A Local Block Coordinate Descent Algorithm for the CSC Model

Ev Zisselman
Technion
Israel Institute of Technology
ev_zis@campus.technion.ac.il

Jeremias Sulam
Johns Hopkins University
jsulam1@jhu.edu

Michael Elad
Technion
Israel Institute of Technology
elad@cs.technion.ac.il

Abstract

The Convolutional Sparse Coding (CSC) model has recently gained considerable traction in the signal and image processing communities. By providing a global, yet tractable, model that operates on the whole image, the CSC was shown to overcome several limitations of the patch-based sparse model while achieving superior performance in various applications. Contemporary methods for pursuit and learning the CSC dictionary often rely on the Alternating Direction Method of Multipliers (ADMM) in the Fourier domain for the computational convenience of convolutions, while ignoring the local characterizations of the image. In this work we propose a new and simple approach that adopts a localized strategy, based on the Block Coordinate Descent algorithm. The proposed method, termed Local Block Coordinate Descent (LoBCoD), operates locally on image patches. Furthermore, we introduce a novel stochastic gradient descent version of LoBCoD for training the convolutional filters. This Stochastic-LoBCoD leverages the benefits of online learning, while being applicable even to a single training image. We demonstrate the advantages of the proposed algorithms for image inpainting and multi-focus image fusion, achieving state-of-the-art results.

1. INTRODUCTION

Sparse representation has been shown to be a very powerful model for many real-world signals, leading to impressive results in various restoration tasks such as denoising [10], deblurring [7], inpainting [11, 25], super-resolution [7, 40] and recognition [37], to name a few. The core assumption of this model is that signals can be expressed as 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. Concretely, for a signal $X \in \mathbb{R}^N$, the model assumption is that $X = \mathbf{D}\Gamma + V$, where V is a noise vector with bounded energy $\|V\|_2 < \epsilon$, which allows for a slight deviation from the model and/or may account for noise in the signal. The vector $\Gamma \in \mathbb{R}^M$ is the sparse representation of the signal, obtained by solving the pursuit problem [1, 9]:

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

where $\|\Gamma\|_0$ counts the number of non-zeros in Γ. The solution of problem (1) can be approximated using greedy algorithms such as Orthogonal Matching Pursuit (OMP) [4] or convex relaxation algorithms such as Basis Pursuit (BP) [5]. Over the years, various methods have been proposed to adaptively learn the dictionary $\mathbf{D}$ from real data. Prime examples are K-SVD [1], MOD [12], Double sparsity [29], Online dictionary learning [24], Trainlets [33], and more.

When dealing with high-dimensional signals, learning the dictionary suffers from the curse of dimensionality, rendering this task infeasible. To cope with this problem, many algorithms suggest training a local model on fully-overlapping patches taken from the signal X. This patch-based technique has gained much popularity due to its simplicity and high-performance [7, 10, 25, 40]. Yet, patch-based approaches are known to be sub-optimal as they ignore the relations between neighboring patches [28, 32].

An alternative approach to meet this challenge is posed by the Convolutional Sparse Coding (CSC) model. This model assumes that the signal can be represented as a superposition of a few local filters, convolved with sparse feature-maps. The CSC model utilizes a structured dictionary (union of narrowly banded convolutional matrices) that facilitates a global handling of the signal. This model has been the subject of an extensive research in the past several years, shown to lead to superior performance in applications such as super-resolution [17], inpainting [18], image separation [26], source separation [20] and audio processing [16].

Contemporary CSC based algorithms often rely on the ADMM [2] formulation for representation-extraction and filter-training. While the majority of works employ ADMM in the Fourier domain [3, 18, 35], a recent approach (SBDL) proposed by Papyan at el. [26], adopts a local point of view and trains the filters in terms of only local computations in the signal domain. The SBDL algorithm demonstrates state-of-the-art performance compared to the Fourier-based methods, albeit still relying on the ADMM algorithm. As such, this approach incurs additional memory and sensitivity to additional parameters, it only accommodates a batch-learning mode, and its convergence is questionable[1].

In this work we propose intuitive and easy-to-implement algorithms, based on the block coordinate descent approach, for solving the global pursuit and the CSC filter learning problems, all done with local computations in the original domain. The proposed pursuit algorithm operates without auxiliary variables nor extra parameters for tuning. We call this algorithm Local Block Coordinate Descent (LoBCoD). In addition, we introduce a stochastic gradient descent variant of LoBCoD for training the convolutional filters. This algorithm leverages the benefits of online learning, while being applicable even to a single training-image. The LoBCoD algorithm and its stochastic version show faster convergence and achieve a better solution to the CSC problem compared to the previous ADMM-based methods (global or local).

The rest of this paper is organized as follows: Section 2 reviews the CSC model and discusses previous methods. The proposed pursuit algorithm is presented in Section 3. In Section 4 we discuss dictionary update methods and introduce the stochastic LoBCoD algorithm. We compare these methods with previously published approaches in section 5. Section 6 extends our methods to image inpainting and multi-focus image fusion, followed by empirical results in Section 7. Section 8 concludes this work.

