Toward Fast Transform Learning

Olivier Chabiron, François Malgouyres, Jean-Yves Tourneret, Nicolas Dobigeon

Noname manuscript No.
(will be inserted by the editor)

Toward Fast Transform Learning

Olivier Chabiron · François Malgouyres · Jean-Yves Tourneret · Nicolas Dobigeon

the date of receipt and acceptance should be inserted later

Abstract Dictionary learning is a matrix factorization problem. It aims at finding a dictionary of atoms that best represents an image or a class of images according to a given objective, usually sparsely. It has led to many state-of-the-art algorithms in image processing. In practice, all algorithms performing dictionary learning iteratively estimate the dictionary and a sparse representation of the images using this dictionary. However, the numerical complexity of dictionary learning restricts its use to atoms with a small support since the computations using the constructed dictionaries require too much resources to be deployed for large scale applications.

In order to alleviate these issues, this paper introduces a new strategy to learn dictionaries composed of atoms obtained by translating the composition of K convolutions with S-sparse kernels of known support. The dictionary update step associated with this strategy is a non-convex optimization problem. The purpose of the present paper is to study this non-convex problem. We first reformulate the problem to reduce the number of its irrelevant stationary points. A Gauss-Seidel type algorithm, referred to as Alternative Least Square Algorithm, is introduced for its resolution. The search space of the considered optimization problem is of dimension KS, which is typically smaller than the size of the target atom and is much smaller than the size of the image. Moreover, the complexity of the algorithm is linear with respect to the size of the image, allowing larger atoms to be learned (as opposed to small patches). The conducted experiments show that, when K is large (say $K = 10$), we are able to approximate with a very good accuracy many atoms such as wavelets, curvelets, sinc functions or cosines. We also argue empirically that surprisingly the algorithm generally converges to a global minimum for large values of K and S.

Olivier Chabiron is supported by ANR-11-LABX-0040-CIMI within the program ANR-11-IDEX-0002-02.

This work was performed during the Thematic Trimester on image processing of the CIMI Excellence Laboratory which was held in Toulouse, France, during the period May-June-July 2013.

Olivier Chabiron (E-mail: olivier.chabiron@n7.fr) · Jean-Yves Tourneret · Nicolas Dobigeon
Institut de Recherche en Informatique de Toulouse, IRIT-CNRS UMR 5505, ENSEEIHT, Toulouse, France

François Malgouyres
Institut de Mathématiques de Toulouse, IMT-CNRS UMR 5219, Université de Toulouse, Toulouse, France

Keywords dictionary learning · matrix factorization · fast transform · sparse representation · global optimization · Gauss-Seidel

1 Introduction

1.1 Problem Formulation

We consider $d \in \mathbb{N}$ and d-dimensional signals living in a domain $\mathcal{P} \subset \mathbb{Z}^d$ (i.e., $d = 1$ for 1D signals, $d = 2$ for 2D images,...). Typically, $\mathcal{P} = \{0,\dots,N-1\}^d$, where $N \in \mathbb{N}$ is the number of "pixels" along each axis. We consider an ideal target atom $H \in \mathbb{R}^{\mathcal{P}}$ which we want to recover. To fix ideas, one might think of the target atom as a curvelet in 2D or an apodized modified discrete cosine in 1D. A weighted sum $u \in \mathbb{R}^{\mathcal{P}}$ of translations of the target atom corrupted by additive noise is observed. More precisely, we are interested in measurements defined by

$$u = \alpha * H + b, \tag{1}$$

where $b \in \mathbb{R}^{\mathcal{P}}$ is an additive noise, $*$ stands for the circular discrete convolution[1] in dimension d and $\alpha \in \mathbb{R}^{\mathcal{P}}$ is a code of known coefficients. A simple example is obtained when α is a Dirac delta function. In this situation u reduces to a noisy version of H. Another interesting situation is when α is a sparse code. This situation turns out to be more favorable since, in that case, H is seen several times with different realizations of the noise. The typical framework we have in mind includes situations where α is a sparse code, and where α contains coefficients that have been estimated by dictionary learning (DL) strategies such as those described in Section 1.2. In such situations, a DL algorithm alternates an estimation of α and an estimation of H. Of course, α is only approximatively known and the stability of the proposed estimation of H with respect to the noise affecting α is crucial. Note finally that no assumption or constraint about the code α is required. However, the performance of an estimator of H from the data u defined in (1) clearly depends on the conditioning of the convolution with respect to the value of α.

The problem addressed in this paper consists of both estimating the unknown target atom H and expressing it as a composition of convolutions of sparse kernels. More precisely, we consider an integer $K \geq 2$ and K convolutions of sparse kernels $(h^k)_{1\leq k\leq K} \in (\mathbb{R}^{\mathcal{P}})^K$. We assume that all these kernels have less than a fixed number S of non-zero elements (i.e., that they are at most S-sparse). Furthermore, we assume that the support of the kernels (i.e., the locations in $\mathcal{P}$ of their non-zero elements) are known or pre-set. Similarly to the code α, the location of the non-zero elements can be designed manually or can be estimated by some other means. For instance, the supports could be obtained by alternating support and kernel estimations.

In order to manipulate the kernel supports, we define, for all $k \in \{1,\dots,K\}$, an injective support mapping $S^k \in \mathcal{P}^S$. The range of the support mapping is defined by

$$\mathrm{rg}\left(S^k\right) = \{S^k(1),\dots,S^k(S)\}.$$

The set of constraints on the support of h^k (denoted by $\mathrm{supp}\left(h^k\right)$) takes the form

$$\mathrm{supp}\left(h^k\right) \subset \mathrm{rg}\left(S^k\right) \quad ,\forall k \in \{1,\dots,K\}. \tag{2}$$

For 1D signals, examples of simple support mappings include $S^k(s) = k(s-1)$, $\forall s \in \{1,\dots,S\}$. A similar support is displayed in Figure 1 for 2D images. In addition to the support constraint (2), the convolution of the K kernels $\mathbf{h} = (h^k)_{1\leq k\leq K} \in (\mathbb{R}^{\mathcal{P}})^K$, should approximate the target atom H, i.e.,

$$h^1 * \cdots * h^K \approx H.$$

The motivations for considering such a decomposition are detailed in Section 1.2. They are both to approximate a large target atom H with a model containing few degrees of freedom and to obtain target atoms whose manipulation is numerically efficient. As an illustration, we mention the approximation of a curvelet target atom by a composition of convolutions that will receive a specific attention in our experiments (see Section 4.3.1).

Therefore, we propose to solve the following optimization problem

$$(P_0): \quad \begin{cases} \mathrm{argmin}_{\mathbf{h}\in(\mathbb{R}^{\mathcal{P}})^K} \| \alpha * h^1 * \cdots * h^K - u\|_2^2, \\ \text{subject to} \quad \mathrm{supp}\left(h^k\right) \subset \mathrm{rg}\left(S^k\right), \\ \forall k \in \{1,\dots,K\}. \end{cases}$$

[1] All the signals in $\mathbb{R}^{\mathcal{P}}$ are extended by periodization to be defined at any point in $\mathbb{Z}^d$.

where $\|.\|_2$ stands for the usual Euclidean[2] norm in $\mathbb{R}^{\mathcal{P}}$. For instance, in the favorable case where the code α is a Dirac delta function and $b = 0$ (noiseless case), the solution of (P_0) approximates the target atom H by a composition of sparse convolutions. At the other extreme, when the convolution with α is ill-conditioned and the noise is significant, the solution of (P_0) estimates the target atom H and regularizes it according to the composition of sparse convolution model.

The problem (P_0) is non convex. Thus, depending on the values of $K \geq 2$, $(S^k)_{1\leq k\leq K} \in (\mathcal{P}^S)^K$, $\alpha \in \mathbb{R}^{\mathcal{P}}$ and $u \in \mathbb{R}^{\mathcal{P}}$, it might be difficult or impossible to find a good approximation of a global minimizer of (P_0). The main objective of this paper is to study if such a problem lends itself to global optimization. Another important objective is to assess empirically if the computed compositions of convolutions provide good approximations of some atoms usually encountered in applications. The current paper gives empirical answers to these questions. In order to do so, it contains the description of an algorithm for solving (P_0) and its performance analysis.

Before describing the proposed algorithm, we mention some links between the optimization problem (P_0) and some known issues in sparse representation.

1.2 Motivations

The primary motivation for considering the observation model (1) comes from DL, which was pioneered by Lewicki and Sejnowski (2000); Olshausen and Field (1997) and has received a growing attention during the last ten years. It can be viewed as a way of representing data using a sparse representation. We invite the reader to consult the book written by Elad (2010) for more details about sparse representations and DL. Given a set of L images[3] $(u^l)_{1\leq l\leq L} \in (\mathbb{R}^{\mathcal{P}})^L$, the archetype of the DL strategy is to look for a dictionary as the solution of the following optimization problem

$$\text{argmin}_{\mathbb{H},(\alpha^l)_{1\leq l\leq L}} \quad \sum_{l=1}^{L} \|\mathbb{H}\alpha^l - u^l\|_2^2 + \lambda\|\alpha^l\|_*,$$

where $\mathbb{H}$ is a matrix whose columns have a bounded norm and form the atoms of the dictionary, $\lambda \geq 0$ is a regularization parameter and $\|.\|_*$ is a sparsity-inducing norm such as the counting function (or ℓ_0 pseudo-norm) or the usual ℓ_1 norm. The DL optimization problem is sometimes formulated by imposing a constraint on $\|\alpha^l\|_*$. The resulting non-convex problem can be solved (or approximatively solved) by many methods including the "method of optimal direction" (MOD) (Engan et al, 1999) and, in a different manner, by K-SVD (Aharon et al, 2006). To better reflect the distribution of images, it can also be useful to increase the number of images and to use an online strategy (Mairal et al, 2010). Finally, note that an alternative model has been presented for task driven DL by Mairal et al (2012). Algorithmically, all these approaches rely on alternatively updating the codes $(\alpha^l)_{1\leq l\leq L}$ and the dictionary $\mathbb{H}$.

The problem considered in the current paper mimics an update step of the dictionary. In this context, α is fixed and the target atom H is a column of the dictionary $\mathbb{H}$. The dictionary $\mathbb{H}$ is made of translations of the target atom H. The main novelty of the proposed approach is to impose the learned atoms to be a composition of convolutions of sparse kernels. The interest for such a constraint is that it provides numerically effective dictionaries and permits to consider larger atoms. Indeed, the reconstruction operator

$$\begin{aligned} &\mathbb{R}^{\mathcal{P}} \longrightarrow \mathbb{R}^{\mathcal{P}} \\ &\alpha \longmapsto \alpha * h^1 * \ldots * h^K \end{aligned}$$

and its adjoint can be computed by K convolutions with kernels of size S. As a consequence, the computation of the reconstruction operator and its adjoint have a computational complexity of $O(KS\#\mathcal{P})$, where $\#\mathcal{P}$ denotes the cardinality of the set $\mathcal{P}$. Depending on the support mappings

[2] $\mathbb{R}^{\mathcal{P}}$ and $\mathbb{R}^S$ are endowed with the usual scalar product denoted $\langle .,. \rangle$ and the usual Euclidean norm denoted $\|\cdot\|_2$. We use the same notation whatever the vector space. We expect that the notation will not be ambiguous, once in context.

[3] Usually, DL is applied to small images such as patches extracted from large images.

$(S^k)_{1\leq k\leq K}$, this complexity can be much smaller than a convolution with a kernel filling the "reachable support"

$$\mathcal{S} = \left\{ p \in \mathcal{P},\ \exists p_1 \in \mathrm{rg}\left(S^1\right), \ldots, p_K \in \mathrm{rg}\left(S^K\right), \sum_{k=1}^{K} p_k = p \right\}. \quad (3)$$

In the latter case, the computational complexity is indeed equal to $O(\#\mathcal{S}\#\mathcal{P})$ or $O(\#\mathcal{P}\log(\#\mathcal{P}))$ if the convolutions are computed using a Fast Fourier Transform (FFT).

Moreover, when several target atoms are considered, the convolutions of sparse kernels can be arranged according to a tree structure to save even more computing resources. The typical example of an existing dictionary having a similar structure is the dictionary made of undecimated wavelets (Starck et al, 2007) or undecimated wavelet packets.

Let us detail an example of a fast transform learning model that can benefit from the current study. Consider a tree and associate to each edge $\mathfrak{e} \in \mathfrak{E}$ of the tree a sparse kernel $h^{\mathfrak{e}} \in \mathbb{R}^{\mathcal{P}}$ and a support mapping $S^{\mathfrak{e}} \in \mathcal{P}^S$; denote as $\mathfrak{L}$ the set of all the leaves $\mathfrak{l}$ of the tree; denote as $\alpha^{\mathfrak{l}} \in \mathbb{R}^{\mathcal{P}}$ the coefficients of the leaf $\mathfrak{l} \in \mathfrak{L}$ and as $c(\mathfrak{l})$ the path containing all the edges linking the root of the tree to leaf $\mathfrak{l}$, for every $\mathfrak{l} \in \mathfrak{L}$. The reconstruction of a code $\alpha = (\alpha^{\mathfrak{l}})_{\mathfrak{l}\in\mathfrak{L}} \in \left(\mathbb{R}^{\mathcal{P}}\right)^{\mathfrak{L}}$ with the fast transform defined by the proposed tree can be defined as

$$\mathbb{H}\alpha = \sum_{\mathfrak{l}\in\mathfrak{L}} \alpha^{\mathfrak{l}} * \left(*_{\mathfrak{e}\in c(\mathfrak{l})} h^{\mathfrak{e}} \right),$$

where $*_{\mathfrak{e}\in c(\mathfrak{l})} h^{\mathfrak{e}}$ denotes the composition of convolutions between all the kernels associated with the edges of the path $c(\mathfrak{l})$. The adjoint of $\mathbb{H}$ is easily established given this formula.

A dictionary learning problem can then be defined as follows:

$$\begin{array}{ll} \text{argmin} & \sum_{l=1}^{L} \|\mathbb{H}\alpha^l - u^l\|_2^2 + \gamma\|\alpha^l\|_* \\ \text{subject to} & \mathbf{h} \in (\mathbb{R}^{\mathcal{P}})^{\mathfrak{E}}, (\alpha^l) \in ((\mathbb{R}^{\mathcal{P}})^{\mathfrak{L}})^L, \\ \text{and} & \mathrm{supp}\left(h^{\mathfrak{e}}\right) \subset \mathrm{rg}\left(S^{\mathfrak{e}}\right), \forall \mathfrak{e} \in \mathfrak{E}, \\ \text{and} & \|h^{\mathfrak{e}}\|_2 \leq 1, \forall \mathfrak{e} \in \mathfrak{E}. \end{array}$$

When $L = 1$ and the tree only contains one leaf, the dictionary update is exactly the problem we are considering in this paper (modulo the constraint $\|h^{\mathfrak{e}}\|_2 \leq 1$). In particular, it seems impossible to solve the dictionary update of the above problem if we are not able to solve the problem (P_0). In other words, solving the problem (P_0) is a step toward fast transform learning (hence the name of the paper).

