Dictionary Learning for Analysis-Synthesis Thresholding

Ron Rubinstein, *Member, IEEE*, and Michael Elad, *Fellow, IEEE*

***Abstract*—Thresholding is a classical technique for signal denoising. In this process, a noisy signal is decomposed over an orthogonal or overcomplete dictionary, the smallest coefficients are nullified, and the transform pseudo-inverse is applied to produce an estimate of the noiseless signal. The dictionaries used is this process are typically fixed dictionaries such as the DCT or Wavelet dictionaries. In this work, we propose a method for incorporating adaptive, trained dictionaries in the thresholding process. We present a generalization of the basic process which utilizes a pair of overcomplete dictionaries, and can be applied to a wider range of recovery tasks. The two dictionaries are associated with the analysis and synthesis stages of the algorithm, and we thus name the process *analysis-synthesis thresholding*. The proposed training method trains both the dictionaries and threshold values simultaneously given examples of original and degraded signals, and does not require an explicit model of the degradation. Experiments with small-kernel image deblurring demonstrate the ability of our method to favorably compete with dedicated deconvolution processes, using a simple, fast, and parameterless recovery process.**

***Index Terms*—Analysis dictionary learning, signal deblurring, sparse representation, thresholding.**

I. Introduction

SPARSE representation of signals is a widely-used technique for modeling natural signals, with applications ranging from denoising, deconvolution, and general inverse problem solution, through compression, detection, classification, and more [1]. Such techniques typically take a *synthesis* approach, where the underlying signal is described as a sparse combination of signals from a prespecified dictionary. Specifically, $\mathbf{x} = \mathbf{D}\boldsymbol{\gamma}$, where $\mathbf{x} \in \mathbb{R}^N$ is the original signal, $\mathbf{D} \in \mathbb{R}^{N \times L}$ is a dictionary with atom signals as its columns, and $\boldsymbol{\gamma} \in \mathbb{R}^L$ is the sparse synthesis representation, assumed to contain mostly zeros. The dictionary $\mathbf{D}$ is generally overcomplete ($L \geq N$), a property known to improve expressiveness and increase sparsity. The cardinality of $\boldsymbol{\gamma}$, i.e., the number of non-zeros in this vector, is denoted by $\|\boldsymbol{\gamma}\|_0$. We assume that $\|\boldsymbol{\gamma}\|_0$ is substantially smaller than N, the signal dimension, implying that the model produces an effective dimensionality reduction.

Recently, a new *analysis* sparse representation approach has been proposed, and is attracting increasing attention [2], [3]. This new model considers a dictionary $\boldsymbol{\Omega} \in \mathbb{R}^{L \times N}$ with atom signals as its *rows*, and aims to sparsify the analysis representation of the signal, $\boldsymbol{\gamma} = \boldsymbol{\Omega}\mathbf{x}$, where $\boldsymbol{\gamma}$ has many vanishing coefficients (i.e., $\|\boldsymbol{\Omega}\mathbf{x}\|_0$ is small). Recently produced theoretical results reveal intriguing relations between this model and its synthesis counterpart [3], [4], and research in this direction is still ongoing. More on this matter appears in Section II below.

While the analysis and synthesis sparsity-based models are conceptually different, both share a common ancestor known as *thresholding*. This classical technique, originally proposed for Wavelet-based denoising [5], utilizes an orthogonal or overcomplete analysis dictionary $\boldsymbol{\Omega}$, and suggests a process of the form

$$\hat{\mathbf{x}} = \boldsymbol{\Omega}^{\dagger} S_{\lambda}(\boldsymbol{\Omega}\mathbf{y}). \tag{1}$$

Here, $\mathbf{y}$ is the noisy signal, $\boldsymbol{\Omega}^{\dagger}$ is the dictionary pseudo-inverse, and S_{λ} is an element-wise attenuation of the analysis coefficients governed by λ (which should be carefully selected based on the noise power). Typical choices for $\boldsymbol{\Omega}$ include the DCT, Wavelet, Curvelet, and Contourlet dictionaries, among others [6].

As we show in the next section, the thresholding process is essentially optimal for finding the sparsest representation, if the dictionary (analysis or synthesis) is orthogonal [5]. In contrast, for overccomplete dictionaries, this method is less rigorously justified, though more recent works have provided important clues as to the origins of its success (see e.g., [7]).

Several common choices for the operator S_{λ} exist. A particular widely-used choice, which is also closely related to the analysis and synthesis sparse representation models, is the hard thresholding operator $S_{\lambda}(\alpha) = \alpha \cdot \mathbf{1}(|\alpha| \geq \lambda)$. This operator nullifies the smallest coefficients in $\boldsymbol{\Omega}\mathbf{y}$, essentially performing an ℓ^0 sparsification of the analysis coefficients. At the same time, by multiplying with the synthesis dictionary $\boldsymbol{\Omega}^{\dagger}$ the process can also be viewed as a form of synthesis sparse representation, and thus, it is essentially a hybrid model. This duality has initially led to some confusion between the two approaches, though the distinction between them is now widely recognized [2].

A. This Work

The nature of the hard thresholding process as a combination of analysis and synthesis models makes it an appealing target for research. In this work we consider specifically the task of

Manuscript received April 17, 2013; revised November 20, 2013 and September 11, 2014; accepted September 19, 2014. Date of publication September 23, 2014; date of current version October 24, 2014. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Zhi-Quan (Tom) Luo. This work was supported by the European Community FP7- ERC program, Grant agreement no. 320649.

The authors are with the Department of Computer Science, The Technion—Israel Institute of Technology, Haifa 32000, Israel.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2014.2360157

training a dictionary for (1). This task is closely related to the ℓ^0 analysis dictionary learning problem [8]–[12], though the optimization in our case focuses on recovery performance rather than on model fitting. Similar task-driven approaches have been recently employed for ℓ^1 analysis dictionary training [13], [14], though the non-explicit form of the analysis estimator in these cases leads to a more complex bi-level optimization problem.

This paper is organized as follows: We begin in Section II with more background on the synthesis and the analysis sparsity-inspired models and their inter-relations. We introduce the role of the thresholding algorithm in both these models, and discuss its optimality. All this leads to the definition of a simple generalization of the process posed in (1), which disjoins the analysis and synthesis dictionaries, and thus allows more general inverse problems to be solved. We continue in Section III by presenting our dictionary learning algorithm for this generalized thresholding process, and apply it to small-kernel image deblurring in Section IV. In Section V we discuss a few specific related works, focusing on a family of machine-learning techniques which employ formulations similar to (1) for the task of unsupervised feature training. We summarize and conclude in Section VI.

II. Analysis-Synthesis Thresholding

A. Synthesis, Analysis, and Thresholding

Sparsity-inspired models have proven valuable in signal and image processing, due to their ability to reduce the dimension of the underlying data representation. These models are universal in the sense that they may fit to various and diverse sources of information. In the following we shall briefly introduce the synthesis and analysis sparsity-based models, their relationships, and then discuss the thresholding algorithm, which serves them both. More on these topics can be found in [1]–[3].

In its core form, a sparsity-based model states that a given signal $\mathbf{x} \in \mathbb{R}^N$ is believed to be created as a linear combination of *few* atoms from a pre-specified dictionary, $\mathbf{D} \in \mathbb{R}^{N \times L}$. This is encapsulated by the relation $\mathbf{x} = \mathbf{D}\boldsymbol{\gamma}$, where $\boldsymbol{\gamma} \in \mathbb{R}^L$ is the signal's representation, assumed to be k-sparse, i.e., $\|\boldsymbol{\gamma}\|_0 = k \ll N$. This model is referred to as the synthesis approach, since the signal is synthesized by the relation $\mathbf{x} = \mathbf{D}\boldsymbol{\gamma}$.

Given a noisy version of the original signal, $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where $\mathbf{n}$ stands for an additive white Gaussian noise, denoising of $\mathbf{y}$ amounts to a search for the k-sparse representation vector $\boldsymbol{\gamma}$ such that $\|\mathbf{y} - \mathbf{D}\boldsymbol{\gamma}\|_2$ is minimized, i.e.,

$$\hat{\boldsymbol{\gamma}} = \underset{\boldsymbol{\gamma}}{\text{Argmin}} \|\mathbf{y} - \mathbf{D}\boldsymbol{\gamma}\|_2 \quad \text{Subject To } \|\boldsymbol{\gamma}\|_0 = k. \tag{2}$$

Once the representation is found, the denoised signal is obtained by $\hat{\mathbf{x}} = \mathbf{D}\hat{\boldsymbol{\gamma}}$. If the true support (non-zero locations) of the representation has been found in $\hat{\boldsymbol{\gamma}}$, the noise in the resulting signal is reduced by a factor of k/N, implying a highly effective noise attenuation.

The problem posed in (2) is referred to as a pursuit task. It is known to be NP-hard in general, and thus approximation algorithms are proposed for its solution. Among the various existing possibilities, the thresholding algorithm stands out due to its simplicity. It suggests approximating $\boldsymbol{\gamma}$ by the formula $\hat{\boldsymbol{\gamma}} = S_\lambda(\mathbf{D}^T\mathbf{y})$, where S_λ nulls all the small entries (positive or negative) in $\mathbf{D}^T\mathbf{y}$, leaving intact only the k largest elements in this vector. Thus, the denoised signal under this approach becomes $\hat{\mathbf{x}} = \mathbf{D}S_\lambda(\mathbf{D}^T\mathbf{y})$.

