K-SVD DICTIONARY-LEARNING FOR THE ANALYSIS SPARSE MODEL

Ron Rubinstein, Tomer Faktor and Michael Elad

Departments of Computer Science and Electrical Engineering
Technion – Israel Institute of Technology, Haifa 32000, Israel
{ronrubin@cs, tomerfa@tx, elad@cs}.technion.ac.il

ABSTRACT

The synthesis-based sparse representation model for signals has drawn a 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 Dictionary* multiplies the signal, leading to a sparse outcome. Our goal is to learn the analysis dictionary from a set of signal examples, and 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, which include two greedy tailored pursuit algorithms and a penalty function for the dictionary update stage. We demonstrate its effectiveness in several experiments, showing a successful and meaningful recovery of the analysis dictionary.

Index Terms— Sparse Representations, Analysis Model, Backward Greedy (BG) Pursuit, Dictionary Learning, K-SVD.

1. INTRODUCTION

Signal models are fundamental for handling various processing tasks, such as denoising, solving inverse problems, compression, sampling, and more. A very popular approach to signal modeling is the synthesis-based sparse representation model, where a signal $\mathbf{x} \in \mathbb{R}^d$ is assumed to be composed as a linear combination of a *few* atoms from a given dictionary $\mathbf{D} \in \mathbb{R}^{d \times n}$ [1, 2]. The main activity in studying this model concentrated so far on estimating the representation from a corrupted signal and learning the dictionary $\mathbf{D}$ from signal examples. A popular technique for dictionary learning is the K-SVD algorithm [3], which leads to state-of-the-art results in various image processing applications [2].

While the *synthesis* model has been intensively studied, there is an *analysis* viewpoint to sparse representations that has been left aside almost untouched [4]. The analysis model relies on a linear operator (a matrix) $\mathbf{\Omega} \in \mathbb{R}^{p \times d}$, which we will refer to as the analysis dictionary. The key property of this model is our expectation that the analysis representation vector $\mathbf{\Omega x} \in \mathbb{R}^p$ is supposed to be sparse with ℓ zeros, and these zeros define the subspace this signal belongs to. While this may sound similar to the synthesis counterpart approach, it is in fact very different when dealing with a redundant dictionary, which means that $p > d$.

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 this trend has started to change [5–9]. In this paper we focus on the analysis model and concentrate on the development of an algorithm that would learn a possibly redundant analysis dictionary Ω from a set of signal examples. The objective is to find a suitable dictionary Ω so that the analysis coefficients for all the signals in the training set are adequately sparse.

Analysis dictionary training is a challenging problem in signal modeling, with origins in the pioneering work of Black and Roth [10], which trained an analysis sparsity-based image prior. Note however that their approach assumes an underdetermined dictionary ($p < d$), in contrast to our work. The problem of learning an analysis dictionary has already started to attract attention [8, 9, 11]. The authors of [8] suggest to 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. Their approach relies heavily on a randomized initialization strategy and loses its efficiency rapidly as the signal dimension d grows. The work reported in [9] poses the task of learning $\mathbf{\Omega}$ as a constrained optimization problem. The approach suggested there puts a rather arbitrary constraint for regularizing the learning problem – the dictionary is constrained to be a uniform normalized tight frame – limiting the possible $\mathbf{\Omega}$ to be learned. Finally, in [11] analysis dictionary learning is formulated as a bilevel-programming optimization problem.

In this paper we adopt an approach that is based directly on ℓ^0 sparsity. This exact sparsity measure distinguishes our work from previous ones and allows us the development of an efficient training algorithm, which is parallel to the synthesis-model K-SVD in its rationale and computational steps.

2. A CLOSER LOOK AT THE ANALYSIS MODEL

In the synthesis model the representation $\boldsymbol{\alpha}$ is obtained by a complex and non-linear pursuit process that seeks the sparsest solution to the linear system of equations $\mathbf{D}\boldsymbol{\alpha} = \mathbf{x}$. This representation can be arbitrarily sparse, $\|\boldsymbol{\alpha}\|_0 = k \ll d$ and its support (the k nonzero indices) describes the atoms constructing the signal $\mathbf{x}$, which in turn define the subspace this signal belongs to.