2. Convolutional sparse coding

The CSC model assumes that a signal[2] $X \in \mathbb{R}^N$ can be represented by the sum of m convolutions. These are built by feature maps $\{Z_i\}_{i=1}^m$, each of length of the original signal N, convolved with m small support filters $\{d_i\}_{i=1}^m$ of length $n \ll N$. In the dictionary learning problem, one minimizes the following cost function over both the filters and the feature maps[3]:

$$\min_{d_i, Z_i} \frac{1}{2}\|X - \sum_{i=1}^m d_i * Z_i\|_2^2 + \lambda \sum_{i=1}^m \|Z_i\|_1. \tag{2}$$

Given the filters, the above problem becomes the CSC pursuit task of finding the representations $\{Z_i\}_{i=1}^m$. Consider a global dictionary $\mathbf{D}$ to be the concatenation of m banded circulant matrices, where each matrix represents a convolution with one filter d_i. By permuting its columns, the global dictionary $\mathbf{D}$ consists of all shifted versions of a local dictionary $\mathbf{D}_L$ of size $n \times m$, containing the filters $\{d_i\}_{i=1}^m$ as its columns, and the global sparse vector Γ is simply the interlaced concatenation of all the feature maps $\{Z_i\}_{i=1}^m$. Such a structure is depicted in Figure 1. Using the above formulation, the convolutional dictionary learning problem (2) can be rewritten as

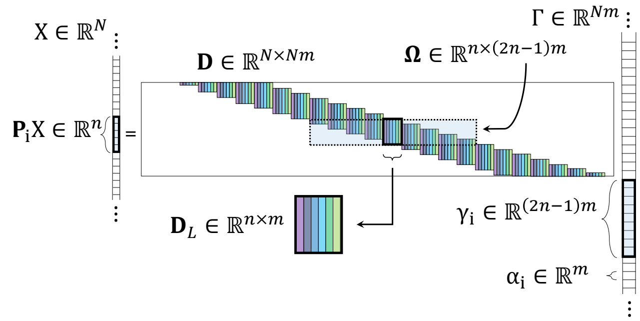


Figure 1: The CSC model and its local components.

$$\min_{\mathbf{D}, \Gamma} \frac{1}{2}\|X - \mathbf{D}\Gamma\|_2^2 + \lambda\|\Gamma\|_1. \tag{3}$$

Similar to our earlier comment, when $\mathbf{D}$ is known, we obtain the CSC pursuit problem, defined as

$$\min_{\Gamma} \frac{1}{2}\|X - \mathbf{D}\Gamma\|_2^2 + \lambda\|\Gamma\|_1. \tag{4}$$

Herein, we review some of the definitions from [27] as they will serve us later for the description of our algorithms.

The global sparse vector Γ can be broken into N non-overlapping m dimensional local vectors α_i, referred to as *needles*. This way, one can express the global vector X as $X = \sum_{i=1}^N \mathbf{P}_i^T \mathbf{D}_L \alpha_i$, where $\mathbf{P}_i^T \in \mathbb{R}^{N \times n}$ is the operator that positions $\mathbf{D}_L \alpha_i$ in the i-th location and pads the rest of the entries with zeros. On the other hand, a patch $\mathbf{P}_i X = \mathbf{P}_i \mathbf{D}\Gamma$ taken from the signal X equals to $\boldsymbol{\Omega}\gamma_i$ (see Figure 1), where $\boldsymbol{\Omega} \in \mathbb{R}^{n \times (2n-1)m}$ is a *stripe* dictionary containing $\mathbf{D}_L$ in its center, and γ_i is the *stripe* vector containing the local vector α_i in its center. In other words, a stripe γ_i is the sparse vector that codes all the content in the patch $\mathbf{P}_i X$, whereas a needle α_i only codes part of the information within it.

The theoretical work in [27] suggested an analysis of the CSC global model, augmented by a localized sparsity measure. Inspired by this analysis, herein we maintain such a local-global decomposition and propose a global algorithm that operates locally on image patches.

3. Proposed Method: CSC Pursuit

3.1. Local Block Coordinate Descent

In this section we focus on the pursuit of the representations, leaving the study of updating the dictionary for Section 4. The convolutional sparse coding problem presented in the previous section is solved by minimizing the global

[1] While SBDL's pursuit method is provably converging, this is no longer true when the dictionary is updated within the ADMM.

[2] The description given focuses on 1D signals for simplicity of the presentation, and all our treatment applies to higher dimensions just as well.

[3] We assume that the filters are normalized to a unit l_2-norm.

objective of Equation (4). In this paper, we adopt a local strategy and split the global sparse vector Γ into local vectors, needles α_i, and express the global CSC problem in term of such needles and the local dictionary $\mathbf{D}_L$ by

$$\min_{\mathbf{D}_L,\{\alpha_i\}} \frac{1}{2}\|X - \sum_{i=1}^{N} \mathbf{P}_i^T \mathbf{D}_L \alpha_i\|_2^2 + \lambda \sum_{i=1}^{N} \|\alpha_i\|_1. \quad (5)$$

However, rather than optimizing with respect to all the needles together, we suggest to treat the needles sequentially, and optimize with respect to each block α_i separately. As such, the update rule of each needle can be written as

$$\min_{\alpha_i} \frac{1}{2}\|(X - \sum_{\substack{j=1 \\ j\neq i}}^{N} \mathbf{P}_j^T \mathbf{D}_L \alpha_j) - \mathbf{P}_i^T \mathbf{D}_L \alpha_i\|_2^2 + \lambda\|\alpha_i\|_1. \quad (6)$$

