Convolutional Dictionary Learning via Local Processing

Vardan Papyan
vardanp@campus.technion.ac.il

Yaniv Romano
yromano@tx.technion.ac.il

Jeremias Sulam
jsulam@cs.technion.ac.il

Michael Elad
elad@cs.technion.ac.il

Technion - Israel Institute of Technology
Technion City, Haifa 32000, Israel

Abstract

Convolutional sparse coding is an increasingly popular model in the signal and image processing communities, tackling some of the limitations of traditional patch-based sparse representations. Although several works have addressed the dictionary learning problem under this model, these relied on an ADMM formulation in the Fourier domain, losing the sense of locality and the relation to the traditional patch-based sparse pursuit. A recent work suggested a novel theoretical analysis of this global model, providing guarantees that rely on a localized sparsity measure. Herein, we extend this local-global relation by showing how one can efficiently solve the convolutional sparse pursuit problem and train the filters involved, while operating locally on image patches. Our approach provides an intuitive algorithm that can leverage standard techniques from the sparse representations field. The proposed method is fast to train, simple to implement, and flexible enough that it can be easily deployed in a variety of applications. We demonstrate the proposed training scheme for image inpainting and image separation, achieving state-of-the-art results.

1. Introduction

The celebrated sparse representation model has led to impressive results in various applications over the last decade [10, 1, 29, 30, 8]. In this context one assumes that a signal $\mathbf{X} \in \mathbb{R}^N$ is a linear combination of a few columns, also called atoms, taken from a matrix $\mathbf{D} \in \mathbb{R}^{N \times M}$ termed a dictionary; i.e. $\mathbf{X} = \mathbf{D}\mathbf{\Gamma}$ where $\mathbf{\Gamma} \in \mathbb{R}^M$ is a sparse vector. Given $\mathbf{X}$, finding its sparsest representation, called sparse pursuit, amounts to solving the following problem

$$\min_{\mathbf{\Gamma}} \|\mathbf{\Gamma}\|_0 \text{ s.t. } \|\mathbf{X} - \mathbf{D}\mathbf{\Gamma}\|_2 \leq \epsilon, \tag{1}$$

where ϵ stands for the model mismatch or an additive noise strength. The solution for the above can be approximated using greedy algorithms such as Orthogonal Matching Pursuit (OMP) [6] or convex formulations such as BP [7]. The task of learning the model, i.e. identifying the dictionary $\mathbf{D}$ that best represents a set of training signals, is called dictionary learning and several methods have been proposed for tackling it, including K-SVD [1], MOD [13], online dictionary learning [21], trainlets [26], and more.

When dealing with high-dimensional images, traditional image processing algorithms train a local model for fully overlapping patches extracted from $\mathbf{X}$, process these independently using the trained local prior and then average the results to obtain a global estimate. This approach, which we refer to in what follows as *patch averaging*, gained much popularity and success due to its simplicity and high-performance [10, 30, 8, 20]. A different approach is the Convolutional Sparse Coding (CSC), which works with a global model by imposing a specific structure on the dictionary involved [15, 4, 19, 27, 17, 16, 18]. In particular, this assumes that $\mathbf{D}$ is a banded convolutional dictionary, implying that the signal is a superposition of a few local atoms, or filters, shifted to different positions. Unlike patch averaging that restores independently the same information in the image several times, CSC treats the information jointly and only once. Several works have presented algorithms for training convolutional dictionaries [4, 17, 27], circumventing some of the computational burdens of this problem by relying on ADMM solvers that operate in the Fourier domain. In doing so, these methods lost the connection to the patch-based processing paradigm, as widely practiced in many signal and image processing applications.

In this work, we propose a novel approach for training the CSC model, called slice-based dictionary learning. Unlike current methods, we leverage a localized strategy enabling the solution of the global problem in terms of only local computations in the original domain. The main advantages of our method over existing ones are:

1. It operates *locally* on patches, while solving faithfully

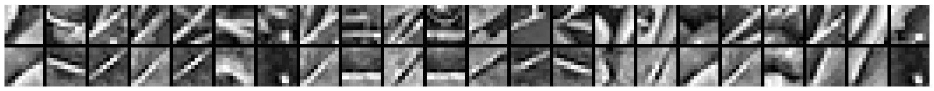


Figure 1: Top: Patches extracted from natural images. Bottom: Their corresponding slices. Observe how the slices are far simpler, and contained by their corresponding patches.

the *global* CSC problem;

2. It is easy to implement and intuitive to understand;
3. It reveals how one should modify current (and any) dictionary learning algorithms to solve the CSC problem in a variety of applications;
4. It can leverage standard techniques from the sparse representations field, such as OMP, LARS, K-SVD, MOD, online dictionary learning and trainlets;
5. When compared to state-of-the-art methods, it can be applied to standard-sized images, converges faster and provides a better model; and
6. It can naturally allow for a different number of non-zeros in each spatial location, according to the local signal complexity.

The rest of this paper is organized as follows: Section 2 reviews the CSC model. The proposed method is presented in Section 3 and contrasted with conventional approaches in Section 4. Section 5 shows how our method can be employed to tackle the tasks of image inpainting and separation, and later in Section 6 we demonstrate empirically our algorithms. We conclude this work in Section 7.

2. Convolutional sparse coding

The CSC model assumes that a global signal $\mathbf{X}$ can be decomposed as $\mathbf{X} = \sum_{i=1}^{m} \mathbf{d}_i * \mathbf{\Gamma}_i$, where $\mathbf{d}_i \in \mathbb{R}^n$ are local filters that are convolved with their corresponding features maps (or sparse representations) $\mathbf{\Gamma}_i \in \mathbb{R}^N$. Alternatively, following Figure 2, the above can be written in matrix

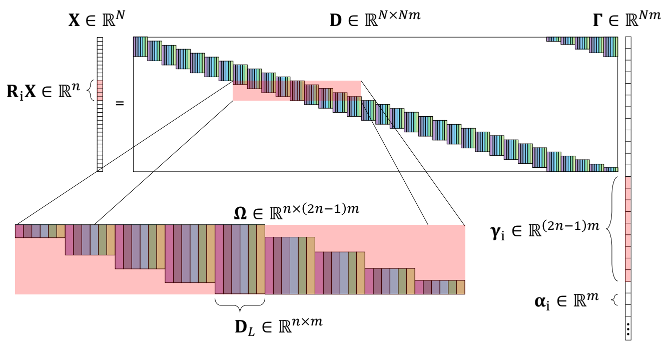

$\mathbf{X} \in \mathbb{R}^N$
$\mathbf{D} \in \mathbb{R}^{N\times Nm}$
$\mathbf{\Gamma} \in \mathbb{R}^{Nm}$
$\mathbf{R}_i\mathbf{X} \in \mathbb{R}^n$
=
$\mathbf{\Omega} \in \mathbb{R}^{n\times(2n-1)m}$
$\boldsymbol{\gamma}_i \in \mathbb{R}^{(2n-1)m}$
$\boldsymbol{\alpha}_i \in \mathbb{R}^m$
$\mathbf{D}_L \in \mathbb{R}^{n\times m}$