What is the origin of the thresholding algorithm? In order to answer this, consider the case where $\mathbf{D}$ is square and orthogonal (i.e., $\mathbf{D}\mathbf{D}^T = \mathbf{I}$). In this case, the pursuit task simplifies because of the relation[1] $\|\mathbf{y} - \mathbf{D}\boldsymbol{\gamma}\|_2 = \|\mathbf{D}^T\mathbf{y} - \boldsymbol{\gamma}\|_2$. With this modification, the original pursuit task collapses into a set of 1D optimization problems that are easy to handle, leading to a hard-thresholding operation on the elements of $\mathbf{D}^T\mathbf{y}$as the exact minimizer. This implies that in this case the thresholding algorithm leads to the optimal solution for the problem posed in (2).

The above is not the only approach towards practicing sparsity in signal models. An appealing alternative is the analysis model: A signal $\mathbf{x}$ is believed to emerge from this model if $\boldsymbol{\Omega}\mathbf{x}$ is known to be $(N-k)$-sparse, where $\boldsymbol{\Omega} \in \mathbb{R}^{L \times N}$ is the analysis dictionary. We refer to the vector $\boldsymbol{\Omega}\mathbf{x}$ as the analysis representation.

Although this may look similar to the above-described synthesis approach, the two options are quite different in general. This difference is exposed by the following view: in the synthesis approach we compose the signal by gathering atoms (columns of $\mathbf{D}$) with different weights. A k-sparse signal implies that it is a combination of k such atoms, and thus it resides in a k dimensional subspace spanned by these vectors. In contrast, the analysis model defines the signal by carving it from the whole $\mathbb{R}^N$-space, by stating what directions it is orthogonal to. If the multiplication $\boldsymbol{\Omega}\mathbf{x}$ is known to be $(N-k)$-sparse, this means that there are $(N-k)$ rows in $\boldsymbol{\Omega}$ that are orthogonal to $\mathbf{x}$, which in turn means that the signal of interest resides in a k-dimensional subspace that is the orthogonal complement to these rows.

The difference between the two models is also clearly seen when discussing the denoising task. Given a noisy version of an analysis signal, $\mathbf{y} = \mathbf{x} + \mathbf{n}$, denoising it amounts to a search for the signal $\mathbf{x}$ which is the closest to $\mathbf{y}$ such that $\|\boldsymbol{\Omega}\mathbf{x}\|_0 = N - k$, i.e.,

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{Argmin}} \|\mathbf{y} - \mathbf{x}\|_2 \quad \text{Subject To } \|\boldsymbol{\Omega}\mathbf{x}\|_0 = N - k. \tag{3}$$

Clearly, the denoising process here is quite different from the one discussed above. Nevertheless, just as before, if the true support (non-zero locations) of the analysis representation $\boldsymbol{\Omega}\mathbf{x}$ has been found correctly, the noise in the resulting signal is reduced by the same factor of k/N. Furthermore, this problem, termed the analysis pursuit, is a daunting NP-hard problem in general, just as the synthesis purusit, and various approximation algorithms were developed to evaluate its solution.

What happens when $\boldsymbol{\Omega}$ is square and orthogonal? In this case the analysis pursuit simplifies and becomes equivalent to the synthesis model. This is easily seen by denoting $\boldsymbol{\Omega}\mathbf{x}$ as $\boldsymbol{\gamma}$ which implies $\mathbf{x} = \boldsymbol{\Omega}^T\boldsymbol{\gamma}$. Plugging these two relations into the problem posed in (3), we obtain the very same problem posed in the synthesis case, for which the very same thresholding

[1]The L_2 norm is invariant to unitary rotations, a property known as the Parseval theorem.

algorithm is known to lead to the optimal solution. Thus, in this case, the thresholding solution becomes $\hat{\mathbf{x}} = \mathbf{\Omega}^T S_\lambda(\mathbf{\Omega}\mathbf{y})$.

To summarize, we have seen that both the synthesis and the analysis models find the thresholding algorithm as optimal in the orthogonal case. When moving to non-orthogonal dictionaries, and indeed, to redundant ones, this optimality claim is no longer valid. Nevertheless, the threshodling algorithm remains a valid approximation technique for handling both pursuit problems (2) and (3), and in some cases, one could even get reasonable approximations with this method [1], [7]. In such cases, the general formula to use would be the one posed in (1), where a pseudo-inverse is replacing the plain transpose operation.

B. Analysis-Synthesis Decoupling

The denoising process posed in (1) can be easily extended to handle more general recovery tasks, by decoupling the analysis and synthesis dictionaries in the recovery process:

$$\hat{\mathbf{x}} = \mathbf{D} S_\lambda(\mathbf{\Omega}\mathbf{y}). \tag{4}$$

Here, $\mathbf{D} \in \mathbb{R}^{M\times L}$, $\mathbf{\Omega} \in \mathbb{R}^{L\times N}$, and $M \neq N$ in general. This generalization can model, for example, degradations of the form $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$, where $\mathbf{H}$ is linear (possibly rank-deficient) and $\mathbf{n}$ is white Gaussian noise, by setting $\mathbf{D} = \mathbf{H}^\dagger \mathbf{\Omega}^\dagger$. Additional study of the degradations supported by this formulation is left for future research; instead, in this work we adopt an alternative *model-less* approach in which the training process itself learns the degradation model from examples.

One advantage of the proposed dictionary decoupling is that it results in a simpler dictionary training task, due to the elimination of the pseudo-inverse constraint between the two dictionaries. The decoupling of the dictionaries makes the process a true analysis-synthesis hybrid, which we thus name *analysis-synthesis thresholding*.

Threshold Selection: One point which must be addressed in any shrinkage process of the form (1) or (4) is the choice of the threshold λ. Common techniques for threshold selection include SureShrink [15], VisuShrink [16], BayesShrink [17], K-Sigma shrink [18], and FDR-shrink [19], among others. Alternatively, in this work we adopt a *learning* approach to this task, and train the threshold as a part of the dictionary learning process. We note that in practice, the threshold value will generally depend on the noise level. For simplicity, in the following we address this by training an individual triplet $(\mathbf{\Omega}, \mathbf{D}, \lambda)$ for each noise level. In a more practical setting, it is likely that a single dictionary pair could be trained for several noise levels, adapting only the threshold values to the different noise levels using the proposed threshold training process.

III. Dictionary Training Process

A. Training Target

The recovery process (4) gives rise to a supervised learning formulation for estimating the signal recovery parameters. Specifically, given a set of training pairs $\{(\mathbf{x}_i, \mathbf{y}_i)\}$ consisting of original signals $\mathbf{x}_i$ and their degraded versions $\mathbf{y}_i$, we seek a triplet $(\mathbf{D}, \mathbf{\Omega}, \lambda)$ which best recovers the $\mathbf{x}_i$'s from the $\mathbf{y}_i$'s. Letting $\mathbf{X} = [\mathbf{x}_1 \mathbf{x}_2 \dots \mathbf{x}_R]$ and $\mathbf{Y} = [\mathbf{y}_1 \mathbf{y}_2 \dots \mathbf{y}_R]$, the training process takes the form[2]:

$$\{\hat{\mathbf{\Omega}}, \hat{\mathbf{D}}, \hat{\lambda}\} = \underset{\mathbf{\Omega},\mathbf{D},\lambda}{\text{Argmin}} \|\mathbf{X} - \mathbf{D} S_\lambda(\mathbf{\Omega}\mathbf{Y})\|_F^2. \tag{5}$$

As such, the analytical form of the thresholding estimator in (4) leads to a closed-form expression for the parameter tuning optimization target posed here. This should be contrasted with more effective pursuit methods such as Basis Pursuit [1], which approximate the sparsest representation via yet another optimization goal, and thus lead to a complex bi-level optimization task, when plugged into the above learning-based objective function.

Returning to (5), we note that this problem statement is in fact ill-posed, as we are free to rescale $\mathbf{\Omega} \to (\alpha\mathbf{\Omega})$, $\mathbf{D} \to (1/\alpha\,\mathbf{D})$, $\lambda \to (\alpha\lambda)$ for any $\alpha > 0$. Thus, we could normalize this problem by selecting, e.g., $\lambda = 1$, and allowing the optimization to set the norms of the rows in $\mathbf{\Omega}$ to fit this threshold. Alternatively, the normalization we choose here—mainly for presentation clarity—is to fix the norms of all rows in $\mathbf{\Omega}$ to unit length, and allow the threshold to vary. However, to accommodate this normalization, we must clearly allow different threshold values for different rows in $\mathbf{\Omega}$. Thus, we define $\boldsymbol{\lambda} = (\lambda_1, \dots \lambda_L)$, where λ_i is the threshold for the i-th row of $\mathbf{\Omega}$. With this notation, we reformulate our training target as:

$$\{\hat{\mathbf{\Omega}}, \hat{\mathbf{D}}, \hat{\boldsymbol{\lambda}}\} = \underset{\mathbf{\Omega},\mathbf{D},\boldsymbol{\lambda}}{\text{Argmin}} \|\mathbf{X} - \mathbf{D} S_{\boldsymbol{\lambda}}(\mathbf{\Omega}\mathbf{Y})\|_F^2 \quad \text{Subject To} \quad \|\boldsymbol{\omega}_i\|_2 = 1 \forall i, \tag{6}$$

where $S_{\boldsymbol{\lambda}}$ is a function operating on matrices with L rows, thresholding the i-th row by λ_i. The vectors $\boldsymbol{\omega}_i (i = 1 \dots L)$ represent the rows of $\mathbf{\Omega}$, arranged as column vectors.

