K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation

Michal Aharon, Michael Elad, and Alfred Bruckstein

***Abstract*—In recent years there has been a growing interest in the study of sparse representation of signals. Using an overcomplete dictionary that contains prototype signal-atoms, signals are described by sparse linear combinations of these atoms. Applications that use sparse representation are many and include compression, regularization in inverse problems, feature extraction, and more. Recent activity in this field has concentrated mainly on the study of pursuit algorithms that decompose signals with respect to a given dictionary. Designing dictionaries to better fit the above model can be done by either selecting one from a prespecified set of linear transforms or adapting the dictionary to a set of training signals. Both of these techniques have been considered, but this topic is largely still open. In this paper we propose a novel algorithm for adapting dictionaries in order to achieve sparse signal representations. Given a set of training signals, we seek the dictionary that leads to the best representation for each member in this set, under strict sparsity constraints. We present a new method—the K-SVD algorithm—generalizing the K-means clustering process. K-SVD is an iterative method that alternates between sparse coding of the examples based on the current dictionary and a process of updating the dictionary atoms to better fit the data. The update of the dictionary columns is combined with an update of the sparse representations, thereby accelerating convergence. The K-SVD algorithm is flexible and can work with any pursuit method (e.g., basis pursuit, FOCUSS, or matching pursuit). We analyze this algorithm and demonstrate its results both on synthetic tests and in applications on real image data.**

***Index Terms*—Atom decomposition, basis pursuit, codebook, dictionary, FOCUSS, gain-shape VQ, K-means, K-SVD, matching pursuit, sparse representation, training, vector quantization.**

I. Introduction

A. *Sparse Representation of Signals*

RECENT years have witnessed a growing interest in the search for sparse representations of signals. Using an overcomplete dictionary matrix $\mathbf{D} \in \mathbb{R}^{n\times K}$ that contains K prototype signal-atoms for columns, $\{\mathbf{d}_j\}_{j=1}^{K}$, a signal $\mathbf{y} \in \mathbb{R}^n$ can be represented as a sparse linear combination of these atoms. The representation of $\mathbf{y}$ may either be exact $\mathbf{y} = \mathbf{Dx}$ or approximate, $\mathbf{y} \approx \mathbf{Dx}$, satisfying $\|\mathbf{y} - \mathbf{Dx}\|_p \leq \epsilon$. The vector $\mathbf{x} \in \mathbb{R}^K$ contains the representation coefficients of the signal $\mathbf{y}$. In approximation methods, typical norms used for measuring the deviation are the ℓ^p-norms for $p = 1, 2,$ and ∞. In this paper, we shall concentrate on the case of $p = 2$.

If $n < K$ and $\mathbf{D}$ is a full-rank matrix, an infinite number of solutions are available for the representation problem, hence constraints on the solution must be set. The solution with the fewest number of nonzero coefficients is certainly an appealing representation. This sparsest representation is the solution of either

$$(P_0) \quad \min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \mathbf{y} = \mathbf{Dx} \tag{1}$$

or

$$(P_{0,\epsilon}) \quad \min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \|\mathbf{y} - \mathbf{Dx}\|_2 \leq \epsilon \tag{2}$$

where $\|\cdot\|_0$ is the l^0 norm, counting the nonzero entries of a vector.

Applications that can benefit from the sparsity and overcompleteness concepts (together or separately) include compression, regularization in inverse problems, feature extraction, and more. Indeed, the success of the JPEG2000 coding standard can be attributed to the sparsity of the wavelet coefficients of natural images [1]. In denoising, wavelet methods and shift-invariant variations that exploit overcomplete representation are among the most effective known algorithms for this task [2]–[5]. Sparsity and overcompleteness have been successfully used for dynamic range compression in images [6], separation of texture and cartoon content in images [7], [8], inpainting [9], and more.

Extraction of the sparsest representation is a hard problem that has been extensively investigated in the past few years. We review some of the most popular methods in Section II. In all those methods, there is a preliminary assumption that the dictionary is known and fixed. In this paper, we address the issue of designing the proper dictionary in order to better fit the sparsity model imposed.

B. *The Choice of the Dictionary*

An overcomplete dictionary $\mathbf{D}$ that leads to sparse representations can either be chosen as a prespecified set of functions or designed by adapting its content to fit a given set of signal examples.

Choosing a prespecified transform matrix is appealing because it is simpler. Also, in many cases it leads to simple and fast algorithms for the evaluation of the sparse representation. This is indeed the case for overcomplete wavelets, curvelets, contourlets, steerable wavelet filters, short-time Fourier transforms, and more. Preference is typically given to tight frames that can easily be pseudoinverted. The success of such dictionaries in applications depends on how suitable they are to sparsely describe the signals in question. Multiscale analysis with oriented basis

Manuscript received December 26, 2004; revised January 21, 2006. This work was supported in part by The Technion under V.P.R. funds and by the Israel Science Foundation under Grant 796/05. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Steven L. Grant.

The authors are with the Department of Computer Science, The Technion—Israel Institute of Technology, Haifa 32000, Israel (e-mail: michalo@cs.technion.ac.il; elad@cs.technion.ac.il; freddy@cs.technion.ac.il).

Digital Object Identifier 10.1109/TSP.2006.881199

functions and a shift-invariant property are guidelines in such constructions.

In this paper, we consider a different route for designing dictionaries $\mathbf{D}$ based on learning. Our goal is to find the dictionary $\mathbf{D}$ that yields sparse representations for the training signals. We believe that such dictionaries have the potential to outperform commonly used predetermined dictionaries. With ever-growing computational capabilities, computational cost may become secondary in importance to the improved performance achievable by methods that adapt dictionaries for special classes of signals.

C. *Our Paper's Contribution and Structure*

In this paper, we present a novel algorithm for adapting dictionaries so as to represent signals sparsely. Given a set of training signals $\{\mathbf{y}_i\}_{i=1}^N$, we seek the dictionary $\mathbf{D}$ that leads to the best possible representations for each member in this set with strict sparsity constraints. We introduce the K-SVD algorithm that addresses the above task, generalizing the K-means algorithm. The K-SVD is an iterative method that alternates between sparse coding of the examples based on the current dictionary and an update process for the dictionary atoms so as to better fit the data. The update of the dictionary columns is done jointly with an update of the sparse representation coefficients related to it, resulting in accelerated convergence. The K-SVD algorithm is flexible and can work with any pursuit method, thereby tailoring the dictionary to the application in mind. In this paper, we present the K-SVD algorithm, analyze it, discuss its relation to prior art, and prove its superior performance. We demonstrate the K-SVD results in both synthetic tests and applications involving real image data.