Figure 2: The CSC model and its constituent elements.

form as $\mathbf{X} = \mathbf{D\Gamma}$; where $\mathbf{D} \in \mathbb{R}^{N\times Nm}$ is a banded convolutional dictionary built from shifted versions of a local matrix $\mathbf{D}_L$, containing the atoms $\{\mathbf{d}_i\}_{i=1}^m$ as its columns, and $\mathbf{\Gamma} \in \mathbb{R}^{Nm}$ is a global sparse representation obtained by interlacing the $\{\mathbf{\Gamma}_i\}_{i=1}^m$. In this setting, a patch $\mathbf{R}_i\mathbf{X}$ taken from the global signal equals $\mathbf{\Omega}\boldsymbol{\gamma}_i$, where $\mathbf{\Omega} \in \mathbb{R}^{n\times(2n-1)m}$ is a *stripe* dictionary and $\boldsymbol{\gamma}_i \in \mathbb{R}^{(2n-1)m}$ is a *stripe* vector. Here we defined $\mathbf{R}_i \in \mathbb{R}^{n\times N}$ to be the operator that extracts the i-th n-dimensional patch from $\mathbf{X}$.

The work in [23] suggested a theoretical analysis of this global model, driven by a localized sparsity measure. Therein, it was shown that if all the stripes $\boldsymbol{\gamma}_i$ are sparse, the solution to the convolutional sparse pursuit problem is unique and can be recovered by greedy algorithms, such as the OMP [6], or convex formulations such as the Basis Pursuit (BP) [7]. This analysis was then extended in [24] to a noisy regime showing that, under similar sparsity assumptions, the global problem formulation and the pursuit algorithms are also stable. Herein, we leverage this local-global relation from an algorithmic perspective, showing how one can efficiently solve the convolutional sparse pursuit problem and train the dictionary (i.e., the filters) involved, while only operating locally.

Note that the global sparse vector $\mathbf{\Gamma}$ can be broken into a set of non-overlapping m-dimensional sparse vectors $\boldsymbol{\alpha}_{i=1}^N$, which we call *needles*. The essence of the presented algorithm is in the observation that one can express the global signal as $\mathbf{X} = \sum_{i=1}^N \mathbf{R}_i^T \mathbf{D}_L \boldsymbol{\alpha}_i$, where $\mathbf{R}_i^T \in \mathbb{R}^{N\times n}$ is the operator that puts $\mathbf{D}_L\boldsymbol{\alpha}_i$ in the i-th position and pads the rest of the entries with zeros. Denoting by $\mathbf{s}_i$ the i-th *slice* $\mathbf{D}_L\boldsymbol{\alpha}_i$, we can write the above as $\mathbf{X} = \sum_{i=1}^N \mathbf{R}_i^T \mathbf{s}_i$. It is important to stress that the slices do not correspond to patches extracted from the signal, $\mathbf{R}_i\mathbf{X}$, but rather to much simpler entities. They represent only a fraction of the i-th patch, since $\mathbf{R}_i\mathbf{X} = \mathbf{R}_i \sum_{j=1}^N \mathbf{R}_j^T \mathbf{s}_j$, i.e. a patch is constructed from several overlapping slices. Unlike current works in signal and image processing, which train a local dictionary on the patches $\{\mathbf{R}_i\mathbf{X}\}_{i=1}^N$, in what follows we define the learning problem with respect to the slices, $\{\mathbf{s}_i\}_{i=1}^N$, instead. In other words, we aim to train $\mathbf{D}_L$ instead of $\mathbf{\Omega}$. As a motivation, we present in Figure 1 a set of patches $\mathbf{R}_i\mathbf{X}$ extracted from natural images and their corresponding slices $\mathbf{s}_i$, obtained from the proposed algorithm, which will be presented in Section 3. Indeed, one can observe that the slices are simpler than the patches, as they contain less information.

3. Proposed method: slice-based dictionary learning

The convolutional dictionary learning problem refers to the following optimization[1] objective,

$$\min_{\mathbf{D},\mathbf{\Gamma}} \frac{1}{2}\|\mathbf{X}-\mathbf{D}\mathbf{\Gamma}\|_2^2+\lambda\|\mathbf{\Gamma}\|_1, \tag{2}$$

for a convolutional dictionary $\mathbf{D}$ as in Figure 2 and a Lagrangian parameter λ that controls the sparsity level. Employing the decomposition of $\mathbf{X}$ in terms of its slices, and the separability of the ℓ_1 norm, the above can be written as the following constrained minimization problem,

$$\begin{aligned}\min_{\substack{\mathbf{D}_L,\{\boldsymbol{\alpha}_i\}_{i=1}^N,\\ \{\mathbf{s}_i\}_{i=1}^N}} \; & \frac{1}{2}\|\mathbf{X}-\sum_{i=1}^N \mathbf{R}_i^T\mathbf{s}_i\|_2^2+\lambda\sum_{i=1}^N\|\boldsymbol{\alpha}_i\|_1 \\ \text{s.t. } & \mathbf{s}_i=\mathbf{D}_L\boldsymbol{\alpha}_i.\end{aligned} \tag{3}$$

One could tackle this problem using half-quadratic splitting [14] by introducing a penalty term over the violation of the constraint and gradually increasing its importance. Alternatively, we can employ the ADMM algorithm [3] and solve the augmented Lagrangian formulation (in its scaled form),

$$\begin{aligned}\min_{\substack{\mathbf{D}_L,\{\boldsymbol{\alpha}_i\}_{i=1}^N,\\ \{\mathbf{s}_i\}_{i=1}^N,\{\mathbf{u}_i\}_{i=1}^N}} \; & \frac{1}{2}\|\mathbf{X}-\sum_{i=1}^N\mathbf{R}_i^T\mathbf{s}_i\|_2^2 \\ & +\sum_{i=1}^N\left(\lambda\|\boldsymbol{\alpha}_i\|_1+\frac{\rho}{2}\|\mathbf{s}_i-\mathbf{D}_L\boldsymbol{\alpha}_i+\mathbf{u}_i\|_2^2\right),\end{aligned} \tag{4}$$

where $\{\mathbf{u}_i\}_{i=1}^N$ are the dual variables that enable the constrains to be met.