In contrast, in the analysis model the computation of the representation is trivial, obtained by multiplying $\mathbf{x}$ by the possibly redundant analysis dictionary $\mathbf{\Omega} : \mathbb{R}^d \to \mathbb{R}^p$. In this model we put an emphasis on the locations of the zeros in the vector $\mathbf{\Omega x}$, and define the *co-sparsity* ℓ as the number of these zeros, so that $\|\mathbf{\Omega x}\|_0 = p - \ell$. Similarly, we define the *co-support* Λ of the signal $\mathbf{x}$ as the set of rows in $\mathbf{\Omega}$ that are orthogonal to it, namely $\mathbf{\Omega}_\Lambda \mathbf{x} = 0$, where $\mathbf{\Omega}_\Lambda$ is a sub-matrix of $\mathbf{\Omega}$ that contains only the rows indexed in Λ. The signal $\mathbf{x}$ is characterized by its co-support, since it defines the subspace the signal belongs to.

The analysis model assumes that the analysis representation vector $\mathbf{\Omega x}$ should be sparse. How sparse can it be? To answer this

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

question, 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]. In this case we necessarily have $\|\mathbf{\Omega x}\|_0 \geq p-d$, since otherwise there would be d or more rows that are orthogonal to $\mathbf{x}$, implying $\mathbf{x}=0$.

In general, $\mathbf{\Omega}^T$ may have non-full spark. The immediate implication in such cases is that ℓ could go beyond d, and yet the signal would not necessarily be nulled, since the co-support rows would span a subspace that do not cover the complete $\mathbb{R}^d$. An example of such a dictionary is the vertical concatenation of cyclic horizontal and vertical one-sided derivatives, applied on a 2D signal of size $\sqrt{d} \times \sqrt{d}$. Such a dictionary, denoted $\mathbf{\Omega}_{DIF}$, is of size $2d \times d$, thus twice redundant [6, 7].

One can generate an analysis signal, and this is done in the following way: Choose ℓ rows from $\mathbf{\Omega}$ at random - this will be the signal's co-support. Starting with a random vector $\mathbf{u} \sim N(0, \mathbf{I})$, project it onto the subspace orthogonal to $\mathbf{\Omega}_\Lambda$, $\mathbf{x} = (\mathbf{I} - \mathbf{\Omega}_\Lambda^+ \mathbf{\Omega}_\Lambda)\mathbf{u}$, and $\mathbf{x}$ is an analysis signal that satisfies our basic assumptions. In the experiments that follow we shall use such randomly generated signals, when dealing with a synthetic setup (see Section 5).

3. ANALYSIS SPARSE-CODING

Before we study the problem of learning the analysis dictionary $\mathbf{\Omega}$, we have to consider a simpler task called *Analysis Sparse-Coding*. As we shall see in the next section, this is an important building block in the overall 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, we assume that the given signal $\mathbf{y}$ is contaminated, $\mathbf{y} = \mathbf{x} + \mathbf{v}$, where $\mathbf{v}$ is a zero-mean white-Gaussian additive noise. We aim at recovering the clean signal $\mathbf{x}$, and this requires solving a problem of the form

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{Argmin}} \ \|\mathbf{x}-\mathbf{y}\|_2 \quad \text{Subject To} \quad \|\mathbf{\Omega x}\|_0 \leq p-\ell. \tag{1}$$

or

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{Argmin}} \ \|\mathbf{\Omega x}\|_0 \quad \text{Subject To} \quad \|\mathbf{x}-\mathbf{y}\|_2 \leq \epsilon. \tag{2}$$

Here ϵ is the error tolerance, 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 $\mathbf{x}$. In principle, denoising is possible with the analysis model because, once the co-support has been detected, projection onto the complement subspace attenuates the additive noise in the co-support subspace, thus cleaning the signal.

