Analysis K-SVD: A Dictionary-Learning Algorithm for the Analysis Sparse Model

Ron Rubinstein, *Member, IEEE*, Tomer Peleg, *Student Member, IEEE* and Michael Elad, *Fellow, IEEE*

***Abstract*—The synthesis-based sparse representation model for signals has drawn considerable interest in the past decade. Such a model assumes that the signal of interest can be decomposed as a linear combination of a *few* atoms from a given dictionary. In this paper we concentrate on an alternative, analysis-based model, where an analysis operator – hereafter referred to as the *analysis dictionary* – multiplies the signal, leading to a sparse outcome. Our goal is to learn the analysis dictionary from a set of examples. The approach taken is parallel and similar to the one adopted by the K-SVD algorithm that serves the corresponding problem in the synthesis model. We present the development of the algorithm steps: This includes tailored pursuit algorithms – the *Backward Greedy* and the *Optimized Backward Greedy* algorithms, and a penalty function that defines the objective for the dictionary update stage. We demonstrate the effectiveness of the proposed dictionary learning in several experiments, treating synthetic data and real images, and showing a successful and meaningful recovery of the analysis dictionary.**

***Index Terms*—Sparse Representations, Synthesis Model, Analysis Model, Backward Greedy (BG) Pursuit, Optimized Backward Greedy Pursuit (OBG), Dictionary Learning, K-SVD, Image Denosing.**

I. Introduction

A. Synthesis and Analysis Signal Models

Signal models are fundamental for handling various processing tasks, such as denoising, solving inverse problems, compression, interpolation, sampling, and more. Among the many ways we have to model signals, one approach that has found great popularity in the past decade is the synthesis-based sparse representation model. In this model, a signal $\mathbf{x} \in \mathbb{R}^d$ is modeled as being the outcome of the multiplication $\mathbf{x} = \mathbf{D}\boldsymbol{\alpha}$, where $\mathbf{D} \in \mathbb{R}^{d \times n}$ is a dictionary – its columns are signal prototypes (atoms) that are used to build the signal. We typically consider a redundant dictionary with $n > d$. The vector $\boldsymbol{\alpha} \in \mathbb{R}^n$ is the redundant signal's representation, and a fundamental feature in this model is the expectation that this vector is sparse, i.e. $\|\boldsymbol{\alpha}\|_0 = k \ll d$. This implies that the signals we work on are assumed to be composed as linear combinations of a *few* atoms from the dictionary [1], [2].

Vast work on the synthesis model during the past decade has been invested in an attempt to better understand it, and build practical tools for its use. The main activity concentrated on problems such as methods to estimate or approximate the sparse representation from the possibly corrupted signal, deriving theoretical success guarantees for such algorithms, and techniques to learn the dictionary $\mathbf{D}$ from signal examples. Referring specifically to the last point of dictionary learning, two popular techniques for this task are the MOD and K-SVD algorithms [3]–[5], whose deployment has led to state-of-the-art results in various image processing applications [2].

While the *synthesis* model has been extensively studied, there is a dual *analysis* viewpoint to sparse representations that has been left aside almost untouched [6]. The analysis model relies on a linear operator (a matrix) $\boldsymbol{\Omega} \in \mathbb{R}^{p \times d}$, which we will refer to as the *analysis dictionary*, and whose rows constitute *analysis atoms*. The key property of this model is our expectation that the analysis representation vector $\boldsymbol{\Omega}\mathbf{x} \in \mathbb{R}^p$ should be sparse with ℓ zeros. These zeros carve out the low-dimensional subspace that this signal belongs to. We shall assume that the dimension of this subspace, which is denoted by r is indeed small, namely $r \ll d$.

While this description may seem similar to the synthesis counterpart approach, it is in-fact very different when dealing with a redundant dictionary $p > d$. More on this model will be given below, contrasting it with the synthesis alternative. Until recently, relatively little was known about the analysis model, and little attention has been given to it in the literature, compared to the synthesis counterpart model. In the past few years there is a growing interest in the analysis model, as we gain more understanding and insight to its interesting viewpoint. See [7]–[16] for some work that has already commenced on this model.

In this paper we focus on the analysis model and more specifically, on the development of an algorithm that would learn the analysis dictionary $\boldsymbol{\Omega}$ from a set of signal examples $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ \ldots \ , \mathbf{x}_R]$. The objective is to find a suitable dictionary $\boldsymbol{\Omega}$ so that the analysis coefficients $\boldsymbol{\Omega}\mathbf{X}$ are sparse. We note that when dealing with a square (and invertible) matrix $\boldsymbol{\Omega}$, the analysis model is completely equivalent to the synthesis one with $\boldsymbol{\Omega}^{-1} = \mathbf{D}$ [6], and in such a case, the synthesis-dictionary-learning methods can be used to build $\boldsymbol{\Omega}$. In this work, though, we concentrate on the redundant case ($p > d$), where the two models depart, and where the analysis model becomes more interesting and powerful. This case of analysis dictionary training is a challenging problem, which has recently started to attract attention [13]–[16].

R. Rubinstein and M. Elad are with the Computer Science Department, Technion – Israel Institute of Technology, Haifa 32000, Israel (e-mail: {ronrubin,elad}@cs.technion.ac.il). T. Peleg is with the Department of Electrical Engineering, Technion – Israel Institute of Technology, Haifa 32000, Israel (e-mail: tomerfa@tx.technion.ac.il).

This work was supported by the European Commissions FP7-FET program, SMALL project (grant agreement no. 225913).

B. Related Work

One of the first attempts to train an analysis model was the pioneering work of Black and Roth, who adopted a very different point of view in their quest for $\mathbf{\Omega}$ [17], [18]. Roth and Black trained an image prior, termed Field-of-Experts (FoE), for the purpose of regularizing inverse problems in image processing. Their work originates from the area of probabilistic image modeling [19], [20], which aims at representing image patches through the notion of constraint violation and filter responses. When applying the analysis model on image patches, each row in $\mathbf{\Omega}$ can be viewed as a local filter operating on a patch, and once included in the co-support of the patch, this row serves as a constraint. Reviewing this area of research is not the intention of this paper. Instead we focus on the FoE approach and refer the readers to [18] for a comprehensive review.

The FoE prior derives its value from the sparsity of the analysis representations computed for overlapping image patches. The measure of sparsity used is a relaxed ℓ^0-norm. Our approach will be based on a different sparsity measure – the co-rank, which in turn will define a different objective for dictionary learning. Black and Roth use contrastive divergence for learning the analysis atoms, which approximates the maximum likelihood estimator and requires a high computational load as it relies heavily on Monte Carlo sampling. As such, this learning framework differs substantially from our work, which will take a block-coordinate relaxation approach, alternating between an analysis pursuit stage for each signal example and a simple update rule for each of the learned atoms.

The training set used by the FoE approach is a large database of image regions (each consisting of a set of overlapping patches) and the learning algorithm runs "offline" resulting in one generic prior that will be suitable for any natural image. In the context of image denoising, previous work on the synthesis model [21] has shown that adapting the dictionary to a given noisy image can lead to improved image denoising performance with respect to denoising with a "global" dictionary that was trained "offline". The approach we are about to suggest in this paper is capable of learning an adaptive analysis dictionary from a given noisy set of examples.

More recently, three interesting attempts to learn a redundant analysis dictionary have been proposed [13]–[16]. The first work, reported in [13], suggests to incrementally learn $\mathbf{\Omega}$ one row at a time, exploiting the fact that a considerable set of examples is expected to be orthogonal to such a row. Assuming knowledge of this set of signals, the eigenvector that corresponds to the smallest eigenvalue of these examples' autocorrelation matrix is the desired row. For each row, the proposed algorithm thus alternates between the computation of this row from the current subset of chosen examples, and an update of this subset to reject outlier signals. This algorithm relies heavily on a randomized initialization strategy, both for enabling the detection of a variety of rows, and for surpassing deadlock situations in the iterative process. As the dimension of the signal d grows (and with it p, the number of rows in $\mathbf{\Omega}$), this approach loses its efficiency rapidly, requiring (too) many rounds of attempts before a new row is detected. Moreover, this method might suffer from a poor detection quality, since a row that was admitted to the accumulated set cannot be replaced. These two limitations will be demonstrated in Section V-A when comparing our approach with [13] in the task of recovering the dictionary in a synthetic setup.

The work reported in [14], [15] takes a different route towards the task of learning $\mathbf{\Omega}$, posing it as a constrained optimization problem. The goal of sparsifying the representations $\mathbf{\Omega X}$ is formulated by an ℓ^1-norm penalty function on these representations. In order to avoid the trivial solution $\mathbf{\Omega} = 0$ and solutions like an orthonormal basis in the first d rows followed by zero rows, this work proposes to constrain the dictionary to be a uniform normalized tight frame. However, this choice limits the possible $\mathbf{\Omega}$ to be learned, and puts a rather arbitrary constraint for regularizing the learning problem. In our work we aim at handling the most general setup of redundant analysis dictionaries and therefore we would like to be less restrictive as possible with respect to the learned atoms.

The work in [16] proposes to learn $\mathbf{\Omega}$ such that it optimizes the denoising performance on a given set of example pairs (clean and noisy versions of example signals). The learning is achieved by formulating the learning task as a bilevel-programming optimization problem, which in turn is handled using gradient descent. The main focus of [16] is on learning an analysis operator that takes the form of a convolution, which is equivalent to learning one filter (convolution kernel). This is very different from our main goal – learning a set of p analysis atoms, which can be viewed as local filters operating on image patches.

In the process of preparing this paper for publication, another relevant work on analysis dictionary learning was brought to our attention [22]. The approach suggested in this paper shares some basic ideas with the FoE approach, such as learning a "global" dictionary for natural image patches and inserting it to a regularization term of an image recovery formulation. However, the authors of [22] take a very different route towards the dictionary learning problem, posing it in terms of an optimization over manifolds. This allows them to update the analysis dictionary as a whole, in contrast to the separate atom updates practiced in our approach, thus explicitly enforcing basic dictionary properties, such as having distinct rows and full row rank into the learning procedure. Using these optimization tools, they learn a redundant analysis dictionary that obtains competitive results with respect to the synthesis counterpart model for various image processing applications.

C. This Work

In this paper we adopt a different approach to the analysis training problem, based on a *co-rank* measure which determines the dimension of the analysis subspace. This co-rank measure allows us to develop a novel training algorithm, whose uniqueness is in the relations it exhibits with the synthesis formulation. Specifically, the proposed dictionary-training method is parallel to the synthesis-model K-SVD in its rationale and computational steps. Similar to the work in

[14], [15], we consider the learning process as a solution of a constrained optimization task. However, as we shall show next, the constraint we employ uses our knowledge on the signals and their relation to $\mathbf{\Omega}$ in a more direct way. The atom update rule in our proposed approach will be similar to the one suggested in [13]. However, it will be better justified by deriving it directly from the constrained optimization problem. Moreover, the set of signals orthogonal to each row in $\mathbf{\Omega}$ will be determined in a more effective fashion, resulting in a more efficient algorithm.

The analysis model gives rise to a series of research questions, which are far from being solved: (i) What are the desired properties of an analysis dictionary? (ii) How can a signal and its sparse analysis representation be recovered, given the dictionary and a noisy version of the signal? (iii) Can the analysis dictionary be learned from a given data-set of examples? In this work we aim at providing some answers to these questions. Our main contribution is an efficient algorithm for obtaining the analysis dictionary from a given data-set in a K-SVD-like manner. We demonstrate the potential of this approach in a series of experiments on synthetic and real data (images), showing the ability of the algorithm to recover a meaningful result in all cases. We note that our main goal in this work is to highlight the potential and capability of the co-rank analysis approach, and we do not focus here on specific applications.

This paper is organized as follows: In Section II we present the core concept of the co-rank analysis model, and characterize the signals that belong to it. In Section III we consider the *analysis pursuit* problem of denoising a signal using the analysis model, which serves as an important building-block in the Analysis K-SVD algorithm described in detail in Section IV. Finally, Section V provides several experiments that demonstrate the performance of the Analysis K-SVD algorithm.

II. A Closer Look at the Analysis Model

In this section we briefly review the co-rank analysis model, and characterize the signals that belong to it, and which our learning algorithm is to operate on. The content of this section relies in part on explanations found in [8], [9], [13].

The analysis model for the signal $\mathbf{x} \in \mathbb{R}^d$ uses the possibly redundant analysis dictionary $\mathbf{\Omega} \in \mathbb{R}^{p \times d}$ (redundancy here implies $p \geq d$), and assumes that the analysis representation vector $\mathbf{\Omega x}$ should be sparse. In this work we consider specifically ℓ^0 sparsity, which implies that $\mathbf{\Omega x}$ contains many zeros. The *co-sparsity* ℓ of the analysis model is defined as the number of zeros in the vector $\mathbf{\Omega x}$,

$$\|\mathbf{\Omega x}\|_0 = p - \ell. \tag{1}$$

In the synthesis model the representation $\boldsymbol{\alpha}$ is obtained by a complex and non-linear pursuit process that seeks (or approximates) the sparsest solution to the linear system of equations $\mathbf{D}\boldsymbol{\alpha} = \mathbf{x}$. This representation can be arbitrarily sparse, $\|\boldsymbol{\alpha}\|_0 = k \ll d$. The signal $\mathbf{x}$ is characterized by the k non-zero indices in the representation vector $\boldsymbol{\alpha}$, and their associated atoms define the subspace this signal belongs to. The dimension of this subspace equals k and as we mentioned before, it is small with respect to the signal dimension d.