B. Optimization Scheme

We optimize (6) using a sequential approach similar to the K-SVD and Analysis K-SVD [11], [20]. At the j-th step, we keep all but the j-th pair of atoms fixed, and optimize:

$$\{\hat{\boldsymbol{\omega}}_j, \hat{\mathbf{d}}_j, \hat{\lambda}_j\} = \underset{\boldsymbol{\omega}_j,\mathbf{d}_j,\lambda_j}{\text{Argmin}} \|\mathbf{X} - \mathbf{D} S_{\boldsymbol{\lambda}}(\mathbf{\Omega}\mathbf{Y})\|_F^2 \quad \text{Subject To } \|\boldsymbol{\omega}_j\|_2 = 1. \tag{7}$$

To isolate the dependence on the j-th atom pair, we write:

$$\begin{aligned} \|\mathbf{X} - \mathbf{D} S_{\boldsymbol{\lambda}}(\mathbf{\Omega}\mathbf{Y})\|_F^2 &= \|\mathbf{X} - \sum_k \mathbf{d}_k S_{\lambda_k}(\boldsymbol{\omega}_k^T \mathbf{Y})\|_F^2 \\ &= \|\mathbf{E}_j - \mathbf{d}_j S_{\lambda_j}(\boldsymbol{\omega}_j^T \mathbf{Y})\|_F^2, \end{aligned}$$

where $\mathbf{E}_j = \mathbf{X} - \sum_{k\neq j} \mathbf{d}_k S_{\lambda_k}(\boldsymbol{\omega}_k^T \mathbf{Y})$. Thus, our optimization goal for the j-the atom pair becomes:

$$\{\hat{\boldsymbol{\omega}}_j, \hat{\mathbf{d}}_j, \hat{\lambda}_j\} = \underset{\boldsymbol{\omega}_j,\mathbf{d}_j,\lambda_j}{\text{Argmin}} \|\mathbf{E}_j - \mathbf{d}_j S_{\lambda_j}(\boldsymbol{\omega}_j^T \mathbf{Y})\|_F^2 \quad \text{Subject To } \|\boldsymbol{\omega}_j\|_2 = 1. \tag{8}$$

[2]Note that here and elsewhere in the paper, whenever referring to a minimization target with possibly a multitude of solutions, we use the simplified notation $x = \text{argmin}_x f(x)$ to denote the goal of locating any one of the members in the solution-set, assuming all are equally acceptable.

We note that the hard thresholding operator defines a partitioning of the signals in $\mathbf{Y}$ to two sets: letting $J = J(\boldsymbol{\omega}_j, \lambda_j)$ denote the indices of the examples that remain intact after the thresholding (i.e., $|\boldsymbol{\omega}_j^T \mathbf{y}_j| \geq \lambda_j$), we split $\mathbf{Y}$ to the matrices $\mathbf{Y}^J$ and $\mathbf{Y}^{\overline{J}}$, containing the signals indexed by J and the remaining signals, respectively. We similarly split $\mathbf{E}_j$ to the submatrices $\mathbf{E}_j^J$ and $\mathbf{E}_j^{\overline{J}}$. With these notations, the above can be rearranged as:

$$\underset{\boldsymbol{\omega}_j, \mathbf{d}_j, \lambda_j}{\text{Argmin}} \|\mathbf{E}_j^J - \mathbf{d}_j \boldsymbol{\omega}_j^T \mathbf{Y}^J\|_F^2 + \|\mathbf{E}_j^{\overline{J}}\|_F^2 \quad \text{Subject To } \|\boldsymbol{\omega}_j\|_2 = 1. \quad (9)$$

Optimizing this expression is obviously non-trivial as the target function is non-convex and highly discontinuous. The main difficulty is due to the fact that updating $\boldsymbol{\omega}_j$ and λ_j modifies the signal partitioning J, causing a non-smooth change to the cost function. One straightforward approach, taken by the K-SVD algorithm for instance, is to perform the update while constraining the partitioning of the signals to remain fixed. Under such a constraint, the atom update task can be formulated as a convex Quadratic Programming (QP) problem, and can be globally solved. Unfortunately, this approach can only accommodate small deviations of the solution from the initial estimate, and thus we take a different approach here. For completeness, we detail the derivation of the QP formulation in Appendix A.

1) Optimization via Rank-One Approximation: A simple and effective alternative to the above approach is to make the *approximation* that the update process does not change much the partitioning of the signals. This approach assumes that the set J remains roughly constant during the update, and thus, approximates the target function in (9) by

$$\underset{\boldsymbol{\omega}_j, \mathbf{d}_j, \lambda_j}{\text{Argmin}} \|\mathbf{E}_j^{J_0} - \mathbf{d}_j \boldsymbol{\omega}_j^T \mathbf{Y}^{J_0}\|_F^2 + \|\mathbf{E}_j^{\bar{J}_0}\|_F^2 \quad \text{Subject To } \|\boldsymbol{\omega}_j\|_2 = 1. \quad (10)$$

Here, J_0 denotes the current partitioning of the signals. Formally, this approach is equivalent to optimizing (8) under a first-order expansion of S_{λ_j}, which is reasonably accurate for coefficients far from the threshold.

Deriving a formal bound on the error of the proposed approximation is difficult. In fact, when the set J is small, the approximation becomes useless as the signal partitioning may be substantially altered by the update process. However, when the set J covers a significant enough portion of the examples, we expect the majority of examples to follow this assumption due to the nature of the update which favors signals already using the atom. Our simulations support this assumption, and indicate that the typical fraction of signals moving between J and $\bar{J}$ is quite small. A representative case is provided in Fig. 1: we see that the fraction of signals moving between J and $\overline{J}$ in this case is $<12\%$ for the first iteration, and goes down to just 2–6% for the remaining iterations.

By refraining from an explicit constraint on the partitioning, we both simplify the optimization as well as allow some outlier examples to "switch sides" relative to the threshold. Returning to (10), $E_j^{\overline{J}_0}$ in this optimization is now constant, allowing us to reduce the atom update task to:

$$\{\hat{\boldsymbol{\omega}}_j, \hat{\mathbf{d}}_j\} = \underset{\boldsymbol{\omega}_j, \mathbf{d}_j}{\text{Arg min}} \|\mathbf{E}_j^{J_0} - \mathbf{d}_j \boldsymbol{\omega}_j^T \mathbf{Y}^{J_0}\|_F^2 \quad \text{Subject To } \|\boldsymbol{\omega}_j\|_2 = 1. \quad (11)$$

We note that λ_j is omitted in this formulation, as the partitioning is fixed and thus λ_j has no further effect. Nonetheless, we indeed update λ_j following the update of $\boldsymbol{\omega}_j$ and $\mathbf{d}_j$, in order to optimally tune it to the new atom values.

The resulting problem (11) is a simple rank-one approximation, which can be solved using the SVD. Due to the presence of the matrix $\mathbf{Y}^{J_0}$, the solution requires a short derivation which we detail in Appendix B. The resulting procedure is listed in Fig. 2. We should note that the solution proposed assumes that $\mathbf{Y}^{J_0}$ is full rank, a condition easily satisfied if enough examples are available.

2) Threshold Update: Once $\boldsymbol{\omega}_j$ and $\mathbf{d}_j$ are updated according to (11), we recompute the threshold λ_j to match the new atoms. Optimizing the target function (8) for λ_j translates to the following optimization task:

$$\hat{\lambda}_j = \underset{\lambda_j}{\text{Argmin}} \|\mathbf{E}_j - \mathbf{d}_j S_{\lambda_j}\left(\boldsymbol{\omega}_j^T \mathbf{Y}\right)\|_F^2. \quad (12)$$

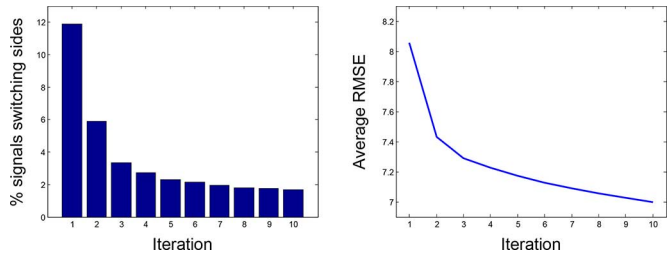

Fig. 1. Fraction of examples "switching sides" relative to the threshold during dictionary training. Results are for a pair of dictionaries with 256 atoms each, used to denoise 8×8 image patches ($\sigma = 10$). Training patches are extracted from eight arbitrary images in the CVG Granada dataset [21], 40 000 patches from each image. Left: Percentage of examples switching sides during each algorithm iteration (median over all atom pairs). Right: Corresponding target function evolution.

1: Input: Matrices $\mathbf{E}, \mathbf{Y} \in \mathbb{R}^{N \times R}$
2: Output: Solution to:

$$\underset{\mathbf{d}, \boldsymbol{\omega}}{\text{Argmin}} \|\mathbf{E} - \mathbf{d}\boldsymbol{\omega}^T \mathbf{Y}\|_F^2 \quad \text{Subject To} \quad \|\boldsymbol{\omega}\|_2 = 1$$

3: **procedure**:
4: Compute the SVD: $\mathbf{Y} = \mathbf{U}\mathbf{S}\mathbf{V}^T$
5: $\boldsymbol{\Delta} := \text{diag}(s_1^{-1}, \ldots, s_N^{-1})$
6: $\widetilde{\mathbf{Y}} := \boldsymbol{\Delta}\mathbf{U}^T\mathbf{Y}$
7: $\{\mathbf{d}, \tilde{\boldsymbol{\omega}}\} := \underset{\mathbf{d}, \tilde{\boldsymbol{\omega}}}{\text{Argmin}} \|\mathbf{E}\widetilde{\mathbf{Y}}^T - \mathbf{d}\tilde{\boldsymbol{\omega}}^T\|_F^2$
8: $\boldsymbol{\omega}^T := \tilde{\boldsymbol{\omega}}^T \boldsymbol{\Delta}\mathbf{U}^T$
9: $\mathbf{d} := \mathbf{d} \cdot \|\boldsymbol{\omega}\|_2$
10: $\boldsymbol{\omega}^T := \boldsymbol{\omega}^T / \|\boldsymbol{\omega}\|_2$
11: **end**

Fig. 2. Rank-one approximation used in the dictionary training.