In Section II, we survey pursuit algorithms that are later used by the K-SVD, together with some recent theoretical results justifying their use for sparse coding. In Section III, we refer to recent work done in the field of sparse-representation dictionary design and describe different algorithms that were proposed for this task. In Section IV, we describe our algorithm, its possible variations, and its relation to previously proposed methods. The K-SVD results on synthetic data are presented in Section V, and some preliminary applications involving real image data are given in Section VI. We conclude and discuss future possible research direction in Section VII.

II. Sparse Coding: Prior Art

Sparse coding is the process of computing the representation coefficients $\mathbf{x}$ based on the given signal $\mathbf{y}$ and the dictionary $\mathbf{D}$. This process, commonly referred to as "atom decomposition," requires solving (1) or (2), and this is typically done by a "pursuit algorithm" that finds an approximate solution. In this section, we briefly discuss several such algorithms and their prospects for success. A more detailed description of those methods can be found in [10]. Sparse coding is a necessary stage in the K-SVD method we develop later in this paper, hence it is important to have a good overview of methods for achieving it.

Exact determination of sparsest representations proves to be an NP-hard problem [11]. Thus, approximate solutions are considered instead, and in the past decade or so several efficient pursuit algorithms have been proposed. The simplest ones are the *matching pursuit* (MP) [12] and the *orthogonal matching pursuit* (OMP) algorithms [13]–[16]. These are greedy algorithms that select the dictionary atoms sequentially. These methods are very simple, involving the computation of inner products between the signal and dictionary columns, and possibly deploying some least squares solvers. Both (1) and (2) are easily addressed by changing the stopping rule of the algorithm.

A second well-known pursuit approach is the *basis pursuit* (BP) [17]. It suggests a convexification of the problems posed in (1) and (2) by replacing the ℓ^0-norm with an ℓ^1-norm. The focal underdetermined system solver (FOCUSS) is very similar, using the ℓ^p-norm with $p \leq 1$ as a replacement for the ℓ^0-norm [18]–[21]. Here, for $p < 1$, the similarity to the true sparsity measure is better but the overall problem becomes nonconvex, giving rise to local minima that may mislead in the search for solutions. Lagrange multipliers are used to convert the constraint into a penalty term, and an iterative method is derived based on the idea of iterated reweighed least squares that handles the ℓ^p-norm as an ℓ^2 weighted norm.

Both the BP and the FOCUSS can be motivated based on *maximum a posteriori* (MAP) estimation, and indeed several works used this reasoning directly [22]–[25]. The MAP can be used to estimate the coefficients as random variables by maximizing the posterior $P(\mathbf{x}|\mathbf{y},\mathbf{D}) \propto P(\mathbf{y}|\mathbf{D},\mathbf{x})P(\mathbf{x})$. The prior distribution on the coefficient vector $\mathbf{x}$ is assumed to be a super-Gaussian (i.i.d.) distribution that favors sparsity. For the Laplace distribution, this approach is equivalent to BP [22].

Extensive study of these algorithms in recent years has established that if the sought solution $\mathbf{x}$ is sparse enough, these techniques recover it well in the exact case [16], [26]–[30]. Further work considered the approximated versions and has shown stability in recovery of $\mathbf{x}$ [31], [32]. The recent front of activity revisits those questions within a probabilistic setting, obtaining more realistic assessments on pursuit algorithm performance and success [33]–[35]. The properties of the dictionary $\mathbf{D}$ set the limits on the sparsity of the coefficient vector that consequently leads to its successful evaluation.

III. Design of Dictionaries: Prior Art

We now come to the main topic of the paper, the training of dictionaries based on a set of examples. Given such a set $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$, we assume that there exists a dictionary $\mathbf{D}$ that gave rise to the given signal examples via sparse combinations, i.e., we assume that there exists $\mathbf{D}$, so that solving (P_0) for each example $\mathbf{y}_k$ gives a sparse representation $\mathbf{x}_k$. It is in this setting that we ask what the proper dictionary $\mathbf{D}$ is.

A. *Generalizing the* K*-Means?*

There is an intriguing relation between sparse representation and clustering (i.e., vector quantization). This connection has previously been mentioned in several reports [36]–[38]. In clustering, a set of descriptive vectors $\{\mathbf{d}_k\}_{k=1}^K$ is learned, and each sample is represented by one of those vectors (the one closest to it, usually in the ℓ^2 distance measure). We may think of this as an extreme sparse representation, where only one atom is allowed in the signal decomposition, and furthermore, the coefficient multiplying it must be one. There is a variant of the vector quantization (VQ) coding method, called gain-shape VQ, where this coefficient is allowed to vary [39]. In contrast, in sparse representations as discussed in this paper, each example is represented as a linear combination of several vectors $\{\mathbf{d}_k\}_{k=1}^K$. Thus, sparse representations can be referred to as a generalization of the clustering problem.

Since the K-means algorithm (also known as the generalized Lloyd algorithm—GLA [39]) is the most commonly used procedure for training in the vector quantization setting, it is natural to consider generalizations of this algorithm when turning to the problem of dictionary training. The clustering problem and its K-means solution will be discussed in more detail in Section IV-A, since our work approaches the dictionary training problem by generalizing the K-means. Here we shall briefly mention that the K-means process applies two steps per each iteration: i) given $\{\mathbf{d}_k\}_{k=1}^K$, assign the training examples to their nearest neighbor; and ii) given that assignment, update $\{\mathbf{d}_k\}_{k=1}^K$ to better fit the examples.

The approaches to dictionary design that have been tried so far are very much in line with the two-step process described above. The first step finds the coefficients given the dictionary—a step we shall refer to as "sparse coding." Then, the dictionary is updated assuming known and fixed coefficients. The differences between the various algorithms that have been proposed are in the method used for the calculation of coefficients and in the procedure used for modifying the dictionary.

B. Maximum Likelihood Methods

The methods reported in [22]–[25] use probabilistic reasoning in the construction of $\mathbf{D}$. The proposed model suggests that for every example $\mathbf{y}$ the relation

$$\mathbf{y} = \mathbf{D}\mathbf{x} + \mathbf{v} \tag{3}$$

holds true with a sparse representation $\mathbf{x}$ and Gaussian white residual vector $\mathbf{v}$ with variance σ^2. Given the examples $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$, these works consider the likelihood function $P(\mathbf{Y}|\mathbf{D})$ and seek the dictionary that maximizes it. Two assumptions are required in order to proceed: the first is that the measurements are drawn independently, readily providing

$$P(\mathbf{Y}|\mathbf{D}) = \prod_{i=1}^N P(\mathbf{y}_i|\mathbf{D}). \tag{4}$$

The second assumption is critical and refers to the "hidden variable" $\mathbf{x}$. The ingredients of the likelihood function are computed using the relation

$$P(\mathbf{y}_i|\mathbf{D}) = \int P(\mathbf{y}_i, \mathbf{x}|\mathbf{D})d\mathbf{x} = \int P(\mathbf{y}_i|\mathbf{x}, \mathbf{D}) \cdot P(\mathbf{x})d\mathbf{x}. \tag{5}$$