To conclude with the motivations, having a numerically effective scheme for using a dictionary is crucial since the computational complexity of most algorithms favoring sparsity is proportional to the computational complexity of the matrix-vector multiplications involving $\mathbb{H}$ and its transpose. In particular, for the DL algorithms alternating a sparse coding step and a dictionary update step, the sparse coding steps require less computational resources. These resources are therefore available for the dictionary update.

1.3 Related Works

Before going ahead, it is interesting to describe the structures of the dictionaries that have been considered in DL. Structured and parametric dictionaries have recently been considered with increasing interest. Interested readers can find a concise bibliographical note on that subject by Rubinstein et al (2010a). In particular, the structures studied so far include orthobases (Dobigeon and Tourneret, 2010) and unions of orthobases (Lesage et al, 2005), translation invariant dictionaries (Mailhé et al, 2008), concatenation of learned and fixed dictionaries (Peyré et al, 2010), dictionaries composed of patches with multiple sizes (Mairal et al, 2008), dictionaries divided into ordered pieces (Thiagarajan et al, 2011), structures induced by structured codes (Jenatton et al, 2010, 2011), and tight frames (Cai et al, 2013). Other interesting dictionaries are characterized by several layers. These dictionaries can be constructed as the composition of a fixed transform and learned dictionaries (Rubinstein et al, 2010b; Ophir et al, 2011). Dictionaries made of two layers based on a sparsifying transform and a sampling matrix (both layers can be learned by the algorithm investigated by Duarte-Carvajalino and Sapiro (2009))

have also been considered. Another attempt requires two layers to build separable atoms (Rigamonti et al, 2013). To the best of our knowledge, there only exists a few attempts for building dictionaries involving an arbitrary number of layers. In a slightly different context, dictionaries structured by Kronecker products have been proposed by Tsiligkaridis et al (2013). Interestingly, despite the non-convexity of the corresponding energy, it is possible to find some of its global minima (Wiesel, 2012). Finally, dictionaries structured by wavelet-like trees (similar to one we are targeting in this paper) using a dictionary update based on a gradient descent have been studied by Sallee and Olshausen (2002).

When compared to the dictionaries mentioned in this Section, the structure of the proposed dictionary aims at obtaining a numerically efficient translation invariant dictionary, whose elementary atoms H can have large supports. Moreover, the update of the proposed structured dictionary reduces to a global optimization problem. Surprisingly, the proposed algorithm provides interesting solutions for relatively large values of the number of layers K, e.g., $K = 10$ seems very reasonable.

It is interesting to mention that the decomposition of H as a convolution of K kernels makes the problem similar to the design of filter-banks that has received a considerable attention in the wavelet community. For instance, filters defined as convolutions of high-pass and low-pass kernels with perfect reconstruction properties have been studied in (Delsarte et al, 1992) and (Macq and Mertes, 1993). These filters are determined by maximizing an appropriate coding gain for image compression applications. Other methods for designing FIR and IIR filters are also mentioned in the review paper (Lu and Antoniou, 2000) (based on weighted least-squares or on a minimax approach). Finally, we would like to point out that the filters resulting from our algorithm can vary from scale to scale, as for for the "non-stationary" wavelet transform (Uhl, 1996) or wavelet-packets (Cohen and Séré, 1996). The main novelty of the proposed work is that our filters are constructed as a composition of convolutions with sparse kernels, which cannot be obtained with the existing methods.

1.4 Paper Organization

The paper is organized as follows. Section 1 formulates the proposed dictionary update and provides motivations with references to previous works. A more practical problem formulation is introduced in Section 2. Section 3 presents an algorithm for approximating a dictionary atom as a composition of convolutions, in order to build a fast transform. The algorithm is based on an alternating least squares strategy whose steps are detailed carefully. Simulation results illustrating the performance of the proposed algorithm and its convergence properties are provided in Sections 4 and 5. Conclusions and future work are reported in Section 6.

2 Reformulating (P_0)

The problem (P_0) is not very tractable because it has many stationary points. Denote as $\mathbf{h} = (h^k)_{1\leq k\leq K} \in (\mathbb{R}^{\mathcal{P}})^K$ the sequence of kernels and as E the objective function of (P_0)

$$E(\mathbf{h}) = \| \alpha * h^1 * \cdots * h^K - u\|_2^2.$$

The gradient of E is

$$\nabla E(\mathbf{h}) = \left(\frac{\partial E}{\partial h^1}(\mathbf{h}), \ldots, \frac{\partial E}{\partial h^K}(\mathbf{h}) \right),$$

where $\frac{\partial E}{\partial h^k}$ denotes the partial differential of the energy function E, for any $k \in \{1, \ldots, K\}$. The latter can be calculated easily, leading to

$$\frac{\partial E}{\partial h^k}(\mathbf{h}) = 2\tilde{H}^k * (\alpha * h^1 * \cdots * h^K - u), \tag{4}$$

where

$$H^k = \alpha * h^1 * \cdots * h^{k-1} * h^{k+1} * \cdots * h^K, \tag{5}$$

and where the operator $\tilde{}$ is defined for any $h \in \mathbb{R}^{\mathcal{P}}$ as

$$\tilde{h}_p = h_{-p}, \quad \forall p \in \mathcal{P}. \tag{6}$$

Note that the notation H^k has been used instead of $H^k(\mathbf{h})$ to improve readability.

As soon as $h^{k_1} = h^{k_2} = 0$ for two distinct values of k_1 and $k_2 \in \{1,\dots,K\}$, we have $H^k = 0$, for all $k \in \{1,\dots,K\}$, and thus

$$\frac{\partial E}{\partial h^k}(\mathbf{h}) = 0 \quad \forall k \in \{1,\dots K\}.$$

As a consequence, nothing prevents a minimization algorithm solving (P_0) to get stuck at one of these stationary points, although it is usually not a global minimizer of (P_0).

Furthermore, $\forall \mathbf{h} \in (\mathbb{R}^{\mathcal{P}})^K$ and $\forall (\mu_k)_{1\leq k\leq K} \in \mathbb{R}^K$ such that $\prod_{k=1}^K \mu_k = 1$, we have

$$E\left[(\mu_k h^k)_{1\leq k\leq K}\right] = E(\mathbf{h}),$$

while, for any $k \in \{1,\dots,K\}$,

$$\frac{\partial E}{\partial h^k}\left[(\mu_k h^k)_{1\leq k\leq K}\right] = \frac{1}{\mu_k}\frac{\partial E}{\partial h^k}(\mathbf{h}).$$

This results in an unbalanced situation where the partial differentials and the gradient are large along directions of small kernels. These kernels are therefore favoured which does not seem justified.

To address the two issues mentioned above and reduce the number of irrelevant stationary points, we propose to include an additional constraint for the norms of the kernels $h^k \in \mathbb{R}^{\mathcal{P}}$, $\forall k \in \{1,\dots,K\}$. More precisely, we consider a norm-to-one constraint $\|h^k\|_2 = 1$, $\forall k \in \{1,\dots,K\}$ and introduce an additional scaling factor $\lambda \geq 0$, to scale the result according to the target atom. To simplify notations, we write

$$\mathcal{D} = \Big\{ \mathbf{h} = (h^k)_{1\leq k\leq K} \in (\mathbb{R}^{\mathcal{P}})^K \,|\, \forall k \in \{1,\dots,K\}, \\ \|h^k\|_2 = 1 \text{ and } \operatorname{supp}\left(h^k\right) \subset \operatorname{rg}\left(S^k\right) \Big\}$$

and define the following optimization problem

$$(P_1): \quad \operatorname{argmin}_{\lambda\geq 0, \mathbf{h}\in\mathcal{D}} \|\lambda\, \alpha * h^1 * \cdots * h^K - u\|_2^2.$$

Let us now analyze the properties of the optimization problem (P_1).

Proposition 1 (Existence of a solution) *For any* $(u, \alpha, (S^k)_{1\leq k\leq K}) \in \left(\mathbb{R}^{\mathcal{P}} \times \mathbb{R}^{\mathcal{P}} \times (\mathcal{P}^S)^K\right)$, *if*

$$\forall \mathbf{h} \in \mathcal{D}, \quad \alpha * h^1 * \dots * h^K \neq 0, \tag{7}$$

then the problem (P_1) *has a minimizer.*

This property relies on the regularity of the objective function and the compacity/coercivity of the problem. Its proof is detailed in Appendix.

Note that there might be refined alternatives to the condition (7). However, the investigation of the tightest condition for the existence of a minimizer of (P_1) is clearly not the subject of this paper. Concerning the existence of a solution, note that the objective function of (P_1) is not necessarily coercive, e.g., it is not coercive if there exists $\mathbf{h} \in (\mathbb{R}^{\mathcal{P}})^K$ such that $\alpha * h^1 * \dots * h^K = 0$. In this situation, a minimizing sequence might be such that $\lambda\alpha * h^1 * \dots * h^K$ and $(h^k)_{1\leq k\leq K}$ have accumulation points whereas $\alpha * h^1 * \dots * h^K$ and λ go towards 0 and infinity. Note finally that we typically expect the condition (7) to hold as soon as the supports $(S^k)_{1\leq k\leq K} \in (\mathcal{P}^S)^K$ and $\operatorname{supp}(\alpha)$ are sufficiently localized. In our experiments, we have never encountered a situation where $\alpha * h^1 * \dots * h^K$ equals zero.

We also have:

Proposition 2 ((P_1) is equivalent to (P_0)) *Let* $(u, \alpha, (S^k)_{1\leq k\leq K}) \in \left(\mathbb{R}^{\mathcal{P}} \times \mathbb{R}^{\mathcal{P}} \times (\mathcal{P}^S)^K\right)$ *be such that* (7) *holds. For any* $(\lambda, \mathbf{h}) \in \mathbb{R} \times (\mathbb{R}^{\mathcal{P}})^K$, *we consider the kernels* $\mathbf{g} = (g^k)_{1\leq k\leq K} \in (\mathbb{R}^{\mathcal{P}})^K$ *defined by*

$$g^1 = \lambda\, h^1 \text{ and } g^k = h^k, \quad \forall k \in \{2,\dots,K\}. \tag{8}$$

The following statements hold:

1. *if* $(\lambda, \mathbf{h}) \in \mathbb{R} \times (\mathbb{R}^{\mathcal{P}})^K$ *is a stationary point of* (P_1) *and* $\lambda > 0$ *then* $\mathbf{g}$ *is a stationary point of* (P_0).
2. *if* $(\lambda, \mathbf{h}) \in \mathbb{R} \times (\mathbb{R}^{\mathcal{P}})^K$ *is a global minimizer of* (P_1) *then* $\mathbf{g}$ *is a global minimizer of* (P_0).

The proof relies on the homogeneity of the problems (P_0) and (P_1). The proof of the proposition is detailed in Appendix.

To conclude this part, it is interesting to mention some structural properties of problem (P_1). The objective function of (P_1) is a polynomial of degree $2K$. Thus, it is infinitely differentiable

and non-negative. The objective function of (P_1) is non-convex. However, for any $k \in \{1,\dots,K\}$, the objective function of (P_1) is marginally quadratic and convex with respect to h^k. Finally, $\mathcal{D}$ is a smooth but non convex set. It is not difficult to check that the following mapping provides an orthogonal projection onto $\mathcal{D}$:

$$(\mathbb{R}^{\mathcal{P}})^K \longrightarrow \mathcal{D}$$
$$(h^k)_{1\le k\le K} \longmapsto (\overline{h}^k)_{1\le k\le K},$$

where

$$\overline{h}^k = \begin{cases} \frac{h^k \mathbb{1}_{\mathrm{rg}(S^k)}}{\|h^k \mathbb{1}_{\mathrm{rg}(S^k)}\|_2} & \text{, if } \|h^k \mathbb{1}_{\mathrm{rg}(S^k)}\|_2 \neq 0, \\ \frac{1}{\sqrt{S}} \mathbb{1}_{\mathrm{rg}(S^k)} & \text{, otherwise,} \end{cases}$$

where $\mathbb{1}_{\mathrm{rg}(S^k)}$ is the characteristic function of $\mathrm{rg}\left(S^k\right)$.

3 The alternating least squares algorithm

3.1 Principle of the algorithm

The objective function in (P_1) being non-convex, there is in general no guarantee to find a global or a local minimum of (P_1). However, it makes sense to build a method finding a stationary point of (P_1). Also, because the considered problem has similarities with the best rank 1 approximation of tensors, we have considered an algorithm inspired from a well known algorithm solving this tensor problem: The alternating least squares (ALS) algorithm (De Lathauwer et al, 2000). This ALS algorithm alternates minimizations with respect to the kernels h^k, $\forall k \in \{1,\dots,K\}$. The resulting algorithm is often referred to as a "Gauss-Seidel" or "block coordinate descent". Although our convergence analysis will not rely on these results let us mention that some convergence properties of these algorithms have been studied in (Luo and Tseng, 1992; Grippo and Sciandrone, 2000; Razaviyayn M. et al, 2013; Attouch et al, 2013). As we will see, the ALS algorithm takes advantage of the fact that, when all the kernels but one are fixed, the objective function is a quadratic function of this latter kernel. As a consequence, every step of the algorithm will have a closed form solution and thus has a low complexity.

Using a better minimization algorithm might help to reduce the time required for the optimization. Among the alternating strategies, we can think of proximal Gauss-Seidel strategy (see (Attouch et al, 2010)) or proximal alternating linerarized minimization (see (Bolte et al, 2013)) or finally a variant (see Chouzenoux et al (2013)). Also, gradient descent or quasi-Newton algorithms might provide good convergence rates. Finally, the reader can find standard results on all the issues related to optimization in (Bertsekas, 2003).