In contrast, in the analysis model the computation of the representation is trivial, obtained by the multiplication $\mathbf{\Omega x}$. In this model we put an emphasis on the zeros of $\mathbf{\Omega x}$, and define the *co-support* Λ of $\mathbf{x}$ as the set of $\ell = |\Lambda|$ rows that are orthogonal to it. In other words, $\mathbf{\Omega}_\Lambda \mathbf{x} = 0$, where $\mathbf{\Omega}_\Lambda$ is a sub-matrix of $\mathbf{\Omega}$ that contains only the rows indexed in Λ. For a given analysis dictionary $\mathbf{\Omega}$, we define the *co-rank* of a signal $\mathbf{x}$ with co-support Λ as the rank of $\mathbf{\Omega}_\Lambda$. The signal $\mathbf{x}$ is thus characterized by its co-support, which determines the subspace it is orthogonal to, and consequently the complement space to which it belongs. Just like in the synthesis model, we assume that the dimension of the subspace the signal belongs to, denoted by r, is small, namely $r \ll d$. The co-rank of such an analysis signal is $d - r$.

How sparse can the analysis representation vector be? Let us first assume that the rows in $\mathbf{\Omega}$ are in general-position, implying that every subset of d or less rows are necessarily linearly independent. This is equivalent to the claim that the spark of $\mathbf{\Omega}^T$ is full [2]. Naturally, for this case, $\ell < d$, since otherwise there would be d independent rows orthogonal to $\mathbf{x}$, implying $\mathbf{x} = 0$. Thus, in this case the analysis model leads necessarily to a mild sparsity, $\|\mathbf{\Omega x}\|_0 > p - d$, and for a highly redundant analysis operator, the cardinality of the analysis representation vector $\mathbf{\Omega x}$ is expected to be quite high. In this case, the dimension of the subspace the signal belongs to is $r = d - \ell$.

A more interesting case is when $\mathbf{\Omega}^T$ has *non-full spark*, implying that linear dependencies exist between the dictionary atoms. The immediate implication is that ℓ *could go beyond* d, and yet the signal would not necessarily be nulled. An example of such a dictionary is the set of cyclic horizontal and vertical one-sided derivatives, applied on a 2D signal of size $\sqrt{d} \times \sqrt{d}$. The corresponding analysis dictionary, denoted $\mathbf{\Omega}_{DIF}$, is of size $2d \times d$, thus twice redundant. Figure 1 shows this dictionary for $d = 25$. In [9] this dictionary is discussed in detail, showing that its rows exhibit strong linear dependencies.

Generating a random analysis signal amounts to the following process: Choose a set of row indices $\Lambda \subseteq \{1, \ldots, p\}$ — this will be the signal's co-support. Starting with a random vector $\mathbf{u}$, project it onto the subspace orthogonal to $\mathbf{\Omega}_\Lambda$:

$$\mathbf{x} = (\mathbf{I} - \mathbf{\Omega}_\Lambda^\dagger \mathbf{\Omega}_\Lambda)\mathbf{u}, \tag{2}$$

and $\mathbf{x}$ is an analysis signal that satisfies our sparsity assumption. For a general-positioned $\mathbf{\Omega}$ we choose ℓ rows from $\mathbf{\Omega}$ at random. Otherwise we choose $d - r$ linearly independent rows from $\mathbf{\Omega}$. This choice is still done in a random fashion, but is naturally more restricted. In the experiments that follow we shall use such randomly generated signals, when dealing with synthetic experiments.

As mentioned above, when the rows in $\mathbf{\Omega}$ are not in general-position, the co-sparsity ℓ can be greater than d. In this case, once a signal $\mathbf{x}$ has been generated using the process (2), computation of its analysis representation $\mathbf{\Omega x}$, could reveal additional rows that are orthogonal to the signal, due to linear dependence on the chosen subset Λ. To demonstrate this

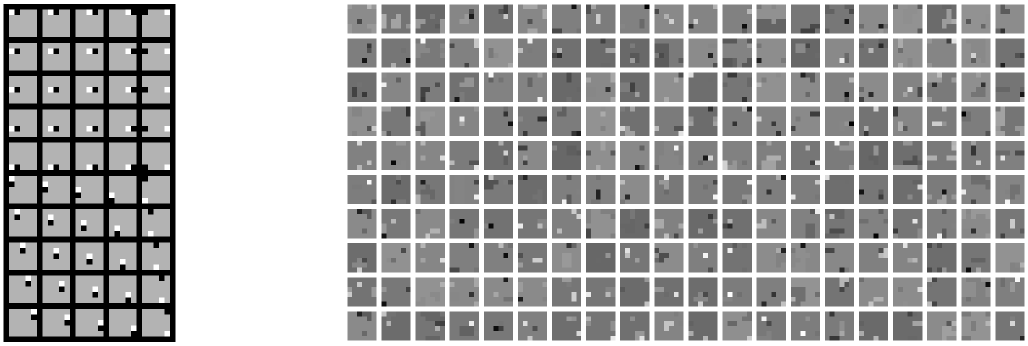

Figure 1. Left: The dictionary $\mathbf{\Omega}_{DIF}$ of size 50×25, corresponding to horizontal and vertical cyclic one-sided derivatives of image patches of size 5×5 pixels. Right: Examples of sparse analysis signals (5×5 patches) residing in 4-dimensional subspaces related to the dictionary $\mathbf{\Omega}_{DIF}$, i.e., orthogonal to 21 linearly-independent atoms each.

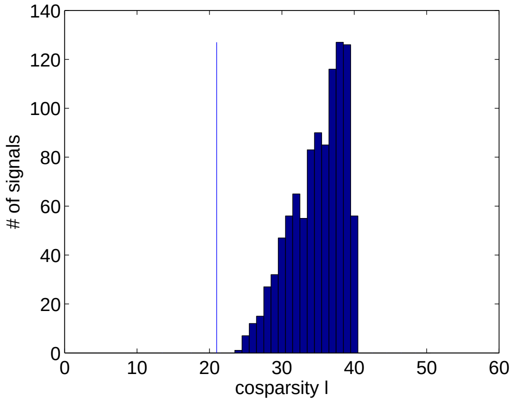

Figure 2. A histogram of the effective co-sparsities of the 1000 analysis signals generated from $\mathbf{\Omega}_{DIF}$ of size 50×25. The reference value of $\ell = 21$ is indicated by the thin vertical line. As can be seen, the effective co-sparsities are all strictly higher.

behavior, we generated 1000 unit-norm analysis signals residing in 4-dimensional subspaces related to $\mathbf{\Omega}_{DIF} \in \mathbb{R}^{50 \times 25}$. Figure 1 presents a set of such randomly created signals. For these signals, Figure 2 presents a histogram of the effective co-sparsities. As can be seen, though the signals are each orthogonal to subspaces of rank 21, their actual co-sparsities are much higher, varying in the range 23 to 40. Thus, we see that by allowing linear dependencies between the rows in $\mathbf{\Omega}$, co-sparsities much higher than the signal dimension can be achieved.

An equivalent way to interpret the co-rank analysis and ℓ^0 synthesis models is as Unions of Subspaces (UoS) signal models [23]. As we have seen, in both cases the sparse signals reside within some UoS defined by the dictionary atoms. In the synthesis case, these subspaces are formed by the *spans* of all sets of atoms with rank $\leq k$ for some choice of k. In contrast, in the analysis case these subspaces are the *orthogonal complements* of the sets of atoms with rank $= d-r$. We note that when the dictionaries are in general position, the number of such subsets is $\binom{n}{k}$ and $\binom{p}{d-r}$, respectively. In general, the UoS's associated with the two models will be very different. For example, if $p = n = 2d$, $k = r \ll d$ and the rows in $\mathbf{\Omega}$ are in general-position, the subspaces united by the two models are of the same dimension (k or r), but their number is entirely different, with many more subspaces included in the analysis model.

III. Analysis Sparse-Coding

A. *Defining the Pursuit Problem*

Before we study the problem of learning the analysis dictionary $\mathbf{\Omega}$, we have to consider a simpler task called *analysis sparse-coding* or *analysis pursuit*. As we shall see in the next section, this is an important building-block in the overall dictionary-learning procedure.

A convenient property of the analysis approach is that given a signal $\mathbf{x}$, we can readily compute its analysis coefficients $\mathbf{\Omega x}$, and thus determine the cardinality of its analysis representation. However, if we assume an additive contamination of the signal, $\mathbf{y} = \mathbf{x} + \mathbf{v}$, then computation of the analysis representation $\mathbf{\Omega x}$ is no longer simple. We shall assume that $\mathbf{v}$ is a zero-mean white-Gaussian additive noise vector. Recovering the correct signal $\mathbf{x}$ from its noisy version (and thereby computing the analysis representation), $\mathbf{y}$, requires solving a problem of the form

$$\left\{\hat{\mathbf{x}},\ \hat{\Lambda}\right\} = \underset{\mathbf{x},\,\Lambda}{\text{Argmin}}\ \ \|\mathbf{x}-\mathbf{y}\|_2 \quad \text{Subject To} \qquad (3)$$
$$\mathbf{\Omega}_{\Lambda}\mathbf{x} = 0$$
$$\text{Rank}\left(\mathbf{\Omega}_{\Lambda}\right) = d - r$$

or

$$\left\{\hat{\mathbf{x}},\ \hat{\Lambda}\right\} = \underset{\mathbf{x},\,\Lambda}{\text{Argmin}}\ \ \text{Rank}(\mathbf{\Omega}_{\Lambda}) \quad \text{Subject To} \qquad (4)$$
$$\mathbf{\Omega}_{\Lambda}\mathbf{x} = 0$$
$$\|\mathbf{x}-\mathbf{y}\|_2 \leq \epsilon.$$

In Equation (3) we require a co-rank of $d-r$ for the obtained solution, while in Equation (4), we constrain the solution to be ϵ-close to the given noisy signal, where this error tolerance is derived from the noise power. The above problems can be considered as denoising schemes, as $\hat{\mathbf{x}}$ is an attempt to estimate the true noiseless signal $\mathbf{x}$. The two problems (3) and (4) are equivalent, of course, given the correct correspondence between r and ϵ, and the choice between them depends on the available information regarding the process that generated $\mathbf{y}$. We refer to these problems as the analysis sparse-coding or analysis-pursuit problems.

In principle, denoising is possible with the analysis model because, once the co-support has been detected, projection on the complement subspace attenuates the additive noise in the co-support subspace, thus cleaning the signal. Indeed, the higher the dimension of the true co-support, the better the denoising is expected to perform.

In an *oracle* setup, the true co-support Λ is known, and thus can be used for obtaining a signal recovery,

$$\hat{\mathbf{x}} = \left(\mathbf{I} - \mathbf{\Omega}_\Lambda^\dagger \mathbf{\Omega}_\Lambda\right)\mathbf{y}. \tag{5}$$

The mean denoising error in the oracle setup is given by

$$E\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 = \text{tr}\left(\mathbf{I} - \mathbf{\Omega}_\Lambda^\dagger \mathbf{\Omega}_\Lambda\right)\sigma^2 = r\sigma^2, \tag{6}$$

where $\text{tr}(\cdot)$ denotes the trace of a matrix. In the first equality we used the fact that the noise is white and Gaussian and that $\mathbf{\Omega}_\Lambda \mathbf{x} = 0$. The last equality holds since $\mathbf{I} - \mathbf{\Omega}_\Lambda^\dagger \mathbf{\Omega}_\Lambda$ is a projection matrix onto a r-dimensional space, so that it has two eigenvalues – a zero eigenvalue of multiplicity $d - r$ and an eigenvalue 1 with multiplicity r. This should remind the reader of the oracle error in the synthesis case, as described in [2].

Similar to the synthesis sparse approximation problem, the problems posed in Equations (3) and (4) are combinatorial in nature and can thus only be approximated in general. One approach to approximating the solution is to relax the ℓ^0 norm and replace it with an ℓ^1 penalty function, producing

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{Argmin}} \quad \|\mathbf{x} - \mathbf{y}\|_2 \quad \text{Subject To} \quad \|\mathbf{\Omega}\mathbf{x}\|_1 \leq T. \tag{7}$$

This approach is parallel to the basis-pursuit approach for synthesis approximation [24], and the resulting problem may be solved via an iterated re-weighted least squares (IRLS) method [25], or using standard quadratic or conic optimization methods.

B. The Backward-Greedy Algorithm

A second approach parallels the synthesis greedy pursuit algorithms [26], [27], and is the one we shall use in this work. It suggests selecting rows from $\mathbf{\Omega}$ one-by-one in a greedy fashion. The solution can be built by either detecting the rows that correspond to the non-zeros in $\mathbf{\Omega}\mathbf{x}$, or by detecting the zeros. The first approach is the one taken by the Greedy-Analysis-Pursuit (GAP) algorithm, described in [9]. We shall take the alternative (and simpler) approach of finding the co-support Λ one element at a time. We refer to this algorithm as the Backward-Greedy (BG) Algorithm, as it is gathering the zeros in the representation. A detailed description of this algorithm, is given below in Algorithm 1. Note that this algorithm takes as input the co-rank $d-r$ of the desired co-support rather than the exact number of zeros, and thus the actual number of vanishing coefficients in the output representation may be larger than $d - r$.