Input: Matrices $\mathbf{E}, \mathbf{Y} \in \mathbb{R}^{N \times R}$, atoms $\mathbf{d}, \boldsymbol{\omega} \in \mathbb{R}^N$
Output: Solution to $\underset{\lambda}{\text{Argmin}} \|\mathbf{E} - \mathbf{d}S_\lambda(\boldsymbol{\omega}^T\mathbf{Y})\|_F^2$
Preprocess: Sort the columns of $\mathbf{E}$ and $\mathbf{Y}$ in increasing order of $|\boldsymbol{\omega}^T\mathbf{Y}|$
procedure:
  $\forall i: \quad \alpha_i := \|\mathbf{e}_i\|_2^2$
  $\forall i: \quad \beta_i := \|\mathbf{e}_i - \mathbf{d}\boldsymbol{\omega}^T\mathbf{y}_i\|_2^2$
  $s_1 := \sum_{i=1}^R \beta_i$
  for $k = 1 \dots R$ **do**
    $s_{k+1} := s_k + \alpha_k - \beta_k$
  end for
  $\hat{k} := \underset{k}{\text{Argmin}}\ s_k$
  $$\lambda := \begin{cases} 0 & \hat{k} = 1 \\ |\boldsymbol{\omega}^T\mathbf{y}_R| + 1 & \hat{k} = R+1 \\ (|\boldsymbol{\omega}^T\mathbf{y}_{\hat{k}-1}| + |\boldsymbol{\omega}^T\mathbf{y}_{\hat{k}}|)/2 & \text{otherwise} \end{cases}$$
end

Fig. 3. Threshold update used in the dictionary training.

Input: Training signals $\mathbf{X} \in \mathbb{R}^{M \times R}$, degraded signals $\mathbf{Y} \in \mathbb{R}^{N \times R}$, initial dictionaries $\boldsymbol{\Omega}_0 \in \mathbb{R}^{L \times N}$, $\mathbf{D}_0 \in \mathbb{R}^{M \times L}$, initial thresholds $\boldsymbol{\lambda}_0$, number of iterations N_{iter}
Output: Dictionaries $\boldsymbol{\Omega}$, $\mathbf{D}$ and threshold vector $\boldsymbol{\lambda}$ minimizing (6)
Init: Set $\boldsymbol{\Omega} := \boldsymbol{\Omega}_0$, $\mathbf{D} := \mathbf{D}_0$, $\boldsymbol{\lambda} := \boldsymbol{\lambda}_0$
for $n = 1 \dots N_{iter}$ **do**
  for $j = 1 \dots L$ **do**
    $J := \{\, i \in \{1 \dots R\} \mid |\boldsymbol{\omega}_j^T\mathbf{y}_i| \geq \lambda\}$
    $E_j = \mathbf{X} - \sum_{k \neq j} \mathbf{d}_k S_{\lambda_k}(\boldsymbol{\omega}_k^T\mathbf{Y})$
    $\{\mathbf{d}_j, \boldsymbol{\omega}_j\} := \underset{\mathbf{d}, \|\boldsymbol{\omega}\|_2 = 1}{\text{Argmin}} \|\mathbf{E}_j^J - \mathbf{d}\boldsymbol{\omega}^T\mathbf{Y}^J\|_F^2$ *(Fig. 2)*
    $\lambda_j := \underset{\lambda}{\text{Argmin}} \|\mathbf{E}_j - \mathbf{d}_j S_\lambda(\boldsymbol{\omega}_j^T\mathbf{Y})\|_F^2$ *(Fig. 3)*
    $\boldsymbol{\Omega}\{j\text{-th row}\} := \boldsymbol{\omega}_j^T$
    $\mathbf{D}\{j\text{-th col}\} := \mathbf{d}_j$
    $\boldsymbol{\lambda}\{j\text{-th elem}\} := \lambda_j$
  end for
end for

Fig. 4. Full analysis-synthesis dictionary learning algorithm.

This problem can be *globally* solved due to the discrete nature of the hard threshold operator. Without loss of generality, we assume that the signals are ordered such that $|\boldsymbol{\omega}_j^T\mathbf{y}_1| \leq |\boldsymbol{\omega}_j^T\mathbf{y}_2| \leq \cdots \leq |\boldsymbol{\omega}_j^T\mathbf{y}_R|$. Thus, for any threshold $\lambda_j \in (|\boldsymbol{\omega}_j^T\mathbf{y}_1|, |\boldsymbol{\omega}_j^T\mathbf{y}_R|)$, there exists a unique index $k = k(\lambda_j)$ such that $|\boldsymbol{\omega}_j^T\mathbf{y}_{k-1}| < \lambda_j \leq |\boldsymbol{\omega}_j^T\mathbf{y}_k|$. The examples which survive this threshold are $y_k, y_{k+1}, \dots, y_R$, and we can thus rewrite (12) as:

$$\hat{\lambda}_j = \underset{\lambda_j}{\text{Argmin}} \sum_{i=1}^{k(\lambda_j)-1} \|\mathbf{e}_i\|_2^2 + \sum_{i=k(\lambda_j)}^{R} \|\mathbf{e}_i - \mathbf{d}_j\boldsymbol{\omega}_j^T\mathbf{y}_i\|_2^2,$$

where $\mathbf{e}_i$ is the i-th column of $\mathbf{E}_j$. In this formulation, k encloses all the necessary information about λ_j, and the optimization can therefore be carried out over the discrete transition point k, which is a simple task. Introducing the notations $\alpha_i = \|\mathbf{e}_i\|_2^2$ and $\beta_i = \|\mathbf{e}_i - \mathbf{d}_j\boldsymbol{\omega}_j^T\mathbf{y}_i\|_2^2$, the optimization task for k is given by:

$$\hat{k} = \underset{k}{\text{Argmin}} \sum_{i=1}^{k-1} \alpha_i + \sum_{i=k}^{R} \beta_i. \tag{13}$$

This expression is minimized directly by computing the values $s_k = \sum_{i=1}^{k-1} \alpha_i + \sum_{i=k}^{R} \beta_i$ for all k and taking the global minimum. The values s_k are computed via the recursion $s_1 = \sum_{i=1}^{R} \beta_i$ and $s_{k+1} = s_k + \alpha_k - \beta_k$. Once the value $\hat{k}$ is known, any suitable value for λ_j can be selected, e.g., $\lambda_j = (|\boldsymbol{\omega}_j^T\mathbf{y}_{\hat{k}-1}| + |\boldsymbol{\omega}_j^T\mathbf{y}_{\hat{k}}|)/2$. The threshold update process is summarized in Fig. 3.

3) Full Training Process: Putting the pieces together, the atom update process for the j-th atom pair consists of the following steps: (a) finding the set J_0 of signals using the current atom pair; (b) updating $\boldsymbol{\omega}_j$ and $\mathbf{d}_j$ according to (11); and (c) recomputing the threshold λ_j by solving (13). The algorithm processes the dictionary atoms in sequence, and thus benefits from having the updated atoms and error matrix available for the subsequent updates. The full training process is detailed in Fig. 4. Note that the algorithm assumes some initial choice for $\boldsymbol{\Omega}_0, \mathbf{D}_0$ and $\boldsymbol{\lambda}_0$. In practice, our implementation only requires an initial dictionary $\boldsymbol{\Omega}_0$. Our tests indicate that the overall process is sensitive to the choice of the initialization, and thus care must be taken with respect to this matter. We have chosen to use the redundant DCT as $\boldsymbol{\Omega}_0$, as this proves to be effective for natural images. For $\mathbf{D}_0$ we initialize with either $\mathbf{D}_0 = \boldsymbol{\Omega}^\dagger$ or $\mathbf{D}_0 = \mathbf{X}(\boldsymbol{\Omega}_0\mathbf{Y})^\dagger$ (we have found the first to give slightly better results in our experiments below), and for $\boldsymbol{\lambda}_0$ we begin with an arbitrary choice $\boldsymbol{\lambda}_0 = (\hat{\lambda}, \dots, \hat{\lambda})$ where $\hat{\lambda}$ is the median of the coefficients in $|\boldsymbol{\Omega}\mathbf{Y}|$, and run one sweep of the algorithm in Fig. 3 over all threshold values.

As previously mentioned, the proposed atom update is subject to the condition that the set J_0 have some minimal size. In practice, we set this minimum to a liberal 5% of the examples. However, if this minimum is not satisfied, we discard the current atom pair, and apply steps (b) and (c) above with J_0 being the entire set of signals. This heuristic process replaces the atom pair with a new pair which is hopefully used by more examples. A complementary approach, which we did not currently employ, would be to allow a few atoms with a smaller number of associated examples to prevail, and optimize them using the QP process described in the Appendix.

IV. Empirical Evaluation and Discussion

A. Experiment Setup

We evaluated the performance of the proposed technique for small-kernel image deblurring. Our training set consists of eight natural images taken from the CVG-Granada dataset [21]. Four of these are shown in Fig. 5. Each of the training images was

Fig. 5. Four training images from the CVG Granada data set.