More precisely, for any $k \in \{1,\dots,K\}$, we propose to (alternatively) solve the following least squares (LS) problems

$$(P_k): \quad \begin{cases} \mathrm{argmin}_{\lambda\ge 0, h\in\mathbb{R}^{\mathcal{P}}} \|\lambda\alpha * h^1 * \cdots * h^{k-1} \\ \qquad\qquad * h * h^{k+1} * \ldots * h^K - u\|_2^2, \\ \text{subject to} \quad \mathrm{supp}\,(h) \subset \mathrm{rg}\left(S^k\right) \\ \text{and } \|h\|_2 = 1. \end{cases}$$

where the kernels $(h_p^{k'})_{p\in\mathcal{P}}$ are fixed $\forall k' \neq k$. The resulting alternating least square (ALS) algorithm is described in Algo. 1.

Algorithm 1: ALS algorithm

Input:
u: target measurements;
α: known coefficients;
$(S^k)_{1\le k\le K}$: supports of the kernels $(h^k)_{1\le k\le K}$.
Output:
λ and kernels $(h^k)_{1\le k\le K}$ such that $\lambda h^1 * \ldots * h^K \approx H$.

begin
  Initialize the kernels $(h^k)_{1\le k\le K}$;
  while not converged **do**
    for k = 1 ,..., K **do**
      Update λ and h^k with a minimizer of (P_k).

3.2 Resolution of (P_k)

Before studying the existence of a minimizer of (P_k), let us rewrite the problem (P_k) in a simpler form. Since the embedding from $\mathbb{R}^S$ in $\mathrm{rg}\left(S^k\right) \subset$

$\mathbb{R}^{\mathcal{P}}$ and the operator
$\mathbb{R}^{\mathcal{P}} \longrightarrow \mathbb{R}^{\mathcal{P}}$

$$h \longmapsto \alpha * h^1 * \cdots * h^{k-1} * h * h^{k+1} * \ldots * h^K,$$

are linear, their composition can be described by a matrix-vector product $C_k h$, where the vector $h \in \mathbb{R}^S$ and C_k is a $(\#\mathcal{P}) \times S$ matrix. (The matrix C_k will be detailed in Section 3.3.)

A solution of (P_k) can therefore be constructed by embedding in $\mathrm{rg}\left(S^k\right) \subset \mathbb{R}^{\mathcal{P}}$ a solution of the equivalent problem (still denoted (P_k), for simplicity)

$$(P_k): \quad \begin{cases} \operatorname{argmin}_{\lambda \geq 0, h \in \mathbb{R}^S} \|\lambda C_k h - u\|_2^2 \\ \text{subject to} \quad \|h\|_2 = 1. \end{cases}$$

where we consider that u has been vectorized. In order to solve this problem, we define

$$(P'_k): \quad \operatorname{argmin}_{h \in \mathbb{R}^S} \|C_k h - u\|_2^2.$$

The problem (P'_k) is a LS problem which has a minimizer $h^* \in \mathbb{R}^S$. Moreover, the gradient of its objective function is

$$C_k^T (C_k h - u).$$

Finally, by computing a stationary point of the problem (P'_k), we obtain:

$$h^* = (C_k^T C_k)^\dagger C_k^T u, \tag{9}$$

where $(C_k^T C_k)^\dagger$ is the pseudo-inverse of $C_k^T C_k$. Setting

$$\lambda = \|h^*\|_2 \quad \text{and} \quad h^k = \begin{cases} \frac{h^*}{\|h^*\|_2} \text{ , if } \|h^*\|_2 \neq 0, \\ \frac{1}{\sqrt{S}} \mathbb{1} \text{ , otherwise} \end{cases} \tag{10}$$

where $\mathbb{1} \in \mathbb{R}^S$ is a vector of ones. It is easy to check that we always have $h^* = \lambda h^k$. One can also show that any $(\mu, g) \in \mathbb{R} \times \mathbb{R}^S$ satisfying the constraints of (P_k) is such that:

$$\begin{aligned} \|\lambda C_k h^k - u\|_2^2 &= \|C_k h^* - u\|_2^2, \\ &\leq \|C_k(\mu g) - u\|_2^2 = \|\mu C_k g - u\|_2^2. \end{aligned}$$

As a consequence, (P_k) has a minimizer defined by (9) and (10). Moreover, note that if (λ', h') is a solution of (P_k), we can easily check that $\lambda' h'$ is a minimizer of (P'_k). The latter being unique when C_k is full column rank, we know that the solution of (P_k) is unique under that same condition.

Altogether, we obtain the update rule by embedding in $\mathrm{rg}\left(S^k\right) \subset \mathbb{R}^{\mathcal{P}}$ the solution described by (9) and (10). In order to apply these formulas, the main computational difficulties are to compute $C_k^T u$, $C_k^T C_k$ and the pseudo-inverse of $C_k^T C_k$. These computations are the subject of the next paragraph.

3.3 Computing $C_k^T u$ and $C_k^T C_k$

Considering Dirac delta functions for $h \in \mathbb{R}^S$ and the linearity of C_k, we obtain for any $h \in \mathbb{R}^S$

$$(C_k h)_p = \sum_{s=1}^{S} H^k_{p - S^k(s)} h_s, \quad \forall p \in \mathcal{P}$$

where H^k is defined in (5). In other words, each column of C_k is a vectorization of $(H^k_{p-S^k(s)})_{p \in \mathcal{P}}$. For any $p' \in \mathcal{P}$, denote as $\tau_{p'}$ the translation operator such that $(\tau_{p'} v)_p = v_{p-p'}$, $\forall (v, p) \in \mathbb{R}^{\mathcal{P}} \times \mathcal{P}$. Using this notation, the sth column of C_k is a vectorization of $\tau_{S^k(s)} H^k$. Therefore, the sth line of C_k^T is the transpose of a vectorization of $\tau_{S^k(s)} H^k$. We finally have

$$(C_k^T v)_s = \langle \tau_{S^k(s)} H^k, v \rangle, \quad \forall v \in \mathbb{R}^{\mathcal{P}}. \tag{11}$$

Note that the computational complexity for computing H^k is $O((K-1)S\#\mathcal{P})$. Once H^k has been computed, the cost for computing $(C_k^T u)_s$ is $O(\#\mathcal{P})$, $\forall s \in \{1, \ldots, S\}$, and therefore the cost for computing $C_k^T u$ is $O(S\#\mathcal{P})$. Altogether, we obtain a complexity $O(KS\#\mathcal{P})$.

We can immediately deduce the form of $C_k^T C_k$. Indeed, each of its column is obtained by applying (11) in which we replace v by the column vector $\tau_{S^k(s')} H^k$, for some $s' \in \{1, \ldots, S\}$. Therefore the coefficient of $C_k^T C_k$ at the location $(s, s') \in \{1, \ldots, S\}^2$ is

$$(C_k^T C_k)_{s,s'} = \langle \tau_{S^k(s)} H^k, \tau_{S^k(s')} H^k \rangle. \tag{12}$$

This Gram matrix is symmetric, positive semidefinite and of size $S \times S$. Once H^k has been computed, the computational complexity for computing $C_k^T C_k$ is $O(S^2 \#\mathcal{P})$. The computation of its pseudo-inverse is a well studied problem and is a step of the algorithm that can be optimized. An off-the-shelf implementation using a singular value decomposition (SVD) typically requires $O(S^3)$ operations.

Algorithm 2 summarizes all the steps required for the proposed ALS algorithm. The overall computational complexity is typically $O((K+S)KS\#\mathcal{P})$ per iteration of the while loop[4]. It can be reasonably applied in situations where $KS(K+S)$ is not to large. The most demanding case considered in the experiments described in this paper corresponds to $KS^2 = 6250$ (corresponding to $K = 10$ and $S = 25$). In order to choose the number of iterations in the while loop, we have used the relative difference between the values of the objective function of (P_k) for two consecutive iterations. When this difference is lower than 10^{-4}, we consider that we have reached a stationary point, and the algorithm stops.

Algorithm 2: Detailed ALS algorithm

Input:
u: target measurements;
α: known coefficients;
$(S^k)_{1\leq k\leq K}$: supports of the kernels $(h^k)_{1\leq k\leq K}$.
Output:
$(h^k)_{1\leq k\leq K}$: convolution kernels such that
$h^1 * \ldots * h^K \approx H$.

begin
  Initialize the kernels $((h_p^k)_{p\in\mathcal{P}})_{1\leq k\leq K}$;
  while not converged **do**
    for k = 1 ,..., K **do**
      Compute H^k according to (5)
        $O((K-1)S\#\mathcal{P})$
      Compute $C_k^T C_k$ and $C_k^T u$ according to (12) and (11) ;
        $O((S+1)S\#\mathcal{P})$
      Compute h^* according to (9);
        $O(S^3)$
      Update h^k and λ according to (10) ;
        $O(S)$

3.4 Convergence of the algorithm

Before stating the convergence result, let us give a few notations.

First, notice that the result of an iteration of the for loop in Algorithm 2 only depends on the initial kernels $\mathbf{h} \in \mathcal{D}$ and not on the initial scaling factor λ. If we consider an initial condition $\mathbf{h} \in \mathcal{D}$ of the for loop in Algorithm 2, we denote the initial condition of the kth iteration by $T_k(\mathbf{h})$. For instance, we have $T_1(\mathbf{h}) = \mathbf{h}$. We also denote the scaling factor and the kernels resulting from the whole for loop by $T(\mathbf{h})$. More precisely, denoting as $(\lambda^n, \mathbf{h}^n)_{n\in\mathbb{N}}$ the sequence generated by Algorithm 2, we have for all $n \in \mathbb{N}$

$$(\lambda^{n+1}, \mathbf{h}^{n+1}) = T(\mathbf{h}^n).$$

Proposition 3 (Convergence of Algorithm 2) *For any* $(u, \alpha, (S^k)_{1\leq k\leq K}) \in \left(\mathbb{R}^{\mathcal{P}} \times \mathbb{R}^{\mathcal{P}} \times (\mathcal{P}^S)^K\right)$, *if*

$$\alpha * h^1 * \ldots * h^K \neq 0, \quad \forall \mathbf{h} \in \mathcal{D}, \tag{13}$$

then the following statements hold:

1. *The sequence generated by Algorithm 2 is bounded and its limit points are in* $\mathbb{R} \times \mathcal{D}$. *The value of the objective function is the same for all these limit points.*
2. *For any limit point* $(\lambda^*, \mathbf{h}^*) \in \mathbb{R} \times \mathcal{D}$, *if for all* $k \in \{1, \ldots, K\}$, *the matrix* C_k *generated using* $T_k(\mathbf{h}^*)$ *is full column rank and* $C_k^T u \neq 0$, *then* $(\lambda^*, \mathbf{h}^*) = T(\mathbf{h}^*)$ *and* $(\lambda^*, \mathbf{h}^*)$ *is a stationary point of the problem* (P_1).

The proof relies on the fact that the objective function is coercive, smooth, that each iteration of the algorithm is a regular mapping that makes the value of the objective function decrease. It also exploits the fact that every problem (P_k) has a unique solution. The detailed proof of the proposition is given in Appendix.

3.5 Initialization of the algorithm and restart

First, it is interesting to note that the ALS algorithm does not need any initialization for λ. Moreover, the initial kernel values $(h^k)_{1\leq k\leq K}$ must satisfy the constraints and therefore belong to $\mathcal{D}$. When the problem (P_1) has a global minimizer, we denote by $\mathbb{I} \subset \mathcal{D}$ the non-empty convergence set such that the ALS algorithm converges to a global minimizer when it has been initialized with an element of $\mathbb{I}$. Surprisingly, after running intensively

[4] In the practical situations we are interested in, $\#\mathcal{P} \gg S$ and S^3 can be neglected when compared to $(K+S)S\#\mathcal{P}$.

the ALS algorithm, it appears that in many situations $\mathbb{I}$ is actually large. In order to illustrate this aspect, we have chosen a simple initialization. It consists of initializing our algorithm by drawing a random variable uniformly distributed in $\mathcal{D}$. This is easily achieved (Muller, 1959) by using[5]

$$h^k = \frac{h}{\|h\|_2}, \quad \text{with} \quad h \sim \mathcal{N}_S(0, Id),$$

where $\mathcal{N}_S(0,Id)$ is the centered normal distribution in $\mathbb{R}^S$. Our experiments will show that $\mathbb{P}(\mathbf{h} \notin \mathbb{I})$ is often significantly smaller than 1 when $\mathbf{h}$ is uniformly distributed in D. Moreover, an advantage of this random initialization is that we can use a "restart" strategy to explore $\mathcal{D}$. More precisely, we propose to run the ALS algorithm R times, for $R \in \mathbb{N}$, and to return the result for which the objective function is the smallest. The probability that such a strategy fails to provide a global minimizer is equal to the probability that none of the R independent initializations belong to $\mathbb{I}$, i.e.,

$$\mathbb{P}(\text{not global}) = [\mathbb{P}(\mathbf{h} \notin \mathbb{I})]^R$$

which decays rapidly to 0, when $\mathbb{P}(\mathbf{h} \in \mathbb{I})$ is not negligible. For instance, to guarantee

$$\mathbb{P}(\text{not global}) \leq \varepsilon$$

for $\varepsilon > 0$, we must take

$$R \geq R_\varepsilon = \frac{\log(\varepsilon)}{\log(\mathbb{P}(\mathbf{h} \notin \mathbb{I}))}. \tag{14}$$

Note that the number of restarts does not increase significantly when ε decreases. However, when $\mathbb{P}(\mathbf{h} \in \mathbb{I})$ is small (or negligible) we have

$$R_\varepsilon \sim \frac{-\log(\varepsilon)}{\mathbb{P}(\mathbf{h} \in \mathbb{I})}.$$

The proposed "restart" strategy is therefore only reasonable when $\mathbb{P}(\mathbf{h} \in \mathbb{I})$ is not too small.

[5] For simplicity, in the formula below, we do not mention the mapping of $\mathbb{R}^S$ into $\mathbb{R}^{\mathcal{P}}$ necessary to build h^k.

4 Approximation experiments

4.1 Simulation scenario

Our first goal is to empirically assess the ability of a composition of convolutions to approximate a given target atom $H \in \mathbb{R}^{\mathcal{P}}$. We are also interested in observing the influence of the number of kernels K and of the size of the kernels on the approximation error. In order to do so, this section presents results obtained for several 1D and 2D target atoms H (i.e., $d = 1$ or 2) that have been selected from dictionaries commonly used in signal and image processing.