Returning to the initial assumption in (3), we have

$$P(\mathbf{y}_i|\mathbf{x}, \mathbf{D}) = Const \cdot \exp\left\{\frac{1}{2\sigma^2}\|\mathbf{D}\mathbf{x} - \mathbf{y}_i\|^2\right\}. \tag{6}$$

The prior distribution of the representation vector $\mathbf{x}$ is assumed to be such that the entries of $\mathbf{x}$ are zero-mean i.i.d., with Cauchy [24] or Laplace distributions [22], [23]. Assuming for example a Laplace distribution, we get

$$\begin{aligned} &P(\mathbf{y}_i|\mathbf{D}) \\ &= \int P(\mathbf{y}_i|\mathbf{x}, \mathbf{D}) \cdot P(\mathbf{x})d\mathbf{x} \\ &= Const \cdot \int \exp\left\{\frac{1}{2\sigma^2}\|\mathbf{D}\mathbf{x} - \mathbf{y}_i\|^2\right\} \cdot \exp\{\lambda\|\mathbf{x}\|_1\} d\mathbf{x}. \end{aligned} \tag{7}$$

This integration over $\mathbf{x}$ is difficult to evaluate, and indeed, Olshausen and Field [23] handled this by replacing it with the extremal value of $P(\mathbf{y}_i, \mathbf{x}|\mathbf{D})$. The overall problem turns into

$$\begin{aligned} D &= \arg\max_{\mathbf{D}} \sum_{i=1}^N \max_{\mathbf{x}_i} \{P(\mathbf{y}_i, \mathbf{x}_i|\mathbf{D})\} \\ &= \arg\min_{\mathbf{D}} \sum_{i=1}^N \min_{\mathbf{x}_i} \left\{\|\mathbf{D}\mathbf{x}_i - \mathbf{y}_i\|^2 + \lambda\|\mathbf{x}_i\|_1\right\}. \end{aligned} \tag{8}$$

This problem does not penalize the entries of $\mathbf{D}$ as it does for those of $\mathbf{x}_i$. Thus, the solution will tend to increase the dictionary entries' values, in order to allow the coefficients to become closer to zero. This difficulty has been handled by constraining the ℓ^2-norm of each basis element, so that the output variance of the coefficients is kept at an appropriate level [24].

An iterative method was suggested for solving (8). It includes two main steps in each iteration: i) calculate the coefficients $\mathbf{x}_i$ using a simple gradient descent procedure and then ii) update the dictionary using [24]

$$\mathbf{D}^{(n+1)} = \mathbf{D}^{(n)} - \eta \sum_{i=1}^N (\mathbf{D}^{(n)}\mathbf{x}_i - \mathbf{y}_i)\mathbf{x}_i^T. \tag{9}$$

This idea of iterative refinement, mentioned before as a generalization of the K-means algorithm, was later used again by other researchers, with some variations [36], [37], [40]–[42].

A different approach to handle the integration in (7) was suggested by Lewicki and Sejnowski [25]. They approximated the posterior as a Gaussian, enabling an analytic solution of the integration. This allows an objective comparison of different image models (basis or priors). It also removes the need for the additional rescaling that enforces the norm constraint. However, this model may be too limited in describing the true behaviors expected. This technique and closely related ones have been referred to as *approximated ML* techniques [37].

There is an interesting relation between the above method and the independent component analysis (ICA) algorithm [43]. The latter handles the case of a complete dictionary (the number of elements equals the dimensionality) without assuming additive noise. The above method is then similar to ICA in that the algorithm can be interpreted as trying to maximize the mutual information between the inputs (samples) and the outputs (the coefficients) [24], [22], [25].

C. The MOD Method

An appealing dictionary training algorithm, named *method of optimal directions* (MOD), is presented by Engan *et al.* [36],

[40], [41]. This method follows more closely the K-means outline, with a sparse coding stage that uses either OMP or FOCUSS followed by an update of the dictionary. The main contribution of the MOD method is its simple way of updating the dictionary. Assuming that the sparse coding for each example is known, we define the errors $\mathbf{e}_i = \mathbf{y}_i - \mathbf{D}\mathbf{x}_i$. The overall representation mean square error is given by

$$\|\mathbf{E}\|_F^2 = \|[\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N]\|_F^2 = \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2. \tag{10}$$

Here we have concatenated all the examples $\mathbf{y}_i$ as columns of the matrix $\mathbf{Y}$ and similarly gathered the representations coefficient vectors $\mathbf{x}_i$ to build the matrix $\mathbf{X}$. The notation $\|\mathbf{A}\|_F$ stands for the Frobenius norm, defined as $\|\mathbf{A}\|_F = \sqrt{\sum_{ij} A_{ij}^2}$.

Assuming that $\mathbf{X}$ is fixed, we can seek an update to $\mathbf{D}$ such that the above error is minimized. Taking the derivative of (10) with respect to $\mathbf{D}$, we obtain the relation $(\mathbf{Y} - \mathbf{D}\mathbf{X})\mathbf{X}^T = 0$, leading to

$$\mathbf{D}^{(n+1)} = \mathbf{Y}\mathbf{X}^{(n)^T} \cdot (\mathbf{X}^{(n)}\mathbf{X}^{(n)^T})^{-1}. \tag{11}$$

MOD is closely related to the work by Olshausen and Field, with improvements both in the sparse coding and the dictionary update stages. Whereas the work in [23], [24], and [22] applies a steepest descent to evaluate $\mathbf{x}_i$, those are evaluated much more efficiently with either OMP or FOCUSS. Similarly, in updating the dictionary, the update relation given in (11) is the best that can be achieved for fixed $\mathbf{X}$. The iterative steepest descent update in (9) is far slower. Interestingly, in both stages of the algorithm, the difference is in deploying a second order (Newtonian) update instead of a first-order one. Looking closely at the update relation in (9), it could be written as

$$\begin{aligned}\mathbf{D}^{(n+1)} &= \mathbf{D}^{(n)} + \eta\mathbf{E}\mathbf{X}^{(n)^T} \\ &= \mathbf{D}^{(n)} + \eta(\mathbf{Y} - \mathbf{D}^{(n)}\mathbf{X}^{(n)})\mathbf{X}^{(n)^T} \\ &= \mathbf{D}^{(n)}(\mathbf{I} - \eta\mathbf{X}^{(n)}\mathbf{X}^{(n)^T}) + \eta\mathbf{Y}\mathbf{X}^{(n)^T}. \end{aligned} \tag{12}$$

Using infinitely many iterations of this sort, and using small enough η, this leads to a steady-state outcome that is exactly the MOD update matrix (11). Thus, while the MOD method assumes known coefficients at each iteration and derives the best possible dictionary, the ML method by Olshausen and Field only gets closer to this best current solution and then turns to calculate the coefficients. Note, however, that in both methods a normalization of the dictionary columns is required and done.