By defining $R_i = (X - \sum_{\substack{j=1 \\ j\neq i}}^{N} \mathbf{P}_j^T \mathbf{D}_L \alpha_j)$ as the residual image without the contribution of the needle α_i, we can rewrite Equation (6) as

$$\min_{\alpha_i} \frac{1}{2}\|R_i - \mathbf{P}_i^T \mathbf{D}_L \alpha_i\|_2^2 + \lambda\|\alpha_i\|_1. \quad (7)$$

While the above minimization involves global variables, such as the residual R_i, one can show[4] that this can be decomposed into an equivalent and local problem:

$$\min_{\alpha_i} \frac{1}{2}\|\mathbf{P}_i R_i - \mathbf{D}_L \alpha_i\|_2^2 + \lambda\|\alpha_i\|_1. \quad (8)$$

This follows from the observation that the update rule of the needle α_i is affected only by pixels of the corresponding patch $\mathbf{P}_i R_i$ (the part that fully overlaps with $\mathbf{D}_L \alpha_i$).

The main idea of the block coordinate descent algorithm is that every step minimizes the overall penalty w.r.t. a certain block of coordinates, while the other ones are set to their most updated values. Following this idea, every local pursuit (8) proceeds by updating the global reconstructed signal $\widehat{X}$ and the global residual $R = X - \widehat{X}$, as a preprocessing stage that precedes the update of the next needle, based on the most updated values of the previous needles.

An important insight is that needles that have no footprint overlap in the image can be updated efficiently in parallel in the above algorithm without changing the algorithm's outcome. This enables employing efficient batch-implementations of the LARS algorithm [8]. Alternatively, the calculation can be distributed across multiple processors to gain a significant speedup in performance. To formalize these observations, we define the *layer* L_i as the set of needles that have no induced overlap in the image. We sweep through these layers and update their respective needles in parallel, followed by updating the global reconstructed signal $\widehat{X}$ and the global residual R. This way, the number of the layers imposes the number of the inner iterations, which will determine the complexity of our final algorithm. In that manner, the number of the inner iterations depends only on the patch size; for $\sqrt{n} \times \sqrt{n}$ patches, the number of layers is n. This pursuit algorithm is presented in Algorithm 1. Note that this algorithm can clearly be extended to iterate over multiple signals, but for the sake of brevity we assume that the data corresponds to an individual signal X.

[4] The proof is provided in the supplementary material.

Algorithm 1: The stochastic LoBCoD pursuit and dictionary learning algorithm

Input: signal X, initial $\mathbf{D}_L$, initial needles $\{\alpha_i^0\}_{i=1}^N$
Output: needles $\{\alpha_i\}_{i=1}^N$, the trained dictionary $\mathbf{D}_L$
Initialization: $R = X - \widehat{X}$, $\widehat{X} = \sum_{i=1}^N \mathbf{P}_i^T \mathbf{D}_L \alpha_i^0$, $k = 0$
while *not converged* **do**
  $k = k + 1$
  for *j = 1:n* **do**
    Computation of the residual:
$$R_j = R + \sum_{i\in L_j} \mathbf{P}_i^T \mathbf{D}_L \alpha_i^{k-1}$$
    Sparse pursuit: $\forall i \in L_j$ (in parallel)
$$\alpha_i^k = \arg\min_{\alpha_i} \frac{1}{2}\|\mathbf{P}_i R_j - \mathbf{D}_L \alpha_i\|_2^2 + \lambda\|\alpha_i\|_1$$
    Computation of the reconstructed signal:
$$\widehat{X} = \widehat{X} + \sum_{i\in L_j} \mathbf{P}_i^T \mathbf{D}_L (\alpha_i^k - \alpha_i^{k-1})$$
    Computation of the residual signal:
$$R = X - \widehat{X}$$
    Computation of the gradient w.r.t $\mathbf{D}_L$:
$$\nabla_{\mathbf{D}_L} = -\sum_{i\in L_j} \mathbf{P}_i R \cdot (\alpha_i^k)^T$$
    Dictionary update:
$$\mathbf{D}_L = \mathcal{P}_{\mathbb{1}}[\mathbf{D}_L - \eta \nabla_{\mathbf{D}_L}]$$
  end
end

3.2. Boundary Conditions and Initialization

In the formulation of the CSC model, as shown in Figure 1, we assumed that the dictionary is comprised of a set of banded circulant matrices, which impose a circulant boundary conditions on the signals. In practice, however, signals and images do not exhibit circulant boundary behavior. Therefore, our model incorporates a preemptive treatment of the boundaries. We adopt a similar approach to [26], in which the signal boundaries are padded with $n-1$ elements prior to decomposing it with the model. At the end of the process, we discard the added padding by cropping the $n-1$ boundary elements from the reconstructed signal and from the resulting feature maps (sparse representation).

Another beneficial preprocess step is needles initialization. A good initialization would equally spread the contribution of the needles towards signal reconstruction. With that goal, we set the initial value of each needle α_i to be the sparse representation of $\frac{1}{n}\mathbf{P}_i X$, i.e its relative portion of the corresponding patch. This can be done by solving the following local pursuit for every needle:

$$\alpha_i^0 = \arg\min_{\alpha_i} \frac{1}{2}\|\frac{1}{n}\mathbf{P}_i X - \mathbf{D}_L\alpha_i\|_2^2 + \lambda\|\alpha_i\|_1, \quad (9)$$

as a preprocess stage of our algorithm.