For all the experiments in Section 4, we consider a size $N \in \mathbb{N}$, a dimension $d \in \{1,2\}$ and take $\mathcal{P} = \{0,\dots,N-1\}^d$. We consider a target atom $H \in \mathbb{R}^{\mathcal{P}}$, a code $\alpha \in \mathbb{R}^{\mathcal{P}}$ and a zero mean Gaussian noise $b \in \mathbb{R}^{\mathcal{P}}$ of variance σ^2. Throughout these experiments, we explore parameters up to $K = 11$ and $S = 25$. Moreover, for a dimension $d \in \{1,2\}$ and a size $c \in \mathbb{N}$, we always consider the support mappings $(S^k)_{1\leq k\leq K} \in (\mathcal{P}^S)^K$ such that for all $k \in \{1,\dots,K\}$

$$\text{rg}\left(S^k\right) = k\{-c,\dots,0,\dots,c\}^d. \tag{15}$$

For example with two 2D kernels h^1 and h^2 and a size $c = 1$, their support mappings are set to $\text{rg}\left(S^1\right) = \{-1,0,1\} \times \{-1,0,1\}$ and $\text{rg}\left(S^2\right) = \{-2,0,2\} \times \{-2,0,2\}$, which means that both kernels have $S = 9$ authorized non-zero elements. Note that centering these support mappings on $p = 0$ is possible because of the periodization of $\mathbb{R}^{\mathcal{P}}$. Figure 1 shows an example of support mapping obtained for $K = 4$, $d = 2$ and $c = 1$.

It is not difficult to show (for instance, by induction) that the reachable support defined in (3) associated with the support mappings defined in (15) is:

$$\begin{aligned}\mathcal{S} &= \left\{\sum_{k=1}^K -ck,\dots,\sum_{k=1}^K ck\right\}^d \\ &= \left\{-c\frac{K(K+1)}{2},\dots,c\frac{K(K+1)}{2}\right\}^d.\end{aligned}$$

To continue with the previous example, the convolution of h^1 with h^2 can reach the set $\mathcal{S} =$

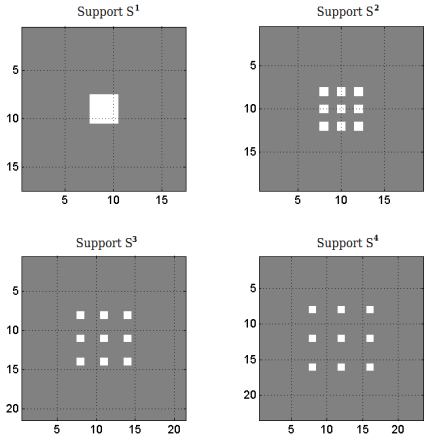

Fig. 1: The supports $\mathrm{rg}\left(S^k\right)$ described by (15), for $d=2$, $k\in\{1,2,3,4\}$ and for $c=1$ (i.e., $S=3\times 3$). The representation is shifted so that the origin element of (15) is at the center of each image. The constraint (2) forces each kernel h^k to take the value 0 outside of $\mathrm{rg}\left(S^k\right)$.

$\{-3,\ldots,3\}^2$, which contains 49 pixels. Therefore, the width of $\mathcal{S}$ is given by $K(K+1)c$ and its size (length or area) is $(K(K+1)c)^d$. Note that the size of $\mathcal{S}$ is usually much smaller than the size of the search space, equal to $K(2c+1)^d$. The ratio between these two quantities corresponds to a "compression ratio" when describing the atom with convolution kernels. This ratio behaves like $\frac{K^{2d-1}}{2^d}$ when both c and K grow. Table 1 shows the compression ratio for a few values of (K,c) and $d\in\{1,2\}$. The gain is clearly more interesting when increasing K compared to increasing c.

For most experiments, the support of H is contained into $\mathcal{S}$. When it is the case, we provide an indicator for the ability of the composition of convolutions to reduce the search space while filling the target atom's support. This indicator G is the ratio between the size of the effective support of H and the size of the actual search space using the K S-sparse kernels, i.e.,

$$\mathrm{G}=\frac{\#\mathrm{supp}_{\mathrm{eff}}(H)}{K(2c+1)^d}$$

where

$$\mathrm{supp}_{\mathrm{eff}}(H)=\left\{p\in\mathcal{P}\,|\,|H_p|\geq 10^{-4}(\max_{p\in\mathcal{P}}|H_p|)\right\}.$$

The role of the effective support is to realistically account for the energy localization in H. We will provide some values of G for the tests presented in this section.

Compression ratio		$K=3$	$K=4$	$K=6$	$K=10$
$d=1$	$c=1_{(S=3)}$	0.67	1.00	1.67	3.00
	$c=2_{(S=5)}$	0.80	1.20	2.00	3.60
	$c=3_{(S=7)}$	0.86	1.29	2.14	3.86
$d=2$	$c=1_{(S=9)}$	1.33	4.00	16.67	90.00
	$c=2_{(S=25)}$	1.92	5.76	24.00	129.60
	$c=3_{(S=49)}$	2.20	6.61	27.55	148.78

Table 1: Compression ratio $\frac{(K(K+1)c)^d}{K(2c+1)^d}$ for various K and c in dimension $d=1$ and $d=2$.

For each experiment, the quantities N, d, H, α, σ, K, c and the number R of restarts are provided. Given these quantities, we compute u according to (1). Then, Algorithm 2 is run for a given number R of restarts and the result with the smallest objective function value is kept. The result of this process is denoted as $(\lambda,(h^k)_{1\leq k\leq K})\in\mathbb{R}\times\mathcal{D}$ in what follows.

Given a result $(\lambda,(h^k)_{1\leq k\leq K})\in\mathbb{R}\times\mathcal{D}$, we evaluate the quality of the approximation of H by $\lambda h^1*\cdots*h^K$ using the peak-signal-to-noise ratio (PSNR). Moreover, in order to consider that the size of the support of H can be much smaller than $\#\mathcal{P}$, the PSNR is normalized according to the size of the effective support of H. More precisely, it is defined by

$$\mathrm{PSNR}_H=10\log_{10}\left(\frac{r^2}{\mathrm{MSE}_H}\right)$$

where $r=\max_{p\in\mathcal{P}}(H_p)-\min_{p\in\mathcal{P}}(H_p)$ is the dynamic range of the atom H and the mean-square-error (MSE) is defined by:

$$\mathrm{MSE}_H=\frac{\|\lambda h^1*\cdots*h^K-H\|_2^2}{\#\mathrm{supp}_{\mathrm{eff}}(H)}. \tag{16}$$

Note that the usual PSNR and MSE are normalised by the whole image size $\#\mathcal{P}$ instead of $\#\mathrm{supp}_{\mathrm{eff}}(H)$. The normalization defined in (16) is motivated by the nature of most atoms studied in this section: though their support may span over the whole set $\mathcal{P}$, most of their energy is concentrated in a small region.

Note that in noisy settings, PSNR values are provided in addition to the noise variance σ^2. These PSNR values inform us on the degradation between $\alpha * H$ and u, and cannot be compared to the values of PSNR_H, which concern the reconstructed atom only. The only exception is the first experiment, of paragraph 4.2.1, where the code α is a Dirac delta function.

We also provide a figure of merit reflecting both the quality of the convergence and the level of regularization induced by the composition of convolutions. The Normalized Reconstruction Error (NRE) is defined as

$$\mathrm{NRE} = \frac{\|\lambda\alpha * h^1 * \cdots * h^K - u\|_2^2}{\|u\|_2^2}. \tag{17}$$

When NRE is large, either the convergence has not been reached or the values of K and S are too small to obtain a good approximation of H. When it is small, the algorithm has converged to a stationary point close to a global minimum and the values of K and S provide a good approximation of u. Note that this last property can be a problem when u is contaminated by a strong noise.

Finally, in order to assess the additional difficulty induced by the convolution with the code α, we provide a measure of conditioning. Indeed, recovering H from u can be a badly conditioned problem (see (1)) yielding instabilities. For every experiment where α is not a Dirac delta function, a histogram of the values of the modulus of its Fourier transform $|\hat{\alpha}|$ is used to measure conditioning. The greater the range over which these values span, the worse the conditioning. The case of a sparse α seems to be the best compromise between conditioning and redundancy, the latter being crucial to get a stable approximation of H in the presence of noise.

Note that NRE can be small whatever the conditioning because the value of the denominator in (17) depends on α. For this reason, PSNR_H is still the most relevant indicator of the success of the algorithm.

4.2 1D targets

4.2.1 Apodized Modified Discrete Cosine

The modified discrete cosine transform (MDCT) has been successfully used in several signal processing applications such as audio coding (Painter and Spanias, 2000). The aim of the proposed experiment is to approximate an apodized modified discrete cosine (MDC) with a composition of convolutions. In order to do so, we apply the inverse MDCT to a Dirac delta function located at a given frequency, in a signal of size 512 (i.e., $d = 1$). We then apodize the MDC using the sine window $(w_p)_{0\leq p\leq 255}$ defined by:

$$w_p = \begin{cases} 0 & \text{if} \quad p \in \{0,\ldots,127\} \\ \sin\left[\pi\frac{(p-128)}{256}\right] & \text{if} \quad p \in \{128,\ldots,383\}. \\ 0 & \text{if} \quad p \in \{384,\ldots,512\} \end{cases}$$

This type of window is, for instance, used in MDCT analysis for time-domain aliasing cancellation (Princen and Bradley, 1986).

Figures 2 and 3 show examples of target atoms H obtained for frequencies 10Hz and 100Hz. The code α used in this experiment is a Dirac delta function located at $p = 256$. In this simple, noiseless case, u equals H. As for all simulations conducted in this section, the kernel supports have been defined according to (15). We have used $R = 50$ restarts because the simulation is very fast.

Moreover, we have considered $5 \leq K \leq 11$ and $5 \leq S = 2c+1 \leq 11$, corresponding to the values of PSNR_H reported in Table 2. One can see that the higher K and S, the better the approximation of H. This result is expected since increasing these parameters confers more flexibility to describe the target atom H, leading to a lower resulting objective function value (after algorithm convergence), which is inversely proportional to PSNR_H. Note that values of PSNR_H above 50 dB are obtained in many cases.

The approximations obtained for frequencies 10 and 100, both with $K = 9$, $S = 2c+1 = 9$, are depicted in Figures 2 and 3, respectively. More precisely, each figure shows the approximation $\lambda h^1 * \cdots * h^K$ and the atom H. Note that the re-

PSNR_H(dB)	$K=5$	$K=7$	$K=9$	$K=11$
$c=2$	14.43	17.32	23.81	38.26
$c=3$	16.23	23.02	46.24	51.48
$c=4$	18.45	34.84	54.32	54.33
$c=5$	21.60	53.70	54.82	55.73

Table 2: MDC approximation for frequency 100Hz: PSNR_H for several values of K and c.

sulting approximations are very accurate, and $G = \frac{256}{9\times 9} = 3.16$.

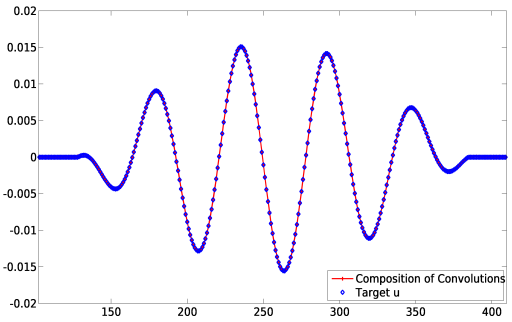

Fig. 2: Approximation of an apodized frequency 10 MDC by the convolution of $K=9$ kernels of sparsity $S=9$ ($\text{PSNR}_H = 58.88$dB).

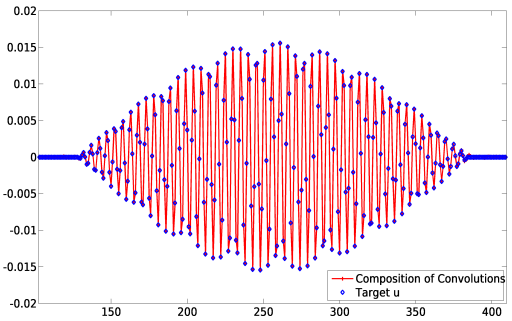

Fig. 3: Approximation of an apodized frequency 100 MDC by the convolution of $K=9$ kernels of sparsity $S=9$ ($\text{PSNR}_H = 54.32$dB).

The same experiment has been conducted for the frequency 100 with $K=9$, $S=2c+1=9$ and $R=25$ restarts, with an additive white Gaussian noise of variance $\sigma^2 \in \left[10^{-6}, 10^{-3}\right]$. Figure 4 shows PSNR_H as a function of the noise variance. Note that PSNR_H is always higher than the PSNR between u and $\alpha * H$ [6]. This means that the model (P_0) reduces noise when u is a noisy apodized MDC. This denoising would be further improved with a sparse code α containing several non-zero coefficients.

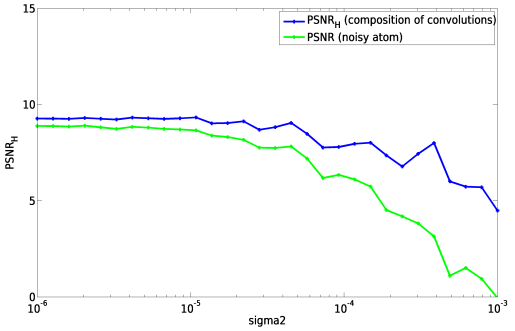

Fig. 4: PSNR_H for the approximation of the apodized frequency 100 MDC by the convolution of $K=9$ kernels of sparsity $S=9$, for $10^{-6} \leq \sigma^2 \leq 10^{-3}$ (blue curve). The green curve is the PSNR between u and $\alpha * H$.

4.2.2 Sinc function

This experiment aims at approximating the sinc function used to perform a linear zoom (Whittaker, 1915). The sinc interpolation has been successfully approximated with splines (Aldroubi et al, 1992). Though the spline interpolation can be interpreted as a composition of convolutions, we use different kernel supports. The zoom factor is $Z=3$ and the signal is of size 128. We therefore have $d=1$ and $N=3\times 128=384$. The target atom H is a sinc function obtained by computing the inverse Fourier transform of the characteristic function of a centered interval of length $N/3$. The signal to be zoomed corresponds to the first 128 values of the 128th column of the Barbara image.

The code α has been built by upsampling this signal by a factor $Z=3$ (see Figure 5). This upsampling has been performed by inserting 2 zeros

[6] In this case the comparison is relevant, because α is a Dirac delta function.

between every couple of neighbors in the initial signal. We are obviously not in a case where α is sparse. Moreover, the histogram of its Fourier transform displayed in Figure 6 shows that the convolution with α is not very well conditioned. Indeed, the ratio between the highest Fourier coefficient and the lowest is 728.

The signal u has been constructed according to (1) for different noise levels. Moreover, as for all experiments in this section, kernel supports have been set according to (15).