D. Maximum A-Posteriori Probability Approach

The same researchers that conceived the MOD method also suggested a MAP probability setting for the training of dictionaries, attempting to merge the efficiency of the MOD with a natural way to take into account preferences in the recovered dictionary. In [37], [41], [42], and [44], a probabilistic point of view is adopted, very similar to the ML methods discussed above. However, rather than working with the likelihood function $P(\mathbf{Y}|\mathbf{D})$, the posterior $P(\mathbf{D}|\mathbf{Y})$ is used. Using Bayes' rule, we have $P(\mathbf{D}|\mathbf{Y}) \propto P(\mathbf{Y}|\mathbf{D})P(\mathbf{D})$, and thus we can use the likelihood expression as before and add a prior on the dictionary as a new ingredient.

These works considered several priors $P(\mathbf{D})$ and proposed corresponding formulas for the dictionary. The efficiency of the MOD in these methods is manifested in the efficient sparse coding, which is carried out with FOCUSS. The proposed algorithms in this family deliberately avoid a direct minimization with respect to $\mathbf{D}$ as in MOD, due to the prohibitive $n \times n$ matrix inversion required. Instead, iterative gradient descent is used.

When no prior is chosen, the update formula is the very one used by Olshausen and Field, as in (9). A prior that constrains $\mathbf{D}$ to have a unit Frobenius norm leads to the update formula

$$\mathbf{D}^{(n+1)} = \mathbf{D}^{(n)} + \eta\mathbf{E}\mathbf{X}^T + \eta \cdot \mathrm{tr}\left(\mathbf{X}\mathbf{E}^T\mathbf{D}^{(n)}\right)\mathbf{D}^{(n)}. \tag{13}$$

As can be seen, the first two terms are the same as in (9). The last term compensates for deviations from the constraint. This case allows different columns in $\mathbf{D}$ to have different norm values. As a consequence, columns with small norm values tend to be underused, as the coefficients they need are larger and as such more penalized.

This led to the second prior choice, constraining the columns of $\mathbf{D}$ to have a unit ℓ^2-norm. The new update equation formed is given by

$$\mathbf{d}_i^{(n+1)} = \mathbf{d}_i^{(n)} + \eta\left(\mathbf{I} - \mathbf{d}_i^{(n)}\mathbf{d}_i^{(n)^T}\right)\mathbf{E} \cdot \mathbf{x}_i^T \tag{14}$$

where $\mathbf{x}_i^T$ is the ith column in the matrix $\mathbf{X}^T$.

Compared to the MOD, this line of work provides slower training algorithms. Simulations reported in [37], [41], [42], [44] on synthetic and real image data seem to provide encouraging results.

E. Unions of Orthonormal Bases

The very recent work reported in [45] considers a dictionary composed as a union of orthonormal bases

$$\mathbf{D} = [\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_L]$$

where $\mathbf{D}_j \in \mathbb{R}^{n\times n}, j = 1, 2, \ldots, L$ are orthonormal matrices. Such a dictionary structure is quite restrictive, but its updating may potentially be made more efficient.

The coefficients of the sparse representations $\mathbf{X}$ can be decomposed to L pieces, each referring to a different orthonormal basis. Thus

$$\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_L]^T$$

where $\mathbf{X}_i$ is the matrix containing the coefficients of the orthonormal dictionary $\mathbf{D}_i$.

One of the major advantages of the union of orthonormal bases is the relative simplicity of the pursuit algorithm needed for the sparse coding stage. The coefficients are found using the block coordinate relaxation algorithm [46]. This is an appealing way to solve $(P_{1,\epsilon})$ as a sequence of simple shrinkage steps, such that at each stage $\mathbf{X}_i$ is computed while keeping all the other pieces of $\mathbf{X}$ fixed. Thus, this evaluation amounts to a simple shrinkage.

Assuming known coefficients, the proposed algorithm updates each orthonormal basis $\mathbf{D}_j$ sequentially. The update of $\mathbf{D}_j$ is done by first computing the residual matrix

$$\mathbf{E}_j = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N] = \mathbf{Y} - \sum_{i \neq j} \mathbf{D}_i \mathbf{X}_i.$$

Then, by computing the singular value decomposition of the matrix $\mathbf{E}_j \mathbf{X}_j^T = \mathbf{U} \mathbf{\Lambda} \mathbf{V}^T$, the update of the jth orthonormal basis is done by $\mathbf{D}_j = \mathbf{U} \mathbf{V}^T$. This update rule is obtained by solving a constrained least squares problem with $\|\mathbf{E}_j - \mathbf{D}_j \mathbf{X}_j\|_F^2$ as the penalty term, assuming fixed coefficients $\mathbf{X}_j$ and error $\mathbf{E}_j$. The constraint is over the feasible matrices $\mathbf{D}_j$, which are forced to be orthonormal. This way the proposed algorithm improves each matrix $\mathbf{D}_j$ separately, by replacing the role of the data matrix $\mathbf{Y}$ in the residual matrix $\mathbf{E}_j$, as the latter should be represented by this updated basis.

Compared to previously mentioned training algorithms, the work reported in [45] is different in two important ways: beyond the evident difference of using a structured dictionary rather than a free one, a second major difference is in the proposed sequential update of the dictionary. This update algorithm reminds of the updates done in the K-means. Interestingly, experimental results reported in [45] show weak performance compared to previous methods. This might be explained by the unfavorable coupling of the dictionary parts and their corresponding coefficients, which is overlooked in the update.

F. Summary of the Prior Art

Almost all previous methods can essentially be interpreted as generalizations of the K-means algorithm, and yet, there are marked differences between these procedures. In the quest for a successful dictionary training algorithm, there are several desirable properties.