The process begins by setting $\hat{\mathbf{x}} = \mathbf{y}$ and initializing the co-support to be an empty set of rows. In each iteration, the inner-products $\mathbf{\Omega}\hat{\mathbf{x}}$ are computed, and the row with the *smallest* non-zero inner-product is selected and added to the set. The solution $\hat{\mathbf{x}}$ is then updated by projecting $\mathbf{y}$ on the orthogonal space to the selected rows. Finally, the co-support is refined by recalculating the representation vector $\mathbf{\Omega}\hat{\mathbf{x}}$ and finding the additional coefficients that fall below some small threshold ϵ_0. This can reveal additional rows that are orthogonal to the current estimate of the signal, namely the rows that are spanned by the existing set of rows $\mathbf{\Omega}_{\Lambda_i}$. The process described above repeats until the target subspace dimension is achieved.

Algorithm 1 BACKWARD-GREEDY

```
Input: Analysis dictionary $\mathbf{\Omega} \in \mathbb{R}^{p \times d}$, signal $\mathbf{y} \in \mathbb{R}^d$, and target co-rank $d - r$
Output: Signal $\hat{\mathbf{x}} \in \mathbb{R}^d$ with co-rank $d-r$ and minimizing $\|\mathbf{y} - \hat{\mathbf{x}}\|_2$
Initialization: Set $i = 0$, $\Lambda_0 := \emptyset$, $\hat{\mathbf{x}}_0 := \mathbf{y}$
for $i = 1 \ \ldots \ d - r$ do
    Sweep: $\hat{k}_i := \underset{k \notin \Lambda_{i-1}}{\text{Argmin}} \ |\mathbf{w}_k^T \hat{\mathbf{x}}_{i-1}|$
    Update Co-Support: $\Lambda_i := \Lambda_{i-1} \cup \{\hat{k}_i\}$
    Project: $\hat{\mathbf{x}}_i := \left(\mathbf{I} - \mathbf{\Omega}_{\Lambda_i}^\dagger \mathbf{\Omega}_{\Lambda_i}\right)\mathbf{y}$
    Refine Co-Support:
        $\Lambda_i := \{k \,|\, 1 \leq k \leq p, \ |\mathbf{w}_k^T \hat{\mathbf{x}}_i| < \epsilon_0\}$
end for
return $\hat{\mathbf{x}} = \hat{\mathbf{x}}_{d-r}$
```

In practice, the above algorithm can be implemented efficiently by accumulating an orthogonalized set of the co-support rows. This means that once $\hat{k}_i$ has been found and the row $\mathbf{w}_{\hat{k}_i}^T$ is about to join the co-support, it is first orthogonalized with respect to the already accumulated rows using a modified Gram-Schmidt process. Denoting by $\{\mathbf{q}_j\}_{j=1}^{i-1}$ the orthogonal set accumulated so far (as column vectors), the orthogonalization of $\mathbf{w}_{\hat{k}_i}^T$ is obtained by

$$\mathbf{q}_i = \mathbf{w}_{\hat{k}_i} - \sum_{j=1}^{i-1} (\mathbf{q}_j^T \mathbf{w}_{\hat{k}_i})\mathbf{q}_j. \tag{8}$$

This should be followed by a normalization of this vector, $\mathbf{q}_i = \mathbf{q}_i / \|\mathbf{q}_i\|_2$.

The above-described orthogonalization process is done for one purpose: avoiding the matrix inversion in the update of $\hat{\mathbf{x}}_i$. The "Projection" step in Algorithm 1 translates comfortably to

$$\hat{\mathbf{x}}_i = \left(\mathbf{I} - \mathbf{\Omega}_{\Lambda_i}^\dagger \mathbf{\Omega}_{\Lambda_i}\right)\mathbf{y} = \left[\mathbf{I} - \sum_{j=1}^{i} \mathbf{q}_j \mathbf{q}_j^T\right]\mathbf{y}. \tag{9}$$

The correctness of this update formula can be easily verified by multiplying $\hat{\mathbf{x}}_i$ by $\mathbf{q}_j$ $(j = 1, 2, \ldots, i)$, leading to $\mathbf{q}_j^T \hat{\mathbf{x}}_i = 0$, as required. This can be further simplified by observing that

$$\begin{aligned}\hat{\mathbf{x}}_i &= \left[\mathbf{I} - \sum_{j=1}^{i} \mathbf{q}_j \mathbf{q}_j^T\right] \mathbf{y} = \left[\mathbf{I} - \sum_{j=1}^{i-1} \mathbf{q}_j \mathbf{q}_j^T\right] \mathbf{y} - \mathbf{q}_i \mathbf{q}_i^T \mathbf{y} \\ &= \hat{\mathbf{x}}_{i-1} - \mathbf{q}_i \mathbf{q}_i^T \mathbf{y}. \end{aligned} \tag{10}$$

Finally, we can suggest a slight modification,

$$\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_{i-1} - \mathbf{q}_i \mathbf{q}_i^T \mathbf{y} = \left[\mathbf{I} - \mathbf{q}_i \mathbf{q}_i^T\right] \hat{\mathbf{x}}_{i-1}. \tag{11}$$

This last change is justified by the fact that a projection of $\mathbf{y}$ onto $\mathbf{q}_i$ is the same as the projection of $\hat{\mathbf{x}}_{i-1}$. Though mathematically the two expressions are the same, we have observed that the latter option exhibits better numerical stability.

C. The Optimized Backward-Greedy Algorithm

We can propose an improved pursuit algorithm in the spirit of the BG, by considering the following approach: At the i^{th} step, rather than choosing $\hat{k}_i := \underset{k \notin \Lambda_{i-1}}{\text{Argmin}} \; |\, \mathbf{w}_k^T \, \hat{\mathbf{x}}_{i-1} \,|$, we can test all possible values of $\hat{k}_i \notin \Lambda_{i-1}$, and for each, compute the complete update ("Update Co-Support" and "Project" steps) in Algorithm 1. Then we choose $\hat{k}_i$ that leads to the smallest decrease in the signal's energy, i.e. choosing $\hat{k}_i$ that minimizes $|\, \mathbf{q}_i^T \hat{\mathbf{x}}_{i-1} \,|$. This should remind the reader of the OOMP algorithm [28], also known as the Least-Squares OMP algorithm [2]. We shall refer hereafter to this algorithm as the Optimized-BG (OBG), detailed in Algorithm 2.

Algorithm 2 OPTIMIZED-BACKWARD-GREEDY

```
1:  Input: Analysis dictionary $\mathbf{\Omega} \in \mathbb{R}^{p \times d}$, signal $\mathbf{y} \in \mathbb{R}^d$, and
    target co-rank $d - r$
2:  Output: Signal $\hat{\mathbf{x}} \in \mathbb{R}^d$ with co-rank $d-r$ and minimizing
    $\|\mathbf{y} - \hat{\mathbf{x}}\|_2$
3:  Initialization: Set $i = 0$, $\Lambda_0 := \emptyset$, $\hat{\mathbf{x}}_0 := \mathbf{y}$
4:  for $i = 1 \ \ldots \ d - r$ do
5:      for $k \notin \Lambda_{i-1}$ do
6:          Provisional Update Co-Support
                $\Lambda_i^{temp} := \Lambda_{i-1} \cup \{k\}$
7:          Provisional Project:
                $\hat{\mathbf{x}}_i^{temp} := \left(\mathbf{I} - \mathbf{\Omega}^{\dagger}_{\Lambda_i^{temp}} \mathbf{\Omega}_{\Lambda_i^{temp}}\right) \mathbf{y}$
8:          Provisional Error: $e_k = \|\hat{\mathbf{x}}_i^{temp} - \hat{\mathbf{x}}_{i-1}\|_2$
9:      end for
10:     Sweep: $\hat{k}_i := \underset{k \notin \Lambda_{i-1}}{\text{Argmin}} \; e_k$
11:     Update Co-Support: $\Lambda_i := \Lambda_{i-1} \cup \{\, \hat{k}_i \,\}$
12:     Project: $\hat{\mathbf{x}}_i := \left(\mathbf{I} - \mathbf{\Omega}^{\dagger}_{\Lambda_i} \mathbf{\Omega}_{\Lambda_i}\right) \mathbf{y}$
13:     Refine Co-Support:
            $\Lambda_i := \{k \,|\, 1 \le k \le p, \, |\, \mathbf{w}_k^T \, \hat{\mathbf{x}}_i \,| \, < \epsilon_0\}$
14: end for
15: return $\hat{\mathbf{x}} = \hat{\mathbf{x}}_{d-r}$
```

In practice, the OBG algorithm can be implemented efficiently using the accumulated set of orthogonalized rows $\{\mathbf{q}_j\}$. Using Equation (11) we get that the provisional steps (lines 6–8) in Algorithm 2 can be replaced by computing $\mathbf{q}_i^{(k)}$ for every $k \notin \Lambda_{i-1}$ using (8) and the eventual "Sweep" step can be replaced by

$$\hat{k}_i := \text{Arg} \min_{k \notin \Lambda_{i-1}} \; |\, (\mathbf{q}_i^{(k)})^T \, \hat{\mathbf{x}}_{i-1} \,|. \tag{12}$$

The computational complexity of the two pursuit algorithms, using their efficient implementations discussed above, is $O(d^2 p)$ for BG and is $O(d^3 p)$ for OBG, where we have used the assumption that $r \ll d$ – see Appendix A. Finally, we can design alternative error-based versions for both the BG and OBG pursuit algorithms, where the process described above proceeds until the error $\|\hat{\mathbf{x}}_i - \mathbf{y}\|_2$ is *above* a pre-specified threshold. Suppose this happens at the i^{th} step of the algorithm, then the algorithm returns $\hat{\mathbf{x}}_{i-1}$.

D. Experimenting with Analysis Sparse-Coding

In order to illustrate how the BG and OBG operate in practice, we provide the following brief experiment. Continuing with the experiment setup reported in Section II, we take the 1000 generated example signals and contaminate them with additive white Gaussian noise. We set the noise standard-deviation to be $\sigma = 0.04$, which corresponds to a signal-to-noise ratio (SNR) of 25, and apply BG and OBG. We test the two algorithms in their rank-constrained and error-constrained variants. For the rank-constrained case, we use a varying target subspace dimension r in the range $[3, 10]$ (recall that the true subspace dimension in our case is 4). For the error-constrained case, we use an error threshold $\eta\sqrt{d}\sigma$ with a varying η in the range $[0.5, 1.5]$. We use a threshold $\epsilon_0 = 10^{-4}$ on the coefficients to find the effective co-support.

Figure 3 presents the mean denoising results obtained, measured as the ratio $\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 / (d\sigma^2)$. A ratio below 1 implies an effective noise reduction. The best performance in each of the tests is obtained for $r = 4$ and $\eta = 1.1$ respectively. Interestingly, we see that the BG appears to give a better denoising performance than OBG in this case. The figure also displays the oracle's performance, computed using Equation (5), assuming knowledge of the true co-support.

The observant reader might ask at this stage: why bother with the more complicated OBG, when the simpler BG performs better? The answer resides in a closer inspection of the obtained results. While indeed giving better denoising (which should be explained), the OBG does better in terms of finding a set of clean signals $\mathbf{x}$ that are closer to $\mathbf{y}$, while satisfying the co-sparsity requirement. Figure 4 presents this "representation" error as a function of the chosen dimensionality of the signals $\mathbf{x}$, and as can be seen, the OBG error is smaller (and better). Notice that the error obtained for both methods is below the oracle's, implying that these greedy methods do succeed in extracting good set of rows for the co-support. This

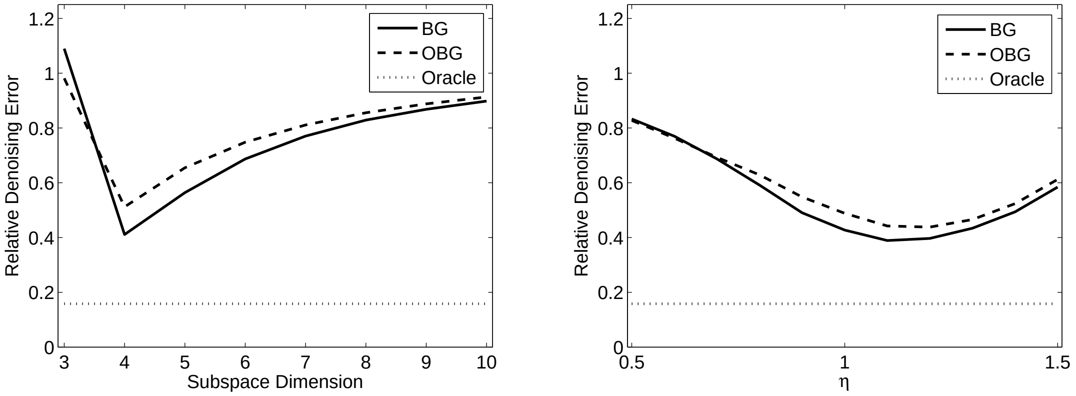

Figure 3. Denoising experiment with sparse analysis signals of dimension 4 created from $\mathbf{\Omega}_{DIF}$: The noise attenuation performance obtained for BG and OBG with varying target subspace dimension (left) and with varying error threshold (right). The oracle result was obtained using the process in equation (5) with the true vanishing support of the signals.

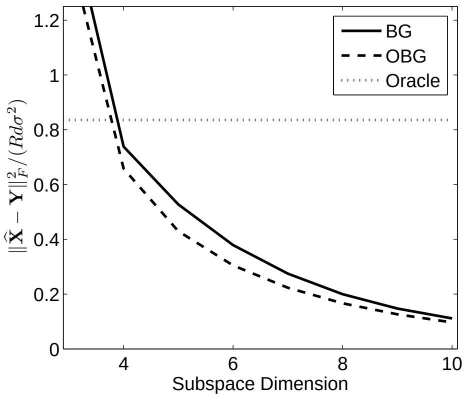

Figure 4. Denoising experiment with sparse analysis signals of dimension 4 created from $\mathbf{\Omega}_{DIF}$: The "representation" error obtained for BG and OBG with varying target subspace dimension. Note that the error is normalized by the number of signal examples R.

superiority of the OBG will become pronounced and critical when turning to the dictionary learning task.