First, we have considered $K = 9$ and $c = 4$ (i.e., $S = 9$) and run $R = 50$ restarts of Algorithm 2 for noiseless and noisy signals ($\sigma^2 = 5$). Figures 7 and 8 shows the target sinc atom H and the approximation $\lambda * h^1 * \cdots * h^K$, for the noiseless and noisy cases. In the noiseless case (Figure 7), we see that the resulting composition of convolutions $\lambda * h^1 * \cdots * h^K$ is a good approximation of the sinc function. In the noisy case (Figure 8), the approximation is less accurate, which is expected since there is no regularization and the convolution with α is ill-conditioned.

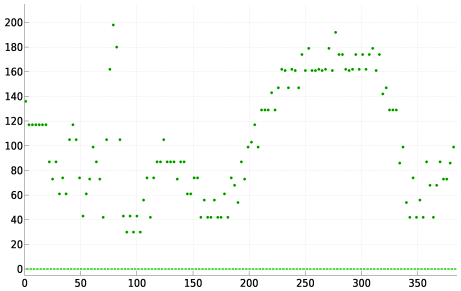

Fig. 5: Code α used in the the approximation of a 1D sinc function.

The same experiment has been run for $K \in \{3,5,7,9\}$ and $c \in \{1,2,3\}$ (i.e. $S \in \{3,5,7\}$), $R = 50$, for both cases $\sigma^2 = 0$ and $\sigma^2 = 5$. In the latter case, the PSNR between u and $\alpha * H$ is 28.20 dB.

Tables 3 and 5 contain the values of PSNR_H obtained for these parameters. In the noisy case (Table 3), PSNR_H is only a little smaller than that of the noiseless case (Table 5), which suggests that the method is robust to noise. To confirm this, a single case ($K = 9$, $c = 3$) is run for an increasing noise variance $0 < \sigma^2 < 20$. Figure 9 shows that

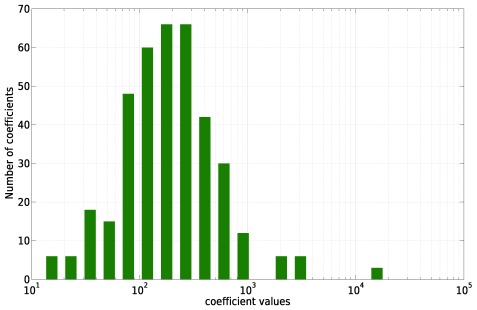

Fig. 6: Histogram of $|\hat{\alpha}|$, the modulus of the Fourier transform of the code.

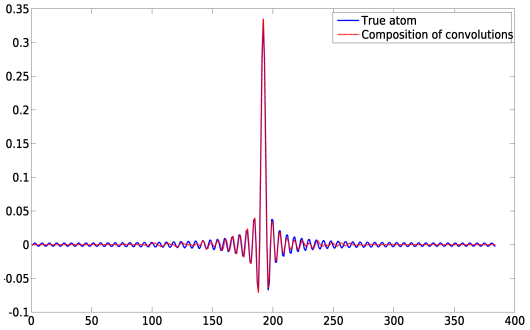

Fig. 7: Approximation of a noiseless 1D sinc function with $(K,c) = (9,4)$. The target sinc atom H and the composition of convolutions $\lambda h^1 * \cdots * h^K$. $\mathrm{PSNR}_H = 44.47$dB.

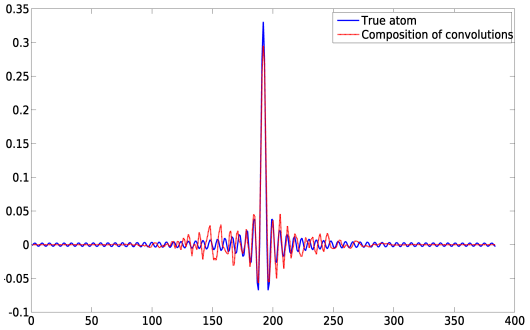

Fig. 8: Approximation of a noisy ($\sigma^2 = 5$) 1D sinc function with $(K,c) = (9,4)$. The target sinc atom H and the composition of convolutions $\lambda h^1 * \cdots * h^K$. $\mathrm{PSNR}_H = 35.68$dB.

when the noise variance increases (and PSNR between u and $\alpha * H$ decreases), PSNR_H decreases at the same rate.

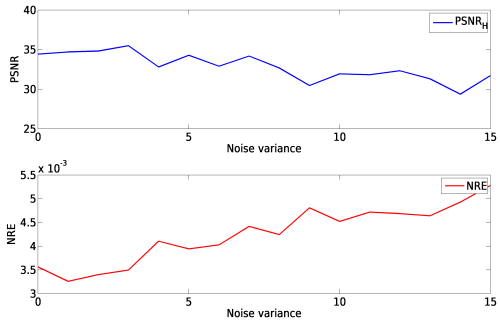

Fig. 9: Evolution of PSNR_H and NRE, for the sinc target atom, with respect to the noise variance σ^2, for $K = 9$, $c = 3$.

Moreover, in the presence of noise, increasing parameters K ans S does not clearly improve PSNR_H. This is due to the lack of regularization, when K and c are large. We do not observe this phenomenon in Table 5, which contains PSNR_H results for the noiseless case.

Tables 4 and 6 show, for the same experiments, the convergence criterion defined in (17). We observe that increasing K and c improves the criterion NRE, even in the noisy case, which is expected because of the conditioning of the convolution with α. This simulation shows that it is possible to have a good reconstruction of the signal u with a poor approximation of the atom H when the convolution with α is poorly conditioned.

PSNR_H(dB)	$K=3$	$K=5$	$K=7$	$K=9$
$c=1$	31.66	33.14	34.53	35.77
$c=2$	37.34	38.03	37.32	36.67
$c=3$	37.69	37.61	36.63	36.82

Table 3: Sinc approximation: PSNR_H for $\sigma^2 = 5$ ($R = 50$).

Finally, it is interesting to test the stability of the proposed model to an imperfect knowledge of α. For this purpose, a Gaussian noise $b_\alpha \sim \mathcal{N}(0, \sigma_\alpha^2)$ has been added to the code α used to solve (P_1) (u is still built with a noiseless α). We

$\text{NRE} \times 10^{-3}$	$K=3$	$K=5$	$K=7$	$K=9$
$c=1$	5.0	4.6	4.1	4.1
$c=2$	3.5	3.5	3.4	3.4
$c=3$	3.5	3.4	3.3	3.2

Table 4: Sinc approximation: NRE for $\sigma^2 = 5$ ($R = 50$).

PSNR_H(dB)	$K=3$	$K=5$	$K=7$	$K=9$
$c=1$	31.46	33.91	32.89	33.79
$c=2$	37.59	38.95	39.29	39.49
$c=3$	39.14	41.86	41.93	42.07

Table 5: Sinc approximation: PSNR_H for $\sigma^2 = 0$ ($R = 50$).

$\text{NRE} \times 10^{-3}$	$K=3$	$K=5$	$K=7$	$K=9$
$c=1$	2.0	1.2	1.2	1.1
$c=2$	0.3	0.2	0.2	0.2
$c=3$	0.2	0.1	0.1	0.1

Table 6: Sinc approximation: NRE for $\sigma^2 = 0$ ($R = 50$).

have set $K = 9$ and $c = 3$, i.e., $S = 7$, and have run the algorithm for several noise levels $0 \leq \sigma_\alpha^2 \leq 15$. Figure 10 shows that PSNR_H is stable with respect to σ_α^2, even though NRE tends to increase with σ_α^2. This suggest that the model is robust to an imperfect knowledge of α.

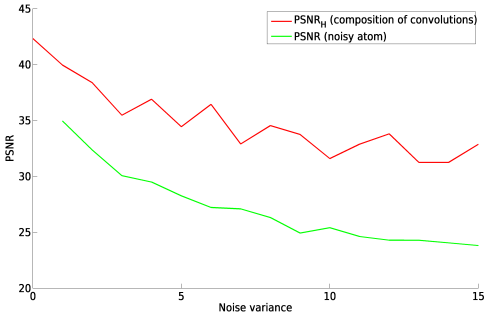

Fig. 10: Evolution of PSNR_H, for the sinc target atom, with respect to the noise variance σ_α on the code α, for $K = 9$, $c = 3$.

Finally, it is important to note that all the kernels used in these 1D experiments have the same support. Despite this constraint, the optimized kernels approximate very different target atoms such as MDC at frequency 10 and 100 and a sinc function. This shows that the proposed model based

on compositions of convolutions is reasonably rich and versatile.

4.3 2D targets

4.3.1 Curvelet

The aim of this experiment is to approximate a curvelet atom H in an image (i.e., $d = 2$) of size $N \times N$ with $N = 128$. The curvelet is obtained by applying the inverse curvelet transform to a Dirac delta function, using the MCALAB toolbox (Fadili et al, 2010). The code α corresponds to a Dirac delta function located at the barycenter of the curvelet. Once again, the support mapping is the one described in (15), with either $c = 1$ or $c = 2$. Note that this support mapping does not take the anisotropy of the curvelet into account. This is an unfavorable situation. All values of K satisfying $3 \leq K \leq 11$ have been tested. We consider a noiseless case so that u is a simple translation of H. We have used $R = 10$ restarts.

Figure 11 shows the target atom H and $\lambda h^1 * \cdots * h^K$, for $K = 7$ and $c = 2$. For these parameters, the size ratio between the effective support of the curvelet and the actual search space is $\mathrm{G} = 42.72$. We observe that, although $\mathrm{PSNR}_H = 44.30$ dB, the accuracy of the approximation is not the same in different parts of the image. In particular, the tails of curvelet are not properly captured.

Fig. 11: Curvelet approximation with $K = 7$ and $S = 5 \times 5$. Comparison between $\lambda h^1 * \ldots * h^7$ (top) and the target curvelet atom H (bottom). We have $\mathrm{PSNR}_H = 44.30$.

PSNR_H(dB)	$K=3$	$K=5$	$K=7$	$K=9$	$K=11$
$S = 3 \times 3$	33.06	36.55	36.52	37.22	37.01
$S = 5 \times 5$	39.99	45.81	44.30	40.74	38.05

Table 7: Curvelet approximation: PSNR_H for several values of K and S.

NRE	$K=3$	$K=5$	$K=7$	$K=9$	$K=11$
$c=1$	1.99	0.89	0.90	0.76	0.80
$c=2$	0.40	0.11	0.15	0.34	0.63

Table 8: Curvelet approximation: NRE for several values of K and c.

Table 7 contains the values of PSNR_H for various values of K and c. In this experiment, we were expecting that increasing K and S would improve the accuracy. It is not exactly what we observe in Table 7. For $S = 5 \times 5$, increasing K beyond a certain value actually makes PSNR_H decrease. This result can be explained by a lack of convergence, as is confirmed in Table 8. This problem could easily be corrected by an initialization exploiting the results obtained for smaller values of K and c.

Finally, Figure 12 shows the kernels $(h^k)_{1 \leq k \leq K}$ computed for $K = 7$ and $S = 5 \times 5$. We can observe that many kernel coefficients are close to zero, i.e., only the coefficients along the main direction of the curvelet have significant values. It is obvious

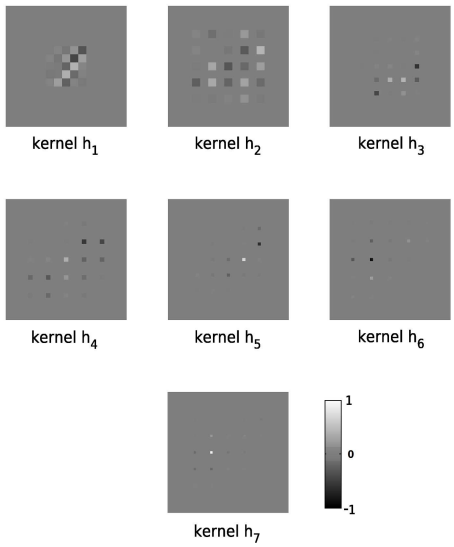

Fig. 12: Curvelet approximation for $K = 7$ and $S = 5 \times 5$. Zoom on the computed kernels $(h^k)_{1 \leq k \leq 7}$. The colormap is flattened around 0 to highlight the higher coefficients.

that the simple isotropic dilation of the supports defined by (15) is not appropriate for this curvelet. This raises the question of the adaptation of the support mappings $(S^k)_{1 \leq k \leq K}$ to the atom's geometry.

4.3.2 Cosine

The aim of this experiment is to approximate an atom representing a 2D cosine function in an image of size 64×64 (i.e., $d = 2$ and $N = 64$). In the context of image processing, such an atom can be seen as a large local cosine or a Fourier atom. Both are widely used in image processing. The interest of this atom is that it covers the whole image and is of a rather large support. Beside, patches of this size are difficult to handle with existing dictionary learning strategies. The considered atom is given by

$$H_p = \cos\left(2\pi \frac{\langle p, (2,5) \rangle}{N}\right) \quad , \forall p \in \{0, \ldots, 63\}^2.$$

The code α is a sparse vector whose support elements are randomly chosen. More precisely, for all $p \in \mathcal{P}$, there is a probability 10^{-1} that $\alpha_p \neq 0$. The values of the non-zero elements are then set according to the centered normal distribution $\mathcal{N}(0,1)$. In other words, for a given p, the elements of code α_p are assumed to be independent and identically distributed according to a Bernoulli-Gaussian distribution, that has been widely used in sparse signal and image deconvolution (Champagnat et al, 1996; Kail et al, 2012; Quinsac et al, 2011). Therefore, u contains a few weighted translations of the cosine atom H[7], which should result in a better approximation of H using Algorithm 2. Figures 13 and 14 show the code and the histogram of its Fourier transform. Note that the ratio between the largest and the smallest Fourier coefficients (in modulus) is 91, which corresponds to a reasonable conditioning. The target u is built with additive Gaussian noise of variance $\sigma^2 = 0.5$, which corresponds to a normalized PSNR_H between $\alpha * H$ and u of 22.08.

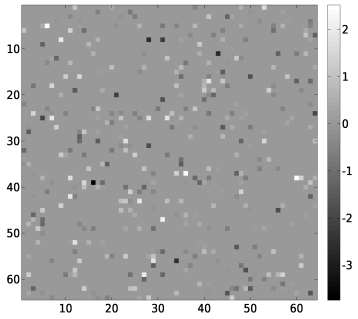

Fig. 13: Cosine experiment: Code α.