- *Flexibility:* The algorithm should be able to run with any pursuit algorithm, and this way enable choosing the one adequate for the run-time constraints or the one planned for future usage in conjunction with the obtained dictionary. Methods that decouple the sparse-coding stage from the dictionary update readily have such a property. Such is the case with the MOD and the MAP-based methods.
- *Simplicity:* Much of the appeal of a proposed dictionary training method has to do with how simple it is, and more specifically, how similar it is to K-means. We should have an algorithm that may be regarded as a natural generalization of the K-means. The algorithm should emulate the ease with which the K-means is explainable and implementable. Again, the MOD seems to have made substantial progress in this direction, although, as we shall see, there is still room for improvement.
- *Efficiency:* The proposed algorithm should be numerically efficient and exhibit fast convergence. The methods described above are all quite slow. The MOD, which has a second-order update formula, is nearly impractical for very large number of dictionary columns because of the matrix inversion step involved. Also, in all the above formulations, the dictionary columns are updated before turning to reevaluate the coefficients. As we shall see later, this approach inflicts a severe limitation on the training speed.
- *Well-Defined Objective:* For a method to succeed, it should have a well-defined objective function that measures the quality of the solution obtained. This almost trivial fact was overlooked in some of the preceding work in this field. Hence, even though an algorithm can be designed to greedily improve the representation mean square error (MSE) and the sparsity, it may happen that the algorithm leads to aimless oscillations in terms of a global objective measure of quality.

IV. The K-SVD Algorithm

In this section, we introduce the K-SVD algorithm for training of dictionaries. This algorithm is flexible and works in conjunction with any pursuit algorithm. It is simple and designed to be a truly direct generalization of the K-means. As such, when forced to work with one atom per signal, it trains a dictionary for the gain-shape VQ. When forced to have a unit coefficient for this atom, it exactly reproduces the K-means algorithm. The K-SVD is highly efficient, due to an effective sparse coding and a Gauss–Seidel-like accelerated dictionary update method. The algorithm's steps are coherent with each other, both working towards the minimization of a clear overall objective function.

We start our discussion with a description of the K-means, setting the notation for the rest of this section. While this may seem superfluous, we will use the very description of the K-means to derive the K-SVD as its direct extension. We then discuss some of the K-SVD properties and implementation issues.

A. K-*Means Algorithm for Vector Quantization*

A codebook that includes K codewords (representatives) is used to represent a wide family of vectors (signals) $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$ $(N \gg K)$ by nearest neighbor assignment. This leads to efficient compression or description of those signals as clusters in $\mathbb{R}^n$ surrounding the chosen codewords. As a side note, we remind the reader that based on the expectation maximization procedure, the K-means can be extended to suggest a fuzzy assignment and a covariance matrix per each cluster, so that the data are modelled as a mixture of Gaussians [47].

The dictionary of VQ codewords is typically trained using the K-means algorithm, and as we have argued before, this has a close resemblance to the problem studied in this paper. We denote the codebook matrix by $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K]$, the codewords being the columns. When $\mathbf{C}$ is given, each signal is represented as its closest codeword (under ℓ^2-norm distance). We can write $\mathbf{y}_i = \mathbf{C}\mathbf{x}_i$, where $\mathbf{x}_i = \mathbf{e}_j$ is a vector from the trivial basis, with all zero entries except a one in the jth position. The index j is selected such that

$$\forall_{k \neq j} \; \|\mathbf{y}_i - \mathbf{C}\mathbf{e}_j\|_2^2 \leq \|\mathbf{y}_i - \mathbf{C}\mathbf{e}_k\|_2^2.$$

This is considered as an extreme case of sparse coding in the sense that only one atom is allowed to participate in the construction of $\mathbf{y}_i$, and the coefficient is forced to be 1. The repre-

```
Task: Find the best possible codebook to represent the data samples $\{\mathbf{y}_i\}_{i=1}^N$ by nearest neighbor, by solving
    $\min_{\mathbf{C},\mathbf{X}} \left\{ \|\mathbf{Y}-\mathbf{CX}\|_F^2 \right\}$ subject to $\forall i,\ \mathbf{x}_i = \mathbf{e}_k$ for some $k$.
Initialization : Set the codebook matrix $\mathbf{C}^{(0)} \in \mathbf{R}^{n\times K}$. Set $J = 1$.
Repeat until convergence (use stop rule):
  • Sparse Coding Stage: Partition the training samples $\mathbf{Y}$ into $K$ sets
        $(R_1^{(J-1)},\ R_2^{(J-1)},\ \dots,\ R_K^{(J-1)})$,
    each holding the sample indices most similar to the column $\mathbf{c}_k^{(J-1)}$,
        $R_k^{(J-1)} = \left\{ i \mid \forall_{\ l\neq k},\ \ \|\mathbf{y}_i - \mathbf{c}_k^{(J-1)}\|_2 < \|\mathbf{y}_i - \mathbf{c}_l^{(J-1)}\|_2 \right\}$.
  • Codebook Update Stage: For each column $k$ in $\mathbf{C}^{(J-1)}$, update it by
        $\mathbf{c}_k^{(J)} = \frac{1}{|R_k|} \sum_{i\in R_k^{(J-1)}} \mathbf{y}_i$.
  • Set $J = J + 1$.
```

Fig. 1. The K-means algorithm.

sentation MSE per $\mathbf{y}_i$ is defined as $e_i^2 = \|\mathbf{y}_i - \mathbf{C}\mathbf{x}_i\|_2^2$, and the overall MSE is

$$E = \sum_{i=1}^{K} e_i^2 = \|\mathbf{Y} - \mathbf{CX}\|_F^2. \tag{15}$$

The VQ training problem is to find a codebook $\mathbf{C}$ that minimizes the error E, subject to the limited structure of $\mathbf{X}$, whose columns must be taken from the trivial basis

$$\min_{\mathbf{C},\mathbf{X}} \left\{ \|\mathbf{Y}-\mathbf{CX}\|_F^2 \right\} \quad \text{subject to} \quad \forall i,\ \mathbf{x}_i = \mathbf{e}_k \text{ for some } k. \tag{16}$$

The K-means algorithm is an iterative method used for designing the optimal codebook for VQ [39]. In each iteration there are two stages: one for sparse coding that essentially evaluates $\mathbf{X}$ and one for updating the codebook. Fig. 1 gives a more detailed description of these steps.

The sparse coding stage assumes a known codebook $\mathbf{C}^{(J-1)}$ and computes a feasible $\mathbf{X}$ that minimizes the value of (16). Similarly, the dictionary update stage fixes $\mathbf{X}$ as known and seeks an update of $\mathbf{C}$ so as to minimize (16). Clearly, at each iteration, either a reduction or no change in the MSE is ensured. Furthermore, at each such stage, the minimization step is optimal under the assumptions. As the MSE is bounded from below by zero, and the algorithm ensures a monotonic decrease of the MSE, convergence to at least a local minimum solution is guaranteed. Note that we have deliberately chosen not to discuss stopping rules for the above-described algorithm, since those vary a lot but are quite easy to handle [39].