3.1. Local sparse coding and dictionary update

The minimization of Equation (4) with respect to all the needles $\{\boldsymbol{\alpha}_i\}_{i=1}^N$ is separable, and can be addressed independently for every $\boldsymbol{\alpha}_i$ by leveraging standard tools such as LARS. This also allows for having a different number of non-zeros per slice, depending on the local complexity. Similarly, the minimization with respect to $\mathbf{D}_L$ can be done using any patch-based dictionary learning algorithm such as the K-SVD, MOD, online dictionary learning or trainlets. Note that in the dictionary update stage, while minimizing for $\mathbf{D}_L$, one could refrain from iterating until convergence, and instead perform only a few iterations before proceeding with the remaining variables. In addition, when employing the K-SVD dictionary update, the $\{\boldsymbol{\alpha}_i\}$ are also updated subject to the support found in the sparse pursuit stage.

[1]Hereafter, we assume that the atoms in the dictionary are normalized to a unit ℓ_2 norm.

3.2. Slice update via local Laplacian

The minimization of Equation (4) with respect to all the slices $\{\mathbf{s}_i\}_{i=1}^N$ amounts to solving the quadratic problem

$$\min_{\{\mathbf{s}_i\}_{i=1}^N} \frac{1}{2}\|\mathbf{X}-\sum_{i=1}^N\mathbf{R}_i^T\mathbf{s}_i\|_2^2+\frac{\rho}{2}\sum_{i=1}^N\|\mathbf{s}_i-\mathbf{D}_L\boldsymbol{\alpha}_i+\mathbf{u}_i\|_2^2. \tag{5}$$

Taking the derivative with respect to the variables $\mathbf{s}_1,\mathbf{s}_2,\dots\mathbf{s}_N$ and nulling them, we obtain the following system of linear equations

$$\begin{gathered}\mathbf{R}_1(\sum_{i=1}^N\mathbf{R}_i^T\mathbf{s}_i-\mathbf{X})+\rho(\mathbf{s}_1-\mathbf{D}_L\boldsymbol{\alpha}_1+\mathbf{u}_1)=\mathbf{0} \\ \vdots \\ \mathbf{R}_N(\sum_{i=1}^N\mathbf{R}_i^T\mathbf{s}_i-\mathbf{X})+\rho(\mathbf{s}_N-\mathbf{D}_L\boldsymbol{\alpha}_N+\mathbf{u}_N)=\mathbf{0}.\end{gathered} \tag{6}$$

Defining

$$\bar{\mathbf{R}}=\begin{bmatrix}\mathbf{R}_1\\ \vdots\\ \mathbf{R}_N\end{bmatrix} \quad \bar{\mathbf{S}}=\begin{bmatrix}\mathbf{s}_1\\ \vdots\\ \mathbf{s}_N\end{bmatrix} \quad \bar{\mathbf{Z}}=\begin{bmatrix}\mathbf{D}_L\boldsymbol{\alpha}_1-\mathbf{u}_1\\ \vdots\\ \mathbf{D}_L\boldsymbol{\alpha}_N-\mathbf{u}_N\end{bmatrix}, \tag{7}$$

the above can be written as

$$\begin{aligned}&\mathbf{0}=\bar{\mathbf{R}}\left(\bar{\mathbf{R}}^T\bar{\mathbf{S}}-\mathbf{X}\right)+\rho\left(\bar{\mathbf{S}}-\bar{\mathbf{Z}}\right)\\ &\implies \bar{\mathbf{S}}=\left(\bar{\mathbf{R}}\bar{\mathbf{R}}^T+\rho\mathbf{I}\right)^{-1}\left(\bar{\mathbf{R}}\mathbf{X}+\rho\bar{\mathbf{Z}}\right).\end{aligned} \tag{8}$$

Using the Woodbury matrix identity and the fact that $\bar{\mathbf{R}}^T\bar{\mathbf{R}}=\sum_{i=1}^N\mathbf{R}_i^T\mathbf{R}_i=n\mathbf{I}$, where $\mathbf{I}$ is the identity matrix, the above is equal to

$$\begin{aligned}\bar{\mathbf{S}}&=\left(\frac{1}{\rho}\mathbf{I}-\frac{1}{\rho^2}\bar{\mathbf{R}}\left(\mathbf{I}+\frac{1}{\rho}\bar{\mathbf{R}}^T\bar{\mathbf{R}}\right)^{-1}\bar{\mathbf{R}}^T\right)\left(\bar{\mathbf{R}}\mathbf{X}+\rho\bar{\mathbf{Z}}\right)\\ &=\left(\mathbf{I}-\bar{\mathbf{R}}\left(\rho\mathbf{I}+n\mathbf{I}\right)^{-1}\bar{\mathbf{R}}^T\right)\left(\frac{1}{\rho}\bar{\mathbf{R}}\mathbf{X}+\bar{\mathbf{Z}}\right).\end{aligned} \tag{9}$$

Plugging the definitions of $\bar{\mathbf{R}}$, $\bar{\mathbf{S}}$ and $\bar{\mathbf{Z}}$, we obtain

$$\begin{aligned}\mathbf{s}_i=&\left(\frac{1}{\rho}\mathbf{R}_i\mathbf{X}+\mathbf{D}_L\boldsymbol{\alpha}_i-\mathbf{u}_i\right)\\ &-\mathbf{R}_i\left(\frac{1}{\rho+n}\sum_{j=1}^N\mathbf{R}_j^T\left(\frac{1}{\rho}\mathbf{R}_j\mathbf{X}+\mathbf{D}_L\boldsymbol{\alpha}_j-\mathbf{u}_j\right)\right).\end{aligned} \tag{10}$$