IV. The Analysis K-SVD Algorithm

A. The Learning Goal

We now turn to describe the main part of this work – learning the analysis dictionary. We consider the following setting: Given a training set $\mathbf{Y} = [\mathbf{y}_1 \ \mathbf{y}_2 \ \ldots \ \mathbf{y}_R] \in \mathbb{R}^{d \times R}$, we assume that every example is a noisy version of a signal residing in an r-dimensional subspace related to the dictionary $\mathbf{\Omega}$. Thus, $\mathbf{y}_i = \mathbf{x}_i + \mathbf{v}_i$, where $\mathbf{v}_i$ is an additive zero-mean and white Gaussian noise vector, and $\mathbf{x}_i$ satisfies a co-rank of $d-r$ with respect to the dictionary $\mathbf{\Omega}$. For simplicity we shall assume that all example signals have the same co-rank $d-r$, although the treatment we give below can cope with more general scenarios. Our goal is to find the dictionary $\mathbf{\Omega}$ giving rise to these signals. Taking into account the noise in the measured signals, we formulate an optimization task as follows:

$$\left\{\hat{\mathbf{\Omega}}, \hat{\mathbf{X}}, \left\{\hat{\Lambda}_i\right\}_{i=1}^{R}\right\} = \underset{\mathbf{\Omega}, \mathbf{X}, \{\Lambda_i\}_{i=1}^R}{\text{Argmin}} \ \|\mathbf{X} - \mathbf{Y}\|_F^2 \quad \text{Subject To} \tag{13}$$

$$\begin{aligned} &\mathbf{\Omega}_{\Lambda_i}\mathbf{x}_i = 0, \quad \forall 1 \leq i \leq R \\ &\text{Rank}\left(\mathbf{\Omega}_{\Lambda_i}\right) = d - r, \quad \forall 1 \leq i \leq R \\ &\|\mathbf{w}_j\|_2 = 1, \quad \forall 1 \leq j \leq p. \end{aligned}$$

Here, $\mathbf{x}_i$ are our estimates of the noiseless signals, arranged as the columns of the matrix $\mathbf{X}$, and Λ_i are their co-supports. The vectors $\mathbf{w}_j$ denote the rows of $\mathbf{\Omega}$ (held as column vectors). The normalization constraint on the rows of $\mathbf{\Omega}$ is introduced to avoid degeneracy, but otherwise has no practical influence on the result. We note the similarity of this problem to the ℓ^0 synthesis training problem [4], given by

$$\left\{\hat{\mathbf{D}}, \hat{\mathbf{\Gamma}}\right\} = \underset{\mathbf{D}, \mathbf{\Gamma}}{\text{Argmin}} \ \|\mathbf{D\Gamma} - \mathbf{Y}\|_F^2 \quad \text{Subject To} \tag{14}$$

$$\begin{aligned} &\|\gamma_i\|_0 \leq k, \quad \forall i \\ &\|\mathbf{d}_j\|_2 = 1, \quad \forall j. \end{aligned}$$

This similarity will become useful in developing a learning method for the analysis case, which shares the same overall structure with some synthesis training algorithms. Indeed, the training problem posed in Equation (13) is highly non-convex (as is the problem posed in Equation (3)), and as such we can only hope for a local solution in general. Thus, we adopt a simple iterative approximation method, described next, which draws its spirit from earlier works on synthesis dictionary learning.

B. Block-Coordinate Descent Algorithm

Assuming an initial estimation $\mathbf{\Omega}_0$ of the analysis operator, the optimization scheme is based on a two-phase block-coordinate-relaxation approach. In the first phase we optimize for $\mathbf{X}$ while keeping $\mathbf{\Omega}$ fixed, and in the second phase we update $\mathbf{\Omega}$ using the computed signals $\hat{\mathbf{X}}$ (in fact, we will be using the detected co-supports and not $\hat{\mathbf{X}}$ directly). The process repeats until some stopping criterion (typically a fixed number of iterations) is achieved.

Given the analysis dictionary $\boldsymbol{\Omega}_0$, optimizing for $\mathbf{X}$ can be done individually for each column of $\mathbf{X}$, defining an ordinary co-rank analysis approximation problem for each signal $\mathbf{y}_i$,

$$\left\{\hat{\mathbf{x}}_i,\ \hat{\Lambda}_i\right\} = \underset{\mathbf{x}_i,\,\Lambda_i}{\text{Argmin}}\ \|\mathbf{x}_i - \mathbf{y}_i\|_2 \quad \text{Subject To} \quad \begin{aligned} &\boldsymbol{\Omega}_{\Lambda_i}\mathbf{x}_i = 0 \\ &\text{Rank}\left(\boldsymbol{\Omega}_{\Lambda_i}\right) = d - r \end{aligned} \tag{15}$$

which may be solved using a pursuit method as described in Section III.

Once $\hat{\mathbf{X}}$ is computed, we turn to update $\boldsymbol{\Omega}$ in the second step. The optimization is carried out sequentially for each of the rows $\mathbf{w}_j$ in $\boldsymbol{\Omega}$. We note that, similar to the K-SVD algorithm [4], the update of $\mathbf{w}_j$ should be affected only by those columns of $\hat{\mathbf{X}}$ that are orthogonal to it, while the remaining signal examples should have no influence. Thus, letting $\hat{\mathbf{X}}_J$ denote the sub-matrix of $\hat{\mathbf{X}}$ containing the columns found to be orthogonal to $\mathbf{w}_j$, and denoting by $\mathbf{Y}_J$ the corresponding sub-matrix of $\mathbf{Y}$, the update step for $\mathbf{w}_j$ can be written as

$$\left\{\hat{\mathbf{w}}_j,\ \hat{\mathbf{X}}_J\right\} = \underset{\mathbf{w}_j,\,\mathbf{X}_J}{\text{Argmin}}\ \|\mathbf{X}_J - \mathbf{Y}_J\|_F^2 \quad \text{Subject To} \quad \begin{aligned} &\boldsymbol{\Omega}_{\Lambda_i}\mathbf{x}_i = 0, \quad \forall i \in J \\ &\text{Rank}\left(\boldsymbol{\Omega}_{\Lambda_i}\right) = d - r, \quad \forall i \in J \\ &\|\mathbf{w}_j\|_2 = 1\ . \end{aligned} \tag{16}$$

Adopting the approach taken by the K-SVD, we maintain the constraint on $\boldsymbol{\Omega}\mathbf{x}_i$ by constraining each $\mathbf{x}_i$ to remain orthogonal to the rows in $\boldsymbol{\Omega}$ it has been found to already be orthogonal to. This is parallel to the K-SVD atom update process, where the representation supports are kept fixed. To formalize this, we use the notation $\boldsymbol{\Omega}_i$ to denote the sub-matrix of $\boldsymbol{\Omega}$ containing the rows from $\boldsymbol{\Omega}$ that $\mathbf{x}_i$ is currently orthogonal to, *excluding* the row $\mathbf{w}_j$. We can thus write the optimization problem for $\mathbf{w}_j$ as

$$\left\{\hat{\mathbf{w}}_j,\ \hat{\mathbf{X}}_J\right\} = \underset{\mathbf{w}_j,\,\mathbf{X}_J}{\text{Argmin}}\ \|\mathbf{X}_J - \mathbf{Y}_J\|_F^2 \quad \text{Subject To} \quad \begin{aligned} &\boldsymbol{\Omega}_i\mathbf{x}_i = 0, \quad \forall i \in J \\ &\mathbf{w}_j^T\mathbf{X}_J = 0 \\ &\|\mathbf{w}_j\|_2 = 1\ . \end{aligned} \tag{17}$$

This suggests that the "dictionary-update" stage uses only the co-supports (rather than the processed signals $\hat{\mathbf{X}}$) that were found in the "sparse-coding" stage. Unfortunately, in general, solving the problem posed in Equation (17) is a difficult task. However, as an alternative, we propose the following approximation to the above optimization goal:

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\text{Argmin}}\ \|\mathbf{w}_j^T\mathbf{Y}_J\|_2^2 \quad \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1. \tag{18}$$

For this problem, the solution is the singular vector corresponding to the *smallest* singular value of $\mathbf{Y}_J$, which can be efficiently computed from the SVD of $\mathbf{Y}_J$, or using some inverse power method. As we show in the results section, this simple approach turns out to be very effective in recovering analysis dictionaries from sparse example sets. In Appendix B we discuss this approximation choice and its relation to the original goal posed in Equation (17).

One advantage of this specific approximation method is that it disjoints the updates of the rows in $\boldsymbol{\Omega}$, enabling all rows to be updated in parallel. Another desirable property of the resulting algorithm is that it assumes a similar structure to the Synthesis K-SVD algorithm – replacing the maximum eigenvalue computation with a minimum eigenvalue one. We term the resulting algorithm *Analysis K-SVD* due to its resemblance to the original K-SVD algorithm.

C. Summary: The Algorithm

The Analysis K-SVD is an iterative scheme that has a simple intuitive interpretation. Each iteration consists of two stages. In the first stage we find for each signal the rows in $\boldsymbol{\Omega}$ that determine the subspace it resides in (the set of rows that the signal is "most orthogonal" to). This is followed by updating each row in $\boldsymbol{\Omega}$ to be the vector that is most orthogonal to all signals associated to it in the first stage. A detailed description is provided in Algorithm 3.

Algorithm 3 ANALYSIS K-SVD

```
Input: Training signals $\mathbf{Y} \in \mathbb{R}^{d \times R}$, initial dictionary $\boldsymbol{\Omega}_0 \in \mathbb{R}^{p \times d}$, target co-rank $d-r$ and number of iterations $k$
Output: Dictionary $\boldsymbol{\Omega}$ and signal set $\mathbf{X}$ minimizing (13)
Initialization: Set $\boldsymbol{\Omega} := \boldsymbol{\Omega}_0$
for $n = 1 \ \ldots \ k$ do
    Analysis Pursuit:
        $\forall i: \ \left\{\hat{\mathbf{x}}_i,\ \hat{\Lambda}_i\right\} := \underset{\mathbf{x},\,\Lambda}{\text{Argmin}}\ \|\mathbf{x} - \mathbf{y}_i\|_2$ Subject To $\boldsymbol{\Omega}_\Lambda\mathbf{x} = 0$, $\text{Rank}\left(\boldsymbol{\Omega}_\Lambda\right) = d - r$
    for $j = 1 \ \ldots \ p$ do
        Extract Relevant Examples: $J :=$ indices of the columns of $\hat{\mathbf{X}}$ orthogonal to $\mathbf{w}_j$
        Compute Row:
            $\hat{\mathbf{w}}_j := \underset{\mathbf{w}}{\text{Argmin}}\ \|\mathbf{w}^T\mathbf{Y}_J\|_2$ Subject To $\|\mathbf{w}\|_2 = 1$
        Update Row: $\boldsymbol{\Omega}\{j\text{-th row}\} := \mathbf{w}_j^T$
    end for
end for
```

In practice, our implementation incorporates the following improvement to Algorithm 3, intended to resolve deadlock situations in the iterative process: During the dictionary update step, each row in $\boldsymbol{\Omega}$ which is found to have too few associated examples or is too close to another row in $\boldsymbol{\Omega}$, namely the maximal inner product with the rest of the rows is larger (in absolute value) than $1 - \delta$, is regarded as a false one and is therefore replaced by a new row which is generated in a random fashion. One possible generation process, which we use here, is randomly selecting a set of $d-1$ examples and setting the row as the vector that spans their one-dimensional null-space.

Algorithm 3 constitutes our basic scheme for analysis dictionary learning. Depending on the specific setup at hand, we may want to encourage certain properties in the learned dictionary. To this end, we can apply a post-processing on the learned atoms or explicitly modify the atom update rule (18). We shall consider such options in the experiments in the next section.

V. Simulation Results

In this section we present a set of experiment results with the proposed training algorithm. The intention of these experiments is to demonstrate the core capability of the proposed learning method, and to expose the potential that exists in the analysis model when coupled with a learned dictionary. In the first part we present experiments on synthetic signals, demonstrating the ability of the method to recover the true underlying analysis dictionary $\mathbf{\Omega}$ given a training set of analysis signals. In the second part we show results for a piecewise-constant (PWC) image and observe the emergence of meaningful structures in the trained dictionary. We also show that the Analysis K-SVD approach may outperform the Synthesis K-SVD, total-variation (TV) denoising and the BM3D method in denoising of such a PWC image.

We then turn to natural images and test the image denoising performance of the analysis dictionary learned by our approach. In this setup we get results that outperform those obtained by the FoE approach, and are comparable to those obtained by the Synthesis K-SVD algorithm. We discuss the limitations of our approach and point to future research directions that will better exploit the capabilities of the analysis model and its learned dictionary.

A. Synthetic Experiments

To demonstrate the performance of the proposed algorithm in recovering an underlying dictionary $\mathbf{\Omega}$, we perform a set of synthetic experiments. In these experiments, the analysis dictionary $\mathbf{\Omega} \in \mathbb{R}^{p\times d}$ is known and a set of R sparse signal examples, each residing in an r-dimensional subspace, are generated as described in Section II. These sparse analysis signals are normalized to unit length and are optionally subjected to additive white Gaussian noise, producing the final training set.