B. K-SVD—Generalizing the K-Means

The sparse representation problem can be viewed as a generalization of the VQ objective (16), in which we allow each input signal to be represented by a linear combination of codewords, which we now call dictionary elements. Therefore the coefficients vector is now allowed more than one nonzero entry, and these can have arbitrary values. For this case, the minimization corresponding to (16) is that of searching the best possible dictionary for the sparse representation of the example set $\mathbf{Y}$

$$\min_{\mathbf{D},\mathbf{X}} \left\{ \|\mathbf{Y}-\mathbf{DX}\|_F^2 \right\} \quad \text{subject to} \quad \forall i,\ \|\mathbf{x}_i\|_0 \le T_0. \tag{17}$$

A similar objective could alternatively be met by considering

$$\min_{\mathbf{D},\mathbf{X}} \sum_i \|\mathbf{x}_i\|_0 \quad \text{subject to} \quad \|\mathbf{Y}-\mathbf{DX}\|_F^2 \le \epsilon \tag{18}$$

for a fixed value ϵ. In this paper, we mainly discuss the first problem (17), although the treatment is very similar.

In our algorithm, we minimize the expression in (17) iteratively. First, we fix $\mathbf{D}$ and aim to find the best coefficient matrix $\mathbf{X}$ that can be found. As finding the truly optimal $\mathbf{X}$ is impossible, we use an approximation pursuit method. Any such algorithm can be used for the calculation of the coefficients, as long as it can supply a solution with a fixed and predetermined number of nonzero entries T_0.

Once the sparse coding task is done, a second stage is performed to search for a better dictionary. This process updates one column at a time, fixing all columns in $\mathbf{D}$ except one, $\mathbf{d}_k$, and finding a new column $\tilde{\mathbf{d}}_k$ and new values for its coefficients that best reduce the MSE. This is markedly different from all the K-means generalizations that were described in Section III. All those methods freeze $\mathbf{X}$ while finding a better $\mathbf{D}$. Our approach is different, as we change the columns of $\mathbf{D}$ sequentially and allow changing the relevant coefficients. In a sense, this approach is a more direct generalization of the K-means algorithm because it updates each column separately, as done in K-means. One may argue that in K-means the nonzero entries in $\mathbf{X}$ are fixed during the improvement of $\mathbf{c}_k$, but as we shall see next, this is true because in the K-means (and the gain-shape VQ), the column update problems are decoupled, whereas in the more general setting, this should not be the case.

The process of updating only one column of $\mathbf{D}$ at a time is a problem having a straightforward solution based on the singular value decomposition (SVD). Furthermore, allowing a change in the coefficient values while updating the dictionary columns accelerates convergence, since the subsequent column updates will be based on more relevant coefficients. The overall effect is very much in line with the leap from gradient descent to Gauss–Seidel methods in optimization.

Here one might be tempted to suggest skipping the step of sparse coding and using only updates of columns in $\mathbf{D}$, along with their coefficients, applied in a cyclic fashion, again and again. This, however, will not work well, as the support of the representations will never be changed, and such an algorithm will necessarily fall into a local minimum trap.

C. K-SVD—Detailed Description

We shall now discuss the K-SVD in detail. Recall that our objective function is

$$\min_{\mathbf{D},\mathbf{X}} \left\{ \|\mathbf{Y}-\mathbf{DX}\|_F^2 \right\} \quad \text{subject to} \quad \forall i,\ \|\mathbf{x}_i\|_0 \le T_0. \tag{19}$$

Let us first consider the sparse coding stage, where we assume that $\mathbf{D}$ is fixed, and consider the above optimization problem as a search for sparse representations with coefficients summarized in the matrix $\mathbf{X}$. The penalty term can be rewritten as

$$\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2=\sum_{i=1}^{N}\|\mathbf{y}_i-\mathbf{D}\mathbf{x}_i\|_2^2.$$

Therefore the problem posed in (19) can be decoupled to N distinct problems of the form

$$\min_{\mathbf{x}_i}\left\{\|\mathbf{y}_i-\mathbf{D}\mathbf{x}_i\|_2^2\right\} \text{ subject to } \quad \|\mathbf{x}_i\|_0\le T_0, \qquad \text{for } i=1,2,\ldots,N. \tag{20}$$

This problem is adequately addressed by the pursuit algorithms discussed in Section II, and we have seen that if T_0 is small enough, their solution is a good approximation to the ideal one that is numerically infeasible to compute.

We now turn to the second, and slightly more involved, process of updating the dictionary together with the nonzero coefficients. Assume that both $\mathbf{X}$ and $\mathbf{D}$ are fixed and we put in question only one column in the dictionary $\mathbf{d}_k$ and the coefficients that correspond to it, the kth row in $\mathbf{X}$, denoted as $\mathbf{x}_T^k$ (this is not the vector $\mathbf{x}_k$ which is the kth column in $\mathbf{X}$). Returning to the objective function (19), the penalty term can be rewritten as

$$\begin{aligned}\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2&=\left\|\mathbf{Y}-\sum_{j=1}^{K}\mathbf{d}_j\mathbf{x}_T^j\right\|_F^2\\&=\left\|\left(\mathbf{Y}-\sum_{j\neq k}\mathbf{d}_j\mathbf{x}_T^j\right)-\mathbf{d}_k\mathbf{x}_T^k\right\|_F^2\\&=\left\|\mathbf{E}_k-\mathbf{d}_k\mathbf{x}_T^k\right\|_F^2.\end{aligned}\tag{21}$$

We have decomposed the multiplication $\mathbf{D}\mathbf{X}$ to the sum of K rank-1 matrices. Among those, $K-1$ terms are assumed fixed, and one—the kth—remains in question. The matrix $\mathbf{E}_k$ stands for the error for all the N examples when the kth atom is removed. Note the resemblance between this error and the one defined in [45].

Here, it would be tempting to suggest the use of the SVD to find alternative $\mathbf{d}_k$ and $\mathbf{x}_T^k$. The SVD finds the closest rank-1 matrix (in Frobenius norm) that approximates $\mathbf{E}_k$, and this will effectively minimize the error as defined in (21). However, such a step will be a mistake, because the new vector $\mathbf{x}_T^k$ is very likely to be filled, since in such an update of $\mathbf{d}_k$ we do not enforce the sparsity constraint.