4. CSC Dictionary Learning

When addressing the question of learning the CSC filters, the common strategy is to alternate between sparse-coding and dictionary update steps for a fixed number of iterations. The dictionary update step aims to find the minimum of the quadratic term of Equation (5) subject to the constraint of normalized dictionary columns:

$$\begin{aligned} &\min_{\mathbf{D}_L} \frac{1}{2}\|X - \sum_{i=1}^{N} \mathbf{P}_i^T \mathbf{D}_L \alpha_i\|_2^2 \\ &\text{s.t } \{\|d_i\|_2 = 1\}_{i=1}^m. \end{aligned} \quad (10)$$

One can do so in a batch manner which requires access to the entire dataset at every iteration, or in an online (stochastic) manner that enables access to only small part of the dataset at every update step. This way it is also applicable for streaming data scenarios, when the probability distribution of the data changes over time.

4.1. Batch Update

Usually, for offline applications where the whole dataset is given and can be stored in memory, the batch approach is generally simpler, and thus we start with its description. The typical approach is to alternate between sparse coding (4) and dictionary update (10). For the latter, solving problem (10) requires finding the optimum $\mathbf{D}_L$ that satisfies the normalization constraint. One can find this optimal solution using projected steepest descent: perform steepest descent with a small step size and project the solution to the constraint set after each iteration, until convergence. To that end, the gradient of the objective in problem (10) w.r.t. $\mathbf{D}_L$ is[5]:

$$\nabla_{\mathbf{D}_L} = -\sum_{i=1}^{N} \mathbf{P}_i (X - \widehat{X}) \cdot \alpha_i^T. \quad (11)$$

The final update step for the local dictionary $\mathbf{D}_L$ is obtained by advancing in the direction of this gradient (11) and normalizing the columns of the resulting $\mathbf{D}_L$ in each iteration, until convergence.

[5]This derivation can be found in the supplementary material.

This batch dictionary update rule follows the line of thought of the MOD algorithm [12], and thus improves the solution in each step. However, it exhibits a very slow convergence rate since each dictionary update can be performed only after finishing the entire sparse coding (pursuit) stage, which is markedly inefficient, as the pursuit is the most time consuming part of the algorithm. This brings us to the Stochastic-LoBCoD alternative.

4.2. Local Stochastic Gradient Descent Approach

The traditional Stochastic Gradient Descent (SGD) approach restricts the computation of the gradient to a subset of the data and advances in the direction of this noisy gradient with every update step. Building upon this concept and the fact that Equation (11) reveals a separable gradient w.r.t the patches and their corresponding needles, we can update the dictionary in a stochastic manner. Rather than concluding the entire pursuit stage and then advancing in the direction of the global gradient, we can take a small step size η and update the dictionary after finding the sparse representation of only a small group of needles. According to Section 3, every iteration updates a group of needles, referred to as a layer L_i, which in turn could now serve to update the dictionary. This way, our algorithm convergences faster and adopts the stochastic behavior of the SGD while still operating on a single image.

The filters should be normalized after every dictionary update by projecting them onto the l_2 unit ball. Here, due to the choice of small step size, we simply normalize the atoms after every dictionary update:

$$\mathbf{D}_L = \mathcal{P}_{\mathbb{1}}[\mathbf{D}_L - \eta\nabla_{\mathbf{D}_L}].$$

Where $\mathcal{P}_{\mathbb{1}}[\cdot]$ denotes the operator that projects the dictionary atoms onto the unit ball. The final algorithm that incorporates the dictionary update is summarized in Algorithm 1.

Note that, although this dictionary update rule introduces an extra parameter (the step size η), determining its value is rather intuitive and can be performed automatically by setting it to $1-2\%$ of the norm of the gradient. Furthermore, this update rule may also leverage any stochastic optimization algorithm such as Momentum, Adagrad, Adadelta, Adam [30] etc., with their authors' recommended parameter values. This choice of parameter setting is sufficient, as will be demonstrated empirically in Section 7. In the rest of this work we will use this dictionary update rule, as it shows superior results.

5. Relation to Other Methods

In this section we describe the advantages of the proposed approach over Fourier and ADMM based methods.

Parallel computation: Our algorithm is trivial to parallelize efficiently across multiple processors by virtue of operating directly on the image patches. One can split the

Method	Time Complexity
[22] (Sparse)	$\underbrace{\boldsymbol{qImNlog(N)}}_{\text{T-pursuit}} + \underbrace{INmk}_{\text{SGD using sparse matrix}}$
[22] (Freq.)	$\underbrace{\boldsymbol{qImNlog(N)}}_{\text{T-pursuit}} + \underbrace{ImNlog(N) + INm}_{\text{SGD in the Fourier domain}}$
SBDL	$\underbrace{\boldsymbol{INnm} + IN(k^3 + mk^2)}_{\text{LARS}} + \underbrace{nm^2}_{\text{Gram}} + \underbrace{INk(n+m) + nm^2}_{\text{K-SVD}}$
Ours	$\underbrace{\boldsymbol{INnm} + IN(k^3 + mk^2)}_{\text{LARS}} + \underbrace{n^2m^2}_{\text{Gram}} + \underbrace{IN(n + nk + m)}_{\text{Stochastic-LoBCoD}}$

Table 1: Complexity analysis of our method compared to the online algorithms presented in [22] and the SBDL algorithm [26]. I: number of signals, N: signal dimension, m: number of filters, n: patch size, k: maximum number of non-zeros per needle, q: number of inner iterations for the pursuit algorithm. The dominant terms are in red color.

computations between N/n processors, corresponding to the number of the needles in every layer.