We begin with a setup where the ground-truth analysis dictionary $\mathbf{\Omega} \in \mathbb{R}^{50\times 25}$ is generated with random Gaussian entries, and the data set consists of $R = 50,000$ analysis signals each residing in a 4-dimensional subspace. We test both the noise-free setup and a noisy setup with noise level $\sigma = 0.2/\sqrt{d} = 0.04$ (SNR=25). We remark that in the noiseless case, and when using the rank-constrained variants of BG and OBG, we first of all find the rows in $\hat{\Omega}$ that are orthogonal to the input signal, update the initial co-support $\hat{\Lambda}_0$, and then run the main loop in these algorithms until the rank of $\mathbf{\Omega}_{\Lambda_i}$ equals $d-r$. This may take less than $d-r$ iterations when the input is already orthogonal to one or more rows in Ω, which occurs as the trained dictionary approaches the true analysis dictionary.

The algorithm is initialized with a dictionary $\mathbf{\Omega}_0$ in which each row is orthogonal to a random set of $d-1$ examples. We apply 100 iterations of the Analysis K-SVD algorithm using the OBG algorithm[1] with a target subspace dimension $r = 4$ for the sparse-coding stage. We fix the parameter $\delta = 0.05$ to enable the recovery of distinct rows in $\mathbf{\Omega}$. A row $\mathbf{w}_j$ in the true dictionary $\mathbf{\Omega}$ is regarded as recovered if

$$\underset{i}{\mathrm{Min}}(1 - |\hat{\mathbf{w}}_i^T \mathbf{w}_j|) < 0.01, \tag{19}$$

where $\hat{\mathbf{w}}_i$ are the atoms of the trained dictionary.

The results of these experiments are shown in Figure 5 on the top, demonstrating the ability of the proposed approach to produce a good estimate of the true underlying operator $\mathbf{\Omega}$ given a sparse analysis training set. As can be seen in the right figure, 94% of the rows in the true $\mathbf{\Omega}$ were recovered in the noise-free setup and 86% in the noisy one. Referring to the left figure, which shows the average representation error per element $\|\hat{\mathbf{X}} - \mathbf{Y}\|_F/\sqrt{Rd}$, we see that this error goes below the noise level σ, indicating a successful training.

To demonstrate the improved efficiency of the Analysis K-SVD, we compare it with [13] on the same experimental setup. Each iteration in their approach starts with a random row, generating a candidate row by a sequential process (10 inner iterations) that alternates between determining the set of relevant signals (there is a ratio ℓ/p between the size of this set and the total number of signals in use at a given iteration) and computing the least eigenvector corresponding to their autocorrelation matrix. A row is accepted to the set of learned atoms if its maximal inner product with the relevant signals is smaller than θ_0 and its maximal inner product with the rows accepted so far is smaller than $1-\delta$.

We apply the randomized version of their algorithm (Algorithm 3 in [13]) with parameters $\ell = d-4 = 21$, $\theta_0 = 1.5\sigma$ (in the noise-free setup: $\theta_0 = 10^{-3}$) and $\delta = 0.05$. Note that the parameter ℓ in this case equals the co-sparsity of each signal example. The δ parameter is similar to the one used in our algorithm – we use it for replacing similar rows in $\mathbf{\Omega}$, while they use it for rejecting rows. The algorithm terminates when p rows have been accumulated or after 300 iterations. The results are shown in Figure 6 on the bottom. In the noise-free setup we observed that while all the accepted rows were correct, the algorithm requires an increasing number of iterations to detect a new distinct row. The final recovery rate after 300 iterations is 74%, which is still inferior to the Analysis K-SVD algorithm. In the noisy case, however, many false rows are accepted to the accumulated set, and once admitted they cannot be replaced. This results in a very poor recovery rate of 44%.

Next we repeat the experiments with the Analysis K-SVD algorithm for the dictionary $\mathbf{\Omega}_{DIF} \in \mathbb{R}^{50\times 25}$ (Figure 1), keeping the setup the same as before, apart from one modification: In the last 25 iterations (out of 100 iterations)

[1] We have found the OBG to perform much better than the BG for dictionary training, and thus focus on this option here.

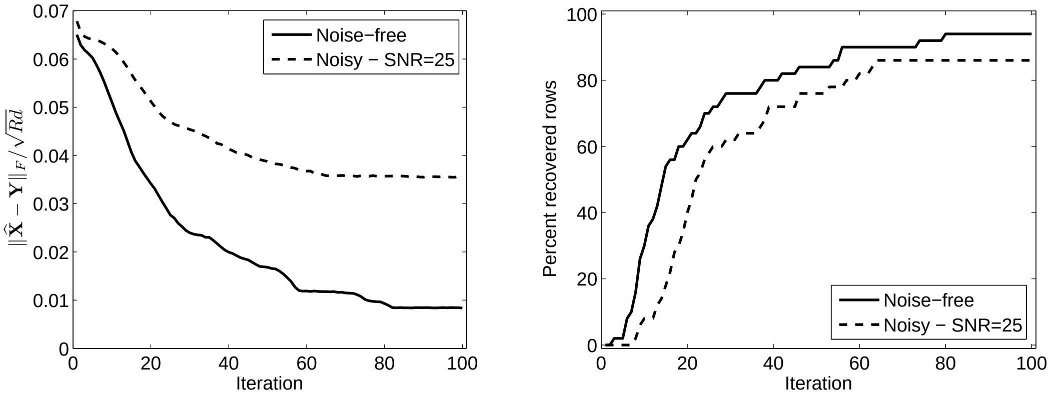

Figure 5. Synthetic experiment results of the Analysis K-SVD algorithm for a random dictionary $\mathbf{\Omega} \in \mathbb{R}^{50\times25}$ and a training set of $R = 50,000$ analysis signals residing in 4-dimensional subspaces.

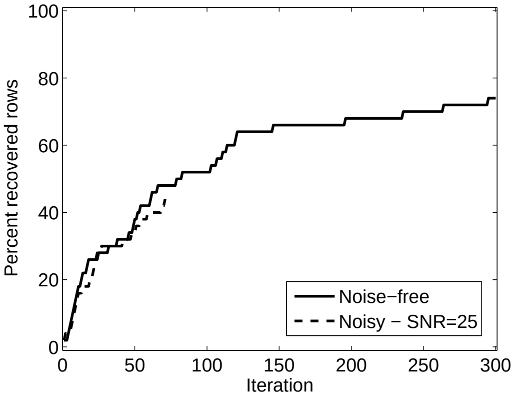

Figure 6. Synthetic experiment results of analysis dictionary learning with the algorithm suggested in [13] for a random dictionary $\mathbf{\Omega} \in \mathbb{R}^{50\times25}$ and a training set of $R = 50,000$ analysis signals residing in 4-dimensional subspaces.

of the algorithm, we employ a slight modification to the training iterations, intended to encourage *linear-dependence* in the dictionary. Specifically, at the end of each iteration, we nullify near-zero entries in $\mathbf{\Omega}$, and remove the mean from rows whose mean is near-zero. These two simple post-processing steps have been found to be advantageous in several training experiments, as these steps promote linear dependencies in the analysis dictionary. Intuitively, such dictionaries enable larger co-supports to be achieved, which can likely be more stably recovered. We note that this is in sharp contrast with synthesis dictionaries, where linear independence is highly desirable.

The results of these experiments are depicted in Figure 7, showing good recovery results for this setup as well. As can be seen, at the end of the first phase of the algorithm (after 75 iterations) 84% of the rows in $\mathbf{\Omega}_{DIF}$ were recovered in the noise-free setup and 80% in the noisy one. At the the end of the second phase (after 100 iterations) the recovery rates increase to 92% for the noise-free setup and 84% in the noisy case. The learned analysis dictionaries at the end of the first and second phases are shown at the bottom row of this figure.

During the second phase of the algorithm – the last 25 iterations – there is a steep increase in the co-sparsity ℓ due to the linear dependencies emerging within the rows of $\mathbf{\Omega}$: The average ℓ becomes 34.1 for the noise-free setup and 30.6 for the noisy one, which is close to the true average co-sparsity, 35. This results in a significant improvement in the denoising performance for the noisy setup, aligning with our hypothesis that linear dependencies in the dictionary are beneficial for analysis-based denoising. Note that the representation error per element increases in the second phase, however it is still below the noise level, as in the previous experiment.

We should note that in this last experiment we encouraged the rows in the recovered dictionary to be sparse and have zero-mean, using a simple post-processing. However, it may be possible to incorporate these constraints explicitly into the learning goal of Equation (13). Further work is required to explore this direction, a well as other options for encouraging linear dependencies in the recovered dictionary.

Finally, we compare our results on $\mathbf{\Omega}_{DIF}$ with [13]. All the parameters of their method remain the same as for the random $\mathbf{\Omega}$, apart from ℓ that should be adjusted to account for the expected linear dependence within the rows of $\mathbf{\Omega}$. We tested three values of ℓ: 28, 30 and 32, all higher than $d - r = 21$. The recovery rates for these three choices of ℓ are depicted in Figure 8. We can see that for $\ell = 32$ excellent recovery rates were obtained (98% recovery rate in the noisy setup!). However, this requires a large number of iterations. Specifically, in the noisy case the algorithm terminates after 300 iterations before it has finished accumulating p rows. More importantly, when using lower values of ℓ, the recovery rates drop to around 60% and 80% (for $\ell = 28$ and $\ell = 30$ respectively). This shows the high sensitivity of this algorithm to its parameters. Note that in contrast to the co-rank r used by the Analysis K-SVD algorithm and common to all the signal examples, the parameter ℓ related with the co-sparsity of the signals has a high variability. Therefore, it is not straightforward to determine its value.

B. Experiments with Images

We now turn to present experimental results on images, with the aim of evaluating the behavior of the training algorithm for signals with more meaningful content, where there

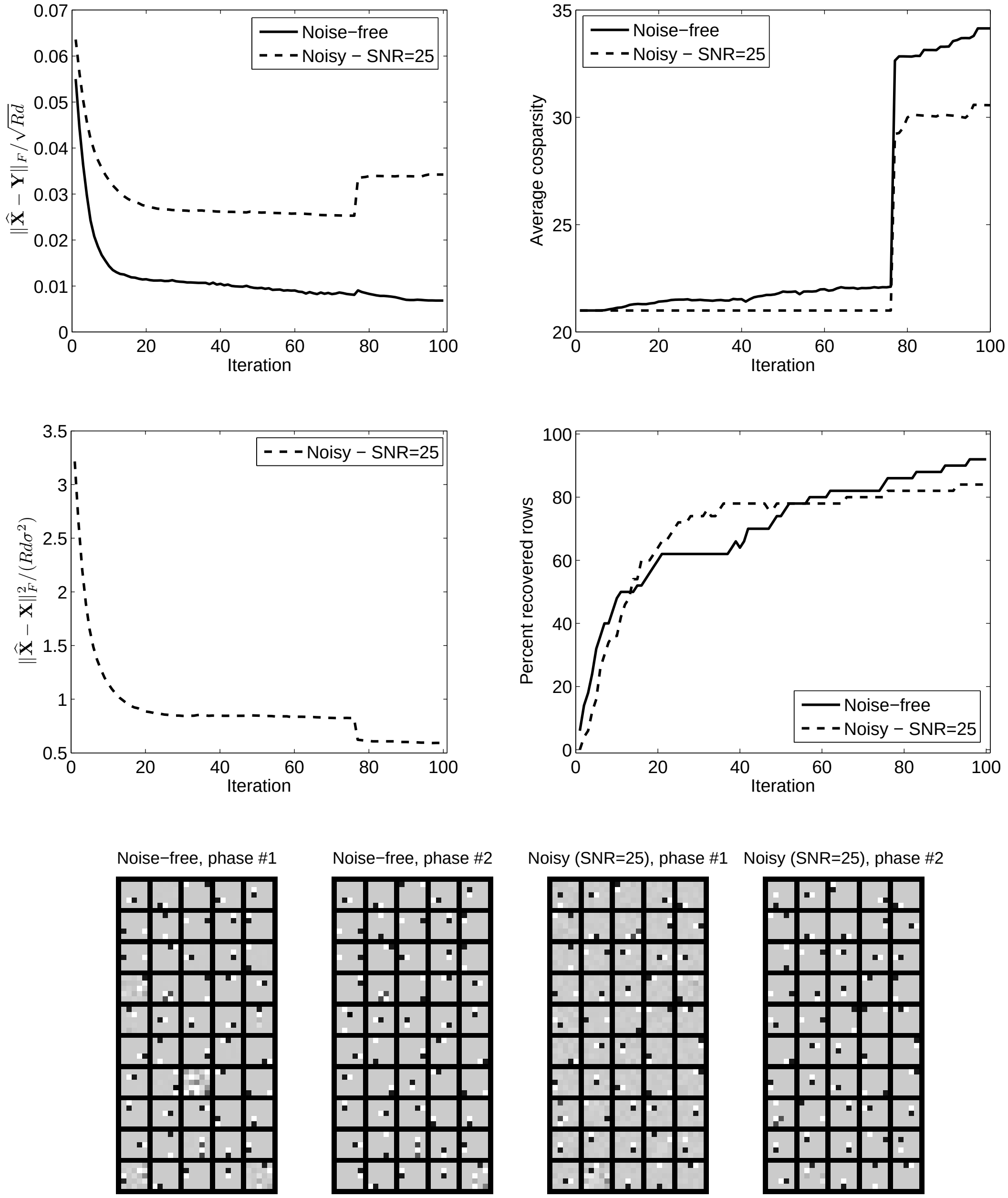

Figure 7. Synthetic experiment results of the Analysis K-SVD algorithm for a dictionary $\mathbf{\Omega}_{DIF} \in \mathbb{R}^{50\times25}$ and a training set of $R = 50,000$ analysis signals residing in 4-dimensional subspaces.