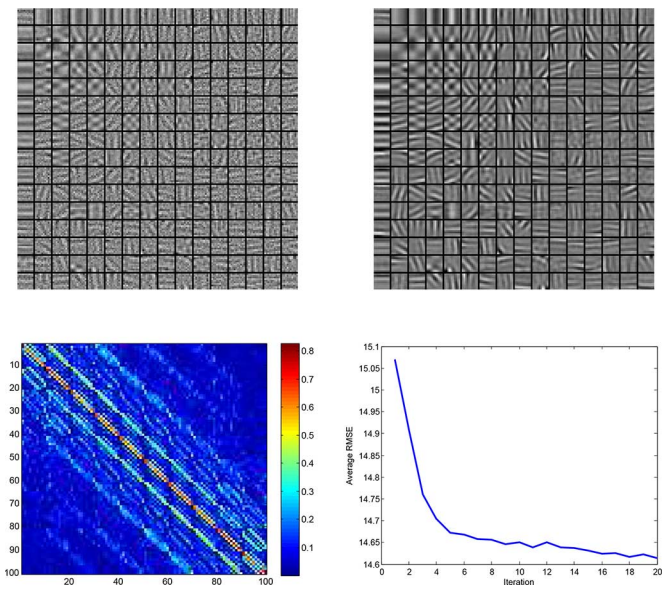

Fig. 6. Results of the training algorithm for image deblurring. Top left: trained $\mathbf{\Omega}$. Top right: trained $\mathbf{D}$. The squares in each grid represent the atoms of the dictionary, contrast-normalized and reshaped to the block size. Bottom left: absolute values of the entries in $\mathbf{D\Omega}$. Bottom right: Error evolution during the algorithm iterations (y-axis is the average RMSE of the recovered patches). Parameters for this experiment are listed in Table I (case 1).

TABLE I
DEBLURRING EXPERIMENT PARAMETERS. THE TWO DEGRADATION CASES ARE TAKEN FROM [22]

	Case 1 (Fig. 7)	Case 2 (Fig. 8)
Blur kernel size	5×5	11×11
Blur standard deviation	1.5	1.75
Noise power	$\sigma = 8.25$ (BSNR=15dB)	$\sigma = 1.15$ (BSNR=30dB)
Block size	10×10	12×12
Dictionary size	256×100	576×144
Training iterations	20	20
Numer of training blocks	320,000	320,000

Fig. 7. Deblurring results for *Lena*. See Table I for the full experiment parameters. RMSE values are 10.78 (blurry), 7.55 (ForWaRD), 6.76 (LPA-ICI), 6.12 (AKTV) and 6.63 (Thresholding). (a) Original. (b) Blurry and noisy. (c) ForWaRD. (d) LPA-ICI. (e) AKTV. (f) Thresholding.

subjected to blur and additive white Gaussian noise, producing eight pairs of original and degraded input images. We then extracted from each image 40 000 random training blocks along with their degraded versions, for a total of 320 000 example pairs. We subtracted from each example pair the mean of the degraded block, to obtain the final training set.

The initial dictionary $\mathbf{\Omega}_0$ was an overcomplete DCT dictionary, and training was performed for 20 iterations. An example result of the training process is shown in Fig. 6. The top row shows the trained $\mathbf{\Omega}$ (left) and $\mathbf{D}$ (right). The bottom-left figure shows the absolute values of the entries in the matrix $\mathbf{D\Omega}$, which exhibits a diagonal structure characteristic of a local deconvolution operator. The bottom-right figure depicts the error evolution during the algorithm iterations. We note that while the error reduction is not completely monotonic due to the approximations made in the algorithm derivation, overall the error goal is effectively decreased.

To evaluate deblurring performance on new images, we used the following procedure: given a new blurry image, we extracted all overlapping blocks, subtracted their means, and applied the learned thresholding process to each block. We then reintroduced the blocks means, and constructed the recovered image by averaging the overlapping blocks. We tested our method on seven standard test images, all of which were not included in the training set: *Barbara, Cameraman, Chemical Plant, House, Lena, Peppers* and *Man*.

B. Results

Results of our deblurring process for two example cases are shown in Figs. 7 and 8; see Table I for the full list of experiment parameters. The two cases are taken from [22], whose inputs are made available online [23]. The figures compare our results with those of ForWaRD [24], LPA-ICI [25] and AKTV [22].[3] The first case (Fig. 7) represents strong noise and small blur, while the second case (Fig. 8) represents moderate noise and moderate blur. In the current work we limit ourselves to handling small to moderate blur kernels, as large kernels require much larger block sizes which are impractical in the current formulation. We thus do not replicate the two additional cases considered in [22], which employ very large blur kernels. A few options for addressing larger blur kernels are mentioned in the conclusion.

As can be seen, our results in both cases exceed ForWaRD and LPA-ICI in raw RMSE by a small margin, and lose only to the AKTV. Visually, our result in Fig. 7 maintains more of the noise than the other methods, though subjectively it also appears less "processed", and we note that lines and curves, for

[3]We note that while later works on image deblurring have been recently published [26], our goal here is mainly to assess the performance of our simple learning approach and demonstrate its unique properties, rather than compete with the state-of-the-art.

TABLE II
FULL RESULTS FOR THE TWO CASES IN TABLE I. VALUES REPRESENT RMSE

Case 1			
Image	Degraded	Thresh.	ForWaRD
Barbara	17.64	**15.44**	15.84
Cameraman	17.37	**13.26**	13.46
Chem. Plant	14.48	**10.46**	11.48
House	8.76	**3.90**	5.03
Lena	10.77	**6.61**	7.44
Peppers	10.78	**6.85**	7.33
Pirate	12.36	**8.87**	9.54

Case 2			
Image	Degraded	Thresh.	ForWaRD
Barbara	16.57	**14.74**	14.81
Cameraman	17.78	11.87	**11.82**
Chem. Plant	15.05	8.67	**8.58**
House	4.94	2.54	**2.50**
Lena	8.92	**5.47**	5.54
Peppers	8.60	**5.93**	6.06
Pirate	11.11	**7.62**	7.73

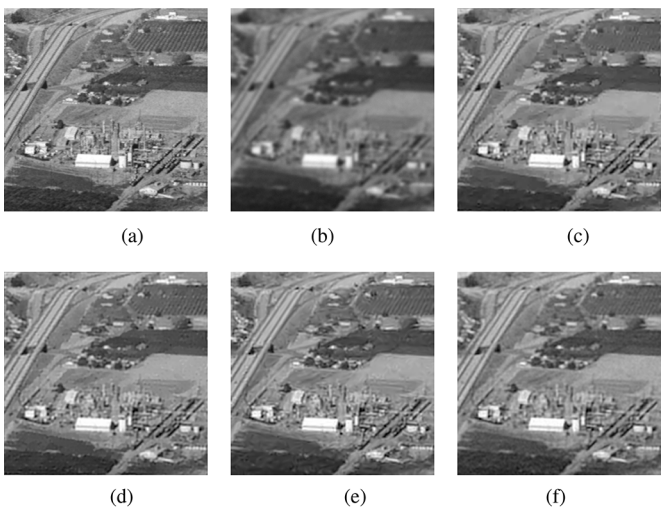

Fig. 8. Deblurring results for *Chemical Plant*. See Table I for the full experiment parameters. RMSE values are 15.09 (blurry), 8.98 (ForWaRD), 8.98 (LPA-ICI), 8.57 (AKTV) and 8.76 (Thresholding). (a) Original. (b) Blurry and noisy. (c) ForWaRD. (d) LPA-ICI. (e)AKTV. (f) Thresholding.

instance, appear straighter and less "jaggy". Continuing with Fig. 8, our result in this case seems more visually pleasing than that of ForWaRD and LPA-ICI, and reproduces more fine details (see for instance the field area at the top right). Compared to the AKTV, our result maintains slightly more noise, though it also avoids introducing the artificial smear and "brush stroke" effects characteristic of the AKTV, and likely associated with its steering regularization kernel.

C. Discussion

Compared to the other methods in this experiment, our deblurring process is simple and efficient, and involves essentially no parameter tuning (except for the block size). In these respects, the ForWaRD algorithm is the most comparable to our system as it is fast and its parameters can be automatically tuned, as described in [24]. The ForWaRD algorithm is also the most similar to our work as it is based on a scaling (shrinkage) process of the image coefficients in the Fourier and Wavelet domains. The LPA-ICI and AKTV, on the other hand, both involve parameters which must be manually tuned to optimize performance. Also, while the LPA-ICI is relative fast, the AKTV is particularly computationally intensive, requiring e.g., in the case shown in Fig. 7, at least 12 minutes to achieve a reasonable result, and nearly an hour to reproduce the final result shown in the figure. In comparison, our method on the same hardware[4] completed in just 8 seconds, due to the diversion of most of the computational burden to the offline training phase. Furthermore, our recovery method is highly parallelizable and consists of primitive computations only, and thus, can likely be optimized to achieve real-time performance.

Another notable difference between our method and the others is its "model-less" nature, as previously mentioned. Indeed, all three methods (ForWaRD, LPA-ICI and AKTV) assume accurate knowledge of the blur kernel, which is typical of deconvolution frameworks. Our method is fundamentally different in that it replaces this assumption with a very different one—the availability of a set of training images which undergo the same degradation, and implicitly represent the convolution kernel. In practice, this difference may not be as significant as it seems, as in both cases a real-world application would require either a prior calibration process, or an online degradation estimation method. Nevertheless, in some cases, acquiring a training set may be a simpler and more robust process (e.g., using a pair of low and high quality equipment) than a precise measurement of the point spread function.

Finally, our method is inherently indifferent to boundary conditions, which plague some deconvolution methods. Our deconvolution process can be applied with no modification to images undergoing non-circular convolution, and will produce no visible artifacts near the image borders. Of the three methods we compared to, only the AKTV provides a similar level of immunity to boundary conditions.

