangle_{3D}, \quad n = 1, \ldots, k.$$

The required arrays $\mathbf{B}_n^k$, $n = 1, \ldots, k$ should be upgraded and updated to account for each newly selected atom. Starting from $k = 1, \mathbf{R}^0 = \mathbf{I}$, $\mathbf{B}_1^1 = \mathbf{W}_1 = \mathbf{A}_1 = \mathbf{d}_{\ell_1^x}^x \otimes \mathbf{d}_{\ell_1^y}^y \otimes \mathbf{d}_{\ell_1^z}^z$, where

$$\ell_1^x, \ell_1^y, \ell_1^z = \arg\max_{\substack{n=1,\ldots,M_x \\ i=1,\ldots,M_y \\ s=1,\ldots,M_z}} \left| \langle \mathbf{d}_n^x \otimes \mathbf{d}_i^y \otimes \mathbf{d}_s^z, \mathbf{R}^{k-1} \rangle_{3D} \right|,$$

at iteration $k + 1$ the indices $\ell_{k+1}^x, \ell_{k+1}^y, \ell_{k+1}^z$ corresponding to the new atom $\mathbf{A}_{k+1} = \mathbf{d}_{\ell_{k+1}^x}^x \otimes \mathbf{d}_{\ell_{k+1}^y}^y \otimes \mathbf{d}_{\ell_{k+1}^z}^z$ are selected as in (4). The required reciprocal set $\mathbf{B}_n^{k+1}$, $n = 1, \ldots, k+1$ is adapted

and upgraded by extending the recursion formula given in [31] as follows.

$$\mathbf{B}_n^{k+1} = \mathbf{B}_n^k - \mathbf{B}_{k+1}^{k+1}\langle \mathbf{A}_{k+1}, \mathbf{B}_n^k \rangle_{3D}, \quad n = 1, \ldots, k,$$

where

$$\mathbf{B}_{k+1}^{k+1} = \mathbf{W}_{k+1}/\|\mathbf{W}_{k+1}\|_{3D}^2,$$

with

$$\mathbf{W}_{k+1} = \mathbf{A}_{k+1} - \sum_{n=1}^{k} \frac{\mathbf{W}_n}{\|\mathbf{W}_n\|_{3D}^2} \langle \mathbf{W}_n, \mathbf{A}_{k+1} \rangle_{3D},$$

including, for numerical accuracy, the re-orthogonalization step:

$$\mathbf{W}_{k+1} \leftarrow \mathbf{W}_{k+1} - \sum_{n=1}^{k} \frac{\mathbf{W}_n}{\|\mathbf{W}_n\|_{3D}^2} \langle \mathbf{W}_n, \mathbf{W}_{k+1} \rangle_{3D}.$$

Although the image approximation is carried out by partitioning the images into relatively small 3D blocks, memory requirements of the OMP3D method are high. Indeed, the above are $2(k+1)$ nonseparable arrays each of dimension $N = N_x N_y N_z$ which need to be stored in double precision. Hence, we consider next a low memory implementation of the orthogonal projection step, which avoids having to store the arrays $\mathbf{W}_n$, $n = 1, \ldots, k$ and $\mathbf{B}_n^k$, $n = 1, \ldots, k$ and fully exploits the separability of the dictionary.

3.3   Self Projected Matching Pursuit in 3D (SPMP3D)

The Self Projected Matching Pursuit (SPMP) methodology was introduced in [18] and conceived to be used with separable dictionaries in 2D (SPMP2D). Because the technique is based on calculations of inner products, it can be easily extended to operate in 3D (SPMP3D).

Suppose that at iteration k the selection process has chosen the atoms labeled by the triple of indices $\{\ell_n^x, \ell_n^y, \ell_n^z\}_{n=1}^k$ and let $\tilde{\mathbf{I}}^k$ be the atomic decomposition

$$\tilde{\mathbf{I}}^k = \sum_{n=1}^{k} a(n)\mathbf{d}_{\ell_n^x}^x \otimes \mathbf{d}_{\ell_n^y}^y B \otimes \mathbf{d}_{\ell_n^z}^z, \tag{7}$$

where the coefficients $a(n)$, $n = 1, \ldots, k$ are arbitrary numbers. Every array $\mathbf{I} \in \mathbb{R}^{N_x \times N_y \times N_z}$ can be expressed as

$$\mathbf{I} = \tilde{\mathbf{I}}^k + \tilde{\mathbf{R}}. \tag{8}$$

For $\tilde{\mathbf{I}}^k$ to be the optimal representation of $\mathbf{I}$ in $\mathbb{V}_k = \text{span}\{\mathbf{d}_{\ell_n^x}^x \otimes \mathbf{d}_{\ell_n^y}^y \otimes \mathbf{d}_{\ell_n^z}^z\}_{n=1}^k$, in the sense of minimizing the norm of the residual $\tilde{\mathbf{R}}$, it should be true that $\hat{P}_{\mathbb{V}_k}\tilde{\mathbf{R}} = 0$. The SPMP3D method fulfills this property by approximating $\tilde{\mathbf{R}}$ in $\mathbb{V}_k$, via the MP method, and subtracting that component from $\tilde{\mathbf{R}}$. The following algorithm describes the whole procedure. Starting from $k = 0$ and $\mathbf{R}^0 = \mathbf{I}$, at each iteration, implement the steps below.

i) Increase $k \leftarrow k+1$ and apply the criterion (4) for selecting the triple of indices $(\ell_k^x, \ell_k^y, \ell_k^z)$. Save this triple in the array $L(k, 1:3) = (\ell_k^x, \ell_k^y, \ell_k^z)$, and set

$$c(k) = \langle \mathbf{d}_{\ell_k^x}^x \otimes \mathbf{d}_{\ell_k^y}^y \otimes \mathbf{d}_{\ell_k^z}^z, \mathbf{R}^{k-1} \rangle_{3D}$$

and implement the update of the residue $\mathbf{R}^k = \mathbf{R}^{k-1} - c(k)\mathbf{d}_{\ell_k^x}^x \otimes \mathbf{d}_{\ell_k^y}^y \otimes \mathbf{d}_{\ell_k^z}^z$ as follows: For $s = 1, \ldots N_z$ calculate

$$\Delta\mathbf{R}^k(:, :, s) = \mathbf{R}^{k-1}(:, :, s) - c(k)\mathbf{d}_{\ell_k^x}^x (\mathbf{d}_{\ell_k^y}^y)^\top d_{\ell_k^z}^z(s),$$

to update $\mathbf{R}^k$ as

$$\mathbf{R}^k = \mathbf{R}^{k-1} - \Delta\mathbf{R}^k.$$

ii) Given the indices $L(n, 1:3) = (\ell_n^x, \ell_n^y, \ell_n^z)$, $n = 1, \ldots, k$ of the previously selected atoms, and a tolerance ϵ for the projection error, realize the orthogonal projection up to that error as follows. Set $j = 1$, $\tilde{\mathbf{R}}^0 = \mathbf{R}^k$ and at iteration j apply the steps a) - c) below.

a) For $n = 1, \ldots, k$ evaluate

$$\alpha(n) = \langle \mathbf{d}_{\ell_n^x}^x \otimes \mathbf{d}_{\ell_n^y}^y \otimes \mathbf{d}_{\ell_n^z}^z, \tilde{\mathbf{R}}^{j-1} \rangle_{3D}, \tag{9}$$

and single out the value k^* such that

$$k^* = \arg\max_{n=1,\ldots,k}|\alpha(n)|. \tag{10}$$

The value k^* signalizes the indices $\ell_{k^*}^x, \ell_{k^*}^y, \ell_{k^*}^z$ corresponding to the already selected atoms with maximum correlation with the residual $\tilde{\mathbf{R}}^{j-1}$.