The above-mentioned two problems are equivalent, of course, given the correct correspondence between ℓ 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 the analysis-pursuit. Similar to the synthesis sparse approximation problem, problems (1) and (2) are combinatorial in nature and can thus only be approximated in general. For simplicity, in the following we focus on formulation (1), though the techniques are equally applicable to (2).

One approach to approximating the solution is to relax the ℓ^0 norm and replace it with an ℓ^1 penalty function. This approach is parallel to the basis-pursuit approach for synthesis approximation [1]. A second approach parallels the synthesis greedy pursuit algorithms [1], 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 $\mathbf{\Omega x}$, or by detecting the zeros. The first approach is the one taken by the Greedy-Analysis-Pursuit (GAP) algorithm, described in [7]. We shall take the alternative 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 concentrating on the zeros in the representation. A detailed description of this algorithm, is given in Algorithm 1.

Algorithm 1 BACKWARD-GREEDY-ALGORITHM

1: Input: Analysis Dictionary $\mathbf{\Omega} \in \mathbb{R}^{p \times d}$, signal $\mathbf{y} \in \mathbb{R}^d$, and target co-sparsity ℓ
2: Output: Signal $\hat{\mathbf{x}} \in \mathbb{R}^d$ satisfying $\|\mathbf{\Omega}\hat{\mathbf{x}}\|_0 = p - \ell$ and minimizing $\|\mathbf{y} - \hat{\mathbf{x}}\|_2$
3: Initialization: Set $i = 0$, co-support set $\Lambda_i := \emptyset$, $\hat{\mathbf{x}}_i := \mathbf{y}$
4: **for** $i = 1 \ \ldots \ \ell$ **do**
5: $\quad \hat{k}_i := \underset{k \notin \Lambda_{i-1}}{\text{Argmin}} \ | \mathbf{w}_k^T \hat{\mathbf{x}}_{i-1} |$
6: $\quad \Lambda_i := \Lambda_{i-1} \cup \{ \hat{k}_i \}$
7: $\quad \hat{\mathbf{x}}_i := \left[\mathbf{I} - \mathbf{\Omega}_{\Lambda_i}^+ \mathbf{\Omega}_{\Lambda_i} \right] \mathbf{y}$
8: **end for**
9: **return** $\hat{\mathbf{x}}_\ell$

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 for all the rows not indexed in the co-support, and the row with the *smallest* 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. This process repeats until the target sparsity is achieved. Alternatively, this process may proceed until the error $\|\hat{\mathbf{x}}_i - \mathbf{y}\|_2$ *exceeds* a pre-specified threshold.

In practice, the matrix inversion in Step 7 of the above algorithm (updating $\hat{\mathbf{x}}_i$) can be avoided and this step 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 $\mathbf{w}_{\hat{k}_i}$ is about to join the co-support, it is first orthogonalized with respect to the already accumulated rows using a 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

$$\mathbf{q}_i = \mathbf{w}_{\hat{k}_i}^T - \sum_{j=1}^{i-1} (\mathbf{q}_j^T \mathbf{w}_{\hat{k}_i}) \mathbf{q}_j. \tag{3}$$

This is followed by normalization of this vector, $\mathbf{q}_i = \mathbf{q}_i / \|\mathbf{q}_i\|_2$.[1] Consequently, Step 7 in the algorithm translates comfortably to

$$\hat{\mathbf{x}}_i = \left[\mathbf{I} - \sum_{j=1}^{i} \mathbf{q}_j \mathbf{q}_j^T \right] \mathbf{y} = \hat{\mathbf{x}}_{i-1} - \mathbf{q}_i \mathbf{q}_i^T \mathbf{y} = \left[\mathbf{I} - \mathbf{q}_i \mathbf{q}_i^T \right] \hat{\mathbf{x}}_{i-1}. \tag{4}$$

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

We can propose the following improvement to Algorithm 1: At the i^{th} iteration, rather than choosing the index $k \notin \Lambda_{i-1}$ that

[1] When dealing with analysis dictionaries that may have non-full spark, this normalization should be done with care, as it may happen that $\mathbf{q}_i$ emerging from (3) is zero, reflecting the fact that it is spanned by the existing set. In such a case, the row $\mathbf{w}_{\hat{k}_i}$ should be simply omitted as it contributes nothing to the accumulated basis.