Although seemingly complicated at first glance, the above is simple to interpret and implement in practice. This expression indicates that one should (i) compute the estimated slices $\mathbf{p}_i=\frac{1}{\rho}\mathbf{R}_i\mathbf{X}+\mathbf{D}_L\boldsymbol{\alpha}_i-\mathbf{u}_i$, then (ii) aggregate them to obtain the global estimate $\hat{\mathbf{X}}=\sum_{j=1}^N\mathbf{R}_j^T\mathbf{p}_j$, and finally

```
Algorithm 1: Slice-based dictionary learning
Input  : Signal $\mathbf{X}$, initial dictionary $\mathbf{D}_L$
Output: Trained dictionary $\mathbf{D}_L$, needles $\{\boldsymbol{\alpha}_i\}_{i=1}^N$ and
        slices $\{\mathbf{s}_i\}_{i=1}^N$
Initialization:
        $\mathbf{s}_i = \frac{1}{n}\mathbf{R}_i\mathbf{X}, \quad \mathbf{u}_i = \mathbf{0}$   (11)
for $iteration = 1 : T$ do
    Local sparse pursuit (needle):
        $\boldsymbol{\alpha}_i = \arg\min_{\hat{\boldsymbol{\alpha}}_i} \frac{\rho}{2}\|\mathbf{s}_i - \mathbf{D}_L\hat{\boldsymbol{\alpha}}_i + \mathbf{u}_i\|_2^2 + \lambda\|\hat{\boldsymbol{\alpha}}_i\|_1$   (12)
    Slice reconstruction:
        $\mathbf{p}_i = \frac{1}{\rho}\mathbf{R}_i\mathbf{X} + \mathbf{D}_L\boldsymbol{\alpha}_i - \mathbf{u}_i$   (13)
    Slice aggregation:
        $\hat{\mathbf{X}} = \sum_{j=1}^{N}\mathbf{R}_j^T\mathbf{p}_j$   (14)
    Slice update via local Laplacian:
        $\mathbf{s}_i = \mathbf{p}_i - \frac{1}{\rho+n}\mathbf{R}_i\hat{\mathbf{X}}$   (15)
    Dual variable update:
        $\mathbf{u}_i = \mathbf{u}_i + \mathbf{s}_i - \mathbf{D}_L\boldsymbol{\alpha}_i$   (16)
    Dictionary update:
        $\mathbf{D}_L, \{\boldsymbol{\alpha}_i\}_{i=1}^N = \arg\min_{\mathbf{D}_L, \{\hat{\boldsymbol{\alpha}}_i\}_{i=1}^N} \sum_{i=1}^{N}\|\mathbf{s}_i - \mathbf{D}_L\hat{\boldsymbol{\alpha}}_i + \mathbf{u}_i\|_2^2$
            s.t. $\mathrm{supp}(\hat{\boldsymbol{\alpha}}_i) = \mathrm{supp}(\boldsymbol{\alpha}_i)$
end   (17)
```

(iii) subtract from $\mathbf{p}_i$ the corresponding patch from the aggregated signal, i.e. $\mathbf{R}_i\hat{\mathbf{X}}$. As a remark, since this update essentially subtracts from $\mathbf{p}_i$ an averaged version of it, it can be seen as a patch-based local Laplacian operation.

3.3. Boundary conditions

In the description of the CSC model (see Figure 2), we assumed for simplicity circulant boundary conditions. In practice, however, natural signals such as images are in general not circulant and special treatment is needed for the boundaries. One way of handling this issue is by assuming that $\mathbf{X} = \mathbf{MD\Gamma}$, where $\mathbf{M} \in \mathbb{R}^{N \times N+2(n-1)}$ is matrix that crops the first and last $n-1$ rows of the dictionary $\mathbf{D}$ (see Figure 2). The change needed in Algorithm 1 to incorporate $\mathbf{M}$ is minor. Indeed, one has to simply replace the patch extraction operator $\mathbf{R}_i$, with $\mathbf{R}_i\mathbf{M}^T$, where the operator $\mathbf{M}^T \in \mathbb{R}^{N+2(n-1) \times N}$ pads a global signal with $n-1$ zeros on the boundary and $\mathbf{R}_i$ extracts a patch from the result. In addition, one has to replace the patch placement operator $\mathbf{R}_i^T$ with $\mathbf{MR}_i^T$, which simply puts the input in the location of the i-th patch and then crops the result.

3.4. From patches to slices

The ADMM variant of the proposed algorithm, named *slice-based dictionary learning*, is summarized in Algorithm 1. While we have assumed the data corresponds to one signal $\mathbf{X}$, this can be easily extended to consider several signals.

At this point, a discussion regarding the relation between this algorithm and standard (patch-based) dictionary learning techniques is in place. Indeed, from a quick glance the two approaches seem very similar: Both perform local sparse pursuit on local patches extracted from the signal, then update the dictionary to represent these patches better, and finally apply patch-averaging to obtain a global estimate of the reconstructed signal. Moreover, both iterate this process in a block-coordinate descent manner in order to minimize the overall objective. So, what is the difference between this algorithm and previous approaches?

The answer lies in the *migration from patches to slices*. While originally dictionary learning algorithms aimed to represent patches $\mathbf{R}_i\mathbf{X}$ taken from the signal, our scheme suggests to train the dictionary to construct slices, which do not necessarily reconstruct the patch fully. Instead, only the summation of these slices results in the reconstructed patches. To illustrate this relation, we show in Figure 3 the decomposition of several patches in terms of their constituent slices. One can observe that although the slices are simple in nature, they manage to construct the rather complex patches. The difference between this illustration and that of Figure 1 is that the latter shows patches $\mathbf{R}_i\mathbf{X}$ and only the slices that are *fully* contained in them.

Note that the slices are not mere auxiliary variables, but rather emerge naturally from the convolutional formulation. After initializing these with patches from the signal,

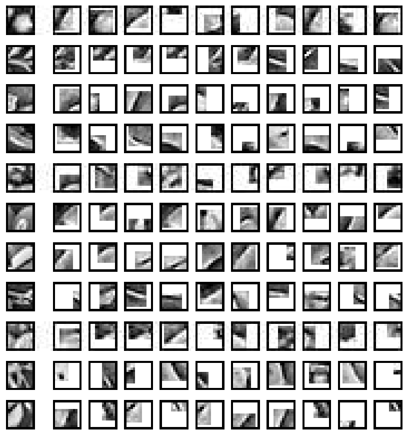


Figure 3: The first column contains patches extracted from the training data, and second to eleventh columns are the corresponding slices constructing these patches. Only the ten slices with the highest energy are presented.

$\mathbf{s}_i = \frac{1}{n}\mathbf{R}_i\mathbf{X}$, each iteration progressively "carves" portions from the patch via the local Laplacian, resulting in simpler constructions. Eventually, these variables are guaranteed to converge to $\mathbf{D}_L\boldsymbol{\alpha}_i$ – the slices we have defined.

Having established the similarities and differences between the traditional patch-based approach and the slice alternative, one might wonder what is the advantage of working with slices over patches. In the conventional approach, the patches are processed independently, ignoring their overlap. In the slice-based case, however, the local Laplacian forces the slices to communicate and reach a consensus on the reconstructed signal. Put differently, the CSC offers a global model, while earlier patch-based methods used local models without any holistic fusion of them.

4. Comparison to other methods

In this section we explain further the advantages of our method, and compare it to standard algorithms for training the CSC model such as [17, 28]. Arguably the main difference resides in our localized treatment, as opposed to the global Fourier domain processing. Our approach enables the following benefits:

1. The sparse pursuit step can be done separately for each slice and is therefore trivial to parallelize.

2. The algorithm can work in a complete online regime where in each iteration it samples a random subset of slices, solves a pursuit for these and then updates the dictionary accordingly. Adopting a similar strategy in the competing algorithms [17, 28] might be problematic, since these are deployed in the Fourier domain on global signals and it is therefore unclear how to operate on a subset of local patches.