b) If $|\alpha(k^*)| < \epsilon$ stop. Otherwise update the coefficient

$$c(k^*) \leftarrow c(k^*) + \alpha(k^*)$$

and for $s = 1, \ldots, N_z$ evaluate

$$\Delta \tilde{\mathbf{R}}^j(:, :, s) = \alpha(k^*)\, \mathbf{d}^x_{\ell^x_{k*}} (\mathbf{d}^y_{\ell^y_{k*}})^\top d^z_{\ell^z_{k*}}(s)$$

to update the residual $\tilde{\mathbf{R}}^j$ as

$$\tilde{\mathbf{R}}^j = \tilde{\mathbf{R}}^{j-1} - \Delta \tilde{\mathbf{R}}^j.$$

This step subtracts from the residual a component in $\mathbb{V}_k$ and add that component to the approximation $\tilde{\mathbf{I}}^k$

c) Increase $j \leftarrow j+1$ and repeat the steps a) - c) to keep subtracting components of $\tilde{\mathbf{R}}^j$ in $\mathbb{V}_k$ until iteration, J say, for which the stopping criterion b) is met. This criterion indicates that, up to tolerance ϵ, the residual has no component in $\mathbb{V}_k$ so that one can set $\mathbf{R}^k = \tilde{\mathbf{R}}^{J-1}$.

Continue with steps i) and ii) to keep enlarging $\mathbb{V}_k$ until, for a required tolerance error ρ, the condition $\|\mathbf{R}^k\|_{3D} < \rho$ is reached.

Remark 1: For each fixed value k the rate of convergence

$$\lim_{j \to \infty} \mathbf{I} - \tilde{\mathbf{R}}^j = \hat{\mathrm{P}}_{\mathbb{V}_k} \mathbf{I}$$

through the steps a) - c) above is given in [34] for the one dimensional case. The proof for 3D is identical to that proof, because a 3D array can be represented as a long 1D vector. What varies is the implementation. A vectorized version of the algorithm would not be applicable in this context.

Implementation details

The bulk of the computational burden in the SPMP3D method lies in the realization of the selection of atoms (4). Algorithm 1 outlines a procedure implementing the process. It should be stressed once again that the algorithm is designed to use as little memory as possible, rather than to reduce complexity. At iteration k the outputs of Algorithm 1 are saved as $c(k) = \alpha$

Algorithm 1 Implementation of the selection of atoms (c.f. (4))

Procedure $[\alpha, \ell^x, \ell^y, \ell^z] = \text{Sel3DAtom}(\mathbf{R}, \mathbf{D}_x, \mathbf{D}_y, \mathbf{D}_z)$

  Input: 3D array $\mathbf{R}$, matrices $\mathbf{D}_x$, $\mathbf{D}_y$ $\mathbf{D}_z$ the columns of which are the atoms in the corresponding dictionaries.

  Output: selected indices ℓ^x, ℓ^y, ℓ^z, and $\alpha = \langle \mathbf{d}^x_{\ell^x} \otimes \mathbf{d}^y_{\ell^y} \otimes \mathbf{d}^z_{\ell^z}, \mathbf{R} \rangle_{3D}$

  {Initiate the algorithm}

  $(N_z, M_z) = \text{size}(\mathbf{D}_z)$, $M_x = \text{size}(\mathbf{D}_x, 2)$; $M_y = \text{size}(\mathbf{D}_y, 2)$

  $q = \text{zeros}(M_x, M_y)$

  $\alpha = 0$

  for $m = 1 : M_z$ **do**

    $q(:,:) = 0$

    for $s = 1 : N_z$ **do**

      $q(:,:) = q(:,:) + \mathbf{D}_x^\top R(:,:,s) \mathbf{D}_y d^z_m(s)$

    end for

    {Realize (4) by finding the partial maximum, and its argument, for each m-plane}

    $[l_1, l_2, \tilde{q}] = \max(|q(:,:)|)$

    if $\tilde{q} > \alpha$ **then**

      $\alpha = \tilde{q}$; $\ell^x = l_1$; $\ell^y = l_2$; $\ell^z = m$

    end if

  end for

---

and $L(k, 1 : 3) = (\ell^x, \ell^y, \ell^z)$. The implementation details for selecting the triple of indices at the projection step are given in Algorithm 2. This is used in Algorithm 3 for the realization of the actual projection to recalculate the coefficients in the atomic decomposition.

Due to computational complexity and memory requirements, pursuit strategies using general dictionaries can only be implemented on an image partitioned into small blocks. We consider nonoverapping blocks. The approximation of each block is carried out independently of the others. When the approximation of all the blocks is concluded, these are assembled together to produce the approximation of the whole image. While the sparsity results yielded by the

Algorithm 2 Selection of the triple of indices from the reduced dictionary (c.f. (10))

Procedure $[\alpha^*, k^*]$=SelTrip$(\mathbf{R}, \mathbf{D}_x, \mathbf{D}_y, \mathbf{D}_z, \mathbf{L})$

Input: As in Algorithm 1 plus the array $\mathbf{L}$, with the triple of indices $L(n, 1 : 3) = (\ell_n^x, \ell_n^y, \ell_n^z)$, $n = 1 \ldots k$

Output: k^* and the corresponding values of α (c.f. (10)) to update the coefficients and residual

{Initiate the algorithm}

$\alpha^* = 0$

for $n = 1 : k$ **do**

    $p = 0$

    for $s = 1 : N_z$ **do**

      $p = p + (\mathbf{d}_{\ell_n^x}^x)^\top R(:, :, s) \mathbf{d}_{\ell_n^y}^y d_{\ell_n^z}^z(s)$

    end for

    if $|p| > |\alpha^*|$ **then**

      $k^* = n$ and $\alpha^* = p$

    end if

end for

OMP3D and the SPMP3D methods are theoretically equivalent, we have seen that the latter implementation is much more economic in terms of storage demands. As discussed in Remark 2 below, this feature makes the SPMP3D algorithm suitable for possible GPU implementations using only the fast access shared memory. Assuming for simplicity in the notation that a 3D image is partitioned into cubes of size N_b^3 and the dictionaries $\mathcal{D}_x$, $\mathcal{D}_y$ and $\mathcal{D}_z$ are all of the same size $N_b \times rN_b$, where $r > 1$ is the redundancy of the 1D dictionary, the SPMP3D algorithm storage needs are as follows.

1. Two N_b^3 arrays for the intensity block in the image partition and the residual of the corresponding approximation.

2. Three matrices of size $N_b \times rN_b$ for each dictionary, in case they are different.

3. A $r^2 \times N_b^2$ array for the selection of indices in Algorithm 1.

4. A vector of k real numbers to store the coefficients of the atomic decomposition and k vectors of size 3 to store the indices of the atoms in the atomic decomposition. The value of k is the total number of atoms in the approximation of the block.

Algorithm 3 Implementation of the self projection steps a) - c).

Procedure $[\tilde{\mathbf{R}}, \bar{\mathbf{c}}]=\mathrm{Proj3D}(\mathbf{R}, \mathbf{D}_x, \mathbf{D}_y, \mathbf{D}_z, \mathbf{L}, \mathbf{c}, \epsilon, \mathrm{MaxJ})$.

Input: As in Algorithm 2, plus the coefficients of the atomic decomposition $\mathbf{c}$, a tolerance parameter ϵ for the numerical error of the projection, and a maximum number of permitted iterations, MaxJ.