Table II lists our deblurring results for all seven standard test images. We compare our results to those of the ForWaRD algorithm, which we chose due to its combination of efficiency, lack of manual parameter tuning, and relation to our method. The thresholding results in these tables were produces using the same trained dictionaries used to produce the results in Figs. 7 and 8. The ForWaRD results were generated using the Matlab package available at [27].

V. RELATED WORKS

Just before we conclude this paper, we would like discuss possible relations between the paradigm proposed here past work. We shall put special emphasis on two very different lines of work which are of some relevance to the approach taken in this paper—the one reported in [28] and follow-up work, which proposes a closely related shrinkage strategy for image processing tasks, and the vast work on feature learning and auto-encoders, which emerged recently in machine learning, in the context of deep-learning [29], [30].

[4]All the simulations reported in this paper have been run on an Intel Core i7 CPU 2.8 Ghz PC, with 8 GB of memory, and using Matlab R2013a without parallelization.

A. Shrink Operator Learning

One aspect of problem (1) which has been previously studied in the literature [28], [31], [32] is the design of the *shrink function* S_λ. In this family of works, the dictionary $\mathbf{\Omega}$ is assumed to be fixed, and the goal is to learn a set of scalar shrink functions (or, more precisely, a set of scalar mappings) given pairs of original and degraded examples. The process trains an individual mapping S_i for each representation coefficient, using a piecewise-linear approximation of the function. The method is found to produce competitive results for image denoising, and is shown to apply to other recovery tasks as well.

One interesting observation, relevant to the current work, is that the resulting trained shrinkage operators in [28] bear notable resemblance to the hard thresholding operators we use in our work (though with a small and intriguing non-monotonicity around the center in some cases, see Fig. 8 there). This is an encouraging result, as it demonstrates the usefulness and near-optimality of the hard thresholding operator for practical applications, as explored in the analysis-then-synthesis system we propose in (4).

B. Feature Learning

Dictionary learning for the tasks (1) and (4), in the context of signal reconstruction, has not yet been addressed in the literature (to the best of the authors' knowledge). However, closely related methods have been receiving substantial attention by the Machine Learning community, in the context of *feature* (or *representation*) *learning* [29], [30]. The automatic learning of signal features—for tasks such as classification, detection, identification and regression—has become enormously successful with the advent of greedy deep learning techniques, due to Hinton *et al.* [33]. In these works, a set of features is extracted from the signal $\mathbf{x}$ via the relation

$$\varphi(\mathbf{x}) = S(\mathbf{\Omega}\mathbf{x} + \mathbf{b}), \tag{14}$$

where S is an element-wise non-linear shrink function. Typical choices for S include the sigmoid and tanh, though smooth soft-thresholding has also been proposed [34]. We note however that as opposed to thresholding functions, the sigmoid and hyperbolic tangent are not true sparsifying functions, and thus do not produce sparse outputs in the ℓ^0 sense.

The feature-extraction process (14) is iteratively applied to $\varphi(\mathbf{x})$ (or rather, to a post-processed version of it, see e.g., [35]), forming a *hierarchy* of increasingly higher-level features $\varphi_1(\mathbf{x}) \dots \varphi_k(\mathbf{x})$. The resulting feature-extracting networks lead to state-of-the-art results in many machine learning tasks [30].

Many heuristic approaches have been proposed to train the parameters $(\mathbf{\Omega}, \mathbf{b})$ of the feature extractors in (14). Among these, two highly successful approaches which are particularly relevant to the present work are *(denoising) auto-encoders* [36] and *predictive sparse decompositions* [37]. Auto-encoders train features using an *information-retention* criterion, maximizing recoverability of the training signals from the extracted features via an affine synthesis process of the form $\hat{\mathbf{x}} = \mathbf{D}\varphi(\mathbf{x}) + \mathbf{c}$ (where $\mathbf{D}$ and $\mathbf{c}$ are trained as well).[5] Denoising auto-encoders build on this concept by seeking features which achieve *denoising* of the training signals, leading to:

$$\{\hat{\mathbf{\Omega}}, \hat{\mathbf{b}}, \hat{\mathbf{D}}, \hat{\mathbf{c}}\} = \underset{\mathbf{\Omega},\mathbf{b},\mathbf{D},\mathbf{c}}{\operatorname{Argmin}} \|\mathbf{X} - (\mathbf{D}S(\mathbf{\Omega}\mathbf{Y} + \mathbf{b}) + \mathbf{c})\|_F^2,$$

where $\mathbf{X}$ and $\mathbf{Y}$ are clean and noisy training signals, respectively. It should be noted that (in a slight abuse of notation) we use matrix-vector additions in the above to denote adding a vector to each column of the matrix.

Closely related, predictive sparse decomposition (PSD) learns features which aim to approximate the solution of an ℓ^1 synthesis sparse coding problem, in a least-squares sense. The method can similarly be used in both noiseless and noisy settings, and is given by:

$$\{\hat{\mathbf{\Omega}}, \hat{\mathbf{b}}, \hat{\mathbf{D}}\} = \underset{\mathbf{\Omega},\mathbf{b},\mathbf{D}}{\operatorname{Argmin}} \|\mathbf{X} - \mathbf{D}\mathbf{\Gamma}\|_F^2 + \lambda\|\mathbf{\Gamma}\|_1 + \alpha\|\mathbf{\Gamma} - S(\mathbf{\Omega}\mathbf{Y} + \mathbf{b})\|_F^2,$$

where $\mathbf{\Gamma}$ acts as a mediator between the analysis feature vectors and the synthesis sparse representations.

In all these formulations, the training process outputs a synthesis dictionary $\mathbf{D}$ alongside the analysis one, though the actual outcome of the algorithm is the analysis dictionary alone. The target function is minimized using a gradient-based technique such as stochastic steepest-descent or Levenberg-Marquardt iterations, and thus, the shrink function is intentionally selected to be smooth.

Despite the mathematical similarity, the present work differs from these feature-learning approaches in several ways. Most evidently, our work specifically employs a *non-smooth* hard thresholding function, which leads to a very different minimization procedure in the spirit of ℓ^0 dictionary learning methods. This has two advantages compared to gradient-based methods—first, our approach requires significantly fewer iterations than gradient-based techniques; and second, our training process has no tuneable parameters, and is very easy to use. Another obvious difference between the two frameworks is the very different nature of their designated goals—whereas the trained features are stacked to multi-layer cascades and ultimately evaluated on *clean* signals for machine-learning tasks, our method is a single-level process, intended for signal reconstruction.

VI. Summary and Conclusions

In this work we have presented a technique for training the analysis and synthesis dictionaries of a generalized thresholding-based image recovery process. Our method assumes a hard-thresholding operator, which leads to ℓ^0-sparse representations. This exact sparsity was exploited to design a simple training algorithm based on a sequence of rank-one approximations, in the spirit of the K-SVD algorithm.

[5]Note that in the case of restricted-range shrink functions such as the sigmoid or tanh, the input signals are typically normalized to enable a meaningful reconstruction.

Our training method is designed to simultaneously learn the dictionaries and the threshold values, making the subsequent recovery process simple, efficient, and parameterless. The thresholding recovery process is also naturally parallelizable, allowing for substantial acceleration. A unique characteristic of the process is its example-based approach to the degradation modeling, which requires no explicit knowledge of the degradation, and instead implicitly learns it from pairs of examples. Our approach can thus be applied in cases where an exact model of the degradation is unavailable, but a limited training set can be produced in a controlled environment.

The proposed technique was applied to small-kernel image deblurring, where it was found to match or surpass two dedicated deconvolution methods—ForWaRD and LPA-ICI—and lose only to the computationally demanding AKTV. Our recovery process is also robust to boundary conditions, which some deconvolution methods are sensitive to. We conclude that the proposed learning algorithm provides a simple and efficient method for designing signal recovery processes, without requiring an explicit model of the degradation.

A. Future Directions

The proposed technique may be extended in several ways. First, our method could be adapted to degradations with wider supports by either incorporating downsampling in the recovery process, or more generally, by training structured dictionaries which can represent much larger image blocks [6], [38] (in this respect, we note that downsampling is in fact just a particular choice of dictionary structure). Other possible improvements include training a single dictionary pair for multiple noise levels, and incorporating the block averaging process directly into the dictionary learning target, as done, for example, in [28].

Other directions for future research include performing a more rigorous mathematical analysis of the properties and success guarantees of the thresholding approach, and developing a unified method for simultaneously training the dictionaries and the shrink functions—which could potentially provide a dramatic improvement in restoration results.

Finally, the relation between our method and recent unsupervised feature-learning techniques—namely auto-encoders and predictive sparse decompositions—gives rise to several intriguing future research directions. Among these, we highlight the applicability of the feature-learning formulations to signal restoration tasks; the use of specialized optimization techniques, similar to those developed for dictionary-learning problems, to accelerate feature-training processes; and the extension of our work as well as other sparse representation methods (particularly analysis-based) to form multi-level feature-extracting hierarchies for machine learning tasks. Among the directions mentioned in this section, we find some of the latter to offer particularly promising opportunities for future research.

Appendix A
Quadratic Programming Atom Update

In this Appendix we describe the formulation of the atom update process (8) as a convex QP problem. Beginning with (8), we take a block-coordinate-relaxation approach and update $\mathbf{d}_j$ independently of $\boldsymbol{\omega}_j$ and λ_j. Thus, the update of $\mathbf{d}_j$ becomes a simple least-squares task, given by

$$\mathbf{d}_j = \mathbf{E}_j\boldsymbol{\gamma}_j/(\boldsymbol{\gamma}_j^T\boldsymbol{\gamma}_j), \tag{15}$$

with $\boldsymbol{\gamma}_j = S_{\lambda_j}(\mathbf{Y}^T\boldsymbol{\omega}_j)$.