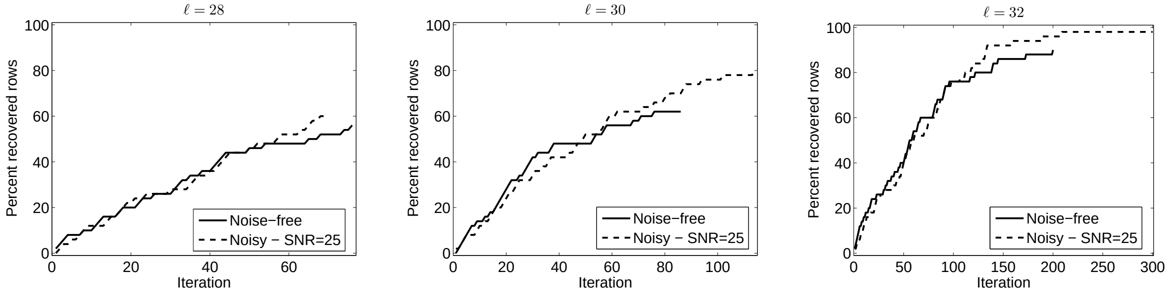

Figure 8. Synthetic experiment results of analysis dictionary learning with the algorithm suggested in [13] for a dictionary $\mathbf{\Omega}_{DIF} \in \mathbb{R}^{50\times25}$ and a training set of $R = 50,000$ analysis signals residing in 4-dimensional subspaces. The algorithm was run with three different choices of the parameter ℓ in the range $[28, 32]$.

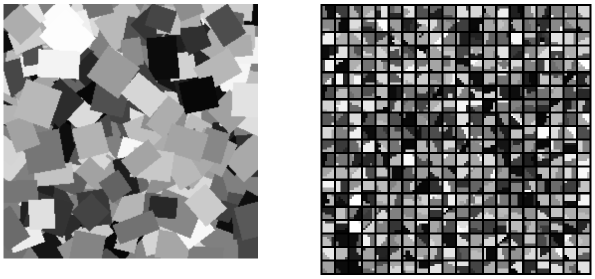

Figure 9. Left: A 256×256 piecewise-constant image. Among all possible patches of size 5×5 extracted from this image, less than half are non-flat. Right: Examples of non-flat patches, used for the training (their noise-free versions are shown).

is no ground-truth reference dictionary. For a given image, contaminated by additive white Gaussian noise, we apply the Analysis K-SVD algorithm (specific implementation details will be provided for each experiment) on a subset of patch examples chosen for training and learn an analysis dictionary.

The learned analysis dictionary is then utilized to denoise each overlapping patch extracted from the noisy image. For the denoising we apply the error-based version of OBG with an error threshold $1.15\sqrt{d}\sigma$, to allow for content-adaptive subspace dimensions in the recovered patches, in parallel with patch denoising with a synthesis dictionary [21]. The patch denoising stage is followed by averaging the overlapping patch recoveries to obtain the final denoised image. This approach is referred to as analysis patch-based image denoising. We remark that [18] takes a different approach towards image denoising with their learned analysis dictionary. The core idea of their approach is that image recovery can be formulated as a regularized optimization problem where the learned analysis dictionary serves in a sparsity-promoting prior for the analysis representations of all overlapping patches.

We start with a piecewise-constant image of size 256×256 contaminated by low noise $\sigma = 5$ ($PSNR = 34.15$dB) and extract all possible 5×5 image patches. We observe that out of a total of $63,504$ patches, less than half were "active" (i.e., non-flat). We randomly choose $20,000$ "active" patches for the dictionary training. The PWC image and examples of "active" patches extracted from it are shown in Figure 9.

We apply 100 iterations of the Analysis K-SVD algorithm on this training set, learning an analysis dictionary of size 50×25 and assuming recovered signals residing in 4-dimensional subspaces. The initialization of the dictionary is the same as in the previous experiments, and the training is composed of two phases: In the first phase we apply 75 iterations of the algorithm using BG for the "sparse-coding" stage[2], and in the second we apply 25 iterations using OBG and the post-processing described in the previous subsection (nulling small entries and removing small means).

The learned analysis dictionary is shown in Figure 10. It exhibits a high resemblance to the $\mathbf{\Omega}_{DIF}$ dictionary (see Figure 1). This observation aligns with our intuition that many finite differences computed on a piecewise-constant signal (in our case – a 2D patch) are expected to be near zero. This dictionary leads to a representation error per element of 3.85 and a denoising error per element of 3.88 on the training set — both are below the noise level $\sigma = 5$, indicating a successful training. The average co-sparsity level on the training set grows from 21 for the initial dictionary to 29.4 for the final one, indicating the emergence of linear dependencies between the rows of the learned dictionary.

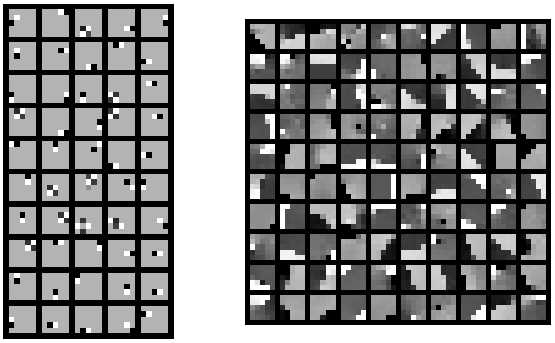

Figure 10. Results of the Analysis and Synthesis K-SVD algorithms for the piecewise-constant image shown in Figure 9 with noise level $\sigma = 5$. Left: learned analysis dictionary $\mathbf{\Omega} \in \mathbb{R}^{50 \times 25}$. Right: learned synthesis dictionary $\mathbf{D} \in \mathbb{R}^{25 \times 100}$. Both dictionaries are learned from a training set of $R = 20,000$ patches. Each signal recovery for this training set resides in a 4-dimensional subspace.

Next we use the learned analysis dictionary in the patch-based image denoising approach mentioned above. Previous work on this approach [21] has found it useful to remove the DC from each noisy patch before it is denoised and add back the DC to the recovered patch before the final averaging process. This procedure guarantees that the DC of each noisy patch is preserved in its recovery. Note that since the learned dictionary in the PWC setup consists of zero-mean atoms, the DC is preserved anyhow, thus we can avoid this procedure.

We compare the suggested approach with three other image denoising methods — the Synthesis K-SVD [21], TV denoising (with the algorithm and software provided in [29]) and BM3D [30] that is known to lead to state-of-the-art results on natural images. For the Synthesis K-SVD we use the same training set as for the Analysis K-SVD: $20,000$ "active" patches of size 5×5, and we train a redundant synthesis dictionary of size 25×100. The initial synthesis dictionary is a set of random patch examples (for details see [4]) and the recovered patches reside in 4-dimensional subspaces (the DC adds a dimension, namely 3 atoms are used in each recovery). The learned synthesis dictionary is shown in Figure 10. We can see that this dictionary consists of atoms containing an edge, as should be expected when looking at the patch examples (see Figure 9). TV denoising solves a regularized image recovery problem using horizontal and vertical one-sided derivatives in the regularization term (these are the same derivatives used to generate the $\mathbf{\Omega}_{DIF}$ dictionary). The BM3D method is applied using the "normal" profile (for details see [30]). In this method

[2] BG was used in the first phase of the algorithm in order to speed-up the overall learning.

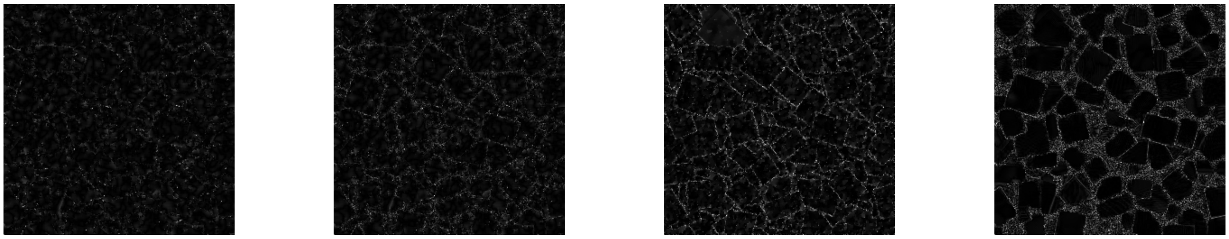

Figure 11. Image denoising results on a noisy piecewise-constant image ($\sigma = 5$, $PSNR = 34.15$dB). Images of the absolute errors are displayed in the dynamic range $[0, 20]$ (from left to right): Analysis K-SVD ($PSNR = 45.67$dB), Synthesis K-SVD ($PSNR = 43.67$dB), TV denoising ($PSNR = 41.12$dB) and BM3D ($PSNR = 40.66$dB).

parameters such as block dimensions and orthogonal basis for the 3D blocks are fixed – they are set to perform best on natural images – rather than adapting them to the data at hand.

The resulting PSNRs of the denoised images are 45.67dB for Analysis K-SVD, 43.67dB for Synthesis K-SVD, 41.12dB for TV denoising and 40.66dB for BM3D. Figure 11 shows the absolute difference images for each of the three methods. Note that these images are displayed in the dynamic range $[0, 20]$ to make the differences more pronounced. The Analysis K-SVD approach leads to a much better denoising result compared to TV-denoising (a performance gap of 4.5dB!), despite the fact that both use first-order derivatives. This difference can be explained by the locality and the different measure applied on the derivative outputs: the co-rank measure on all the derivatives associated with each patch proves to be much more efficient than the ℓ^2-norm computed on the pair of vertical and horizontal derivatives for each pixel. The much-inferior denoising performance of BM3D on this image – with respect to the Analysis and Synthesis K-SVD approaches – can be explained by the fact that it uses a 2D-DCT basis for modeling the patches, which is well-suited for patches of a natural image, but does not give a good fit for piecewise-constant patches. Both the Analysis and Synthesis K-SVD approaches adapt the dictionary to the data, leading to better image denoising despite the fact that they do not exploit self-similarities within the image as the BM3D does.

More interestingly, when we compare the Analysis and Synthesis K-SVD results, we get a gain of 2dB (image PSNR) in favor of the analysis approach. The analysis-based approach also obtains better patch recoveries (before averaging the overlaps), both in terms of the average subspace dimension (1.7 for analysis versus 2.4 for synthesis) and in terms of the denoising error per element (2.08 for analysis versus 2.94 for synthesis), for the same constraint on the representation error $\left(\|\hat{\mathbf{x}}_i - \mathbf{y}_i\|_2 \leq 1.15\sqrt{d}\sigma,\ \forall i \in \{1, \ldots, R\}\right)$. We thus conclude that for this case, the analysis approach provides a better modeling platform than the comparable synthesis approach. Indeed, characterizing additional families of such signals remains an interesting and open question.

We now turn to natural images and test the suggested approach on five images, ‘Lena’, ‘Barbara’, ‘Boats’, ‘House’, and ‘Peppers’, which are commonly used for evaluating image denoising performance (see for example [18], [21]). For each such image, contaminated by low noise $\sigma = 5$, we learn a redundant dictionary of size 63×49 for patches of size 7×7 by applying 20 iterations of the Analysis K-SVD algorithm, using OBG with a target subspace dimension of $r = 7$ in the pursuit stage. We use a training set consisting of $20,000$ informative patch examples extracted from the noisy image and a data-driven dictionary initialization, just as in our experiments with the PWC image.

In this setup we do not apply the atom post-processing suggested before for the PWC image, since for these images the learned atoms do not tend to be sparse. Note that it is not straightforward to encourage strong linear dependence in the learned dictionary, and thus, without these dependencies, the suggested analysis pursuit algorithms are prone to highly instable recoveries. We have tested with the basic Analysis K-SVD algorithm on natural images and obtained results that are comparable with the FoE approach. In order to further improve these results, we suggest to encourage another desired dictionary property to compensate for the instability arising from the lack of strong linear dependencies. Specifically, we consider the following modification to the atom update rule as given in (18), consisting of a regularization term in addition to the original objective function,

$$\begin{aligned} \hat{\mathbf{w}}_j = & \underset{\mathbf{w}_j}{\text{Argmin}} \quad \left\|\mathbf{w}_j^T \mathbf{Y}_J\right\|_2^2 + \gamma \sum_{i \neq j} |I^C \cap J| (\mathbf{w}_j^T \mathbf{w}_i)^2 \\ & \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1. \end{aligned} \tag{20}$$

Here, I and J are the sets of training examples associated with the ith and jth atom (I^C is the complementary set). Similar to the original atom update rule in (18), this optimization problem has a closed-form solution given by the least eigenvector of the matrix

$$\mathbf{Y}_J \mathbf{Y}_J^T + \gamma \sum_{i \neq j} |I^C \cap J| \mathbf{w}_i \mathbf{w}_i^T. \tag{21}$$

Let us take a closer look at the suggested regularization term. It encourages pairs of atoms that do not tend to jointly appear in the co-support to be orthogonal to each other. Intuitively, this leads to a better distinction between the candidate co-supports for a given noisy signal and hence to a more stable recovery. In fact, this modification in the atom update rule is associated with the Restricted Orthogonal Projection Property (ROPP) of analysis dictionaries [31], an important factor dictating

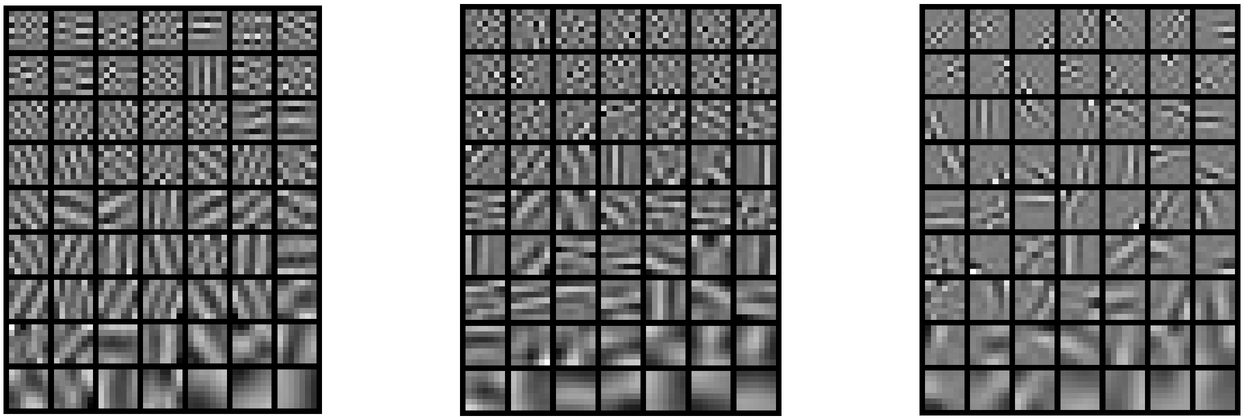

Figure 12. Results of the Analysis K-SVD algorithm with the modified atom upate rule in (20) for several natural images with noise level $\sigma = 5$. The learned analysis dictionaries of size 63×49 are shown for the images 'Barbara' (left), 'House' (middle) and 'Peppers' (right). The atoms are arranged according to their appearance frequency in the co-supports (from left to right, then one row down). All dictionaries are learned from training sets of $R = 20,000$ informative patches extracted from the noisy image. Each signal recovery for this training set resides in a 7-dimensional subspace.