Online learning: The proposed algorithm, due to its local stochastic manner, can work in a streaming mode, where the probability distribution of the patches varies over time. Another aspect of this advantage is our ability to run in an online manner, even for a single input image. This stands in sharp contrast to other recent online methods [21, 34] that allow for online training but only in the case of streaming images. Other approaches took a step further and proposed partitioning the image into sub-images [22], but this is still far from our approach, which can stochastically estimate the gradient for each needle.

Parameter free: Contrary to ADMM-based approaches, our algorithm is unhindered by cumbersome manual parameter-tuning at the pursuit stage. Moreover, it benefits from an intuitively tuned parameter (the step size η) in the dictionary learning stage, as described as Section 4.

Memory efficient: Our algorithm has better storage complexity compare to the ADMM-based approaches [18, 26] since the update of the sparse vector is performed in-place and does not require any auxiliary variables.

Table 1 compares the complexity[6] of executing an epoch in our algorithm with that of the batch algorithm in [26] and the online algorithms in [22]. The conclusion is that our algorithm scales linearly with the global dimension N, while the competing online algorithms grow as O(Nlog(N)).

6. Image Processing via CSC

Having established the foundations for our algorithms, we now detail their extended variants for tackling the task of image inpainting and multi-focus image fusion. We also present adaptations of our algorithm for tackling the tasks of multi-exposure image fusion and salt-and-pepper text image denoising in the supplementary material.

[6]The full explanation can be found in the supplementary material.

6.1. Image Inpainting

The task of image inpainting pertains to filling-in missing pixels at known locations in the image. Assume we are given a corrupted image $Y = \mathbf{A}X$, where $\mathbf{A} \in \mathbb{R}^{N \times N}$ is a binary diagonal matrix that represents the degradation operator, so that $\mathbf{A}(i,i) = 0$ implies that the pixel x_i is masked. The goal of image inpainting is to reconstruct the original image X. Using the CSC formulation, this can be performed by first solving the following problem:

$$\min_{\Gamma} \frac{1}{2}\|Y - \mathbf{AD}\Gamma\|_2^2 + \lambda\|\Gamma\|_1, \tag{12}$$

and then taking the found representation Γ and multiplying by $\mathbf{D}$. By applying the steps described in Section 3, we split the above global optimization problem into a series of more manageable problems, each acting on a block of coordinates, i.e. a *needle*. This yields the following version of Equation (8):

$$\min_{\alpha_i} \frac{1}{2}\|\mathbf{P}_i R_i - \mathbf{A}_i \mathbf{D}_L \alpha_i\|_2^2 + \lambda\|\alpha_i\|_1. \tag{13}$$

Here, $\mathbf{A}_i = \mathbf{P}_i \mathbf{A} \mathbf{P}_i^T$ is the operator that masks the corresponding i-th patch, and $R_i = (Y - \mathbf{A}\sum_{\substack{j=1 \\ j\neq i}}^{N} \mathbf{P}_j^T \mathbf{D}_L \alpha_j)$ is the residual between the corrupted image and the degraded version of the reconstructed image, where the residual R_i does not account for the needle α_i. As mentioned in Section 3, we parallelize the computations of the needles that comprised each layer. The dictionary $\mathbf{D}_L$ can be pretrained on an external, uncorrupted dataset or trained on the corrupted image directly using the following gradient:

$$\nabla_{\mathbf{D}_L} = -\sum_{i \in L_j} \mathbf{P}_i \mathbf{A}^T (Y - \mathbf{A}\widehat{X}) \cdot \alpha_i^T, \tag{14}$$

where $\widehat{X} = \sum_{j=1}^{N} \mathbf{P}_j^T \mathbf{D}_L \alpha_j$ is the reconstructed image. The gradient derivation above is identical to that in Section 4, but with the exception of incorporating the mask $\mathbf{A}$.

6.2. Multi-focus image fusion

Image fusion techniques aim to integrate complimentary information from multiple images, captured with different focal settings, into an all-in-focus image of higher quality. Many patch-based sparse formulations were proposed to address this task, such as choose-max OMP [38], simultaneous OMP [39], and coupled sparse representation [14]. In this work, we adopt a similar scheme to [23], which utilizes the CSC for tackling the task of image-fusion, but with the distinction of solving a unified minimization problem.

Assume we are given a set of source images $\{Y^k\}_{k=1}^{L}$ to fuse, as well as a set of pretrained filters $\{d_i\}_{i=1}^{m}$. We start by decomposing each image Y^k into a base component Y_b^k and an edge component Y_e^k by imposing distinctive priors.

For the base component Y_b^k we penalize the l_2 norm of its gradient, while for the edge component we employ the CSC model such that $Y_e^k = \sum_{i=1}^m d_i * Z_i^k$. Practically, for each image Y^k we solve the unified minimization problem:

$$\min_{\{Z_i^k\}, Y_b^k} \frac{1}{2}\|Y^k - \sum_{i=1}^m d_i * Z_i^k - Y_b^k\|_2^2 + \lambda \sum_{i=1}^m \|Z_i^k\|_1 + \mu \frac{1}{2}\|\nabla Y_b^k\|_2^2. \tag{15}$$

This is done by alternating between minimizing w.r.t. the base component Y_b^k and the feature maps $\{Z_i^k\}_{i=1}^m$. The former boils down to a least square problem, and the latter is solved using our LoBCoD algorithm[7].