3. Our algorithm can be easily modified to allow a different number of non-zeros in each location of the global signal. Such local complexity adaptation cannot be offered by the Fourier-oriented algorithms.

We now turn to comparing the proposed algorithm to alternative methods in terms of computational complexity. Denote by I the number of signals on which the dictionary is trained, and by k the maximal number of non-zeros in a needle[2] $\boldsymbol{\alpha}_i$. At each iteration of our algorithm we employ LARS that has a complexity of $O(k^3 + mk^2 + nm)$ per slice [20], resulting in $O(IN(k^3 + mk^2 + nm) + nm^2)$ computations for all N slices in all the I images. The last term, nm^2, corresponds to the precomputation of the Gram of the dictionary $\mathbf{D}_L$ (which is in general negligible). Then, given the obtained needles, we reconstruct the slices, requiring $O(INnk)$, aggregate the results to form the global estimate,

[2]Although we solve the Lagrangian formulation of LARS, we also limit the maximal number of non-zeros per needle to be at most k.

Method	Time Complexity
[17] $I < m$	$\underbrace{mI^2N + (q-1)mIN}_{\text{linear systems}} + \underbrace{\boldsymbol{qImN\log(N)}}_{\text{FFT}} + \underbrace{qImN}_{\text{thresholding}}$
[17] $I \geq m$	$\underbrace{m^3N + (q-1)m^2N}_{\text{linear systems}} + \underbrace{\boldsymbol{qImN\log(N)}}_{\text{FFT}} + \underbrace{qImN}_{\text{thresholding}}$
Ours	$\underbrace{\boldsymbol{INnm} + IN(k^3 + mk^2)}_{\text{LARS / OMP}} + \underbrace{nm^2}_{\text{Gram}} + \underbrace{INk(n+m) + nm^2}_{\text{K-SVD}}$

Table 1: Complexity analysis. For the convenience of the reader, the dominant term is highlighted in red color.

incurring $O(INn)$, and update the slices, which requires an additional $O(INn)$. These steps are negligible compared to the sparse pursuits and are thus omitted in the final expression. Finally, we update the dictionary using the K-SVD, which is $O(nm^2 + INkn + INkm)$ [25]. We summarize the above in Table 1. In addition, we present in the same table the complexity of each iteration of the (Fourier-based) algorithm in [17]. In this case, q corresponds to the number of inner iterations in their ADMM solver of the sparse pursuit and dictionary update.

The most computationally demanding step in our algorithm is the local sparse pursuit, which is $O(NI(k^3 + mk^2 + nm))$. Assuming that the needles are very sparse, which indeed happens in all of our experiments, this reduces to $O(NImn)$. On the other hand, the complexity in the algorithm of [17] is dominated by the computation of the FFT, which is $O(NImq\log(N))$. We conclude that our algorithm scales linearly with the global dimension, while theirs grows as $N\log(N)$. Note that this also holds for other related methods, such as that of [28], which also depend on the global FFT. Moreover, one should remember the fact that in our scheme one might run the pursuits on a small percentage of the total number of slices, meaning that in practice our algorithm can scale as $O(\mu NInm)$, where μ is a constant smaller than one.

5. Image processing via CSC

In this section, we demonstrate our proposed algorithm on several image processing tasks. Note that the discussion thus far focused on one dimensional signals, however it can be easily generalized to images by replacing the convolutional structure in the CSC model with block-circulant circulant-block (BCCB) matrices.

5.1. Image inpainting

Assume an original image $\mathbf{X}$ is multiplied by a diagonal binary matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$, which masks the entries $\mathbf{X}_i$ in which $\mathbf{A}(i,i) = 0$. In the task of image inpainting, given the corrupted image $\mathbf{Y} = \mathbf{AX}$, the goal is to restore the original unknown $\mathbf{X}$. One can tackle this problem by solving the following CSC problem

$$\min_{\mathbf{\Gamma}} \frac{1}{2}\|\mathbf{Y} - \mathbf{AD\Gamma}\|_2^2 + \lambda\|\mathbf{\Gamma}\|_1, \qquad (18)$$

where we assume the dictionary $\mathbf{D}$ was pretrained. Using similar steps to those leading to Equation (4), the above can be written as

$$\min_{\substack{\{\boldsymbol{\alpha}_i\}_{i=1}^N,\{\mathbf{s}_i\}_{i=1}^N,\\ \{\mathbf{u}_i\}_{i=1}^N}} \frac{1}{2}\|\mathbf{Y}-\mathbf{A}\sum_{i=1}^N \mathbf{R}_i^T\mathbf{s}_i\|_2^2 + \sum_{i=1}^N \Big(\lambda\|\boldsymbol{\alpha}_i\|_1 + \frac{\rho}{2}\|\mathbf{s}_i-\mathbf{D}_L\boldsymbol{\alpha}_i+\mathbf{u}_i\|_2^2\Big). \tag{19}$$

This objective can be minimized via the algorithm described in the previous section. Moreover, the minimization with respect to the local sparse codes $\{\boldsymbol{\alpha}_i\}_{i=1}^N$ remains the same. The only difference regards the update of the slices $\{\mathbf{s}_i\}_{i=1}^N$, in which case one obtains the following expression

$$\mathbf{s}_i = \left(\frac{1}{\rho}\mathbf{R}_i\mathbf{Y}+\mathbf{D}_L\boldsymbol{\alpha}_i-\mathbf{u}_i\right) -\mathbf{R}_i\left(\frac{1}{\rho+n}\mathbf{A}\sum_{j=1}^N\mathbf{R}_j^T\left(\frac{1}{\rho}\mathbf{R}_j\mathbf{Y}+\mathbf{D}_L\boldsymbol{\alpha}_j-\mathbf{u}_j\right)\right). \tag{20}$$

The steps leading to the above equation are almost identical to those in subsection 3.2, and they only differ in the incorporation of the mask $\mathbf{A}$.

5.2. Texture and cartoon separation

In this task the goal is to decompose an image $\mathbf{X}$ into its texture component $\mathbf{X}_T$ that contains highly oscillating or pseudo-random patterns, and a cartoon part $\mathbf{X}_C$ that is a piece-wise smooth image. Many image separation algorithms tackle this problem by imposing a prior on both components. For cartoon, one usually employs the isotropic (or anisotropic) Total Variation norm, denoted by $\|\mathbf{X}_C\|_{TV}$. The modeling of texture, on the other hand, is more difficult and several approaches have been considered over the years [12, 2, 22, 32].