minimizes $|\mathbf{w}_k^T \hat{\mathbf{x}}_{i-1}|$, we can test all of these possible indices, and per each perform the complete update (steps 6 and 7) in Algorithm 1. Then we choose $\hat{k}_i$ that leads to the smallest decrease in the signal's energy, namely the one that minimizes $\|\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_{i-1}\|_2$. This should remind the reader of the Least-Squares OMP algorithm, as described in [2]. We shall refer hereafter to this algorithm as the Optimized-BG (OBG). Using (4) we get that Step 5 in Algorithm 1 should be replaced in the OBG algorithm by:

$$\hat{k}_i := \underset{k \notin \Lambda_{i-1}}{\text{Argmin}} \; |(\mathbf{q}_i^{(k)})^T \hat{\mathbf{x}}_{i-1}|, \tag{5}$$

where $\mathbf{q}_i^{(k)}$ is the orthogonalized and normalized vector corresponding to $\mathbf{w}_k$ at the i^{th} iteration. The computational complexity of the two pursuit algorithms, using their efficient implementations discussed above, is $O(\ell dp)$ for BG and is $O(\ell^2 dp)$ for OBG.

4. THE ANALYSIS K-SVD ALGORITHM

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 orthogonal to ℓ rows from 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 $\|\mathbf{\Omega}\mathbf{x}_i\|_0 = p - \ell$. For simplicity we shall assume that all these signals have the same co-sparsity ℓ, 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 for the learning process – it is given by

$$\left\{\hat{\mathbf{\Omega}}, \hat{\mathbf{X}}\right\} = \underset{\mathbf{\Omega}, \mathbf{X}}{\text{Argmin}} \; \|\mathbf{X} - \mathbf{Y}\|_F^2 \quad \text{Subject To} \quad \|\mathbf{\Omega}\mathbf{x}_i\|_0 \le p - \ell, \;\; \forall 1 \le i \le R \quad \|\mathbf{w}_j\|_2 = 1, \;\; \forall 1 \le j \le p. \tag{6}$$

Here, $\mathbf{x}_i$ are our estimates of the noiseless signals, arranged as the columns of the matrix $\mathbf{X}$. 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 [3]. Indeed, the training problem posed in equation (6) is highly non-convex, and as such we clearly cannot hope to find its global solution in general. Instead, we adopt a simple iterative approximation method, described next, which draws its spirit from earlier work on algorithms for synthesis dictionary learning.

Assuming an initial estimate $\mathbf{\Omega}_0$ 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 $\hat{\mathbf{X}}$. The process repeats until some stopping criterion (typically a fixed number of iterations) is achieved.

Given the current estimate of the analysis dictionary $\mathbf{\Omega}$, optimizing for $\mathbf{X}$ can be done individually for each column of $\mathbf{X}$, defining an ordinary ℓ^0 analysis pursuit problem for each signal $\mathbf{y}_i$, similar to (1). More specifically, the "sparse-coding" stage in the iterative scheme consists of finding for each signal the ℓ rows in $\mathbf{\Omega}$ that are most orthogonal to it. Once $\hat{\mathbf{X}}$ is computed, we turn to update $\mathbf{\Omega}$ in the second step. The optimization is carried out sequentially for each of the rows $\mathbf{w}_j$ in $\mathbf{\Omega}$. We suggest exploiting the fact that a considerable set of examples is expected to be nearly orthogonal to $\mathbf{w}_j$, so that this row can be updated as the one that is most orthogonal to all the signals that were found to be orthogonal to it in the "sparse-coding" stage. Denoting the set of columns in $\hat{\mathbf{X}}$ that were found to be orthogonal to $\mathbf{w}_j$ by J and letting $\mathbf{Y}_J$ be the sub-matrix of $\mathbf{Y}$ containing the columns indexed in J, the update step for $\mathbf{w}_j$ can be written as

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\text{Argmin}} \; \|\mathbf{w}_j^T \mathbf{Y}_J\|_2^2 \quad \text{Subject To} \quad \|\mathbf{w}_j\|_2 = 1, \tag{7}$$

which is a simple SVD problem: the eigenvector that corresponds to the smallest eigenvalue of these examples' autocorrelation matrix is the desired row. Note that this is the same atom update rule as in [8]. The proposed algorithm thus iterates between the computation of this row from the subset of chosen examples, and an update of this subset using greedy analysis pursuit that is based on an exact ℓ^0 sparsity measure.