After decomposing all the images, we aim to fuse their components. For each image, we build an activity map $\tilde{A}^k$, as the sum of the absolute values of $\{Z_i^k\}_{i=1}^m$. For robustness, we convolve $\tilde{A}^k$ with a uniform kernel $U_s \in \mathbb{R}^{s \times s}$:

$$\tilde{\mathbf{A}}^k(u,v) = \sum\nolimits_{i=1}^m \|Z_i^k(u,v)\|_1, \quad \mathbf{A}^k = \tilde{\mathbf{A}}^k * U_s. \tag{16}$$

Based on the observation that a significant value in the activity map A^k indicates a sharp region in the corresponding image Y^k, we reconstruct the all-in-focus components by assembling the most prominent regions based on their values in the corresponding activity maps:

$$Z_i^f(u,v) = Z_i^{k^*}(u,v), \quad Y_b^f(u,v) = Y_b^{k^*}(u,v), \quad k^* = \arg\max_k (\mathbf{A}^k(u,v)). \tag{17}$$

where $\{Z_i^f\}_{i=1}^m$ and Y_b^f are the feature maps and the base component of the fused image Y^f. Finally, the fusion result is obtained by gathering its components:

$$Y^f = Y_b^f + \sum_{i=1}^m d_i * Z_i^f. \tag{18}$$

7. Experiments

The full LoBCoD implementation, documentation and demos that reproduce our results, are available online[8].

7.1. Run Time Comparison

To begin with, and to provide a comparison to other state of the art methods, we evaluate the performance of the proposed algorithm for solving Equation (5) against state of the art batch algorithms for CSC: the SBDL algorithm [26], the algorithm in [36] and the algorithm presented in [15], all using the same settings on the Fruit dataset [13]. For learning the dictionary, we used the ADAM and the Momentum algorithms[9] [30]. Figure 2 presents a comparison of the objective (2) as a function of time for each of the competing algorithms, showing that our method achieves the fastest convergence. Figure 4 shows the obtained dictionaries.

We also compared our method to the online stochastic gradient descent (SGD) based algorithms in [22], which operate in the spatial and in the Fourier domains. Here we randomly selected a training set of 40 images, and a test set of 5 different images from the MIRFLICKER-1M dataset [19]. Figure 3 presents the objective of the test set as a function of time, showing that our algorithm converges faster. Figure 5 shows the dictionaries obtained by the three methods, illustrating similar quality. Note that our algorithm is capable of operating online even if trained on one image, a possibility that is not supported by [22].

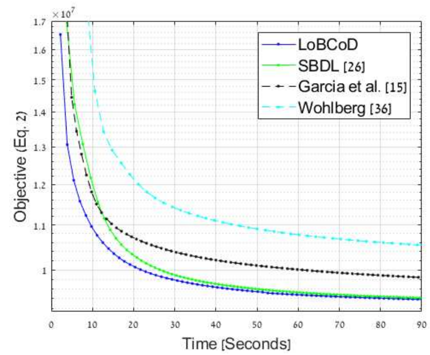


Figure 2: Run time comparison between our method and the batch methods in [26], [15] and [36].

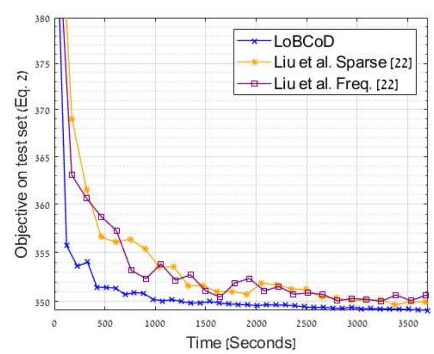


Figure 3: Run time comparison between our method and the online algorithms in [22].

7.2. Image Inpainting

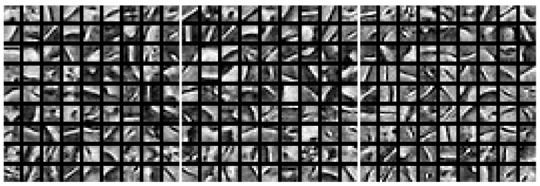
We apply our algorithm to the task of image inpainting, as described in Section 6.1, and compare our results to [26]. In Section 6.1 we described two viable methods for training the dictionary; utilizing an external dataset or training directly on the corrupted source image. For the former, we used the Fruit dataset [13] for both algorithms, as shown in

[7] Additional information can be found in the supplementary material.
[8] https://github.com/EvZissel/LoBCoD

[9] The parameter settings are described in the supplementary material.

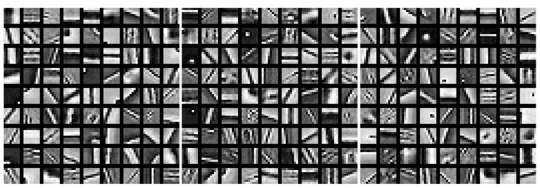
Table 2: Inpainting comparison [dB] between the proposed LoBCoD and the SBDL [26] algorithms.