The support mapping is the same as for the previous experiment (see (15)) with, either $c = 1$ $(S = 3 \times 3)$ or $c = 2$ $(S = 5 \times 5)$. Different Values of K have been tested in the range $3 \leq K \leq 11$, each time with $R = 15$ restarts.

PSNR_H(dB)	$K=3$	$K=5$	$K=7$	$K=9$	$K=11$
$c=1$	11.79	12.27	13.81	25.15	30.09
$c=2$	11.94	15.97	41.44	38.94	39.82

Table 9: 2D Cosine approximation: PSNR_H.

[7] A sum of cosines of same frequency and different phases will yield a cosine of unchanged frequency.

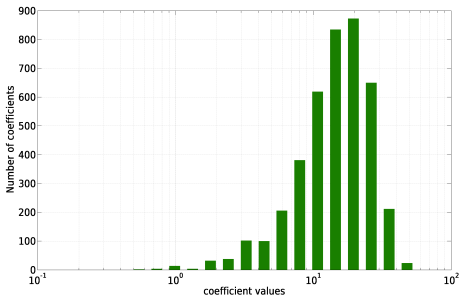

Fig. 14: Cosine experiment: Modulus of the Fourier transform of the code $|\hat{\alpha}|$.

NRE	$K=3$	$K=5$	$K=7$	$K=9$	$K=11$
$c=1$	1.02	0.89	0.41	0.04	0.02
$c=2$	0.96	0.24	0.01	0.01	0.01

Table 10: 2D Cosine approximation: NRE.

Tables 9 and 10 provide the PSNR_H and NRE indicators in the studied range of parameters. In this experiment, we expect to obtain a somewhat regularized atom thanks to the repetitions induced by the sparse (and reasonably conditioned) code α. We observe in Table 9 that PSNR_H rises above 30 if parameters K and S are large enough. Even for $K=9$ and $c=2$, the ratio between the number of variables describing the kernels and the size of the cosine is $G=\frac{64\times 64}{9\times 5\times 5}=18.20$. Table 10 shows a steady improvement of NRE when K and c increase.

Figures 15 and 16 show the cosine image u, its approximation $\lambda\alpha * h^1 * \cdots * h^K$, the actual atom H and $\lambda h^1 * \cdots * h^K$, for $K=7$ and $c=2$. The results obtained here are quite accurate even though the cosine image was corrupted by additive noise.

Figure 17 shows the obtained kernels $(h^k)_{1\leq k\leq K}$. As opposed to the kernels obtained for the curvelet approximation, the energy is more uniformly distributed on the kernel supports. These kernels and the curvelet kernels are provided in a Matlab file available online (see `http://chabiron.perso.enseeiht.fr/FTL_demo/FTL_demo_v1.1.zip`).

This experiment was also run with fixed $K=7$ and $c=2$ for an increasing noise variance, to test the robustness of the proposed model. Figure 18

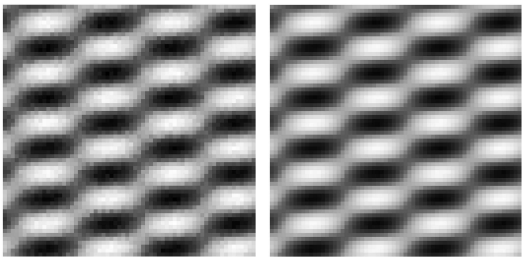

Fig. 15: Cosine approximation with $K=7$, $c=2$, and Gaussian noise of variance $\sigma^2=0.5$. Cosine image u (left) and approximation $\lambda\alpha * h^1 * \cdots * h^K$ (right).

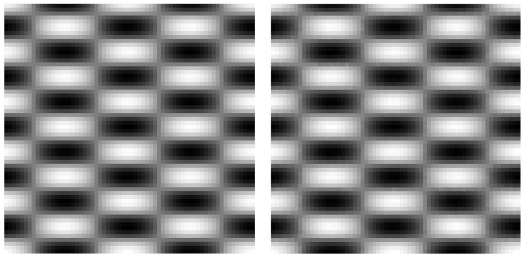

Fig. 16: Cosine approximation with $K=7$, $c=2$, and Gaussian noise of variance $\sigma^2=0.5$. True atom H (left) and approximation $\lambda h^1 * \cdots * h^K$ (right).

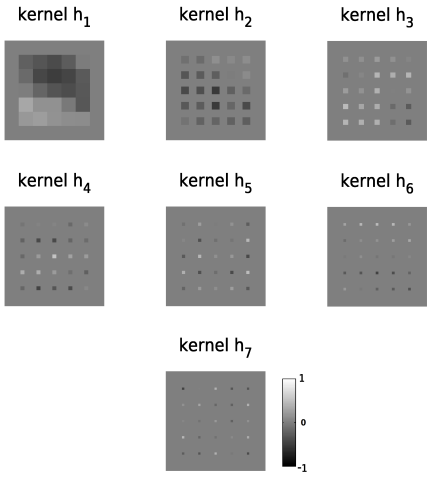

Fig. 17: Cosine approximation with $K=7$, $c=2$, and Gaussian noise of variance $\sigma^2=0.5$. Zoom on the computed kernels $(h^k)_{1\leq k\leq 7}$.

shows the values of PSNR_H associated with the reconstructed image, as a function of the noise variance. Note that PSNR_H decreases at the same rate as the PSNR measuring the degradation between u and $\alpha * H$.

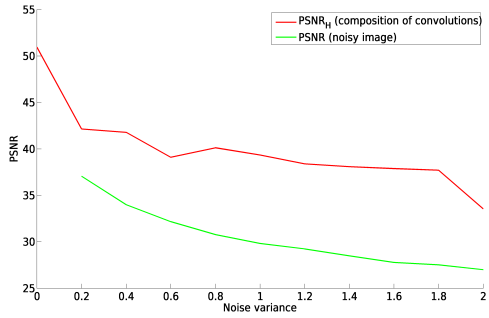

Fig. 18: Evolution of PSNR_H of a reconstructed cosine atom when the noise variance σ^2 varies in $[0,2]$, for $K = 6$ and $c = 2 (S = 5 \times 5)$.

4.3.3 Wavelet decomposition

In this experiment, we consider a scenario reflecting the difficulties of DL. More precisely, we consider $d = 2$, $N = 512$, a target atom H defined as a wavelet atom and a code α resulting from the wavelet coefficients of a natural image. More precisely, the following operations have been conducted

- Select an image (here the Barbara image).
- Compute the wavelet transform of the image using the Daubechies wavelet db4 at level L. We used the official Matlab wavelet toolbox with $L = 3$.
- Select the set of coefficients associated with an orientation and a given decomposition level l such that $1 \leq l \leq L$. The low frequency at level $l = L = 3$ was considered for the first experiment and the horizontal detail at level $l = 3$ for the second experiment.
- Set the non selected wavelet coefficients to zero and compute the inverse wavelet transform. Add white Gaussian noise of variance $\sigma^2 = 5$ to obtain u.
- Define α as a zoom of factor 2^l of the selected coefficients, where the zoom consists of interpolating with zeros.
- Solve problem (P_k) with the code α, the target atom u, $R = 1$, with a support mapping defined by (15) for the parameters $K = 6$ and $S = 3 \times 3$ (i.e. $c = 1$). Note that the knowledge of the supports associated with the composition of convolutions leading to the wavelet atom was not used in this experiment.

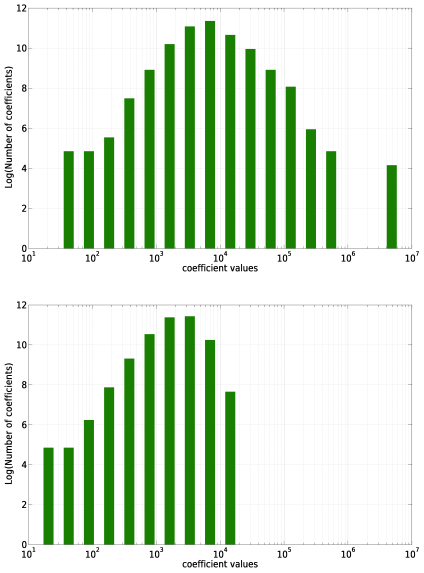

Fig. 19: Logarithm of the histogram of $|\hat{\alpha}|$, the modulus of the Fourier transform of the code. Approximation coefficients (top) and Horizontal detail coefficients (bottom).

The results obtained for the low frequency wavelet atom at level 3 and the horizontal detail wavelet atom are shown in Figures 20, 21, 22 and 23. Note that the conditioning of the convolution with α is less favorable for the estimation of the low frequency wavelet atom. Indeed, α is sparser when selecting detail coefficients, which results in a better conditioned problem. Figure 19 shows histograms of $|\hat{\alpha}|$ for both cases. Note that the ratio between the largest and the smallest Fourier coefficients (in modulus) is 532 for the horizontal detail case, and 6.67×10^4 in the approximation case.

Fig. 20: Estimation of the low frequency wavelet atom at level 3. Target u (left) and $\lambda\alpha * h^1 * \cdots * h^K$ (right). NRE $= 10^{-3}$.

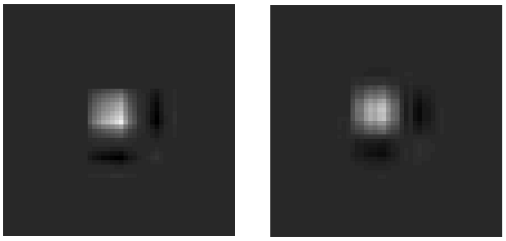

Fig. 21: Estimation of the low frequency wavelet atom at level 3. Atom H (left) and $\lambda * h^1 * \cdots * h^K$ (right). $\text{PSNR}_H = 29.94$.

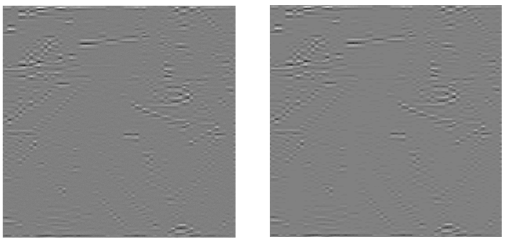

Fig. 22: Estimation of the horizontal detail wavelet atom at level 3. Target u (left) and approximation $\lambda\alpha * h^1 * \cdots * h^K$ (right). NRE $= 0.68$.

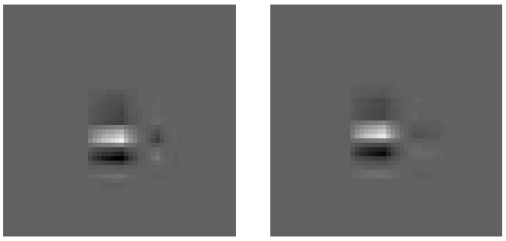

Fig. 23: Estimation of the horizontal detail wavelet atom at level 3. Atom H (left) and approximation $\lambda * h^1 * \cdots * h^K$ (right). $\text{PSNR}_H = 36.61$.

For both experiments, the composition of convolutions is close to the corresponding wavelet atom. However, PSNR_H is larger for the horizontal detail case, which suggests that a good conditioning for α is crucial. Unsurprisingly, convergence is better for the approximation coefficients case (NRE $= 10^{-3}$) than for the horizontal detail case (NRE $= 0.68$), though our primary concern remains accuracy on atom reconstruction. Note, however, that we did not run multiple restarts: a better convergence could still be achieved by increasing R.

Finally, we rerun both experiments with additive Gaussian noise of variance $\sigma_\alpha^2 = 10$ corrupting the code α. Note that this noise degrades the conditioning of the convolution with α. For the horizontal detail coefficients, we obtain $\frac{\max|\hat{\alpha}|}{\min|\hat{\alpha}|} = 4.99 \times 10^3$, NRE $= 1.62$ and $\text{PSNR}_H = 29.09$. For the approximation coefficients, we obtain $\frac{\max|\hat{\alpha}|}{\min|\hat{\alpha}|} = 8.47 \times 10^4$, NRE $= 0.002$ and $\text{PSNR}_H = 27.17$. The horizontal detail case still gives a better result, as expected. Both approximations $\lambda h^1 * \cdots * h^K$ are shown in Figures 24 and 25.

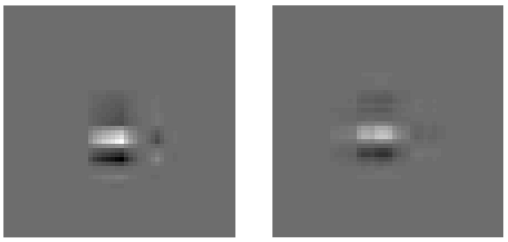

Fig. 24: Approximation obtained with a noisy code α ($\sigma_\alpha^2 = 10$). Horizontal detail atom H (left) and its approximation $\lambda h^1 * \cdots * h^K$ (right) ($\text{PSNR}_H = 29.09$, NRE $= 1.62$).

As for 1D atoms, it is important to note that all the kernels used in these 2D experiments have the same support. Again, despite this constraint, the optimized kernels approximate very different target atoms such as a curvelet, a cosine and a wavelet. This shows that the composition of convolution model is reasonably rich and versatile.

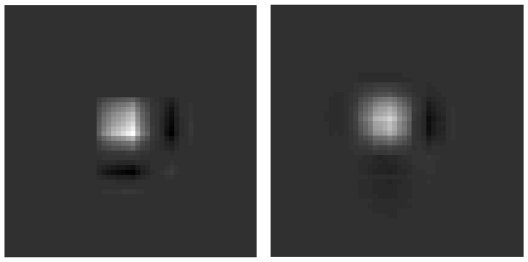

Fig. 25: Approximation obtained with a noisy code α ($\sigma_\alpha^2 = 10$). Approximation atom (left) and its approximation $\lambda h^1 * \cdots * h^K$ (right) (PSNR$_H = 27.17$, NRE $= 2.10^{-3}$).

This potential will be exploited to obtain dictionaries well adapted to a given image class, once the dictionaries are learnt from datasets.

4.4 Dictionary learning experiment

As a follow-up to the previous experiment related to wavelet decomposition, the experiment presented in this section puts the method more in context with the intended application, namely dictionary learning. Basically, it is the same experiment as the one presented in Section 4.3.3 except that the code is learnt through a sparse coding scheme instead of being supposed known.