Image denoising method	Lena	Barbara	Boats	House	Peppers
FoE [18]	38.12	37.19	36.27	38.23	37.63
Analysis K-SVD with the modified atom update rule (20)	38.47	37.75	37.13	39.15	37.89
Synthesis K-SVD [21]	38.60	38.08	37.22	39.37	37.78

Table I
SUMMARY OF IMAGE DENOISING RESULTS (IMAGE PSNR IN dB) ON FIVE NOISY NATURAL IMAGES ($\sigma = 5$, $PSNR = 34.15$dB).

the performance of analysis pursuit algorithms. This property appears in the theoretical performance guarantees derived in [31] for the analysis thresholding algorithm (a simpler version of BG and OBG).

In our experiments we set the parameter γ to 1000. We remark that γ, the number of atoms p, and many other parameters (patch size, noise power, target subspace dimension, number of training iterations) are chosen rather arbitrarily in our experiments, with no aim at optimizing the performance. Further work that aims to extract state-of-the art results from the proposed scheme may explore a proper choice of these parameters. In this work our goal is to demonstrate the core paradigm. Several examples of the learned dictionaries are provided in Figure 12, showing the emergence of meaningful structures. Specifically, some of the learned atoms are directional, others are localized and the bottom ones (the most rarely used in the co-supports) correspond to low frequencies.

The learned dictionaries are utilized for patch-based image denoising. Here we apply the DC-preserving procedure mentioned before, since the learned atoms do not necessarily have zero mean. We compare the denoising performance for the learned analysis dictionaries with FoE [18] and Synthesis K-SVD image denoising [21], which learns a synthesis dictionary of size 64×256 for patches of size 8×8. The image denoising results (image PSNR in dB) appear in Table I. The suggested approach outperforms FoE and is slightly weaker than Synthesis K-SVD, showing that the analysis model is suitable for handling natural images.

To conclude this section, we mention several limitations of the suggested approach. First, the learned analysis dictionaries for the natural images are lacking in strong linear dependencies, which are desired as they lead to larger co-sparsity levels and hence to more stable recoveries. We have already shown how to encourage strong linear dependence in $\mathbf{\Omega}$ for the specific setup of PWC image patches. Extending this property to other setups, and ultimately designing a general framework that will work also for natural image patches, seems to be a challenging and promising direction for future research. Secondly, at this point it is not clear how to combine the two useful dictionary properties – linear dependence and ROPP – and this remains an question for future research. Finally, the suggested dictionary update stage does not correspond to a maximum-likelihood estimator and specifically its objective function does not take into consideration the partition function of the likelihood function. In future research we intend to explore the possibility of adding a regularization term to the dictionary learning objective that will correspond to such a partition function.

VI. CONCLUSIONS

In this work we have presented an efficient algorithm for learning an ℓ^0 analysis dictionary, which is parallel to the Synthesis K-SVD in its rationale and structure. We have demonstrated the effectiveness of this algorithm in several experiments, showing a successful and meaningful recovery of the analysis dictionary for random, piecewise-constant and natural signal data. We have also shown a significant advantage of the suggested approach compared to previous image denoising methods in the case of a piecewise-constant image with low noise-level. This exposes the potential benefit of the suggested approach. For natural images we obtained denoising results which are close to those of the Synthesis K-SVD and we believe that these results can be further improved by better exploiting the capabilities of the analysis model.

Our work gives rise to several questions which are left open for future research. First, what are the desired properties of an analysis dictionary? In this work we have seen evidence that having linear dependencies between sets of rows in the dictionary can improve the recovery quality of pursuit algorithms. This naturally leads to a second research question: How can we encourage such linear dependencies to be formed within the dictionary? Our simulations have demonstrated that forcing sparse and zero-mean rows during the training leads to the emergence of such linear dependencies. It would be desired to obtain a more general formulation of this property and to insert it directly into the learning goal. Having strong linear dependencies is only one desired property of the analysis dictionary and our experiments on natural images imply that other properties, such as ROPP, can be very useful as well. Further work is required to reveal additional dictionary properties and design efficient algorithms for encouraging these properties. Finally, it remains to be seen which applications could benefit from the analysis model and its dictionary learning.

Appendix A: BG and OBG Complexities

In this appendix we analyze the computational complexity of the BG and the OBG algorithms, both assumed to be implemented using the orthogonalization numerical shortcut described in Section III. We start with the BG algorithm and count the number of inner-products between vectors of length d, each requiring d operations. In the i-th iteration ($1 \le i \le d-r$), (i) the "Sweep" step requires $p-i+1$ such vector-multiplications, (ii) orthogonalizing the currently chosen vector to compute $\mathbf{q}_i$ requires $i-1$ vector-multiplications, (iii) the "Project" step requires another such multiplication, and finally, (iv) the "Refine Co-Support" step uses p vector multiplications. Adding all these gives

$$\begin{aligned}\text{Complexity}_{BG} &= d \cdot \sum_{i=1}^{d-r} [p-i+1+i-1+1+p] \qquad \text{(A-1)}\\ &= d(d-r)(2p+1) = \mathcal{O}(d^2 p).\end{aligned}$$

Here we have assumed that $r \ll d$.

Similarly, the OBG performs the following operations in the i-th iteration: (i) the provisional steps are repeated $p-i+1$ times, where first we orthogonalize with $i-1$ vector multiplications, and then compute the provisional error with another vector multiplication. After the provisional stage, (ii) we repeat another orthogonalization with $i-1$ vector multiplications, (iii) one vector multiplication is used for the "Project" step, and (iv) the "Refine Co-Support" step uses p vector multiplications. Adding all these gives

$$\begin{aligned}&\text{Complexity}_{OBG} =\\ &= d \cdot \sum_{i=1}^{d-r} [(p-i+1)(i-1+1)+i-1+1+p] \qquad \text{(A-2)}\\ &= d \cdot \sum_{i=1}^{d-r} \left[(i+1)p - i^2 + 2i\right] = \mathcal{O}(d^3 p).\end{aligned}$$

Appendix B: Approximating the Learning Goal

Equation (17) defines the atom update step,

$$\begin{aligned}\left\{\hat{\mathbf{w}}_j,\ \hat{\mathbf{X}}_J\right\} = \underset{\mathbf{w}_j,\mathbf{X}_J}{\text{Argmin}}\ \|\mathbf{X}_J - \mathbf{Y}_J\|_F^2 \quad \text{Subject To}\\ \mathbf{\Omega}_i \mathbf{x}_i = 0, \quad \forall i \in J\\ \mathbf{w}_j^T \mathbf{X}_J = 0\\ \|\mathbf{w}_j\|_2 = 1\ .\end{aligned}$$

Our goal in this appendix is to show that there is a simpler form that approximates the solution of this problem. We start by eliminating $\mathbf{X}_J$ from this equation, by computing the closed-form solution to $\mathbf{X}_J$ for any fixed choice of $\mathbf{w}_j$. We note that given $\mathbf{w}_j$, the optimization can be carried out separately for each column $\mathbf{x}_i \in \mathbf{X}_J$,

$$\begin{aligned}\hat{\mathbf{x}}_i = \underset{\hat{\mathbf{x}}_i}{\text{Argmin}}\ \|\mathbf{x}_i - \mathbf{y}_i\|_2^2 \quad \text{Subject To} \quad \mathbf{\Omega}_i \mathbf{x}_i = 0 \quad \text{(B-1)}\\ \mathbf{w}_j^T \mathbf{x}_i = 0\ .\end{aligned}$$

The solution to this problem is the projection of $\mathbf{y}_i$ on the space orthogonal to the rows of $\mathbf{\Omega}_i$ and the atom $\mathbf{w}_j$. Since we wish to isolate the dependence of the solution $\hat{\mathbf{x}}_i$ on $\mathbf{w}_j$, we derive the expression of this projection in two steps. First, we apply the projection on the rows of $\mathbf{\Omega}_i$ as $\left(\mathbf{I} - \mathbf{\Omega}_i^\dagger \mathbf{\Omega}_i\right)\mathbf{y}_i = \mathbf{P}_i \mathbf{y}_i = \mathbf{y}_i^\perp$ (this operation is just like in the oracle setup, see Equation (5)). The matrix $\mathbf{P}_i$ is symmetric and nilpotent, i.e. for $k > 1$ $\mathbf{P}_i^k = \mathbf{P}_i$. If we apply a Gram-Schmidt orthogonalization of the rows in $\mathbf{\Omega}_i$, getting the set of vectors $\{\mathbf{q}_k\}_{k=1}^{d-r-1}$, this projection can be rewritten as

$$\mathbf{P}_i = \mathbf{I} - \sum_{k=1}^{d-r-1} \mathbf{q}_k \mathbf{q}_k^T.$$

The next step is to project $\mathbf{y}_i^\perp$ on $\mathbf{w}_j$. First, we orthogonalize it with respect to the previous rows, obtaining $\mathbf{q}_{d-r} = \mathbf{P}_i \mathbf{w}_j / \|\mathbf{P}_i \mathbf{w}_j\|_2$, and then project,

$$\begin{aligned}\hat{\mathbf{x}}_i &= \left(\mathbf{I} - \mathbf{q}_{d-r}\mathbf{q}_{d-r}^T\right)\mathbf{y}_i^\perp = \mathbf{y}_i^\perp - \frac{\mathbf{P}_i \mathbf{w}_j \mathbf{w}_j^T \mathbf{P}_i^T}{\|\mathbf{P}_i \mathbf{w}_j\|_2^2}\mathbf{y}_i^\perp \qquad \text{(B-2)}\\ &= \mathbf{y}_i^\perp - \left[\frac{\mathbf{w}_j^T \mathbf{y}_i^\perp}{\mathbf{w}_j^T \mathbf{P}_i \mathbf{w}_j}\right]\mathbf{P}_i \mathbf{w}_j.\end{aligned}$$

This equation defines the analytic solution for $\hat{\mathbf{x}}_i$ given $\mathbf{w}_j$. Substituting these solutions for $\mathbf{X}$ in Equation (17), we obtain the following optimization problem for the atom $\mathbf{w}_j$,

$$\begin{aligned}\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\text{Argmin}} \sum_{i \in J} \left\| \mathbf{y}_i - \mathbf{y}_i^\perp + \left[\frac{\mathbf{w}_j^T \mathbf{y}_i^\perp}{\mathbf{w}_j^T \mathbf{P}_i \mathbf{w}_j}\right]\mathbf{P}_i \mathbf{w}_j \right\|_2^2 \qquad \text{(B-3)}\\ \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1.\end{aligned}$$

Since $\mathbf{y}_i - \mathbf{y}_i^\perp$ is orthogonal to the third term, and constant in the optimization, we can reduce the minimization to

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\text{Argmin}} \sum_{i\in J} \left\| \left[\frac{\mathbf{w}_j^T\mathbf{y}_i^\perp}{\mathbf{w}_j^T\mathbf{P}_i\mathbf{w}_j}\right] \mathbf{P}_i\mathbf{w}_j \right\|_2^2 \qquad \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1 \ . \tag{B-4}$$

The target function in this minimization simplifies to

$$\begin{aligned} \left\| \left[\frac{\mathbf{w}_j^T\mathbf{y}_i^\perp}{\mathbf{w}_j^T\mathbf{P}_i\mathbf{w}_j}\right] \mathbf{P}_i\mathbf{w}_j \right\|_F^2 &= \frac{(\mathbf{w}_j^T\mathbf{y}_i^\perp)^2}{(\mathbf{w}_j^T\mathbf{P}_i\mathbf{w}_j)^2}\mathbf{w}_j^T\mathbf{P}_i^T\mathbf{P}_i\mathbf{w}_j \\ &= \frac{(\mathbf{w}_j^T\mathbf{y}_i^\perp)^2}{\mathbf{w}_j^T\mathbf{P}_i\mathbf{w}_j} \ . \end{aligned} \tag{B-5}$$

With Equation (B-5), the problem posed in Equation (B-4) finally becomes

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\text{Argmin}} \sum_{i\in J} \frac{(\mathbf{w}_j^T\mathbf{y}_i^\perp)^2}{\mathbf{w}_j^T\mathbf{P}_i\mathbf{w}_j} \qquad \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1. \tag{B-6}$$

The optimization problem (B-6) for $\mathbf{w}_j$ is clearly non-trivial, and does not lead to a closed-form solution for $\mathbf{w}_j$. One option is to work with this problem definition nevertheless, using classical optimization techniques for its solution (e.g. projected steepest-descent). Our tests show that, while this works reasonably well, a better and faster method is within reach, by approximating the above problem.