Output: Orthogonal residual $\tilde{\mathbf{R}}$. Coefficients $\tilde{\mathbf{c}}$ of the optimized atomic decomposition.

for $j = 1 : \mathrm{MaxJ}$ **do**

    {Selection of atoms using Algorithm 2}

    $[\alpha^*, k^*]=\mathrm{SelTrip}(\mathbf{R}, \mathbf{D}_x, \mathbf{D}_y, \mathbf{D}_z, \mathbf{L})$

    {Check stopping criterion}

    if $|\alpha^*| < \epsilon$ **then**

        stop

    end if

    {Update the coefficients}

    $c(k^*) \leftarrow c(k^*) + \alpha^*$

    {Update the residual}

    for $s = 1 : N_z$ **do**

        $\mathbf{R}(:,:,s) \leftarrow \mathbf{R}(:,:,s) - \alpha^*(\mathbf{d}_{\ell_{k^*}^x}^x)^\top R(:,:,s)\mathbf{d}_{\ell_{k^*}^y}^y \, d_{\ell_{k^*}^z}^z(s)$

    end for

end for

{For clarity in the description only, we re-name here the residual and coefficients}

$\tilde{\mathbf{R}} = \mathbf{R}$; $\bar{\mathbf{c}} = \mathbf{c}$

Since the stepwise complexity is dominated by the selection of indices (c.f. (4)), within this setup it is $\mathrm{O}(r^3 N_b^5)$ and for true color images $\mathrm{O}(r^3 N_b^3)$.

Remark 2: By considering blocks of size $8 \times 8 \times 8$ and dictionaries of redundancy $r = 5$ in each dimension, the above listed storage needs of the SPMP3D algorithm comfortably fit the fast access shared memory of a GPU in CUDA, which currently is 48Kb. Indeed, in the worst-case scenario (corresponding to an approximation of zero error using $k = 8^3$ atoms for the approximation of an $8 \times 8 \times 8$ block) SPMP3D would require 38Kb to store most of the arrays in double precision, except for those with the selected indices which contain integer numbers. This still leaves 10Kb for temporary variables to be used within calculations.

3.4 Mixed Dictionaries

A key factor for the success in the construction of sparse representations is to have a good dictionary. While a number of techniques for learning dictionaries from training data have been proposed in the literature [35–42], they are not designed for learning large and highly coherent separable dictionaries. Nevertheless, previous works [18, 32, 33, 43] have demonstrated that highly redundant and highly coherent separable dictionaries, which are easy to construct, achieve remarkable levels of sparsity in the representation of 2D images. Such dictionaries are not specific to a particular class of images. A discrimination is only made to take into account whether the approximation is carried out in the pixel intensity or in the wavelet domain.

As will be illustrated by the numerical examples in the next section, the approximation of the images we are considering are sparser when realized in the wavelet domain (wd). This entails the following steps:

- Apply a wavelet transform to each channel $\mathbf{I}_m, m = 1, \ldots, N_z$ to obtain the arrays $\mathbf{U}_m, m = 1, \ldots, N_z$. For the numerical examples we have used the 9/7 Cohen-Daubechies-Feauveau biorthogonal wavelet transform [44].

- Approximate the array $\mathbf{U} \in \mathbb{R}^{N_x \times N_y \times N_z}$ exactly as it is done in the pixel domain (pd).

- Apply the inverse wavelet transform to the approximated planes to recover the approximated intensity channels.

The mixed dictionary we consider for the 2D approximation consists of two sub-dictionaries: A trigonometric dictionary, $\mathcal{D}_T^x$, which is the common sub-dictionary for the approximation in both domains, and a dictionary of localized atoms, which contains atoms of different shapes when used in each domain.

The trigonometric dictionary is the union of the dictionaries $\mathcal{D}_C^x$ and $\mathcal{D}_S^x$ defined below:

$$\mathcal{D}_C^x = \{ w_c(n) \cos \frac{\pi(2i-1)(n-1)}{2M_x}, i = 1, \ldots, N_x \}_{n=1}^{M_x}$$

$$\mathcal{D}_S^x = \{ w_s(n) \sin \frac{\pi(2i-1)(n)}{2M_x}, i = 1, \ldots, N_x \}_{n=1}^{M_x},$$

13

where $w_c(n)$ and $w_s(n)$, $n = 1, \ldots, M_x$ are normalization factors, and usually $M_x = 2N_x$. Thus, the trigonometric dictionary is constructed as $\mathcal{D}_T^x = \mathcal{D}_C^x \cup \mathcal{D}_S^x$.

For approximations in the pd we add the dictionary, $\mathcal{D}_{\text{Lp}}^x$, which is built by translation of the prototype atoms in the left graph of Fig. 1. This type of dictionary is inspired by a general result holding for continuous spline spaces. Namely, that spline spaces on a compact interval can be spanned by dictionaries of B-splines of broader support than the corresponding B-spline basis functions [45, 46]. Thus, the first 4 prototype atoms $\mathbf{h}_i$, $i = 1, \ldots, 4$ in the left graph of Fig. 1 are generated by discretization of linear B-spline functions of different support. For $m = 1, 2, 3, 4$ those functions are defined as follows:

$$
h_m(x) = \begin{cases} \frac{x}{m} & \text{if} \quad 0 \le x < m \\ 2 - \frac{x}{m} & \text{if} \quad m \le x < 2m \\ 0 & \text{otherwise.} \end{cases} \tag{11}
$$

The remaining prototypes, $\mathbf{h}_5, \mathbf{h}_6$ and $\mathbf{h}_7$, in the left graph of Fig. 1 are generated taken the derivatives of the previous functions: $h_5(x) = (h_2(x))'$, $h_6(x) = (h_3(x))'$ and $h_7 = (h_4(x))'$. The corresponding dictionaries $\mathcal{D}_{H_m}$, $m = 1, \ldots, 7$ are built by discretization of the variable x in (11) and sequential translation of one sampling point, i.e.,

$$
\mathcal{D}_{H_m} = \{w_{h_m}(n)h_m(i-n)|N_x; i = 1, \ldots, N_x\}_{n=1}^M, \quad m = 1, \ldots, 7,
$$

where the notation $h_m(i-n)|N_x$ indicates the restriction to be an array of size N_x. The numbers $w_{h_m}(n)$, $n = 1, \ldots, M$, $m = 1, \ldots, 7$ are normalization factors. The dictionary $\mathcal{D}_{\text{Lp}}^x$ arises by the union of the dictionaries $\mathcal{D}_{H_m}$, $m = 1, \ldots, 7$ i.e., $\mathcal{D}_{\text{Lp}}^x = \cup_{m=1}^7 \mathcal{D}_{H_m}$. The whole mixed dictionary $\mathcal{D}_{\text{pd}}^x$ is finally formed as $\mathcal{D}_{\text{pd}}^x = \mathcal{D}_C^x \cup \mathcal{D}_S^x \cup \mathcal{D}_{\text{Lp}}^x$. For the other dimension we take $\mathcal{D}_{\text{pd}}^y = \mathcal{D}_{\text{pd}}^x$.

For approximations in the wd we use the dictionary of localized atoms $\mathcal{D}_{\text{Lw}}^x$ as proposed in [33], which is built by translation of the prototype atoms $\mathbf{p}_i$, $i = 1, \ldots, 7$ in the right graph of Fig. 1. Notice that $\mathbf{p}_1 = \mathbf{h}_1$ and $\mathbf{p}_3 = \mathbf{h}_5$. The remaining prototypes are given by the vectors:

$\mathbf{p}_2 = (1, 1, 0, 0, \ldots, 0)^\perp \in \mathbb{R}^{N_x}$, $\mathbf{p}_4 = (1, 1, 1, 0, \ldots, 0)^\perp \in \mathbb{R}^{N_x}$, $\mathbf{p}_5 = (-1, 1, 1, 0, \ldots, 0)^\perp \in \mathbb{R}^{N_x}$,
$\mathbf{p}_6 = (1, -1, 1, 0, \ldots, 0)^\perp \in \mathbb{R}^{N_x}$, $\mathbf{p}_7 = (-1, -1, 1, 0, \ldots, 0)^\perp \in \mathbb{R}^{N_x}$,
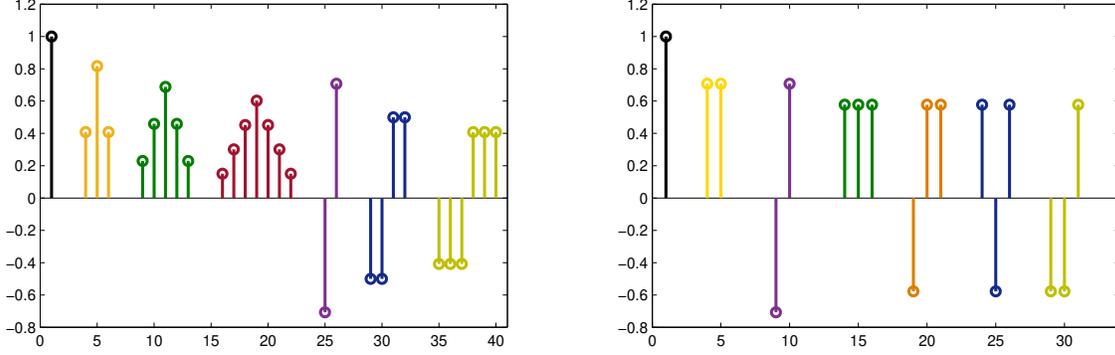
Figure 1: The left graph illustrates the prototype atoms which generate by translation the dictionaries $\mathcal{D}_{H_m}$, $m = 1, \ldots, 7$. The prototypes in the right graph generate by translation the dictionaries $\mathcal{D}_{P_m}$, $m = 1, \ldots, 7$.

The corresponding dictionaries $\mathcal{D}_{P_m}$, $m = 1, \ldots, 7$ are built as in the previous case by sequential translation of one sampling point,

$$\mathcal{D}_{P_m} = \{w_{p_m}(n)p_m(i - n)|N_x; i = 1, \ldots, N_x\}_{n=1}^{M}, \quad m = 1, \ldots, 7,$$

where the numbers $w_{p_m}(n)$, $n = 1, \ldots, M$, $m = 1, \ldots, 7$ are normalization factors. The dictionaries $\mathcal{D}_{P_m}, m = 1, \ldots, 7$ give rise to $\mathcal{D}_{\mathrm{Lw}}^x = \cup_{i=1}^{7}\mathcal{D}_{P_m}$. The latter generates the mixed dictionary $\mathcal{D}_{\mathrm{wd}}^x = \mathcal{D}_{\mathrm{C}}^x \cup \mathcal{D}_{\mathrm{S}}^x \cup \mathcal{D}_{\mathrm{Lw}}^x$ and $\mathcal{D}_{\mathrm{wd}}^y = \mathcal{D}_{\mathrm{wd}}^x$.

The corresponding 2D dictionaries $\mathcal{D}_{\mathrm{pd}} = \mathcal{D}_{\mathrm{pd}}^x \otimes \mathcal{D}_{\mathrm{pd}}^y$ and $\mathcal{D}_{\mathrm{wd}} = \mathcal{D}_{\mathrm{wd}}^x \otimes \mathcal{D}_{\mathrm{wd}}^y$ are very large, but never used as such. All the calculations are carried out using the 1D dictionaries. In order to demonstrate the gain in sparsity attained by the approximation of 3D images by partitioning into 3D blocks, we use dictionaries $\mathcal{D}_{\mathrm{wd}}$ and $\mathcal{D}_{\mathrm{pd}}$ only for the approximation of the single channel 2D images. For the 3D case we maintain the redundancy of the 3D dictionary equivalent to that of the 2D dictionary, by considering the 1D dictionary $\tilde{\mathcal{D}}_{\mathrm{pd}}^x = \mathcal{D}_{\mathrm{C}}^x \cup \mathcal{D}_{\mathrm{S}}^x \cup \mathcal{D}_{P_1}$. Notice that $\mathcal{D}_{P_1}$ is the standard Euclidean basis for $\mathbb{R}^{N_x}$, also called the Dirac's basis, i.e., the basis arising by translation of the first atom in Fig. 1. Notice that $\tilde{\mathcal{D}}_{\mathrm{pd}}^x \subset \mathcal{D}_{\mathrm{pd}}^x$ and $\tilde{\mathcal{D}}_{\mathrm{pd}}^x \subset \mathcal{D}_{\mathrm{wd}}^x$. We also consider $\tilde{\mathcal{D}}_{\mathrm{pd}}^y = \tilde{\mathcal{D}}_{\mathrm{pd}}^x$ and $\tilde{\mathcal{D}}_{\mathrm{pd}}^z = \tilde{\mathcal{D}}_{\mathrm{pd}}^x$, but taking $N_x = N_z$. The redundancy of the resulting

dictionary $\tilde{\mathcal{D}}_{\mathrm{pd}} = \tilde{\mathcal{D}}_{\mathrm{pd}}^x \otimes \tilde{\mathcal{D}}_{\mathrm{pd}}^y \otimes \tilde{\mathcal{D}}_{\mathrm{pd}}^z$ is equivalent to the redundancy of the 2D dictionary $\mathcal{D}_{\mathrm{pd}}$. In 3D we use the same dictionary in both domains $\tilde{\mathcal{D}}_{\mathrm{wd}} = \tilde{\mathcal{D}}_{\mathrm{pd}}$.

4 Numerical Results

The merit of the simultaneous approximation of multiple channel images is illustrated in this section by recourse to two numerical examples. Firstly we make the comparison between the sparsity produced by the joint approximation of the Red-Green-Blue (RGB) channel images partitioned into blocks of size $N_b \times N_b \times 3$ and the sparsity obtained by the independent approximation of each channel partitioned into blocks of size $N_b \times N_b$. Secondly, the full power of the approach is illustrated through the gain in sparsity attained by approximating hyperspectral images partitioned into 3D blocks, vs the plane by plane approximation.

In both cases, once the approximation of each 3D block $\mathbf{I}_q$ in the image partition is completed, for $q = 1, \ldots, Q$ the k_q-term atomic decomposition of the corresponding block is expressed in the form

$$\mathbf{I}_q^{k_q} = \sum_{n=1}^{k_q} c_q(n) \mathbf{d}_{\ell_n^x,q}^x \otimes \mathbf{d}_{\ell_n^y,q}^y \otimes \mathbf{d}_{\ell_n^z,q}^z. \tag{12}$$

The sparsity of the representation of an image of dimension $N = N_x \cdot N_y \cdot N_z$ is measured by the Sparsity Ratio (SR), which is defined as:

$$\mathrm{SR} = \frac{N}{K}, \tag{13}$$

where for the 3D representation $K = \sum_{q=1}^Q k_q$, with k_q the number of atoms in the atomic decomposition (12). For the channel by channel decomposition of a N_z-channel image, each channel is partitioned into $P = (N_x \cdot N_y)/N_b^2$ blocks $\mathbf{I}_{p,z}$, $p = 1, \ldots, P$, which are approximated by the 2D atomic decompositions

$$\mathbf{I}_p^{k_{p,l}} = \sum_{n=1}^{k_{p,l}} c_p^{p,l}(n) \mathbf{d}_{\ell_n^{x,p,l}}^x \otimes \mathbf{d}_{\ell_n^{y,p,l}}^y, \quad l = 1, \ldots, N_z, \tag{14}$$

where the indices $\ell_n^{x,p,l}, \ell_n^{y,p,l}$ are selected for each channel l by the OMP2D algorithm. Accordingly, the number K in (13) is given as $K = \sum_{l=1}^{N_z} \sum_{p=1}^P k_{p,l}$, with $k_{p,l}$ the number of atoms in

16

the atomic decomposition (14).