We consider $d = 2$, $N = 256$, a wavelet atom H and a code α^* resulting from the wavelet coefficients of the barbara image. More precisely, the following operations have been conducted

- Build α^*, H and u exactly as in Section 4.3.3 for the case of horizontal detail coefficients. That is, α is the level 2^3 upsampling of the horizontal detail coefficients, H is the level 3 horizontal detail wavelet atom, and $u = \alpha^* * H$ is the partial wavelet reconstruction of the input image (using the level 3 horizontal coefficients only).
- Initialize α and the kernels $(h^k)_{1\leq k\leq K}$ with random values, where $K = 5$ and $S = 3 \times 3$, with the support mapping defined by (15).
- Iterate between:
  - Solve a Basis Pursuit Denoising (BPDN) problem to update α,
  - Solve a variant of (P_1) that does not contain λ with the updated code α,

Although, this is a preliminary study and we have no proof of convergence, this aims at finding a solution of the following dictionary learning problem:

$$
\begin{array}{ll}
\text{argmin}_{\alpha,(h^k)_{1\leq k\leq K}} & \|\alpha * h^1 * \cdots * h^K - u\|_2^2 + \gamma\|\alpha^l\|_1 \\
\text{subject to} & \mathbf{h} \in (\mathbb{R}^{\mathcal{P}})^K, \alpha \in \mathbb{R}^{\mathcal{P}}, \\
\text{and} & \text{supp}\left(h^k\right) \subset \text{rg}\left(S^k\right), \forall k \in \{1,\ldots,K\}, \\
\text{and} & \|h^k\|_2 \leq 1, \forall k \in \{1,\ldots,K\}, \\
\text{and} & \text{supp}\left(\alpha\right) \subset \mathcal{S}_{\alpha,j}
\end{array}
$$

with $\mathcal{S}_{\alpha,j} = \{p' \in \mathcal{P} | \forall p \in \mathcal{P}, p' = jp\}$, for some chosen integer j. The role of the constraint on the support of α is to improve the incoherence of the set of atoms of our dictionary. Note that the constraints on the kernel norms have been changed to $\|h^k\|_2 \leq 1, \forall k \in \{1,\ldots,K\}$ and that the weight λ has been removed. To solve the sparse coding part of this problem, we use BPDN (Chen et al, 1998) with a simple Iterative Thresholding algorithm (Daubechies et al, 2004) with the renormalization of the dictionary proposed in (Malgouyres and Zeng, 2009).

The experiment is run for different initializations and supports for the code. First, the initial code is chosen as the solution α^* perturbed by additive Gaussian noise of variance $\sigma^2 = 5$, and $j = 8$ is chosen for the support constraint on α. This is supposed to be the most favorable case since α^* has been built as a level 8 upsampling. In another experiment, we use a Gaussian random initialisation for $j \in \{4,8\}$. The L_1 penalty γ has been empirically set to 10.

Table 11 shows the results obtained in terms of PSNR$_H$, NRE and sparsity level, whereas Figure 26 shows the atoms obtained with a randomly initialized code and $j \in \{4,8\}$. Though this experiment is only a first attempt for learning both the code and the atoms with our model, it appears that enforcing incoherence of the atoms of our dictionary will play a central role to fully learn fast transforms.

Initialization	PSNR_H	NRE	Sparsity
Noisy α^*, $j=8$	30.59	0.04	1.0%
Random, $j=8$	19.10	0.04	1.1%
Random, $j=4$	18.21	0.07	1.9%

Table 11: DL experiment: PSNR_H, NRE and sparsity for various support constraints on the code. In some cases, PSNR_H is computed after a translation and/or a sign change.

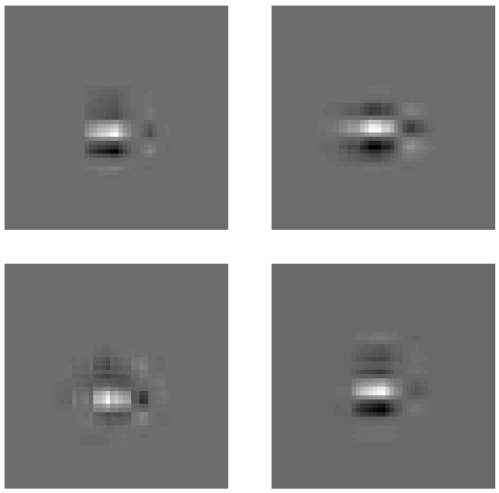

Fig. 26: Estimation of the horizontal detail wavelet atom at level 3 with code unknown. Atom H (top left) and approximations with $j=8$ (top right), $j=4$ (bottom left) and $j=8$ with a favorable initialization (bottom right).

5 Convergence Assessment

5.1 Simulation Scenario

This section evaluates

$$\mathbb{P}\left(\text{not global}\right) \quad \text{and} \quad R_\varepsilon = \frac{\log(\varepsilon)}{\log\left(1-\mathbb{P}\left(\mathbf{h}\in\mathbb{I}\right)\right)}$$

for various supports, kernels and noise levels. All the experiments have been conducted with one-dimensional signals of size $\#\mathcal{P}=128$ and $(K,S)\in\{2,\dots,7\}\times\{2,\dots,10\}$ and random support mappings $\mathbf{S}=(S^k)_{1\leq k\leq K}$. For every $k\in\{1,\dots,K\}$, the support mapping S^k maps $\{1,\dots,S\}$ into S distinct elements randomly drawn according to a uniform distribution in $\{1,...,10\}$. Moreover, for any $(k_1,k_2)\in\{1,\dots,K\}^2$, with $k_1\neq k_2$, $\mathrm{rg}\left(S^{k_1}\right)$ and $\mathrm{rg}\left(S^{k_2}\right)$ are independent random vectors. We also consider K independent random kernels

$$h_p^k \begin{cases} \sim \mathcal{N}(0,1) & \text{, if } p\in \mathrm{rg}\left(S^k\right) \\ =0 & \text{, otherwise.}\end{cases}$$

Finally, the code is set to $\alpha=(1,0,\dots,0)$ (i.e., no translation) and the image u is obtained by convolving the kernels, i.e.,

$$u=\alpha*h^1*\cdots*h^K+b$$

where $b\sim\mathcal{N}(0,\sigma^2\mathbb{1}_{\mathcal{S}})$, σ^2 is the noise variance and the set $\mathcal{S}$ is the "reachable support" defined in (3). Note that u is zero outside of the reachable support.

5.2 Performance measure

Given a problem defined by $(u,\alpha,\mathbf{S})$, a global minimizer $\mathbf{h}^*=(h^{*,k})_{1\leq k\leq K}\in(\mathbb{R}^{\mathcal{P}})^K$ of (P_0) and a solution $\overline{\mathbf{h}}=(\overline{h}^k)_{1\leq k\leq K}\in(\mathbb{R}^{\mathcal{P}})^K$ provided by Algorithm 2, we denote the approximation error by

$$E_a(u,\alpha,\mathbf{S})=\|\alpha*h^{*,1}*\ldots*h^{*,K}-u\|_2^2.$$

For the problem constructed in the previous paragraph, we expect that

$$E_a(u,\alpha,\mathbf{S})\leq\sigma^2\left(\#\mathcal{S}\right),$$

where σ^2 is the noise variance. Moreover, we know that $E_a(u,\alpha,\mathbf{S})=0$ for $\sigma=0$. We also denote the numerical error by

$$E_n\left(\overline{\mathbf{h}},u,\alpha,\mathbf{S}\right)=\|\alpha*\overline{h}^1*\ldots*\overline{h}^K-u\|_2^2-E_a(u,\alpha,\mathbf{S}).$$

The only quantity that we can actually observe is the sum of these two errors

$$\|\alpha*\overline{h}^1*\ldots*\overline{h}^K-u\|_2^2=E_a(u,\alpha,\mathbf{S})+E_n((\overline{\mathbf{h}},u,\alpha,\mathbf{S}).$$

We therefore consider that Algorithm 2 has converged to a global minimum if

$$\|\alpha*\overline{h}^1*\ldots*\overline{h}^K-u\|_2^2\leq\sigma^2\left(\#\mathcal{S}\right)+10^{-4}\|u\|_2^2. \quad (18)$$

Of course, this notion is not very accurate when σ^2 is large.

5.3 Evaluation of $\mathbb{P}$(not global)

For any fixed $(K,S) \in \{2,\dots,6\} \times \{2,\dots,10\}$, we have generated $L = 50K^2$ signals. Each signal is labelled by an index $l \in \{1,\dots,L\}$. For every experiment, we consider $R = 25$ random initializations according to a uniform distribution defined on the set of constraints associated with (P_1), as described in Section 3.5. The corresponding outcome of Algorithm 2 is referred to as the rth result with $r \in \{1,\dots,R\}$. Finally, for any $(l,r) \in \{1,\dots L\} \times \{1,\dots,R\}$, we introduce the following indicator function

$$\mathbb{1}(l,r) = \begin{cases} 1, \text{ if (18) holds for the } r\text{th result} \\ \quad \text{obtained from the } l\text{th input,} \\ 0, \text{ otherwise.} \end{cases}$$

The probability of reaching a global minimum of problem (P_1) is estimated as follows

$$\mathbb{P}(\text{global minimizer}) \simeq \frac{1}{LR} \sum_{l=1}^{L} \sum_{r=1}^{R} \mathbb{1}(l,r).$$

5.4 Results

Figures 27 and 28 show the results obtained in the noiseless ($\sigma^2 = 0$) and noisy ($\sigma^2 = 5 \times 10^{-2}$) cases respectively. In each figure, the curves show $\mathbb{P}$(global minimizer) and the number of restarts needed to ensure a failure probability lower than ε, $R_\varepsilon = \frac{\log(\varepsilon)}{\log\left(1-\mathbb{P}\left((h^k)_{1\le k\le K} \in \mathbb{I}\right)\right)}$ for a given value of K whereas the x axis indicates the support size S. We can see that for very sparse kernels ($S \le 3$), the probability of success is quite high. However, this probability drops significantly when the support size increases. Surprisingly, $\mathbb{P}$(global minimizer) increases when the support size increases. The more kernels we use (i.e., the larger K), the steeper the decrease and increase. These results show that it is possible to obtain convergence to a global minimum with only a few restarts of the proposed algorithm even for relatively large values of K. The last experiments obtained in the noisy case show similar patterns. As a consequence, the described convergence properties seem to be robust to noise.

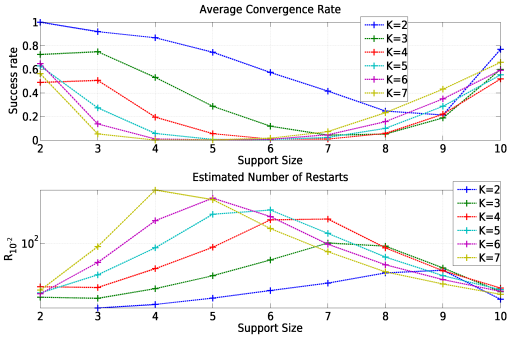

Fig. 27: Convergence test for $\sigma = 0$: Estimated probability of reaching a global minimum (top) for every $K \in \{2,\dots,7\}$ and corresponding number of restarts R_ε to guarantee $\mathbb{P}$(global minimizer) $\ge 99\%$ (bottom). For every $K \in \{2,\dots,7\}$, the results have been averaged over $L = 50K^2$ inputs from which we have computed $R = 25$ outputs.

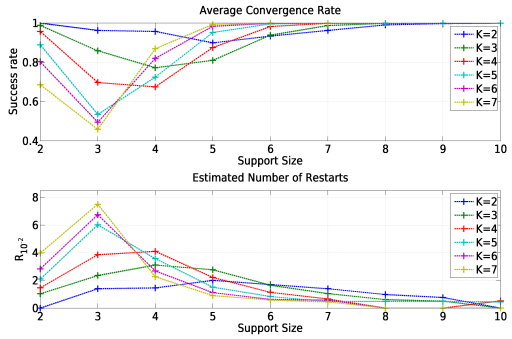

Fig. 28: Convergence test for $\sigma^2 = 5.10^{-2}$: Estimated probability of reaching a L_2 ball of radius $\sigma\sqrt{\#S}$ around a global minimum (top) for every $K \in \{2,\dots,6\}$ and corresponding number of restarts R_ε to guarantee $\mathbb{P}$(global minimizer) $\ge$ 99% (bottom). For every $K \in \{2,\dots,6\}$, the results have been averaged over $L = 50K^2$ inputs, from which we have computed $R = 25$ outputs.

6 Conclusions and perspectives

We introduced a new important problem whose purpose is to mitigate the computational issues encountered in most dictionary learning frameworks (which generally constrain the use of small patches). We proposed to consider atoms defined as a composition of convolutions with sparse kernels. The determination of these atoms required to solve a non-convex optimization problem. Using the sparsity of kernels to reduce the search

space, we studied a computationally efficient algorithm based on alternate least squares minimizations. This algorithm has linear complexity with respect to the image size. It allows the learning of fast transforms by the dictionary update stage and permits to consider larger atoms. Our experiments showed that compositions of convolutions can approximate accurately many atom-like signals and images such as curvelets, cosines and wavelets. This illustrates that the non-convex optimization problem considered in this paper lends itself to global optimization and that (despite the constraint on the kernel supports) the considered setting is sufficiently rich and versatile to approximate a large class of atoms. However, the full potential of these compositions of convolutions for approximation purposes still remains to be assessed.

Future work includes the definition of a tree structure for the proposed composition of kernel convolutions for dictionary learning applications. Designing efficient rules to learn the kernel supports also remains a large and unexplored issue which might have a huge impact on the performance of the proposed algorithm. The typical strategies one can think of for improving the supports is to adapt algorithms like the orthogonal matching pursuit or to add a term in the energy favoring the sparsity of the kernels.

Designing efficient rules to learn the kernel supports also remains a large and unexplored issue which might have a huge impact on the performance of the proposed algorithm. As an example, the learning algorithms investigated in the important literature related to deep learning (and in particular to convolutional networks) would deserve to be studied in the context of convolutions with sparse kernels. Indeed, as explained in (Bengio and LeCun, 2007), even if the convergence of these algorithms is difficult to prove, they do not seem to suffer from the convergence problems that plague deep fully-connected neural nets.

In a similar direction, the experiments of Section 5 show an unexpected behavior of the algorithm. Understanding formally when the functional lends itself to global optimization is a important question that we plan to address in the near future.

Acknowledgement

The authors would like to thank Jose Bioucas-Dias, Jalal Fadili, Rémi Gribonval and Julien Mairal for their fruitful remarks on this work.

Appendix

Proof of Proposition 1

First notice that $\mathcal{D}$ is a compact set. Moreover, when (7) holds, the objective function of (P_1) is coercive in λ. Thus, for any threshold μ, it is possible to build a compact set such that the objective function evaluated at any $(\lambda, \mathbf{h})$ outside this compact set is larger than μ. As a consequence, we can extract a converging subsequence from any minimizing sequence. Since the objective function of (P_1) is continuous in a closed domain, any limit point of this subsequence is a minimizer of (P_1).