	Barbara	Boat	House	Lena	Peppers	C.man	Couple	Finger	Hill	Man	Montage
SBDL external	30.41	31.76	36.17	35.92	**33.69**	28.76	32.16	30.91	33.12	33.04	28.93
Proposed external	**30.93**	**31.82**	**36.58**	**36.15**	33.54	**28.88**	**32.46**	**31.75**	**33.25**	**33.18**	**29.18**
SBDL internal	31.98	32.04	36.19	36.01	34.03	28.85	32.18	30.96	33.21	32.99	28.95
Proposed internal	**32.50**	**32.27**	**36.74**	**36.17**	**34.48**	**29.04**	**32.56**	**31.76**	**33.42**	**33.25**	**29.23**



(a) Ours (b) SBDL [26] (c) [15]

Figure 4: Comparison between the dictionaries obtained using the Stochastic-LoBCoD method vs. the methods in [26] and [15] on the Fruit dataset [13].



(a) Ours (b) [22] Sparse (c) [22] Freq.

Figure 5: Comparison between the dictionaries obtained using the Stochastic-LoBCoD method vs. the online methods in [22] on the MIRFLICKR-1M dataset [19].

Figure 4. The corrupted images were created by applying a randomly generated mask with 50% missing pixels on the original images. All the corrupted test images were mean-subtracted prior to applying both algorithms by subtracting the patch-average of the unmasked pixels. In addition, we tuned λ in Equation (12) for every corrupted test image, to account for their varying complexity. The top two rows of Table 2 present the results using an external dataset in terms of peak signal-to-noise ratio (PSNR) on a set of 11 test images, showing that our method leads to better results. Next, we train the dictionary of both algorithms on the corrupted image itself. The results are presented at the bottom two rows of Table 2, indicating that the Stochastic-LoBCoD algorithm achieves better results.

7.3. Multi-focus image fusion

We conclude by applying our LoBCoD algorithm to the task of multi-focus image fusion, as described in the previous section. We evaluate our proposed method using synthetic data, as well as data from a real dataset, and compare our results to [23]. The dictionaries of both methods were pretrained on the Fruit dataset [13].

For the synthetic experiment, we extracted a portion of

(a) Barbara in-focus (b) Barbara out-of-focus

(c) [23] 41.96dB (d) Proposed 42.27dB

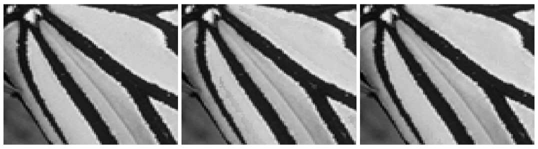
(e) Butterfly in-focus (f) Butterfly out-of-focus

(g) [23] 35.30dB (h) Proposed 36.01dB

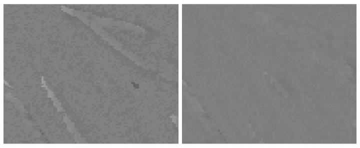
Figure 6: Fusion performance comparison on synthetic images. The PSNR values were computed between the reconstructed and the original images.

the standard image Barbara and created two input images of blurred foreground and blurred background. Image blurring was performed using a 9×9 Gaussian blur kernel with $\sigma = 2$. We repeated the same procedure on the image Butterfly[10], using a 16×16 Gaussian blur kernel with $\sigma = 4$.

[10]The image was taken form the dataset in [6].



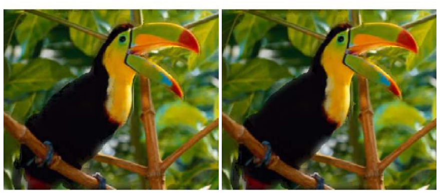
(a) Original (b) [23] (c) Proposed



(d) [23] (e) Proposed

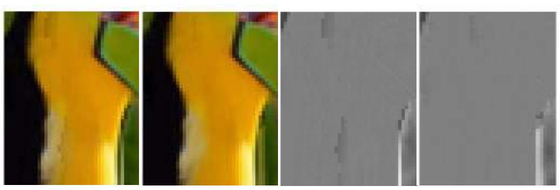
Figure 7: Zoom in on the fusion results of the image Butterfly. Figures (d) and (e) present the error images of (b) and (c), respectively. The error images were computed between the fusion results and the original image (a).

(a) Bird Foreground (b) Bird Background



(c) [23] 39.29dB (d) Proposed 39.81dB

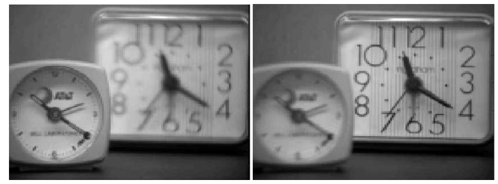
Figure 8: Fusion comparison between the proposed method and the method in [23] on the image Bird.



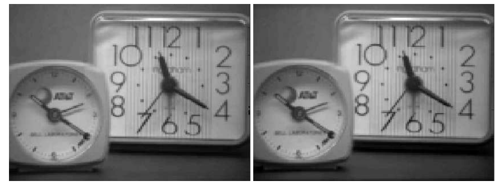
(a) [23] (b) Proposed (c) [23] (d) Proposed

Figure 9: Zoom in on the fusion results of the image Bird. Figure (c) and (d) present the L channel error compared to that of the original image.

Both sets of synthetic blurred images are presented in Figure 6, alongside their reconstructed images, and the corresponding PSNR values. The resulting images demonstrate that our approach leads to visually and quantitatively better results. Figure 7 presents a zoom-in view of our reconstructed image Butterfly, compared to the result of [23] and the original image; showing that for images with prominent blur our method achieves visually better results.



(a) Near in-focus (b) Far in-focus



(c) [23] (d) Proposed

Figure 10: Fusion examples of real images taken from [31].