Notice that the SR is a measure of the reduction of dimensionality for representing an image. The larger the value of the SR the smaller the dimensionality of the atomic decomposition representing the whole image. The required quality of the approximation is ensured with respect to the Mean Structural SIMilarity (MSSIM) index [47,48] and the classical Peak Signal-to-Noise Ratio (PSNR), which for a 3D image is defined as

$$\text{PSNR} = 10 \log_{10} \left(\frac{(\text{Imax})^2}{\text{MSE}} \right), \quad \text{MSE} = \frac{\|\mathbf{I} - \mathbf{I}^K\|_{3\text{D}}}{N_x \cdot N_y \cdot N_z},$$

where Imax is the maximum intensity range and $\mathbf{I}^K$ the image approximation.

4.1   Example I

In this example we use the Kodak data set consisting of 24 true color images shown in Fig. 2.
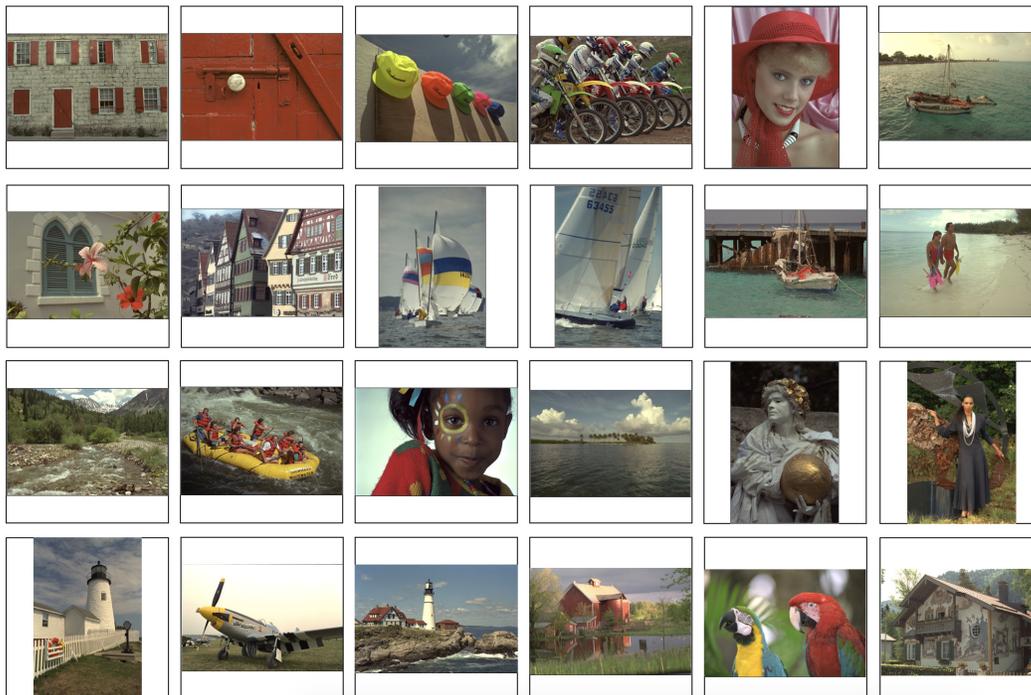

Figure 2: Illustration of the Kodak data set consisting of 24 true color images, credit Rich Franzen [49]. The size of these images is $768 \times 512 \times 3$, for most of them, except for numbers 4, 9, 10, 17, 18 and 19, which are of size $512 \times 768 \times 3$.

The approximations are realized in both domains by maintaining the same redundancy in the 2D and 3D dictionaries. For the independent approximation of the 2D channels the partitions are realized with blocks of size 8×8 and 16×16 (a partition of block size 24×24 does not improve results for this data set). Accordingly, the simultaneous approximation of the 3 color channels involves partitions of block size $8 \times 8 \times 3$ and $16 \times 16 \times 3$ respectively.

As already discussed, for the independent approximation of the 2D channels we consider the dictionaries $\mathcal{D}_{\mathrm{pd}}$ (in the pd) and $\mathcal{D}_{\mathrm{wd}}$ (in the wd) as given in Sec. 3.4. For the simultaneous approximation of the 3 channels we consider the dictionaries $\tilde{\mathcal{D}}_{\mathrm{pd}}$ given in the same section. Both dictionaries have redundancy of 125.

The average values of SR ($\overline{\mathrm{SR}}$), with respect to the 24 images in the set, are given in Table 1 for the approaches and partitions indicated by the first column.

PSNR	45 dB		41 dB	
	$\overline{\mathrm{SR}}$	std	$\overline{\mathrm{SR}}$	std
pd 2D 8×8	6.2	2.0	9.1	3.5
pd 3D $8 \times 8 \times 3$	10.3	2.9	16.1	5.5
wd 2D 8×8	7.1	2.6	11.8	5.8
wd 3D $8 \times 8 \times 3$	11.6	3.8	20.9	9.2
pd 2D 16×16	7.1	2.5	11.1	5.0
pd 3D $16 \times 16 \times 3$	11.6	3.6	18.8	7.5
wd 2D 16×16	7.5	2.7	12.0	6.2
wd 3D $16 \times 16 \times 3$	12.4	3.9	20.4	8.9
Thresholding in the wd	3.2	1.1	4.9	2.6

Table 1: Mean value of the SR, with respect to the 24 images in the set, obtained with the 2D and 3D approximations in both the pd and wd for two different sizes of the image partition. The last row in the table gives the results corresponding to standard nonlinear thresholding of wavelet coefficients, to achieve the same quality of the approximation as with the dictionaries: PSNR = 45dB (left half) and PSNR = 41dB (right half).

All the results in the left half of the table correspond to PSNR = 45 dB and all the results in the right half correspond to PSNR = 41 dB. The third and fifth columns give the standard deviations (std). For completeness we have also produced the $\overline{\mathrm{SR}}$ rendered by

nonlinear thresholding of the wavelets coefficients (last row in the table). Notice that the resulting sparsity is poor in comparison with the other 2D results.

All the results were obtained in the MATLAB environment on a notebook 2.9GHz dual core i7 3520M CPU and 4GB of memory. For the channel by channel approximation a C++ MEX file implementing OMP2D was used. For the 3D approximation SPMP3D was implemented by a C++ MEX file.

As observed in Table 1 the largest $\overline{\mathrm{SR}}$ is achieved in the wd and partition $16 \times 16 \times 3$ (c.f. last but one row of Table 1). However, the results obtained by partition $8 \times 8 \times 3$ are very close (c.f. last row of the upper half of Table 1) and constitute a better tradeoff between SR and approximation time.

Fig. 3 shows the actual values of SRs for this partition in the wd for each of the 24 images in the data set (c.f. Fig. 2). The average time for the 3D approximation was 53 s per image.
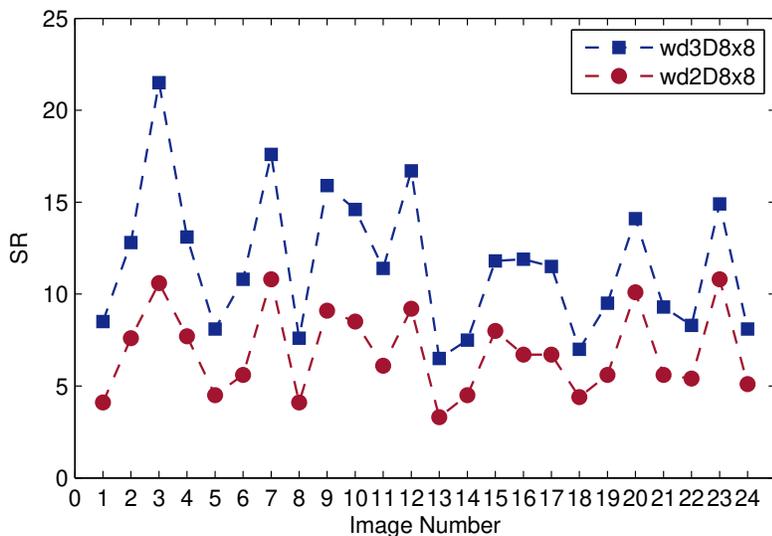

Figure 3: SR for the 45dB approximation, in the wd, of each of the 24 images in the Kodak data set (c.f. Fig. 2 enumerated from top left to bottom right). The results for the independent approximation of each 2D color channel are represented by the filled circles and those corresponding to the simultaneous approximation of the 3 channels are represented by the filled squares. The corresponding partitions are of size 8×8 and $8 \times 8 \times 3$.