Proof of Proposition 2

The proof of 1 hinges on formulating the expression of a stationary point of (P_1), then showing that the Lagrange multipliers associated with the norm-to-one constraint for the $(h^k)_{1\leq k\leq K}$ are all equal to 0. First, considering the partial differential of the objective function of (P_1) with respect to λ and a Lagrange multiplier $\gamma_\lambda \geq 0$ for the constraint $\lambda \geq 0$, we obtain

$$\lambda \|\alpha * h^1 * \cdots * h^K\|_2^2 - \left\langle \alpha * h^1 * \cdots * h^K, u \right\rangle = \frac{\gamma_\lambda}{2}, \tag{19}$$

and

$$\lambda \gamma_\lambda = 0. \tag{20}$$

Then, considering Lagrange multipliers $\gamma_k \in \mathbb{R}$ associated with each constraint $\|h^k\|_2 = 1$, we have for all $k \in \{1, \ldots, K\}$

$$\lambda \tilde{H}^k * (\lambda \alpha * h^1 * \cdots * h^K - u) = \gamma_k h^k, \tag{21}$$

where H^k is defined by (5). Taking the scalar product of (21) with h^k and using both $\|h^k\|_2 = 1$ and (19), we obtain

$$\gamma_k = \lambda \frac{\gamma_\lambda}{2} = 0, \quad \forall k \in \{1,\dots,k\}.$$

Hence, (21) takes the form, for all $k \in \{1,\dots,K\}$

$$\lambda \tilde{H}^k * (\lambda\alpha * h^1 * \cdots * h^K - u) = 0. \tag{22}$$

When $\lambda > 0$, this immediately implies that the kernels $\mathbf{g}$ defined by (8) satisfy

$$\frac{\partial E}{\partial h^k}(\mathbf{h}) = 0, \quad \forall k \in \{1,\dots K\},$$

i.e., the kernels $\mathbf{g} \in (\mathbb{R}^{\mathcal{P}})^K$ form a stationary point of (P_0).

The proof of the item 2 is straightforward since for any $(f^k)_{1\le k\le K} \in (\mathbb{R}^{\mathcal{P}})^K$ satisfying the constraints of (P_0)[8], we have

$$\begin{aligned}
&\|\alpha * g^1 * \ldots * g^K - u\|_2^2 \\
&= \|\lambda\alpha * h^1 * \ldots * h^K - u\|_2^2 \\
&\le \left\| \left(\prod_{k=1}^K \|f^k\|_2 \right) \alpha * \frac{f^1}{\|f^1\|_2} * \ldots * \frac{f^K}{\|f^K\|_2} - u \right\|_2^2 \\
&\le \|\alpha * f^1 * \ldots * f^K - u\|_2^2.
\end{aligned}$$

As a consequence, the kernels $(g^k)_{1\le k\le K}$ defined by (8) form a solution of (P_0).

Proof of Proposition 3

The first item of proposition 3 can be obtained directly since 1) the sequence of kernels generated by the algorithm belongs to $\mathcal{D}$ and $\mathcal{D}$ is compact, 2) the objective function of (P_1) is coercive with respect to λ when (13) holds, and 3) the objective function is continuous and decreases during the iterative process.

To prove the second item of proposition 3, we consider a limit point $(\lambda^*, \mathbf{h}^*) \in \mathbb{R} \times \mathcal{D}$. We denote by F the objective function of (P_1) and denote by $(\lambda^o, \mathbf{h}^o)_{o\in\mathbb{N}}$ a subsequence of $(\lambda^n, \mathbf{h}^n)_{n\in\mathbb{N}}$ which converges to $(\lambda^*, \mathbf{h}^*)$. The following statements are trivially true, since F is continuous and $(F(\lambda^n, \mathbf{h}^n))_{n\in\mathbb{N}}$ decreases:

$$\lim_{o\to\infty} F\left(T(\mathbf{h}^o)\right) = \lim_{o\to\infty} F(\lambda^o, \mathbf{h}^o) = F(\lambda^*, \mathbf{h}^*) \tag{23}$$

However, if for any k inside $\{1,\dots,K\}$, we have $C_k^T u \neq 0$ and the matrix C_k generated using $T_k(\mathbf{h}^*)$ is full column rank, then there exist an open neighborhood of $T_k(\mathbf{h}^*)$ such that these conditions remain true for the matrices C_k generated from kernels $\mathbf{h}$ in this neighborhood. As a consequence, the kth iteration of the for loop is a continuous mapping on this neighborhood. Finally, we deduce that there is a neighborhood of $\mathbf{h}^*$ in which T is continuous.

Since T is continuous in the vicinity of $\mathbf{h}^*$ and $(\mathbf{h}^o)_{o\in\mathbb{N}}$ converges to $(\mathbf{h}^*)$, the sequence $(T(\mathbf{h}^o))_{o\in\mathbb{N}}$ converges to $T(\mathbf{h}^*)$ and (23) guarantees that

$$F\left(T(\mathbf{h}^*)\right) = F(\lambda^*, \mathbf{h}^*).$$

As a consequence, denoting $\mathbf{h}^* = (h^{*,k})_{1\le k\le K}$, for every $k \in \{1,\dots,K\}$, $F(\lambda^*, h^{*,k})$ is equal to the minimal value of (P_k). Since C_k is full column rank, we know that this minimizer is unique (see the end of Section 3.2) and therefore $(\lambda^*, h^{*,k})$ is the unique minimizer of (P_k). We can then deduce that $(\lambda^*, \mathbf{h}^*) = T(\mathbf{h}^*)$.

Finally, we also know that $(\lambda^*, \mathbf{h}^*)$ is a stationary point of (P_k). Combining all the equations stating that, for any k, $(\lambda^*, \mathbf{h}^{*,k})$ is a stationary point of (P_k), we can find that $(\lambda^*, \mathbf{h}^*)$ is a stationary point of (P_1).

[8] We further assume that $\|f^k\|_2 \neq 0$, for all $k \in \{1,\dots,K\}$, since the inequality is otherwise trivial.

References

Aharon M, Elad M, Bruckstein AM (2006) The K-SVD, an algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans Signal Process 54(11):4311–4322

Aldroubi A, Unser M, Eden M (1992) Cardinal spline filters: Stability and convergence to the ideal sinc interpolator. Signal Processing 28(2):127–138

Attouch H, Bolte J, Redont P, Soubeyran A (2010) Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the kurdyka-Łojasiewicz inequality. Math Oper Res 35(2):438–457, DOI 10.1287/moor.1100.0449, URL http://dx.doi.org/10.1287/moor.1100.0449

Attouch H, Bolte J, Svaiter B (2013) Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. Mathematical Programming 137(1-2):91–129

Bengio Y, LeCun Y (2007) Scaling learning algorithms towards ai. in Large-Scale Kernel Machines 34:1–41

Bertsekas D (2003) Convex analysis and optimization. Athena sc

Bolte J, Sabach S, Teboulle M (2013) Proximal alternating linearized minimization for nonconvex and nonsmooth problems. Mathematical Programming, series A pp 1–16, DOI 10.1007/s10107-013-0701-9

Cai JF, Ji H, Shen Z, Ye GB (2013) Data-driven tight frame construction and image denoising. Applied and Computational Harmonic Analysis To appear

Champagnat F, Goussard Y, Idier J (1996) Unsupervised deconvolution of sparse spike trains using stochastic approximation. IEEE Trans Signal Process 44(12):2988–2998

Chen SS, Donoho DL, Saunders MA (1998) Atomic decomposition by basis pursuit. SIAM journal on scientific computing 20(1):33–61

Chouzenoux E, Pesquet J, Repetti A (2013) A block coordinate variable metric forward-backward algorithm. Tech. Rep. 00945918, HAL

Cohen A, Séré E (1996) Time-frequency localization by non-stationary wavelet packets. in Subband and Wavelet Transforms - Theory and Design, ed M T Smith and A Akansu, Kluwer Academic Publisher

Daubechies I, Defrise M, De Mol C (2004) An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. Communications on pure and applied mathematics 57(11):1413–1457

De Lathauwer L, De Moor B, Vandewalle J (2000) On the best rank-1 and rank-(r1,r2,. . .,rn) approximation of higher-order tensors. SIAM J Matrix Anal Appl 21(4):1324–1342

Delsarte P, Macq B, Slock D (1992) Signal adapted multiresolution transform for image coding. IEEE Trans Signal Process 42(11):2955–2966

Dobigeon N, Tourneret JY (2010) Bayesian orthogonal component analysis for sparse representation. IEEE Trans Signal Process 58(5):2675–2685

Duarte-Carvajalino JM, Sapiro G (2009) Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization. IEEE Trans Image Process 18(7):1395–1408

Elad M (2010) Sparse and redundant representations: From theory to applications in signal and image processing. Springer

Engan K, Aase SO, Hakon Husoy J (1999) Method of optimal directions for frame design. In: Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing (ICASSP), Washington, DC, USA, pp 2443–2446

Fadili J, Starck JL, Elad M, Donoho D (2010) MCALab: Reproducible research in signal and image decomposition and inpainting. IEEE Computing in Science and Engineering 12(1):44–62

Grippo GL, Sciandrone M (2000) On the convergence of the block nonlinear Gauss–Seidel method under convex constraints. Operations Research Letters 26(3):127–136

Jenatton R, Mairal J, Obozinski G, Bach F (2010) Proximal methods for sparse hierarchical dictionary learning. In: ICML

Jenatton R, Mairal J, Obozinski G, Bach F (2011) Proximal methods for hierarchical sparse coding. J Mach Learning Research 12:2297–2334

Kail G, Tourneret JY, Dobigeon N, Hlawatsch F (2012) Blind deconvolution of sparse pulse sequences under a minimum distance constraint: A partially collapsed Gibbs sampler method. IEEE Trans Signal Process 60(6):2727–2743

Lesage S, Gribonval R, Bimbot F, Benaroya L (2005) Learning Unions of Orthonormal Bases

with Thresholded Singular Value Decomposition. In: Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing (ICASSP), Philadelphia, PA, United States, vol V, pp 293–296

Lewicki MS, Sejnowski TJ (2000) Learning overcomplete representations. Neural Computation 12(2):337–365

Lu WS, Antoniou A (2000) Design of digital filters and filter banks by optimization: A state of the art review. In: Proc. EUSIPCO 2000, Tampere, Finland, vol 1, pp 351–354

Luo ZQ, Tseng P (1992) On the convergence of the coordinate descent method for convex differentiable minimization. Journal of Optimization Theory and Applications 72(1):7–35

Macq B, Mertes J (1993) Optimization of linear multiresolution transforms for scene adaptive coding. IEEE Trans Signal Process 41(12):3568–3572

Mailhé B, Lesage S, Gribonval R, Bimbot F, Vandergheynst P (2008) Shift-invariant dictionary learning for sparse representations: extending K-SVD. In: Proc. European Signal Process. Conf. (EUSIPCO), Lausanne, Switzerland

Mairal J, Sapiro G, Elad M (2008) Learning multiscale sparse representations for image and video restoration. SIAM Multiscale Modeling and Simulation 7(1):214–241

Mairal J, Bach F, Ponce J, Sapiro G (2010) Online learning for matrix factorization and sparse coding. J Mach Learning Research 11:10–60

Mairal J, Bach F, Ponce J (2012) Task-driven dictionary learning. IEEE Trans Patt Anal Mach Intell 34(4):791–804

Malgouyres F, Zeng T (2009) A predual proximal point algorithm solving a non negative basis pursuit denoising model. International journal of computer vision 83(3):294–311

Muller ME (1959) A note on a method for generating points uniformly on n-dimensional spheres. Comm Assoc Comput Mach 2(4):19–20

Olshausen BA, Field DJ (1997) Sparse coding with an overcomplete basis set: A strategy employed by v1? Vision Research 37(23):3311 – 3325

Ophir B, Lustig M, Elad M (2011) Multi-scale dictionary learning using wavelets. IEEE J Sel Topics Signal Process 5(5):1014–1024

Painter T, Spanias A (2000) Perceptual coding of digital audio. Proc IEEE 88(4):451–515

Peyré G, Fadili J, Starck JL (2010) Learning the morphological diversity. SIAM Journal on Imaging Sciences 3(3):646–669

Princen JP, Bradley AB (1986) Analysis/synthesis filter bank design based on time domain aliasing cancellation. IEEE Trans Acoust, Speech, Signal Process 34(5):1153–1161

Quinsac C, Dobigeon N, Basarab A, Tourneret JY, Kouamé D (2011) Bayesian compressed sensing in ultrasound imaging. In: Proc. of Third International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP11), San Juan, Puerto Rico

Razaviyayn M, Hong M, Luo Z Q (2013) A Unified Convergence Analysis of Block Successive Minimization Methods for Nonsmooth Optimization. SIAM Journal on Optimization 23(2):1126–1153

Rigamonti R, Sironi A, Lepetit V, Fua P (2013) Learning separable filters. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Portland, Oregon

Rubinstein R, Bruckstein AM, Elad M (2010a) Dictionaries for sparse representation. Proc IEEE - Special issue on applications of sparse representation and compressive sensing 98(6):1045–1057

Rubinstein R, Zibulevsky M, Elad M (2010b) Double sparsity: Learning sparse dictionaries for sparse signal approximation. IEEE Trans Signal Process 58(3):1553–1564

Sallee P, Olshausen BA (2002) Learning sparse multiscale image representations. Advances in neural information processing systems pp 1327–1334

Starck JL, Fadili J, Murtagh F (2007) The undecimated wavelet decomposition and its reconstruction. IEEE Trans Image Process 16(2):297–309

Thiagarajan J, Ramamurthy K, Spanias A (2011) Multilevel dictionary learning for sparse representation of images. In: Proc. IEEE Digital Signal Process. Workshop and IEEE Signal Process. Education Workshop (DSP/SPE), Sedona,

Arizona, pp 271–276

Tsiligkaridis T, Hero A, Zhou S (2013) On convergence properties of Kronecker graphical lasso algorithms. IEEE Trans Signal Process 61(7):1743–1755

Uhl A (1996) Image compression using non-stationary and inhomogeneous multiresolution analyses. Image and Vision Computing 14(5):365–371

Whittaker ET (1915) On the functions which are represented by the expansions of the interpolation theory. Proc Royal Soc of Edinburgh 35:181–194

Wiesel A (2012) Geodesic convexity and covariance estimation. IEEE Trans Signal Process 60(12):6182–6189