Moving to the update of $\boldsymbol{\omega}_j$ and λ_j, in the QP approach we constrain the update such that it maintains the partitioning of the training signals about the threshold. Thus, we split $\mathbf{Y}$ to the signals $\mathbf{Y}^J$ which survive the current threshold and the remaining signals $\mathbf{Y}^{\overline{J}}$, and similarly split $\mathbf{E}_j$ to $\mathbf{E}_j^J$ and $\mathbf{E}_j^{\overline{J}}$, obtaining:

$$\begin{aligned}\{\hat{\boldsymbol{\omega}}_j, \hat{\lambda}_j\} = \underset{\boldsymbol{\omega}_j,\lambda_j}{\operatorname{Argmin}} &\|\mathbf{E}_j^J - \mathbf{d}_j\boldsymbol{\omega}_j^T\mathbf{Y}^J\|_F^2 + \|\mathbf{E}_j^{\overline{J}}\|_F^2 \\ &\text{Subject To } |\boldsymbol{\omega}_j^T\mathbf{y}_i| \geq \lambda_j \forall i \in J \\ &|\boldsymbol{\omega}_j^T\mathbf{y}_i| < \lambda_j \forall i \in \overline{J} \\ &\|\boldsymbol{\omega}_j\|_2 = 1. \end{aligned} \tag{16}$$

The constraints ensure that the signal partitioning is maintained by the update process. Note that due to the constraining, J is constant in the optimization.

We now recall that the norm constraint on $\boldsymbol{\omega}_j$ is in fact an arbitrary normalization choice which can be replaced, e.g., with a fixed value for λ_j. Thus, we choose to lift the norm constraint on $\boldsymbol{\omega}_j$ and instead fix the threshold λ_j at its current value. Indeed, the outcome of this optimization can be subsequently re-scaled to satisfy the original unit-norm constraint. Adding the fact that $\mathbf{E}_j^{\overline{J}}$ is fixed in the above optimization (as J is fixed), the update task can be written as:

$$\begin{aligned}\hat{\boldsymbol{\omega}}_j = \underset{\boldsymbol{\omega}_j}{\operatorname{Argmin}} &\|\mathbf{E}_j^J - \mathbf{d}_j\boldsymbol{\omega}_j^T\mathbf{Y}^J\|_F^2 \\ &\text{Subject To } |\boldsymbol{\omega}_j^T\mathbf{y}_i| \geq \lambda_j \forall i \in J \\ &|\boldsymbol{\omega}_j^T\mathbf{y}_i| < \lambda_j \quad \forall i \in \overline{J}. \end{aligned}$$

This formulation does not yet constitute a QP problem, as the first set of constraints is clearly non-convex. To remedy this, we add the requirement that the coefficients $\boldsymbol{\omega}_j^T\mathbf{y}_i$ *do not change sign* during the update process, for the signals in the set J. In other words, we require that $\boldsymbol{\omega}_j$ does not "change sides" relative to the signals in $\mathbf{Y}^J$. While this choice adds further constraining to the problem, in practice many local optimization techniques would be oblivious to the discontinuous optimization regions anyway, and we thus accept the added constraints in return for a manageable optimization task. Of course, an important advantage of this specific choice of constraints is that it necessarily leads to a non-empty feasible region, with the current $\boldsymbol{\omega}_j$ constituting a good starting point for the optimization.

With the updated set of constraints, the optimization domain becomes convex, and the problem can be formulated as a true QP problem. To express the new constraints, we denote by $\sigma_i = \operatorname{sign}(\boldsymbol{\omega}_j^T\mathbf{y}_i)$ the signs of the inner products of the signals with the current atom. We can now write the update process for $\boldsymbol{\omega}_j$ as:

$$\begin{aligned}\hat{\boldsymbol{\omega}}_j = \underset{\boldsymbol{\omega}_j}{\operatorname{Argmin}} &\|\mathbf{E}_j^J - \mathbf{d}_j\boldsymbol{\omega}_j^T\mathbf{Y}^J\|_F^2 \\ &\text{Subject To } \sigma_i\boldsymbol{\omega}_j^T\mathbf{y}_i \geq \lambda_j \quad \forall i \in \mathbf{Y}_j \\ &-\lambda_j < \boldsymbol{\omega}_j^T\mathbf{y}_i < \lambda_j \quad \forall i \in \bar{\mathbf{Y}}_j. \end{aligned} \tag{17}$$

This is a standard QP optimization task, and can be solved using a variety of techniques. Once $\boldsymbol{\omega}_j$ is computed according to (17), we restore the original constraint on $\boldsymbol{\omega}_j$ by normalizing $\{\boldsymbol{\omega}_j, \lambda_j\} \rightarrow \{\alpha_j\boldsymbol{\omega}_j, \alpha_j\lambda_j\}$ with $\alpha_j = 1/\|\boldsymbol{\omega}_j\|_2$, and compute $\mathbf{d}_j$ using (15), which concludes the process.

Appendix B
Rank-One Approximation Solution

In this Appendix we consider the solution to

$$\underset{\mathbf{d},\boldsymbol{\omega}}{\operatorname{argmin}} \|\mathbf{E} - \mathbf{d}\boldsymbol{\omega}^T\mathbf{Y}\|_F^2 \quad \text{Subject To } \|\boldsymbol{\omega}\|_2 = 1, \tag{18}$$

where $\mathbf{E}, \mathbf{Y} \in \mathbb{R}^{N\times R}$, and are assumed to be full-rank. To derive the solution, we first assume that $\mathbf{Y}\mathbf{Y}^T = \mathbf{I}$ (i.e., $\mathbf{Y}^T$ is a tight frame). In this case we have[6]:

$$\begin{aligned}
&\|\mathbf{E} - \mathbf{d}\boldsymbol{\omega}^T\mathbf{Y}\|_F^2 \\
&\quad = \operatorname{tr}\left\{\mathbf{E}^T\mathbf{E} - 2\mathbf{E}^T\mathbf{d}\boldsymbol{\omega}^T\mathbf{Y} + \mathbf{Y}^T\boldsymbol{\omega}\mathbf{d}^T\mathbf{d}\boldsymbol{\omega}^T\mathbf{Y}\right\} \\
&\quad = \operatorname{tr}\left\{\mathbf{E}^T\mathbf{E} - 2\mathbf{Y}\mathbf{E}^T\mathbf{d}\boldsymbol{\omega}^T + \mathbf{Y}\mathbf{Y}^T\boldsymbol{\omega}\mathbf{d}^T\mathbf{d}\boldsymbol{\omega}^T\right\} \\
&\quad = \operatorname{tr}\left\{\mathbf{E}^T\mathbf{E} - 2\mathbf{Y}\mathbf{E}^T\mathbf{d}\boldsymbol{\omega}^T + \boldsymbol{\omega}\mathbf{d}^T\mathbf{d}\boldsymbol{\omega}^T\right\} \\
&\quad + \operatorname{tr}\left\{\mathbf{Y}\mathbf{E}^T\mathbf{E}\mathbf{Y}^T - \mathbf{Y}\mathbf{E}^T\mathbf{E}\mathbf{Y}^T\right\} \\
&\quad = \operatorname{tr}\left\{\mathbf{E}^T\mathbf{E} - \mathbf{Y}\mathbf{E}^T\mathbf{E}\mathbf{Y}^T\right\} \\
&\quad + \operatorname{tr}\left\{\mathbf{Y}\mathbf{E}^T\mathbf{E}\mathbf{Y}^T - 2\mathbf{Y}\mathbf{E}^T\mathbf{d}\boldsymbol{\omega}^T + \boldsymbol{\omega}\mathbf{d}^T\mathbf{d}\boldsymbol{\omega}^T\right\} \\
&\quad = \operatorname{tr}\left\{\mathbf{E}^T\mathbf{E} - \mathbf{Y}\mathbf{E}^T\mathbf{E}\mathbf{Y}^T\right\} + \|\mathbf{E}\mathbf{Y}^T - \mathbf{d}\boldsymbol{\omega}^T\|_F^2.
\end{aligned}$$

Since the left term is constant in the optimization, we see that when $\mathbf{Y}^T$ is a tight frame, (18) is equivalent to:

$$\underset{\mathbf{d},\boldsymbol{\omega}}{\operatorname{Argmin}} \|\mathbf{E}\mathbf{Y}^T - \mathbf{d}\boldsymbol{\omega}^T\|_F^2 \quad \text{Subject To} \quad \|\boldsymbol{\omega}\|_2 = 1, \tag{19}$$

which is a standard rank-one approximation of $\mathbf{E}\mathbf{Y}^T$ whose solution is given by the singular vector pair corresponding to the largest singular value of $\mathbf{E}\mathbf{Y}^T$.

For a general full-rank $\mathbf{Y}$, we compte its SVD, $\mathbf{Y} = \mathbf{U}\mathbf{S}\mathbf{V}^T$. We denote the singular values on the diagonal of $\mathbf{S}$ by $s_1 \ldots s_N$, and let $\boldsymbol{\Delta} = \operatorname{diag}(s_1^{-1}, \ldots, s_N^{-1})$. We note that the matrix

$$\tilde{\mathbf{Y}} = \boldsymbol{\Delta}\mathbf{U}^T\mathbf{Y} = \boldsymbol{\Delta}\mathbf{S}\mathbf{V}^T = \mathbf{I}_{N\times R}\mathbf{V}^T$$

satisfies $\tilde{\mathbf{Y}}\tilde{\mathbf{Y}}^T = \mathbf{I}$, and thus $\tilde{\mathbf{Y}}^T$ is a tight frame.