3D SR=63.5 PSNR=38.4

2D SR=63.5 PSNR=24.8

3D SR=63.5; PSNR=35.9

2D SR=63.5; PSNR=22.1

3D SR=63.5; PSNR=37.6

2D SR=63.5; PSNR=24.9

Figure 4: Approximations of Images 3, 7 and 12 in the Kodak data set, for SR=63.5. The images on the left are the 3D approximations. The images on the right are the 2D channel by channel approximations.

Fig. 4 demonstrates the gain in visual quality obtained when the approximation of Images 3, 7 and 12 are realized simultaneously in 3D, instead of independently for each 2D channel. In both cases the SR is fixed at a high value SR=63.5. While the 3D approximation is still of good quality (c.f. images on the left in Fig. 4) the distortion of the channel by channel approximation is very noticeable even at the scale of the figure (c.f. images on the right in Fig. 4).

As a final remark it is worth noting that the number k_q of atoms in the approximation of each block q of an image partition produces a meaningful summary of local sparsity.

Figure 5: The upper graphs are a representation of the piecewise sparsity corresponding to Image 22 in the Kodak data set. Both graphs are arrays of 64×96 points. Each point corresponds to the number k_q of atoms in the approximation of a block q. The left graph corresponds to the 2D approximation and the right graph to the 3D approximation. The lower graph is the image given as 3 channels of 512×768 pixels each.

The upper graphs of Fig. 5 are a representation of the piecewise sparsity corresponding to Image 22 in the Kodak data set. Both graphs are arrays of 64×96 points. Each point corresponds to the number k_q of atoms in the approximation of a block q. The left graph

corresponds to block size 8×8 in the 2D approximation, by taking the average k_q over the three channels in the block, which is roughly the k_q-value corresponding to the equivalent block in the gray scale image. The right graph corresponds to k_q for each block of size $8 \times 8 \times 3$ in the 3D approximation. Both approximations are realized in the pd. The lower graph is the image given as 3 channels of 512×768 pixels each. It follows from the figure that the points corresponding to the 3D approximation give mode details about the image.

4.2   Example II

We consider now the approximation of the hyper-spectral images illustrated in Fig. 6. Details on the images acquisition and processing are described in [52–54].

Figure 6: Illustration of the hyper-spectral images available on [55] and [56]. From top left to bottom right in Table. 2 are labelled as Ribei., Graff., Rose, and Col. The of size of all four images is $1016 \times 1336 \times 32$ pixels.

All four images are of size $1016 \times 1336 \times 32$, and have been approximated in partitions of block size $N_b \times N_b$, with $N_b = 8, 16,$ and 24 for the 2D approximation, and $8 \times 8 \times 8$ for the

3D approximation. For the 2D channel by channel approximation we use the dictionaries $\mathcal{D}_{\mathrm{pd}}$ and $\mathcal{D}_{\mathrm{wd}}$ as defined in Sec. 3.4. For the 3D approximation we maintain the redundancy as in 2D using the dictionary $\tilde{\mathcal{D}}_{\mathrm{pd}}$ introduced Sec. 3.4 and $\tilde{\mathcal{D}}_{\mathrm{wd}}^{a} = \tilde{\mathcal{D}}_{\mathrm{pd}}^{a}$.

Because the range of intensity varies across the images, in order to compare SRs with different approaches we fix the Signal to Noise Ratio (SNR)

$$\mathrm{SNR} = 10 \log_{10} \left(\frac{\|\mathbf{I}\|_{\mathrm{3D}}^{2}}{\|\mathbf{I} - \mathbf{I}^{K}\|_{\mathrm{3D}}^{2}} \right). \tag{15}$$

Image	Ribei.	Graff.	Rose	Col.
SNR= 31 dB				
PSNR	46.8	48.2	47.8	46.7
$\mathrm{SR}_{\mathrm{2D}}\ N_b = 8$	19.2	19.2	24.1	47.7
Time (min)	1.6	1.6	1.3	0.9
$\mathrm{SR}_{\mathrm{2D}}\ N_b = 16$	27.3	25.5	38.7	110.6
Time (min)	3.4	3.8	2.1	1.1
$\mathrm{SR}_{\mathrm{2D}}\ N_b = 24$	29.6	26.8	44.2	147.5
Time (min)	7.6	9.2	4.5	1.5
$\mathrm{SR}_{\mathrm{3D}}\ N_b = 8$	49.1	59.7	74.6	137.2
Time (min)	18	15	10	6
SNR= 33 dB				
PSNR	48.8	50.2	49.8	48.7
$\mathrm{SR}_{\mathrm{2D}}\ N_b = 8$	15.2	15.4	19.3	41.5
Time (min)	2.3	2.1	1.7	1.1
$\mathrm{SR}_{\mathrm{2D}}\ N_b = 16$	20.4	19.5	29.1	86.4
Time (min)	5.4	5.6	2.9	1.2
$\mathrm{SR}_{\mathrm{2D}}\ Nb = 24$	21.9	20.5	32.7	106.3
Time (min)	12	14	6.8	1.9
$\mathrm{SR}_{\mathrm{3D}}\ N_b = 8$	33.5	41.6	53.2	106.5
Time (min)	25	21	16	8

Table 2: Values of SR for the approximation in the pixel-intensity domain of the images listed in the first row. $\mathrm{SR}_{\mathrm{2D}}$ indicates the SR for the plane by plane approximation in partition of block side $N_b = 8, 16$, and 24. $\mathrm{SR}_{\mathrm{3D}}$ corresponds to a partition in 3D blocks of size $8 \times 8 \times 8$. The times for completing the approximations are given immediately below the sparsity results in minutes.

Every block in the partition is approximated up to the same error. With all the approaches,

two global values of SNR (31 dB and 33 dB) were considered. These values of SNR correspond to the values of PSNR shown in Tables 2 and Table 3. In all of the cases the approximations are of excellent visual quality.

The SRs produced by the 3D approximation are indicated by SR_{3D} and those produced by the 2D plane by plane approximation by SR_{2D}. The times for completing the approximations are given in the row right after the corresponding sparsity result.

Image	Ribei.	Graff.	Rose	Col.
SNR= 31 dB				
PSNR	46.8	48.2	47.8	46.7
$\text{SR}_{2D}\ N_b = 8$	28.6	26.8	38.6	56.5
Time (min)	1.4	1.5	1.2	0.8
$\text{SR}_{2D}\ N_b = 16$	36.5	34.1	63.4	144.8
Time (min)	2.7	3.5	2.3	0.9
$\text{SR}_{2D}\ N_b = 24$	37.2	35.7	71.1	193
Time (min)	9.2	12	4.8	1.8
$\text{SR}_{3D}\ N_b = 8$	86.5	108.0	182.2	371.7
Time (min)	13	10	6	3
SNR= 33 dB				
PSNR	48.8	50.2	49.9	48.7
$\text{SR}_{2D}\ N_b = 8$	22.6	21.8	33.0	56.1
Time (min)	1.7	1.8	1.5	1.0
$\text{SR}_{2D}\ N_b = 16$	26.6	25.8	48.2	118.3
Time (min)	3.5	5.0	2.3	1.1
$\text{SR}_{2D}\ N_b = 24$	21.9	26.8	52.0	144.0
Time (min)	12	15	8.5	1.9
$\text{SR}_{3D}\ N_b = 8$	55.1	70.5	129.5	313.3
Time (min)	23	18	10	1.8

Table 3: Same description as in Table 2, but the approximations are realized by applying first a wavelet transform to each of the 32 channels.