One advantage of this specific approximation method is that it disjoints the updates of the rows in $\mathbf{\Omega}$, enabling all rows to be updated in parallel. Another desirable property of the resulting algorithm is that it has 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 this resemblance. The full algorithm is detailed in Algorithm 2.

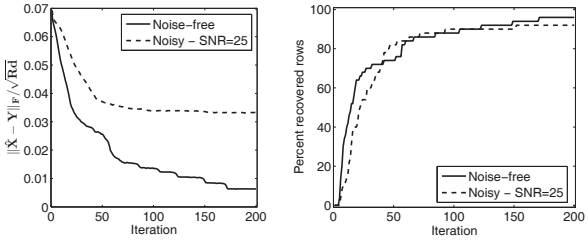
Algorithm 2 ANALYSIS K-SVD

1: Input: Training signals $\mathbf{Y} \in \mathbb{R}^{d \times R}$, initial dictionary $\mathbf{\Omega}_0 \in \mathbb{R}^{p \times d}$, target sparsity ℓ, number of iterations k
2: Output: Dictionary $\mathbf{\Omega}$ and signal set $\hat{\mathbf{X}}$ minimizing (6)
3: Initialization: Set $\mathbf{\Omega} := \mathbf{\Omega}_0$
4: **for** $n = 1 \; \ldots \; k$ **do**
5: $\quad \hat{\mathbf{x}}_i := \underset{\mathbf{x}}{\text{Argmin}} \|\mathbf{y}_i - \mathbf{x}\|_2^2$ Subject To $\|\mathbf{\Omega}\mathbf{x}\|_0 \le p - \ell, \; \forall i$
6: $\quad$ **for** $j = 1 \; \ldots \; p$ **do**
7: $\quad\quad J :=$ *{indices of the columns of* $\hat{\mathbf{X}}$ *orthogonal to* $\mathbf{w}_j$*}*
8: $\quad\quad \hat{\mathbf{w}}_j := \underset{\mathbf{w}}{\text{Argmin}} \|\mathbf{w}^T \mathbf{Y}_J\|_2 \quad \text{Subject To} \quad \|\mathbf{w}\|_2 = 1$
9: $\quad\quad \mathbf{\Omega}\{j\text{-th row}\} := \mathbf{w}_j^T$
10: $\quad$ **end for**
11: **end for**

We can suggest the following improvement to Algorithm 2, intended to surpass deadlock situations where the iterative process becomes stuck at local minimum: In each iteration we check for each row in $\mathbf{\Omega}$ how many signals were found to be orthogonal to it and how much this row has changed in the update stage. A row that remains almost unchanged and is orthogonal to relatively few signals is regarded as a false one and is therefore replaced by another row which is generated in a random fashion. One possible generation process for these rows is described in Section 5, when relating to the initial dictionary.

5. SIMULATION RESULTS

We now present a set of experimental results with the proposed training algorithm. We start with a synthetic setup, where the analysis dictionary is known and a set of signal examples with a known co-sparsity are generated as described in Section 2. These sparse analysis signals are normalized to unit length and are optionally subjected to additive white Gaussian noise, producing the final training set. We performed experiments for a dictionary $\mathbf{\Omega} \in$



(a) Error evolution (b) Recovered rows

Fig. 1. Synthetic experiment results of the Analysis K-SVD algorithm for a random dictionary $\mathbf{\Omega} \in \mathbb{R}^{50\times 25}$ and a training set of $R = 50,000$ analysis signals with co-sparsity $\ell = 21$.