We adapted our approach for fusion of colored images, and showed the result on the image Bird[11]. We generate a pair of blurred background and foreground color images by applying a 16×16 Gaussian blur kernel with $\sigma = 4$ on each RGB channel. We chose to blur the image in the RGB color space to emulate a blur of a camera. Afterwords, both blurred colored images were treated by transforming them to the Lab color space. The PSNR for the Bird image was computed between the L channels of the original and the reconstructed images. We present the results together with their PSNR values in Figure 8 and 9, showing that our approach leads to visually and quantitatively better results.

Lastly, for the real dataset experiment, we ran our proposed algorithm on the image-pair Clocks taken from [31]. Figure 10 presents the resulting fused images, showing comparable results on this image-pair.

8. Conclusions

In this work we have introduced the local block coordinate descent (LoBCoD) algorithm for performing pursuit for the global CSC model, while operating locally on image patches. We demonstrated its advantages over contending state-of-the-art methods in terms of memory requirements, efficient parallel computation, and its exemption from meticulous manual tuning of parameters. In addition, we proposed a stochastic gradient descent version (Stochastic-LoBCoD) of this algorithm for training the convolutional filters. We highlighted its unique qualities as an online algorithm that retains the ability to act on a single image. Finally, we illustrated the advantages of the proposed algorithm on a set of applications and compared it with competing state-of-the-art methods.

9. Acknowledgments

This work was supported by the Israel Science Foundation (ISF) under Grant 335/18.

[11]The image was taken form the dataset in [7]

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

[3] 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.

[4] 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.

[5] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[6] W. Dong, L. Zhang, G. Shi, and X. Li. Nonlocally centralized sparse representation for image restoration. *IEEE Transactions on Image Processing*, 22(4):1620–1630, 2013.

[7] 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.

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

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

[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, 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.

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

[13] R. Fergus, M. D. Zeiler, G. W. Taylor, and D. Krishnan. Deconvolutional networks. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition(CVPR)*, volume 00, pages 2528–2535, 2010.

[14] R. Gao and S. A. Vorobyov. Multi-focus image fusion via coupled sparse representation and dictionary learning. *arXiv preprint arXiv:1705.10574*, 2017.

[15] C. Garcia-Cardona and B. Wohlberg. Subproblem coupling in convolutional dictionary learning. In *Image Processing (ICIP), 2017 IEEE International Conference on*, pages 1697–1701. IEEE, 2017.

[16] R. Grosse, R. Raina, H. Kwong, and A. Y. Ng. Shift-invariant sparse coding for audio classification. In *The Twenty-Third Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 149–158, 2007.

[17] 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.

[18] F. Heide, W. Heidrich, and G. Wetzstein. Fast and flexible convolutional sparse coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5135–5143, 2015.

[19] M. J. Huiskes, B. Thomee, and M. S. Lew. New trends and ideas in visual concept detection: the mir flickr retrieval evaluation initiative. In *Proceedings of the international conference on Multimedia information retrieval*, pages 527–536. ACM, 2010.

[20] H.-W. Liao and L. Su. Monaural source separation using ramanujan subspace dictionaries. *IEEE Signal Processing Letters*, 25(8), 2018.

[21] J. Liu, C. Garcia-Cardona, B. Wohlberg, and W. Yin. Online convolutional dictionary learning. In *Image Processing (ICIP), 2017 IEEE International Conference on*, pages 1707–1711. IEEE, 2017.

[22] J. Liu, C. Garcia-Cardona, B. Wohlberg, and W. Yin. First-and second-order methods for online convolutional dictionary learning. *SIAM Journal on Imaging Sciences*, 11(2):1589–1628, 2018.

[23] Y. Liu, X. Chen, R. K. Ward, and Z. J. Wang. Image fusion with convolutional sparse representation. *IEEE signal processing letters*, 23(12):1882–1886, 2016.

[24] 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.

[25] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transactions on image processing*, 17(1):53–69, 2008.

[26] V. Papyan, Y. Romano, J. Sulam, and M. Elad. Convolutional dictionary learning via local processing. In *ICCV*, pages 5306–5314, 2017.

[27] V. Papyan, J. Sulam, and M. Elad. Working locally thinking globally: Theoretical guarantees for convolutional sparse coding. *IEEE Transactions on Signal Processing*, 65(21):5687–5701, 2017.

[28] Y. Romano and M. Elad. Patch-disagreement as away to improve k-svd denoising. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 1280–1284. IEEE, 2015.

[29] R. Rubinstein, M. Zibulevsky, and M. Elad. Double sparsity: Learning sparse dictionaries for sparse signal approximation. *IEEE Transactions on signal processing*, 58(3):1553–1564, 2010.

[30] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[31] S. Savić. Multifocus image fusion based on empirical mode decomposition. In *Twentieth International Electro technical and Computer Science Conference*, 2011.

[32] J. Sulam and M. Elad. Expected patch log likelihood with a sparse prior. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 99–111. Springer, 2015.

[33] 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.

[34] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni. Scalable online convolutional sparse coding. *IEEE Transactions on Image Processing*, 2018.

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

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

[37] 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.

[38] B. Yang and S. Li. Multifocus image fusion and restoration with sparse representation. *IEEE Transactions on Instrumentation and Measurement*, 59(4):884–892, 2010.

[39] B. Yang and S. Li. Pixel-level image fusion with simultaneous orthogonal matching pursuit. *Information fusion*, 13(1):10–19, 2012.

[40] 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.