Returning to problem (18), we can now write

$$\begin{aligned}
\|\mathbf{E} - \mathbf{d}\boldsymbol{\omega}^T\mathbf{Y}\|_F^2 &= \|\mathbf{E} - \mathbf{d}\boldsymbol{\omega}^T(\boldsymbol{\Delta}\mathbf{U}^T)^{-1}\boldsymbol{\Delta}\mathbf{U}^T\mathbf{Y}\|_F^2 \\
&= \|\mathbf{E} - \mathbf{d}\boldsymbol{\omega}^T\mathbf{U}\boldsymbol{\Delta}^{-1}\tilde{\mathbf{Y}}\|_F^2,
\end{aligned}$$

which leads to the optimization task:

$$\underset{\mathbf{d},\boldsymbol{\omega}}{\operatorname{Argmin}} \|\mathbf{E} - \mathbf{d}\boldsymbol{\omega}^T\mathbf{U}\boldsymbol{\Delta}^{-1}\tilde{\mathbf{Y}}\|_F^2. \tag{20}$$

Since $\tilde{\mathbf{Y}}^T$ is a tight frame, (20) can be solved for $\mathbf{d}$ and $\tilde{\boldsymbol{\omega}}^T := \boldsymbol{\omega}^T\mathbf{U}\boldsymbol{\Delta}^{-1}$ using (19). Once $\tilde{\boldsymbol{\omega}}^T$ is computed, the computation is completed by setting $\boldsymbol{\omega}^T = \tilde{\boldsymbol{\omega}}^T\boldsymbol{\Delta}\mathbf{U}^T$, and renormalizing the obtained $\mathbf{d}$ and $\boldsymbol{\omega}$ such that $\|\boldsymbol{\omega}\|_2 = 1$. The resulting procedure is summarized in Fig. 2.

[6]For simplicity of presentation, we slightly abuse notation and allow differently-sized matrices to be summed within the trace operator. These should be interpreted as summing the matrix traces.

Acknowledgment

The authors would like to thank the anonymous reviewers and the editor-in-chief for their suggestions which greatly improved the paper.

References

[1] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. New York, NY, USA: Springer, 2010.
[2] M. Elad, P. Milanfar, and R. Rubinstein, "Analysis versus synthesis in signal priors," *Inverse Problems*, vol. 23, no. 3, pp. 947–968, 2007.
[3] S. Nam, M. E. Davies, M. Elad, and R. Gribonval, "The cosparse analysis model and algorithms," *Appl. Comput. Harmon. Anal.*, vol. 34, no. 1, pp. 30–56, Jan. 2013.
[4] 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.
[5] D. L. Donoho and J. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.
[6] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proc. IEEE*, vol. 98, no. 6, pp. 1045–1057, 2010.
[7] M. Elad, "Why simple shrinkage is still relevant for redundant representations?," *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5559–5569, 2006.
[8] B. Ophir, M. Elad, N. Bertin, and M. D. Plumbley, "Sequential minimal eigenvalues—An approach to analysis dictionary learning," presented at the Eur. Signal Process. Conf. (EUSIPCO), Barcelona, Spain, Aug. 29, 2011.
[9] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies, "Analysis operator learning for overcomplete cosparse representations," presented at the Eur. Signal Process. Conf. (EUSIPCO), Barcelona, Spain, Aug. 29, 2011.
[10] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies, "Noise aware analysis operator learning for approximately cosparse signals," presented at the Int. Conf. Acoust., Speech, Signal Process. (ICASSP), Kyoto, Japan, Mar. 25–30, 2012.
[11] R. Rubinstein, T. Peleg, and M. Elad, "Analysis K-SVD: A dictionary-learning algorithm for the analysis sparse model," *IEEE Trans. Signal Process.*, vol. 61, no. 3, pp. 661–677, 2013.
[12] S. Hawe, M. Kleinsteuber, and K. Diepold, "Analysis operator learning and its application to image reconstruction," *IEEE Trans. Image Process.*, vol. 22, no. 6, Jun. 2013.
[13] G. Peyré and J. Fadili, "Learning analysis sparsity priors," presented at the Sampl. Theory Appl. (SampTA), Singapore, May 2–6, 2011.
[14] Y. Chen, T. Pock, and H. Bischof, "Learning l_1-based analysis and synthesis sparsity priors using bi-level optimization," presented at the Workshop Anal. Operat. Learn. vs. Diction. Learn.: Fraternal Twins in Sparse Model., South Lake Tahoe, CA, USA, Dec. 7, 2012.
[15] D. L. Donoho and I. M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *J. Amer. Statist. Assoc.*, vol. 90, no. 432, pp. 1200–1224, 1995.
[16] D. L. Donoho, I. M. Johnstone, G. Kerkyacharian, and D. Picard, "Wavelet shrinkage: Asymptopia?," *J. Roy. Statist. Soc. Ser. B (Methodolog.)*, vol. 57, no. 2, pp. 301–369, 1995.
[17] S. G. Chang, B. Yu, and M. Vetterli, "Adaptive wavelet thresholding for image denoising and compression," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1532–1546, 2000.
[18] J. L. Starck, E. J. Candès, and D. L. Donoho, "The curvelet transform for image denoising," *IEEE Trans. Image Process.*, vol. 11, no. 6, pp. 670–684, 2002.
[19] F. Abramovich, Y. Benjamini, D. L. Donoho, and I. M. Johnstone, "Adapting to unknown sparsity by controlling the false discovery rate," *Ann. Statist.*, vol. 34, no. 2, pp. 584–653, 2006.
[20] M. Aharon, M. Elad, and A. M. Bruckstein, "The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.
[21] The CVG Granada Image Database [Online]. Available: http://decsai.ugr.es/cvg/dbimagenes/

[22] H. Takeda, S. Farsiu, and P. Milanfar, "Deblurring using regularized locally-adaptive kernel regression," *IEEE Trans. Image Process.*, vol. 17, no. 4, pp. 550–563, 2008.
[23] Regularized Kernel Regression-Based Deblurring [Online]. Available: http://users.soe.ucsc.edu/~htakeda/AKTV.htm
[24] R. Neelamani, H. Choi, and R. Baraniuk, "ForWaRD: Fourier-wavelet regularized deconvolution for ill-conditioned systems," *IEEE Trans. Signal Process.*, vol. 52, no. 2, pp. 418–433, 2004.
[25] V. Katkovnik, K. Egiazarian, and J. Astola, "A spatially adaptive nonparametric regression image deblurring," *IEEE Trans. Image Process.*, vol. 14, no. 10, pp. 1469–1478, 2005.
[26] A. Danielyan, V. Katkovnik, and K. Egiazarian, "BM3D frames and variational image deblurring," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1715–1728, 2012.
[27] Fourier-Wavelet Regularized Deconvolution (Matlab Software Package) [Online]. Available: http://dsp.rice.edu/software/forward
[28] Y. Hel-Or and D. Shaked, "A discriminative approach for wavelet denoising," *IEEE Trans. Image Process.*, vol. 17, no. 4, pp. 443–457, 2008.
[29] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.
[30] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.
[31] A. Adler, Y. Hel-Or, and M. Elad, "A shrinkage learning approach for single image super-resolution with overcomplete representations," presented at the Eur. Conf. Comput. Vis. (ECCV), Crete-Greece, Sep. 5–11, 2010.
[32] A. Adler, Y. Hel-Or, and M. Elad, "A weighted discriminative approach for image denoising with overcomplete representations," presented at the Int. Conf. Acoust., Speech, Signal Process. (ICASSP), Dallas, TX, USA, Mar. 14–19, 2010.
[33] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
[34] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. L. Cun, "Learning convolutional feature hierarchies for visual recognition," presented at the Conf. Neural Inf. Process. Syst. (NIPS), Vancouver, BC, Canada, Dec. 6–8, 2010.
[35] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?," in *Proc. Int. Conf. Comput. Vision (ICCV)*, Kyoto, Japan, Sep. 2009, pp. 2146–2153.
[36] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010.
[37] K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "Fast inference in sparse coding algorithms with applications to object recognition," 2010, arXiv preprint arXiv:1010.3467 [Online]. Available: http://arxiv.org/abs/1010.3467
[38] R. Rubinstein, M. Zibulevsky, and M. Elad, "Double sparsity: Learning sparse dictionaries for sparse signal approximation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1553–1564, 2010.

Ron Rubinstein (M'13) received the Ph.D. degree in computer science from 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. He currently holds a research scientist position at Intel, and is an adjunct lecturer at the Technion. He participated in the IPAM Short Course on Sparse Representation in May 2007. During 2008, he was a research intern in Sharp Laboratories of America, Camas, WA.

Dr. Rubinstein received the Technion Teaching Assistant's excellence award in 2005 and Lecturer's commendation for excellence in 2008, 2010, and 2011.

Michael Elad (M'98–SM'08–F'12) received the B.Sc. (1986), M.Sc. (1988), and D.Sc. (1997) degrees from the Department of Electrical Engineering, The Technion, Israel.

Since 2003, he has been a faculty member at the Computer-Science Department at The Technion, and since 2010, holds a full-professorship position. He works in the field of signal and image processing, specializing on inverse problems, sparse representations, and superresolution.

Dr. Elad received the Technion's Best Lecturer award six times. He received 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. He is serving as an Associate Editor for SIAM, SIIMS, and ACHA. He is also serving as a Senior Editor for IEEE SIGNAL PROCESSING LETTERS.