Remark 3: In both Table 2 and Table 3 the values of SR_{3D} are significantly larger than the values of SR_{2D}, except for the Col. image and 24×24 blocks. For this image we were able to increase the 3D block size up to $16 \times 16 \times 16$ and the results for SNR $= 31$dB are $\text{SR}_{3D} = 357$ in the pd and $\text{SR}_{3D} = 892$ in the wd (35 min and 10 min respectively). For SNR $= 33$ dB

24

$SR_{3D} = 247$ in the pd and $SR_{3D} = 590$ in the wd (55 min and 20 min respectively).

On comparing the two tables a drastic improvement in the values of SR_{3D} is observed when the approximation is realized in the wavelet domain. This feature is a consequence of the fact that the planes of the natural images are very sparse in the wavelet domain. In order to highlight differences we produce next the SR_{3D} corresponding to the two remote sensing images in Fig. 7. The graph on the left represents the Urban remote sensing hyper-spectral image taken from [57]. The graph on the right is a portion of the University of Pavia image also taken from [57].

Figure 7: Illustration of two remote sensing hyper-spectral images taken from [57]. The graph on the left is the Urban image (size $320 \times 320 \times 128$ pixels). The graph on the right is a portion of the University of Pavia image ($256 \times 256 \times 96$ pixels).

Fig. 8 plots the SR vs four values of SNR, corresponding to the 3D approximations of the Urban and University of Pavia images in both the pd and wd.

Notice that the results in the pd are much closer to the results in the wd than they are in the case of the natural images in Fig. 6. This is because, as illustrated in Fig. 9, the planes of the remote sensing images are not as sparse in the wd as the planes of the natural images are.
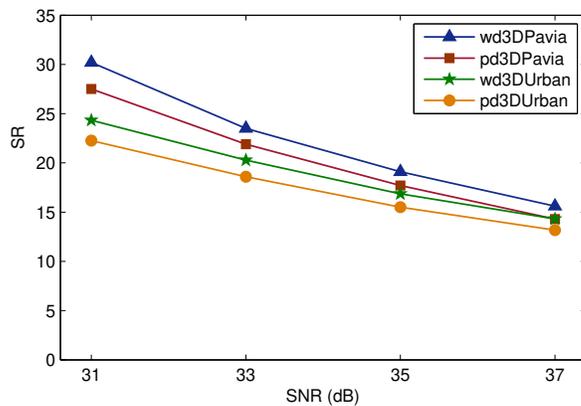
Figure 8: SR vs SNR values for the 3D approximation in both the pd and wd for the Urban and University of Pavia remote sensing images.
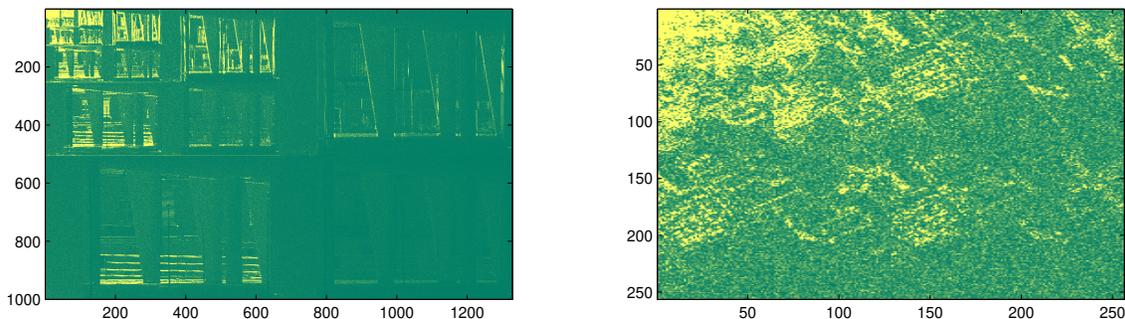

Figure 9: Absolute value of the wavelet transform of a plane in the Col. image (left graph) and in the University of Pavia image (right graph).

5 Conclusions

High quality approximation of 3D images has been considered within the context of data reduction. A remarkable improvement in sparsity achieved by the simultaneous approximation of multiple channels has been illustrated through numerical experiments of different natures. Firstly it was demonstrated that a standard data set of RGB images can be approximated at high quality using far fewer elementary components if each image is treated as a very thin 3D array instead of as 3 independent 2D arrays. Secondly the full power of the approach was demonstrated through the approximation of hyper-spectral images. For the hyper-spectral nat-

ural images the sparsity is remarkably higher if the approximation is realized in the wavelet domain. For the remote sensing images the domain of approximation has less influence because, as opposed to natural images, these images are not as sparse in the wavelet domain as natural images are.

Taking into account the major reduction of dimensionality demonstrated by the numerical examples in this work, we feel confident that the proposed approach will be of assistance to the broad range of image processing applications which rely on a transformation for data reduction as a first step of further processing.

Acknowledgments

References

[1] J. Wright, Yi Ma, J. Mairal, G. Sapiro, T.S. Huang, and S. Yan, "Sparse Representation for Computer Vision and Pattern Recognition", *Proc. of the IEEE* **98**, 1031 – 1044 (2010).

[2] M. Elad, Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing, Springer (2010).

[3] Z. Zhang, Y. Xu, J. Yang, X. Li, and D. Zhang, "A survey of sparse representation: algorithms and applications", *IEEE access* (2015).

[4] J. Mairal, M. Eldar and G. Sapiro, "Sparse Representation for Color Image Restoration", *IEEE Trans. Image Proces.,* **17**, 53 – 69 (2008).

[5] W. Dong, L. Zhang, G. Shi, and X. Li, "Nonlocally Centralized Sparse Representation for Image Restoration", *IEEE Trans. Image Proces.,* **22**, 1620 – 1630 (2013).

[6] J. Gou, B. Hou, W. Ou, Q. Mao, H. Yang, and Y. Liu, "Several Robust Extensions of Collaborative Representation for Image Classification" *Neurocomputing* , DOI: 10.1016/j.neucom.2018.06.089, (2018).

[7] J. Gou, Z. Yi, D. Zhang, Y. Zhan, X. Shen, and L Du, "Sparsity and Geometry Preserving Graph Embedding for Dimensionality Reduction, *IEEE Access* **6**, 75748 − 75766 10.1109/ACCESS.2018.2884027 (2018).

[8] J. Gou, L. Wang, Z. Yi, J. Lv, Q. Mao, and Y-H Yuan, "A New Discriminative Collaborative Neighbor Representation Method for Robust Face Recognition", *IEEE Access*, **6**, 74713 − 74727, 2018 DOI: 10.1109/ACCESS.2018.2883527 (2018).

[9] J. Wright, A. Y. Yang, and A. Ganesh, "Robust Face Recognition via Sparse Representation" *IEEE Trans. Pattern Analysis and Machine Intelligence,* **31,** 210 − 227 (2009).

[10] XT. Yuan, X. Liu, and S. Yan, "Visual Classification With Multitask Joint Sparse Representation", *IEEE Trans. Image Proces.,* **21**, 4349 − 4360 (2012).

[11] J. Yang, J. Wright, and T. Huang, "Image Super-Resolution via Sparse Representation," *IEEE Trans. Image Proces.,* **19**, 2861 − 2873 (2010).

[12] Y. Zhang, J. Liu, W. Yang, and Z. Guo, "Image Super-Resolution Based on Structure-Modulated Sparse Representation", *IEEE Trans. Image Proces.*, **9**, 2797 − 2810 (2015).

[13] K. Dabov, A. Foi, and V. Katkovnik, "Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering", *IEEE Trans. Image Proces.,***16**, 2080 − 2095 (2007).

[14] C. F. Caiafa and A. Cichocki, "Computing sparse representations of multidimensional signals Using Kronecker Bases", *Neural computation*, **25,** 186 − 220 (2013).