$\mathbb{R}^{50\times 25}$ that is generated with random Gaussian entries (note that the rows of this dictionary are in a general position), and a training set of $R = 50,000$ examples with co-sparsity $\ell = 21$. We tested both the noise-free setup and a noisy setup with noise level $\sigma = 0.2/\sqrt{d} = 0.04$, which corresponds to a signal-to-noise ratio of 25.

Each row in the initial dictionary is constructed by randomly selecting a set of $d-1$ examples and computing the vector that spans their one-dimensional null-space. We apply 200 iterations of the Analysis K-SVD algorithm using the OBG algorithm with a target co-sparsity $\ell = 21$ for the "sparse-coding" stage and the improvement mentioned in the end of Section 4. A row $\mathbf{w}_j$ in the true dictionary $\mathbf{\Omega}$ is regarded as recovered if

$$\underset{i}{\text{Min}}(1 - |\hat{\mathbf{w}}_i^T \mathbf{w}_j|) < 0.01, \tag{8}$$

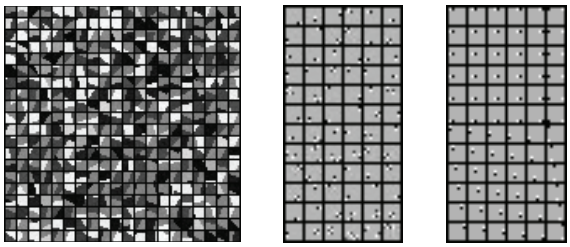
where $\hat{\mathbf{w}}_i$ are the atoms of the trained dictionary. A similar success criterion was used for the synthesis K-SVD [3].

Example results are shown in Figure 1, demonstrating the ability of the proposed approach to obtain a good recovery of the true underlying operator $\mathbf{\Omega}$ given a sparse analysis training set, and its robustness to noise: 96% of the rows in the true $\mathbf{\Omega}$ were recovered in the noise-free setup and 92% in the noisy setup. Note that at the end of the training for the noisy setup, the error per element $\|\hat{\mathbf{X}} - \mathbf{Y}\|_F/\sqrt{Rd}$ goes below the noise level σ, indicating a successful training.

Next, we train an analysis dictionary using $R = 10,000$ patches of size 6×6 extracted from a piece-wise constant image contaminated by additive white Gaussian noise with $\sigma = 5$ (several patch examples are shown in Fig. 2 on the left). We apply 50 iterations of the Analysis K-SVD algorithm with a target co-sparsity $\ell = 32$. The BG algorithm is used in the 30 first iterations and the OBG is used in the remaining iterations. We can observe from the results shown in Fig. 2 that the trained analysis dictionary exhibits a high resemblance to the $\mathbf{\Omega}_{DIF}$ dictionary mentioned in Section 2. This observation aligns with our intuition that many finite differences computed on a piece-wise constant signal (in our case - a 2D patch) are expected to be near zero.

6. CONCLUSIONS

In this work we present an efficient algorithm for learning an 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 succesful recovery of the true analysis dictionary in a synthetic setup and observing the emergence of meaningful structures in the trained dictionary for a setup where there is no "true" reference dictionary. In this work we do not address specific applications for the analysis model and its dictionary learning, as our main goal is to introduce the core approach for obtaining the analysis dictionary from a given data-set of examples. It remains to be seen which applications would benefit the most from this model.



(a) Patch examples (b) $\hat{\mathbf{\Omega}}$ (c) $\mathbf{\Omega}_{DIF}$

Fig. 2. Traning results for $R = 10,000$ patches of size 6×6 extratced from a noisy piece-wise constant image ($\sigma = 5$).

7. 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, 2009.

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

[3] 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, 2006.

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

[5] E.J. Candes, Y.C. Eldar, D. Needell, and P. Randall, "Compressed sensing with coherent and redundant dictionaries," *Appl. Comput. Harmon. Anal.*, vol. 31, no. 1, pp. 59–73, 2011.

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

[7] S. Nam, M.E. Davies, M. Elad, and R. Gribonval, "The cosaprse analysis model and algorithms," submitted to *Appl. Comput. Harmon. Anal.*

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

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

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

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