In this work, we propose to model the texture component using the CSC model. As such, the task of separation amounts to solving the following problem

$$\min_{\substack{\mathbf{D}_T,\boldsymbol{\Gamma}_T,\\ \mathbf{X}_C}} \frac{1}{2}\left\|\mathbf{X}-\mathbf{D}_T\boldsymbol{\Gamma}_T-\mathbf{X}_C\right\|_2^2+\lambda\left\|\boldsymbol{\Gamma}_T\right\|_1+\xi\|\mathbf{X}_C\|_{TV}, \tag{21}$$

where $\mathbf{D}_T$ is a convolutional (texture) dictionary, and $\boldsymbol{\Gamma}_T$ is its corresponding sparse vector. Using similar derivations to those presented in Section 3.2, the above is equivalent to

$$\begin{aligned}\min_{\substack{\mathbf{D}_L,\boldsymbol{\alpha}_T^i,\mathbf{s}_T^i,\\ \mathbf{X}_C,\mathbf{Z}_C}} &\frac{1}{2}\left\|\mathbf{X}-\sum_{i=1}^N\mathbf{R}_i^T\mathbf{s}_T^i-\mathbf{X}_C\right\|_2^2 \\ &+\lambda\sum_{i=1}^N\left\|\boldsymbol{\alpha}_T^i\right\|_1+\xi\|\mathbf{Z}_C\|_{TV} \\ \text{s.t.}\quad &\mathbf{s}_T^i=\mathbf{D}_L\boldsymbol{\alpha}_T^i,\quad \mathbf{X}_C=\mathbf{Z}_C,\end{aligned} \tag{22}$$

where we split the variable $\mathbf{X}_C$ into $\mathbf{X}_C=\mathbf{Z}_C$ in order to facilitate the minimization over the TV norm. Its corresponding ADMM formulation[3] is given by

$$\begin{aligned}\min_{\substack{\mathbf{D}_L,\boldsymbol{\alpha}_T^i,\mathbf{s}_T^i,\mathbf{u}_T^i,\\ \mathbf{X}_C,\mathbf{Z}_C,\mathbf{V}_C}} &\frac{1}{2}\left\|\mathbf{X}-\sum_{i=1}^N\mathbf{R}_i^T\mathbf{s}_T^i-\mathbf{X}_C\right\|_2^2 \\ &+\sum_{i=1}^N\left(\frac{\rho}{2}\left\|\mathbf{s}_T^i-\mathbf{D}_L\boldsymbol{\alpha}_T^i+\mathbf{u}_T^i\right\|_2^2+\lambda\left\|\boldsymbol{\alpha}_T^i\right\|_1\right) \\ &+\frac{\eta}{2}\|\mathbf{X}_C-\mathbf{Z}_C+\mathbf{V}_C\|_2^2+\xi\|\mathbf{Z}_C\|_{TV},\end{aligned} \tag{23}$$

where $\{\mathbf{s}_T^i\}_{i=1}^N$, $\{\boldsymbol{\alpha}_T^i\}_{i=1}^N$ and $\{\mathbf{u}_T^i\}_{i=1}^N$ are the texture slices, needles and dual variables, respectively, and $\mathbf{V}_C$ is the dual variable of the global cartoon $\mathbf{X}_C$. The above optimization problem can be minimized by slightly modifying Algorithm 1. The update for $\{\boldsymbol{\alpha}_i\}_{i=1}^N$ is a sparse pursuit and the update for the $\mathbf{Z}_C$ variable is a TV denoising problem. Then, one can update the $\{\mathbf{s}_T^i\}_{i=1}^N$ and $\mathbf{X}_C$ jointly by

$$\begin{aligned}\mathbf{s}_T^i &= \frac{1}{\rho}\mathbf{p}_T^i-\frac{\frac{1}{\rho}}{1+\frac{n^2}{\rho}+\frac{1}{\eta}}\mathbf{R}_i\left(\frac{1}{\rho}\sum_{j=1}^N\mathbf{R}_j^T\mathbf{p}_T^j+\frac{1}{\eta}\mathbf{Q}_C\right) \\ \mathbf{X}_C &= \frac{1}{\eta}\mathbf{Q}_C-\frac{\frac{1}{\eta}}{1+\frac{n^2}{\rho}+\frac{1}{\eta}}\left(\frac{1}{\rho}\sum_{j=1}^N\mathbf{R}_j^T\mathbf{p}_T^j+\frac{1}{\eta}\mathbf{Q}_C\right),\end{aligned} \tag{24}$$

where $\mathbf{p}_T^i=\mathbf{R}_i\mathbf{X}+\rho\left(\mathbf{D}_L\boldsymbol{\alpha}_T^i-\mathbf{u}_T^i\right)$ and $\mathbf{Q}_C=\mathbf{X}+\eta\left(\mathbf{Z}_C-\mathbf{V}_C\right)$. The final step of the algorithm is updating the texture dictionary $\mathbf{D}_L$ via any dictionary learning method.

6. Experiments

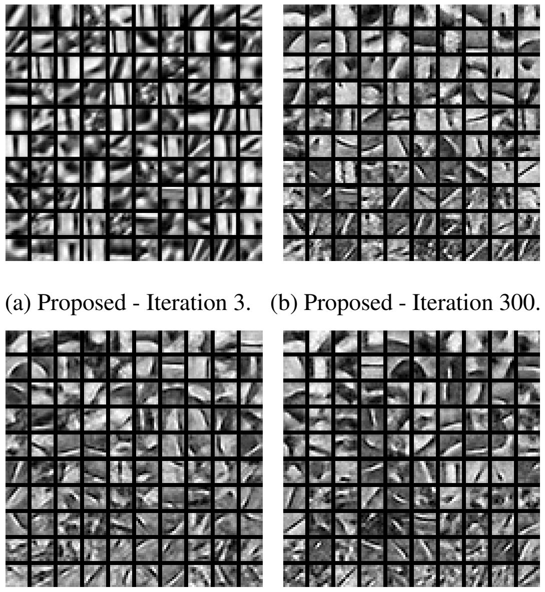
We turn to demonstrate our proposed slice-based dictionary learning. Throughout the experiments we use the LARS algorithm [9] to solve the LASSO problem and the K-SVD [1] for the dictionary learning. The reader should keep in mind, nevertheless, that one could use any other pursuit or dictionary learning algorithm for the respective updates. In all experiments, the number of filters trained are 100 and they are of size 11×11.