A remedy to the above problem, however, is simple and also quite intuitive. Define ω_k as the group of indices pointing to examples $\{\mathbf{y}_i\}$ that use the atom $\mathbf{d}_k$, i.e., those where $\mathbf{x}_T^k(i)$ is nonzero. Thus

$$\omega_k=\{i|\,1\le i\le K,\ \mathbf{x}_T^k(i)\neq 0\}. \tag{22}$$

Define $\mathbf{\Omega}_k$ as a matrix of size $N\times|\omega_k|$, with ones on the $(\omega_k(i),i)$th entries and zeros elsewhere. When multiplying $\mathbf{x}_R^k=\mathbf{x}_T^k\mathbf{\Omega}_k$, this shrinks the row vector $\mathbf{x}_T^k$ by discarding of

Task: Find the best dictionary to represent the data samples $\{\mathbf{y}_i\}_{i=1}^N$ as sparse compositions, by solving

$$\min_{\mathbf{D},\mathbf{X}}\{\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2\}\quad \text{subject to}\quad \forall i,\ \|\mathbf{x}_i\|_0\le T_0.$$

Initialization : Set the dictionary matrix $\mathbf{D}^{(0)}\in\mathbb{R}^{n\times K}$ with ℓ^2 normalized columns. Set $J=1$.
Repeat until convergence (stopping rule):

- *Sparse Coding Stage*: Use any pursuit algorithm to compute the representation vectors $\mathbf{x}_i$ for each example $\mathbf{y}_i$, by approximating the solution of
$$i=1,2,\ldots,N,\quad \min_{\mathbf{x}_i}\{\|\mathbf{y}_i-\mathbf{D}\mathbf{x}_i\|_2^2\}\quad \text{subject to}\quad \|\mathbf{x}_i\|_0\le T_0.$$
- *Codebook Update Stage*: For each column $k=1,2,\ldots,K$ in $\mathbf{D}^{(J-1)}$, update it by
  - Define the group of examples that use this atom, $\omega_k=\{i|\ 1\le i\le N,\ \mathbf{x}_T^k(i)\neq 0\}$.
  - Compute the overall representation error matrix, $\mathbf{E}_k$, by
  $$\mathbf{E}_k=\mathbf{Y}-\sum_{j\neq k}\mathbf{d}_j\mathbf{x}_T^j.$$
  - Restrict $\mathbf{E}_k$ by choosing only the columns corresponding to ω_k, and obtain $\mathbf{E}_k^R$.
  - Apply SVD decomposition $\mathbf{E}_k^R=\mathbf{U}\mathbf{\Delta}\mathbf{V}^T$. Choose the updated dictionary column $\tilde{\mathbf{d}}_k$ to be the first column of $\mathbf{U}$. Update the coefficient vector $\mathbf{x}_R^k$ to be the first column of $\mathbf{V}$ multiplied by $\mathbf{\Delta}(1,1)$.
- Set $J=J+1$.

Fig. 2. The K-SVD algorithm.

the zero entries, resulting with the row vector $\mathbf{x}_R^k$ of length $|\omega_k|$. Similarly, the multiplication $\mathbf{Y}_k^R=\mathbf{Y}\mathbf{\Omega}_k$ creates a matrix of size $n\times|\omega_k|$ that includes a subset of the examples that are currently using the $\mathbf{d}_k$ atom. The same effect happens with $\mathbf{E}_k^R=\mathbf{E}_k\mathbf{\Omega}_k$, implying a selection of error columns that correspond to examples that use the atom $\mathbf{d}_k$.

With this notation, we may now return to (21) and suggest minimization with respect to both $\mathbf{d}_k$ and $\mathbf{x}_T^k$, but this time force the solution of $\tilde{\mathbf{x}}_T^k$ to have the same support as the original $\mathbf{x}_T^k$. This is equivalent to the minimization of

$$\left\|\mathbf{E}_k\mathbf{\Omega}_k-\mathbf{d}_k\mathbf{x}_T^k\mathbf{\Omega}_k\right\|_F^2=\left\|\mathbf{E}_k^R-\mathbf{d}_k\mathbf{x}_R^k\right\|_F^2 \tag{23}$$

and this time it can be done directly via SVD. Taking the restricted matrix $\mathbf{E}_k^R$, SVD decomposes it to $\mathbf{E}_k^R=\mathbf{U}\mathbf{\Delta}\mathbf{V}^T$. We define the solution for $\tilde{\mathbf{d}}_k$ as the first column of $\mathbf{U}$, and the coefficient vector $\mathbf{x}_R^k$ as the first column of $\mathbf{V}$ multiplied by $\mathbf{\Delta}(1,1)$. Note that, in this solution, we necessarily have that i) the columns of $\mathbf{D}$ remain normalized and ii) the support of all representations either stays the same or gets smaller by possible nulling of terms.

We shall call this algorithm "K-SVD" to parallel the name K-means. While K-means applies K computations of means to update the codebook, K-SVD obtains the updated dictionary by K SVD computations, each determining one column. A full description of the algorithm is given in Fig. 2.

In the K-SVD algorithm, we sweep through the columns and use always the most updated coefficients as they emerge from preceding SVD steps. Parallel versions of this algorithm can also be considered, where all updates of the previous dictionary are done based on the same $\mathbf{X}$. Experiments show that while this version also converges, it yields an inferior solution and typically requires more than four times the number of iterations.

An important question that arises is: will the K-SVD algorithm converge? Let us first assume we can perform the sparse coding stage perfectly, retrieving the best approximation to the signal $\mathbf{y}_i$ that contains no more than T_0 nonzero entries. In this case, and assuming a fixed dictionary $\mathbf{D}$, each sparse coding step decreases the total representation error $\|\mathbf{Y}-\mathbf{DX}\|_F^2$, posed in (19). Moreover, at the update step for $\mathbf{d}_k$, an additional reduction or no change in the MSE is guaranteed, while not violating the sparsity constraint. Executing a series of such steps ensures a monotonic MSE reduction, and therefore, convergence to a local minimum is guaranteed.

Unfortunately, the above claim depends on the success of pursuit algorithms to robustly approximate the solution to (20), and thus convergence is not always guaranteed. However, when T_0 is small enough relative to n, the OMP, FOCUSS, and BP approximating methods are known to perform very well.[1] In those circumstances the convergence is guaranteed. We can ensure convergence by external interference—by comparing the best solution using the already given support to the one proposed by the new run of the pursuit algorithm, and adopting the better one. This way we shall always get an improvement. Practically, we saw in all our experiments that a convergence is reached, and there was no need for such external interference.

D. *From* K*-SVD Back to* K*-Means*

What happens when the model order $T_0 = 1$? This case corresponds to the gain-shape VQ, and as such it is important, as the K-SVD becomes a method for its codebook training. When $T_0 = 1$, the coefficient matrix $\mathbf{X}$ has only one nonzero entry per column. Thus, computing the error $\mathbf{E}_k^R$ in (23) yields

$$\mathbf{E}_k^R = \mathbf{E}_k\mathbf{\Omega}_k = \left(\mathbf{Y} - \sum_{j\neq k}\mathbf{d}_j\mathbf{x}_T^j\right)\mathbf{\Omega}_k = \mathbf{Y}\mathbf{\Omega}_k = \mathbf{Y}_k^R. \quad (24)$$

This is because the restriction $\mathbf{\Omega}_k$ takes only those columns in $\mathbf{E}_k$ that use the $\mathbf{d}_k$ atom, and thus necessarily, they use no other atoms, implying that for all j, $\mathbf{x}_T^j\mathbf{\Omega}_k = 0$.