[15] A. Cichocki, D. Mandic, and L. De Lathauweri, G. Zhou, Q. Zhao, C. Caiafai, and H. A. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis", *IEEE Signal Processing Magazine*, **32**, 145–163 (2015).

[16] Q. Dai, S. Yoo, and A. Kappeler "Sparse Representation-Based Multiple Frame Video Super-Resolution", *IEEE Trans. Image Proces.*, **26**, 2080 – 2095 (2017).

[17] H. S. Mousavi and V. Monga, "Sparsity-based Color Image Super Resolution via Exploiting Cross Channel Constraints", *IEEE Trans. Image Proces.,* **26**, 5094 – 5106 (2017).

[18] L. Rebollo-Neira and J.Bowley, Sparse representation of astronomical images, *Journal of The Optical Society of America A,* Vol 30, 758-768 (2013).

[19] D. L. Donoho , Y. Tsaig , I. Drori , and J. Starck, "Stagewise Orthogonal Matching Pursuit", *IEEE Transactions on Information Theory*, **58**, 1094–1121 (2006).

[20] D. Needell and J.A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples", *Applied and Computational Harmonic Analysis*, **26**, 301–321 (2009).

[21] Y. Eldar, P. Kuppinger and H. Biölcskei, "Block-Sparse Signals: Uncertainty Relations and Efficient Recovery", *IEEE Trans. Signal Process.*, **58**, 3042–3054 (2010).

[22] L. Rebollo-Neira, R. Matiol, and S. Bibi, "Hierarchized block wise image approximation by greedy pursuit strategies," *IEEE Signal Process. Letters*, **20**, 1175–1178 (2013).

[23] L. Rebollo-Neira, `http://www.nonlinear-approx.info/examples/node09.html` (Last access April 2019).

[24] JianWen Tao, Shiting Wen, and Wenjun Hu, "Robust domain adaptation image classification via sparse and low rank representation", *Journal of Visual Communication and Image Representation,***33**, 134–148 (2015).

[25] Wei-Zhi Nie, An-An Liu, and Yu-Ting Su, "3D object retrieval based on sparse coding in weak supervision", *Journal of Visual Communication and Image Representation,* **37**, 40–45 (2016).

[26] Z. Gao, S. H. Li, Y. J. Zhu, C. Wang, and H. Zhang, "Collaborative sparse representation leaning model for RGBD action recognition", *Journal of Visual Communication and Image Representation,* **48**, 442–452 (2017).

[27] Youqiang Zhang, Guo Cao, Xuesong Li, and Bisheng Wang, "Cascaded Random Forest for Hyperspectral Image Classification", *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* **11**, 1082–1094 (2018).

[28] Guiji Li, Manman Peng, Ke Nai, Zhiyong Li, Keqin Li, "Visual tracking via context-aware local sparse appearance model", *Journal of Visual Communication and Image Representation,* **56**, Pages 92–105 (2018).

[29] S. Mallat and Z. Zhang, "Matching pursuit with time-frequency dictionaries," *IEEE Trans. Signal Process.,* Vol (41,12) 3397–3415 (1993).

[30] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," *Proc. of the 27th ACSSC,* **1**, 40–44 (1993).

[31] L. Rebollo-Neira and D. Lowe, "Optimized orthogonal matching pursuit approach", *IEEE Signal Process. Letters,* **9**, 137–140 (2002).

[32] L. Rebollo-Neira, J. Bowley, A. Constantinides and A. Plastino, "Self contained encrypted image folding", *Physica A,* **391**, 5858–5870 (2012).

[33] L. Rebollo-Neira, " Effective sparse representation of X-Ray medical images", *International Journal for Numerical Methods in Biomedical Engineering,* https://doi.org/10.1002/cnm.2886 (2017).

[34] L. Rebollo-Neira, M. Rozložník, and P. Sasmal, "Analysis of a low memory implementation of the Orthogonal Matching Pursuit greedy strategy", http://arxiv.org/abs/1609.00053 (2017).

[35] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, Te-Won Lee, and T. J. Sejnowski, "Dictionary Learning Algorithms for Sparse Representation", *Neurocomputing*, **15**, 349–396 (2003).

[36] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation", *IEEE Trans. Signal Process.* **54**, 4311–4322 (2006).

[37] R. Rubinstein, M. Zibulevsky, and M. Elad, "Double Sparsity: Learning Sparse Dictionaries for Sparse Signal Approximation", *IEEE Trans. Signal Process.* **58**, 1553–1564 (2010).

[38] I. Tošić and P. Frossard, "Dictionary Learning: What is the right representation for my signal?", *IEEE Signal Processing Magazine*, **28**, 27–38 (2011).

[39] J. Zepeda, C. Guillemot, E. Kijak, "Image Compression Using Sparse Representations and the Iteration-Tuned and Aligned Dictionary", *IEEE Journal of Selected Topics in Signal Processing,* **5**,1061–1073 (2011).

[40] S. Hawe, M. Seibert, and M. Kleinsteuber, "Separable dictionary learning", Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, 438–445 (2013).

[41] M. Srinivas, R. R. Naidu, C.S. Sastry, and C. Krishna Mohana, "Content based medical image retrieval using dictionary learning", *Neurocomputing*, **168**, 880–895 (2015).

[42] B. Wen, S. Ravishankar, and Y. Bresler, "'Structured Overcomplete Sparsifying Transform Learning with Convergence Guarantees and Applications", *International Journal of Computer Vision*, **114**, 137–167 (2015).

[43] L. Rebollo-Neira, "A competitive scheme for storing sparse representation of X-Ray medical images", *PLOS ONE.* `https://doi.org/10.1371/journal.pone.0201455` (2018).

[44] A. Cohen, I. Daubechies, and J.C. Feauveau, "Biorthogonal bases of compactly supported wavelets", *Comm. Pure and Appl. Math.*, **45**, 485–560 (1992).

[45] M. Andrle and L. Rebollo-Neira, "Cardinal B-spline dictionaries on a compact interval", *Appl. Comput. Harmon. Anal.* , **18**, 336-346 (2005)

[46] L. Rebollo-Neira and Z. Xu, "Adaptive non-uniform B-spline dictionaries on a compact interval", *Signal Processing,* **90,** 2308–2313 (2010).

[47] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, **13**, 600–612 (2004).

[48] I. Kowalik-Urbaniak, D. Brunet, J. Wang, D. Koff, N. Smolarski-Koff, E. Vrscay, B. Wallace, and Z. Wang, "The quest for 'diagnostically lossless' medical image compression: a comparative study of objective quality metrics for compressed medical images", Proc. SPIE 9037, Medical Imaging 2014: Image Perception, Observer Performance, and Technology Assessment, 903717 (March 11, 2014); doi:10.1117/12.2043196.

[49] R. Franzen, `http://r0k.us/graphics/kodak/` (Last access April 2019).

[50] D. H. Foster, S. M. C. Nascimento, K. Amano, "Information limits on neural identification of colored surfaces in natural scenes", *Visual Neuroscience*, **21**, 331–336 (2004).

[51] D. H. Foster, K. Amano, S.M.C. Nascimento, and M. J. Foster, "Frequency of metamerism in natural scenes." *Journal of the Optical Society of America A*, **23**, 2359–2372 (2006).

[52] S. M. C. Nascimento, K. Amano, and D. Forster, "Spatial distributions of local illumination color in natural scenes", *Vision Research*, **120**, 39–44 (2016).

[53] D. H. Foster, `https://personalpages.manchester.ac.uk/staff/d.h.foster/` `Hyperspectral_images_of_natural_scenes_04.html` (Last access April 2019).

[54] D. H. Foster, `http://personalpages.manchester.ac.uk/staff/d.h.foster/` `Local_Illumination_HSIs/Local_Illumination_HSIs_2015.html` (Last access April 2019).

[55] `http://lesun.weebly.com/hyperspectral-data-set.html` (Last access April 2019).