6.1. Slice-based dictionary learning

Following the test setting presented in [17], we run our proposed algorithm to solve Equation (2) with $\lambda=1$ on the Fruit dataset [31], which contains ten images. As in [17], the images were mean subtracted and contrast normalized. We present in Figure 4 the dictionary obtained

[3]Disregarding the training of the dictionary, this is a standard two-function ADMM problem. The first set of variables are $\{\mathbf{s}_T^i\}_{i=1}^N$ and $\mathbf{X}_C$, and the second are $\{\boldsymbol{\alpha}_T^i\}_{i=1}^N$ and $\mathbf{Z}_C$.

after several iterations using our proposed slice-based dictionary learning, and compare it to the result in [17] and also to the method AVA-AMS in [28]. Note that all three methods handle the boundary conditions, which were discussed in Section 3.3. We compare in Figure 5 the objective of the three algorithms as function of time, showing that our algorithm is more stable and also converges faster. In addition, to demonstrate one of the advantages of our scheme, we train the dictionary on a small subset (30%) of all slices and present the obtained result in the same figure.


(a) Proposed - Iteration 3. (b) Proposed - Iteration 300.

(c) [17]. (d) [28].

Figure 4: The dictionary obtained after 3 and 300 iterations using the slice-based dictionary learning method. Notice how the atoms become crisper as the iterations progress. For comparison, we present also the result of [17] and [28].

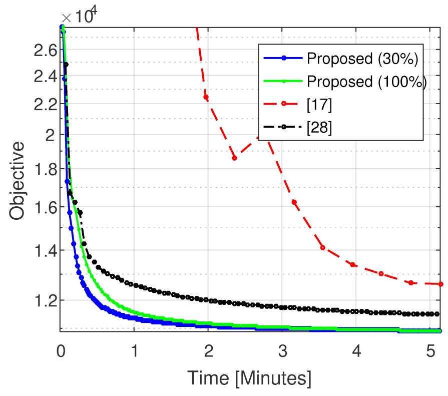


Figure 5: Our method versus the those in [17] and [28].

6.2. Image inpainting

We turn to test our proposed algorithm on the task of image inpainting, as described in Section 5.1. We follow the experimental setting presented in [17] and compare to their state-of-the-art method using their publicly available code. The dictionaries employed in both approaches are trained on the Fruit dataset, as described in the previous subsection (see Figure 4). For a fair comparison, in the inference stage, we tuned the parameter λ for both approaches. Table 2 presents the results in terms of peak signal-to-noise ratio (PSNR) on a set of publicly available standard test images, showing our method leads to quantitatively better results[4]. Figure 6 compares the two visually, showing our method also leads to better qualitative results.

A common strategy in image restoration is to train the dictionary on the corrupted image itself, as shown in [11], as opposed to employing a dictionary trained on a separate collection of images. The algorithm presented in Section 5.1 can be easily adapted to this framework by updating the local dictionary on the slices obtained at every iteration. To exemplify the benefits of this, we include the results[5] obtained by using this approach in Table 2 and Figure 6.

6.3. Texture and cartoon separation

We conclude by applying our proposed slice-based dictionary learning algorithm to the task of texture and cartoon separation. The TV denoiser used in the following experiments is the publicly available software of [5]. We run our method on the synthetic image Sakura and a portion extracted from Barbara, both taken from [22], and on the image Cat, originally from [32]. For each of these, we compare with the corresponding methods. We present the results of all three experiments in Figure 7, together with the

[4]The PSNR is computed as $20\log(\sqrt{N}/\|\mathbf{X}-\hat{\mathbf{X}}\|_2)$, where $\mathbf{X}$ and $\hat{\mathbf{X}}$ are the original and restored images. Since the images are normalized, the range of PSNR values is non-standard.

[5]A comparison with the method of [17] was not possible in this case, as their implementation cannot handle training a dictionary on standard-sized images.

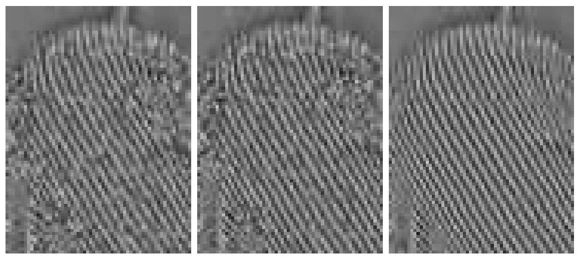


Figure 6: Visual comparison on a cropped region extracted from the image Barbara. Left: [17] (PSNR = 5.22dB). Middle: Ours (PSNR = 6.24dB). Right: Ours with dictionary trained on the corrupted image (PSNR = 12.65dB).

	Barbara	Boat	House	Lena	Peppers	C.man	Couple	Finger	Hill	Man	Montage
Heide et al.	11.00	10.29	10.18	11.77	9.41	9.74	11.99	15.55	10.37	11.60	15.11
Proposed	11.67	10.33	10.56	11.92	9.18	9.95	12.25	16.04	10.66	11.84	15.40
Image specific	15.20	11.60	11.77	12.35	11.45	10.68	12.41	16.07	10.90	11.71	15.67

Table 2: Comparison between the slice-based dictionary learning and the algorithm in [17] on the task of image inpainting.

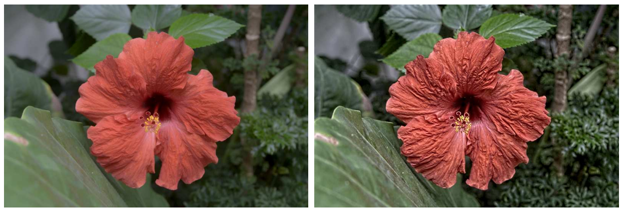

(a) Original image. (b) Enhanced output.

Figure 8: Enhancement of the image Flower via cartoon-texture separation.

trained dictionaries. Lastly, as an application for our texture separation algorithm, we enhance the image Flower by multiplying its texture component by a scalar factor (greater than one) and combining the result with the original image. We treat the colored image by transforming it to the Lab color space, manipulating the L channel, and finally transforming the result back to the original domain. The original image and the obtained result are depicted in Figure 8. One can observe that our approach does not suffer from halos, gradient reversals or other common enhancement artifacts.

7. Conclusion

In this work we proposed the slice-based dictionary learning algorithm. Our method employs standard patch-based tools from the realm of sparsity to solve the global CSC problem. We have shown the relation between our method and the patch-averaging paradigm, clarifying the main differences between the two: (i) the migration from patches to the simpler entities called slices, and (ii) the application of a local Laplacian that results in a global consensus. Finally, we illustrated the advantages of the proposed algorithm in a series of applications and compared it to related state-of-the-art methods.

8. Acknowledgments

The research leading to these results has received funding in part from the European Research Council under EU's 7th Framework Program, ERC under Grant 320649, and in part by Israel Science Foundation (ISF) grant no. 1770/14.