We observe that in the minimization, two forces are simultaneously acting on $\mathbf{w}_j$: The first force comes from the numerator, which pushes $\mathbf{w}_j$ to be orthogonal to $\mathbf{y}_i^\perp$ – the current solution without $\mathbf{w}_j$. At the same time, there is an important second force: The denominator tries to make $\mathbf{w}_j$ *orthogonal* to the rows in $\mathbf{\Omega}_i$, as a large denominator means $\mathbf{w}_j$ is far from the span of the rows of $\mathbf{\Omega}_i$. Thus, we see that this problem naturally incorporates a regularizing force that aims to "spread out" the rows in $\mathbf{\Omega}$.

Since we cannot efficiently solve the problem posed in Equation (B-6), we propose an alternative penalty that contains the above two forces,

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\text{Argmin}} \sum_{i\in J} \left\{ (\mathbf{w}_j^T\mathbf{y}_i^\perp)^2 + (\mathbf{w}_j^T\mathbf{y}_i^\|)^2 \right\} \qquad \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1, \tag{B-7}$$

where $\mathbf{y}_i^\| = \mathbf{y}_i - \mathbf{y}_i^\perp$. The first term is identical to the numerator in (B-6), forcing $\mathbf{w}_j$ to be orthogonal to the current solution $\mathbf{y}_i^\perp$. At the same time, the second term pushes $\mathbf{w}_j$ to be orthogonal to $\mathbf{y}_i^\|$, producing a similar effect to the denominator in (B-6). Therefore, our second option is to work with this new goal (Equation (B-7)) in seeking the update of $\mathbf{w}_j$. Note that the new problem is much simpler, with the need to find the eigenvector that corresponds to the smallest eigenvalue of the positive definite matrix $\sum_{i\in J}\left\{\mathbf{y}_i^\perp(\mathbf{y}_i^\perp)^T + \mathbf{y}_i^\|(\mathbf{y}_i^\|)^T\right\}$. Still, we have to decompose every example signal $\mathbf{y}_i$ into the two parts $\mathbf{y}_i^\perp$, $\mathbf{y}_i^\|$ as part of this computational process. There is yet another approximation step that can be taken to simplify further the above penalty, while preserving the desired forces. The simple observation

$$\begin{aligned} &(\mathbf{w}_j^T\mathbf{y}_i^\perp)^2 + (\mathbf{w}_j^T\mathbf{y}_i^\|)^2 + 2\mathbf{w}_j^T\mathbf{y}_i^\perp\mathbf{w}_j^T\mathbf{y}_i^\| = \\ &(\mathbf{w}_j^T\mathbf{y}_i^\perp + \mathbf{w}_j^T\mathbf{y}_i^\|)^2 = (\mathbf{w}_j^T\mathbf{y}_i)^2 \end{aligned} \tag{B-8}$$

suggests that, if we can neglect the contribution of the cross-term $2\mathbf{w}_j^T\mathbf{y}_i^\perp\mathbf{w}_j^T\mathbf{y}_i^\|$, then the penalty may admit a much simpler form. We found empirically that this is indeed justified, though further work is required to theoretically justify this step. Thus, we end up with the following, third and last option, for updating the analysis dictionary,

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\text{Argmin}} \ \|\mathbf{w}_j^T\mathbf{Y}_J\|_2^2 \quad \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1, \tag{B-9}$$

which has been mentioned in Equation (18).

Note that the problem posed in Equation (B-9) is equivalent to the problem

$$\{\hat{\mathbf{w}}_j, \hat{\mathbf{X}}_J\} = \underset{\mathbf{w}_j, \mathbf{X}_J}{\text{Argmin}} \ \|\mathbf{Y}_J - \mathbf{X}_J\|_F^2 \quad \text{Subject To} \quad \begin{aligned} &\mathbf{w}_j^T\mathbf{X}_J = 0 \\ &\|\mathbf{w}_j\|_2 = 1, \end{aligned} \tag{B-10}$$

which in turn is very close to the original problem we started with in Equation (17), with one delicate (yet important) difference – we have omitted the constraints $\mathbf{\Omega}_i\mathbf{x}_i = 0, \ \forall i \in J$, thus leading to a simple update rule for $\mathbf{w}_j$. This approximation is exact for $r = d-1$, meaning that each example is orthogonal to one row in $\mathbf{\Omega}$. For $r < d-1$, this process becomes inexact, though the approximation does optimize a related goal.

Before we conclude this discussion, we bring this last observation: Looking at Equation (B-10), we may be tempted to propose a small modification to the algorithm – A natural strategy, burrowed from the K-SVD methodology, would be to first orthogonalize each example $\mathbf{y}_i$ relative to all its selected rows from $\mathbf{\Omega}$ (excluding $\mathbf{w}_j$), and then solve

$$\{\hat{\mathbf{w}}_j, \hat{\mathbf{X}}_J\} = \underset{\mathbf{w}_j, \mathbf{X}_J}{\text{Argmin}} \ \|\mathbf{Y}_J^\perp - \mathbf{X}_J\|_F^2 \quad \text{Subject To} \quad \begin{aligned} &\mathbf{w}_j^T\mathbf{X}_J = 0 \\ &\|\mathbf{w}_j\|_2 = 1 \ . \end{aligned} \tag{B-11}$$

Here, the columns of $\mathbf{Y}_J^\perp$ are the orthogonalized signals $\{\mathbf{y}_i^\perp\}$, which essentially constitute a set of "residual signals" that use all but the current row $\mathbf{w}_j$. The solution is given by the left singular vector corresponding to the smallest singular value of $\mathbf{Y}_J^\perp$, which is also the solution to

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\text{Argmin}} \ \sum_{i\in J} \left\{ (\mathbf{w}_j^T\mathbf{y}_i^\perp)^2 \right\} \quad \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1. \tag{B-12}$$

Comparing this with (B-6), we see that this formulation completely lacks the second force of pushing $\mathbf{w}_j$ away from the rest of rows in $\mathbf{\Omega}$. Indeed, our experiments have found this approach to perform poorly compared to optimizing problem (B-9).

References

[1] A.M. Bruckstein, D.L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM Review*, vol. 51, no. 1, pp. 34–81, Feb. 2009.

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

[3] K. Engan, S. Aase, and J. Hakon-Husoy, "Method of optimal directions for frame design," in *Proceedings of ICASSP*, 1999, pp. 2443–2446.

[4] M. Aharon, M. Elad, and A.M. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[5] R. Rubinstein, A. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *IEEE Proceedings*, vol. 98, no. 6, pp. 1045–1057, April 2010.

[6] M. Elad, P. Milanfar, and R. Rubinstein, "Analysis versus synthesis in signal priors," *Inverse Problems*, vol. 23, no. 3, pp. 947–968, June 2007.

[7] E.J. Candes, Y.C. Eldar, D. Needell, and P. Randall, "Compressed sensing with coherent and redundant dictionaries," *Applied Computational Harmonic Analysis*, vol. 31, no. 1, pp. 59–73, July 2011.

[8] S. Nam, M. Davies, M. Elad, and R. Gribonval, "Cosparse analysis modeling – uniqueness and algorithms," in *Proceedings of ICASSP*, May 2010, pp. 5804–5807.

[9] S. Nam, M. Davies, M. Elad, and R. Gribonval, "The cosparse analysis model and algorithms," to appear in *Applied Computational Harmonic Analysis*.

[10] S. Nam, M.E. Davies, M. Elad, and R. Gribonval, "Cosparse analysis modeling," in *Proceedings of SAMPTA*, May 2011.

[11] R. Giryes, S. Nam, R. Gribonval, and M.E. Davies, "Iterative cosparse projection algorithms for the recovery of cosparse vectors," in *Proceedings of EUSIPCO*, Sep. 2011.

[12] S. Vaiter, G. Peyré, C. Dossal, and J. Fadili, "Robust sparse analysis regulatization," Technical Report HAL-00627452.

[13] B. Ophir, M. Elad, N. Bertin, and M.D. Plumbley, "Sequential minimal eigenvalues – an approach to analysis dictionary learning," in *Proceedings of EUSIPCO*, Sep. 2011.

[14] M. Yaghoobi, S. Nam, R. Gribonval, and M.E. Davies, "Analysis operator learning for overcomplete cosparse representations," in *Proceedings of EUSIPCO*, Sep. 2011.

[15] M. Yaghoobi, S. Nam, R. Gribonval, and M.E. Davies, "Noise aware analysis operator learning for approximately cosparse signals," in *Proceedings of ICASSP*, March 2012.

[16] G. Peyré and J. Fadili, "Learning analysis sparsity priors," in *Proceedings of SAMPTA*, May 2011.

[17] S. Roth and M.J. Black, "Fields of experts: A framework for learning image priors," in *Proceedings of CVPR*, June 2005, vol. 2, pp. 860–867.

[18] S. Roth and M.J. Black, "Fields of experts," *International Journal of Computer Vision (IJCV)*, vol. 82, no. 2, pp. 205–229, April 2009.

[19] G. Hinton and Y.W. Teh, "Discovering multiple constraints that are frequently approximately satisfied," in *UAI*, 2001.

[20] M. Welling, G. Hinton, and S. Osindero, "Learning sparse topographic representations with products of student-t distributions," in *Advances in Neural Information Processing Systems, 15*, 2003.

[21] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[22] S. Hawe, M. Kleinsteuber, and K. Diepold, "Analysis operator learning and its application to image reconstruction," submitted to *IEEE Trans. Image Processing*.

[23] Y.M. Lu and M.N. Do, "A theory for sampling signals from a union of subspaces," *IEEE Trans. on Signal Processing*, vol. 56, no. 6, pp. 2334–2345, June 2008.

[24] S.S. Chen, D.L. Donoho, and M.A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, pp. 33–61, 1998.

[25] I.F. Gorodnitsky and B.D. Rao, "Sparse signal reconstruction from limited data using focuss: A re-weighted norm minimization algorithm," *IEEE Trans. Signal Processing*, vol. 45, pp. 600–616, 1997.

[26] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Processing*, vol. 41, pp. 3397–3415, 1999.

[27] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proceedings of the 27th Asilomar Conference on Signals, Systems and Computers*, 1993, vol. 1, pp. 40–44.

[28] L. Rebollo-Neira and D. Lowe, "Optimized orthogonal matching pursuit approach," *IEEE Signal Processing Letters*, vol. 9, no. 4, pp. 137–140, April 2002.

[29] J. Dahl, P.C. Hansen, S.H. Jensen, and T.L. Jensen, "Algorithms and software for total-variation image reconstruction via first-order methods," *Numerical Algorithms*, vol. 53, no. 1, pp. 67–92, 2010.

[30] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Trans. Image Processing*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[31] T. Peleg and M. Elad, "Performance guarantees of the thresholding algorithm for the co-sparse analysis model," to appear in *IEEE Trans. Info. Theory*.

Ron Rubinstein received his Ph.D. degree in Computer Science from the The Technion – Israel Institute of Technology in 2012. His research interests include sparse and redundant representations, analysis and synthesis signal priors, and example-based signal modeling. Ron currently holds a research scientist position at Intel, and is an adjunct lecturer at the Technion. Ron participated in the IPAM Short Course on Sparse Representation in May 2007. During 2008 he participated as a research intern in Sharp Laboratories of America, Camas, WA. Ron has received the Technion Teaching Assistant's excellence award in 2005 and Lecturer's commendation for excellence in 2008, 2010 and 2011.

Tomer Peleg received the B.Sc. degree in electrical engineering (*summa cum laude*) and the B.Sc. degree in physics (*summa cum laude*) both from the Technion, Haifa, Israel, in 2009. He is currently pursuing the Ph.D. degree in electrical engineering at the Technion. From 2007 to 2009, he worked at RAFAEL Research Laboratories, Israel Ministry of Defense. His research interests include statistical signal processing, sparse representations, image processing, inverse problems and graphical models. Since 2012 he holds a fellowship in the Azrieli program for outstanding Israeli graduate students.

Michael Elad (M'98-SM'08-F'12) received his B.Sc. (1986), M.Sc. (1988) and D.Sc. (1997) from the department of Electrical engineering at the Technion, Israel. Since 2003 Michael is a faculty member at the Computer-Science department at the Technion, and since 2010 he holds a full-professorship position.

Michael Elad works in the field of signal and image processing, specializing in particular on inverse problems, sparse representations and super-resolution. Michael received the Technion's best lecturer award six times, he is the recipient of the 2007 Solomon Simon Mani award for excellence in teaching, the 2008 Henri Taub Prize for academic excellence, and the 2010 Hershel-Rich prize for innovation. Michael is an IEEE Fellow since 2012. He is serving as an associate editor for SIAM SIIMS, IEEE-TIT, and ACHA. Michael is also serving as a senior editor for IEEE SPL.