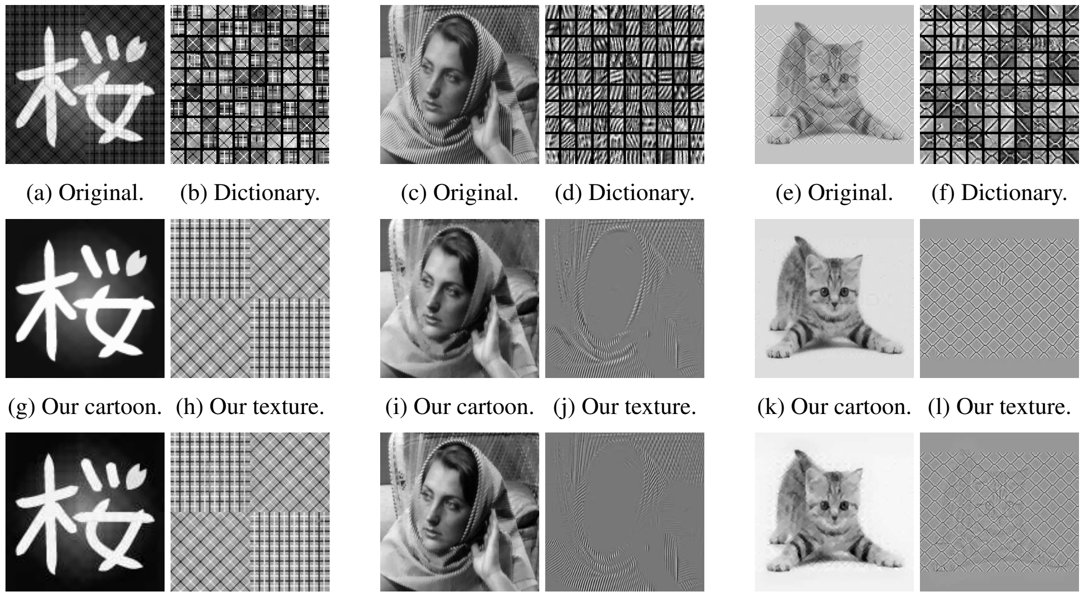

(a) Original. (b) Dictionary. (c) Original. (d) Dictionary. (e) Original. (f) Dictionary.
(g) Our cartoon. (h) Our texture. (i) Our cartoon. (j) Our texture. (k) Our cartoon. (l) Our texture.
(m) [22]. (n) [22]. (o) [22]. (p) [22]. (q) [32]. (r) [32].

Figure 7: Texture and cartoon separation for the images Sakura, Barbara and Cat.

References

[1] M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[2] J.-F. Aujol, G. Gilboa, T. Chan, and S. Osher. Structure-texture image decompositionmodeling, algorithms, and parameter selection. *International Journal of Computer Vision*, 67(1):111–136, 2006.

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[4] H. Bristow, A. Eriksson, and S. Lucey. Fast convolutional sparse coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 391–398, 2013.

[5] S. H. Chan, R. Khoshabeh, K. B. Gibson, P. E. Gill, and T. Q. Nguyen. An augmented lagrangian method for total variation video restoration. *IEEE Transactions on Image Processing*, 20(11):3097–3111, 2011.

[6] S. Chen, S. A. Billings, and W. Luo. Orthogonal least squares methods and their application to non-linear system identification. *International Journal of Control*, 50(5):1873–1896, 1989.

[7] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic Decomposition by Basis Pursuit. *SIAM Review*, 43(1):129–159, 2001.

[8] W. Dong, L. Zhang, G. Shi, and X. Wu. Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization. *IEEE Transactions on Image Processing*, 20(7):1838–1857, 2011.

[9] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, et al. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

[10] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[11] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, Dec. 2006.

[12] M. Elad, J.-L. Starck, P. Querre, and D. L. Donoho. Simultaneous cartoon and texture image inpainting using morphological component analysis (mca). *Applied and Computational Harmonic Analysis*, 19(3):340–358, 2005.

[13] K. Engan, S. O. Aase, and J. H. Husoy. Method of optimal directions for frame design. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 5, pages 2443–2446. IEEE, 1999.

[14] D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, 1995.

[15] R. Grosse, R. Raina, H. Kwong, and A. Y. Ng. Shift-Invariant Sparse Coding for Audio Classification. In *Uncertainty in Artificial Intelligence*, 2007.

[16] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang. Convolutional sparse coding for image super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1823–1831, 2015.

[17] F. Heide, W. Heidrich, and G. Wetzstein. Fast and flexible convolutional sparse coding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5135–5143. IEEE, 2015.

[18] F. Huang and A. Anandkumar. Convolutional dictionary learning through tensor factorization. In *Feature Extraction: Modern Questions and Challenges*, pages 116–129, 2015.

[19] B. Kong and C. C. Fowlkes. Fast convolutional sparse coding (fcsc). *Department of Computer Science, University of California, Irvine, Tech. Rep*, 2014.

[20] J. Mairal, F. Bach, J. Ponce, et al. Sparse modeling for image and vision processing. *Foundations and Trends® in Computer Graphics and Vision*, 8(2-3):85–283, 2014.

[21] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 689–696. ACM, 2009.

[22] S. Ono, T. Miyata, and I. Yamada. Cartoon-texture image decomposition using blockwise low-rank texture characterization. *IEEE Transactions on Image Processing*, 23(3):1128–1142, 2014.

[23] V. Papyan, J. Sulam, and M. Elad. Working locally thinking globally-part I: Theoretical guarantees for convolutional sparse coding. *arXiv preprint arXiv:1607.02005*, 2016.

[24] V. Papyan, J. Sulam, and M. Elad. Working locally thinking globally-part II: Stability and algorithms for convolutional sparse coding. *arXiv preprint arXiv:1607.02009*, 2016.

[25] R. Rubinstein, M. Zibulevsky, and M. Elad. Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit. *Cs Technion*, 40(8):1–15, 2008.

[26] J. Sulam, B. Ophir, M. Zibulevsky, and M. Elad. Trainlets: Dictionary learning in high dimensions. *IEEE Transactions on Signal Processing*, 64(12):3180–3193, 2016.

[27] B. Wohlberg. Efficient convolutional sparse coding. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7173–7177. IEEE, 2014.

[28] B. Wohlberg. Boundary handling for convolutional sparse representations. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 1833–1837. IEEE, 2016.

[29] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[30] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010.

[31] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus. Deconvolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2528–2535. IEEE, 2010.

[32] H. Zhang and V. M. Patel. Convolutional sparse coding-based image decomposition. In *BMVC*, 2016.