The implication of the above outcome is that the SVD in the $T_0 = 1$ case is done directly on the group of examples in ω_k. Also, the K updates of the columns of $\mathbf{D}$ become independent of each other, implying that a sequential process as before, or a parallel one, both lead to the same algorithm. We mentioned before that the K-means update of the cluster centroids could be interpreted as a sequential process, and the discussion here sheds some further light on this interpretation.

We could further constrain our representation stage and, beyond the choice $T_0 = 1$, limit the nonzero entries of $\mathbf{X}$ to be 1. This brings us back to the classical clustering problem as described earlier. In this case, we have that $\mathbf{x}_R^k$ is filled with ones, thus $\mathbf{x}_R^k = \mathbf{1}^T$. The K-SVD then needs to approximate the restricted error matrix $\mathbf{E}_k^R = \mathbf{Y}_k^R$ by a rank-1 matrix $\mathbf{d}_k \cdot \mathbf{1}^T$. The solution is the mean of the columns of $\mathbf{Y}_k^R$, exactly as K-means suggests.

[1]While OMP can be naturally used to get a fixed and predetermined number of nonzeros (T_0), both BP and FOCUSS require some slight modifications. For example, in using FOCUSS to derive exactly T_0 nonzero coefficients, the regularization parameter should be adapted while iterating.

E. K*-SVD—Implementation Details*

Just like the K-means, the K-SVD algorithm is susceptible to local minimum traps. Our experiments show that improved results can be reached if the following variations are applied.

- When using approximation methods with a fixed number of coefficients, we found that FOCUSS proves to be the best in terms of getting the best out of each iteration. However, from a run-time point of view, OMP was found to lead to far more efficient overall algorithm.
- When a dictionary element is not being used "enough" (relative to the number of dictionary elements and to the number of samples), it could be replaced with the least represented signal element, after being normalized (the representation is measured without the dictionary element that is going to be replaced). Since the number of data elements is much larger than the number of dictionary elements, and since our model assumption suggests that the dictionary atoms are of equal importance, such replacement is very effective in avoiding local minima and overfitting.
- Similar to the idea of removal of unpopular elements from the dictionary, we found that it is very effective to prune the dictionary from having too-close elements. If indeed such a pair of atoms is found (based on their absolute inner product exceeding some threshold), one of those elements should be removed and replaced with the least represented signal element.

Similarly to the K-means, we can propose a variety of techniques to further improve the K-SVD algorithm. Most appealing on this list are multiscale approaches and tree-based training where the number of columns K is allowed to increase during the algorithm. We have not yet tested these options, and leave these matters for future work.

V. Synthetic Experiments

As in previously reported works [37], [45], we first try the K-SVD algorithm on synthetic signals, to test whether this algorithm recovers the original dictionary that generated the data and to compare its results with other reported algorithms.

A. *Generation of the Data to Train On*

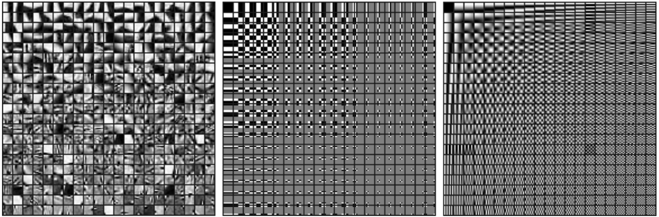
A random matrix $\mathbf{D}$ (referred to later as the *generating dictionary*) of size 20×50 was generated with i.i.d. uniformly distributed entries. Each column was normalized to a unit ℓ^2-norm. Then, 1500 data signals $\{\mathbf{y}_i\}_{i=1}^{1500}$ of dimension 20 were produced, each created by a linear combination of three different generating dictionary atoms, with uniformly distributed i.i.d. coefficients in random and independent locations. White Gaussian noise with varying signal-to-noise ratio (SNR) was added to the resulting data signals.

B. *Applying the* K*-SVD*

The dictionary was initialized with data signals. The coefficients were found using OMP with a fixed number of three coefficients. The maximum number of iterations was set to 80.

C. *Comparison to Other Reported Works*

We implemented the MOD algorithm and applied it on the same data, using OMP with a fixed number of three coefficients



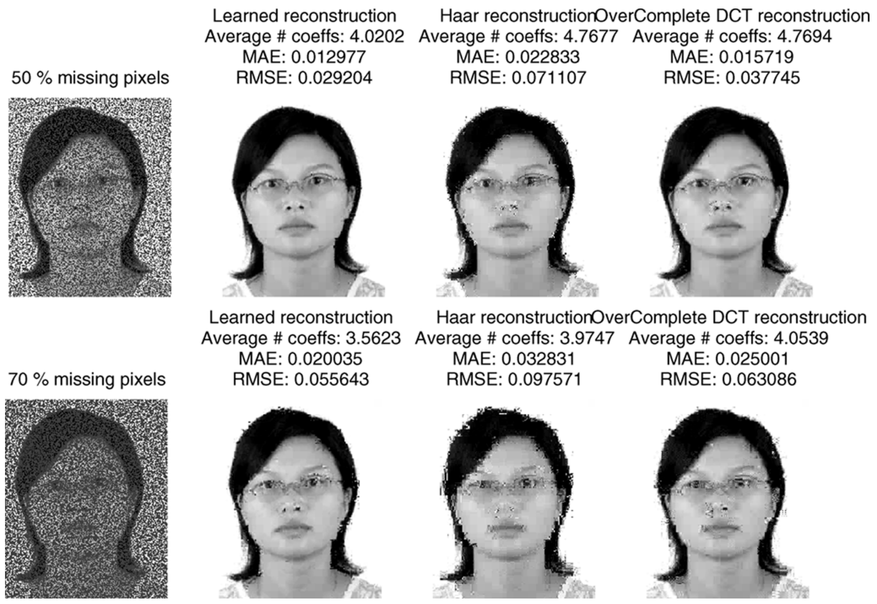


Fig. 7. The corrupted image (left) with the missing pixels marked as points and the reconstructed results by the learned dictionary, the overcomplete Haar dictionary, and the overcomplete DCT dictionary, respectively. The different rows are for 50% and 70% of missing pixels.

ficients, where the maximal number of coefficients is ten. Note that better performance can be obtained by switching to FOCUSS. We concentrated on OMP because of its simplicity and fast execution. The trained dictionary is presented in Fig. 5(a).

4) Comparison Dictionaries: The trained dictionary was compared with the overcomplete Haar dictionary, which includes separable basis functions, having steps of various sizes and in all locations (total of 441 elements). In addition, we build an overcomplete separable version of the DCT dictionary by sampling the cosine wave in different frequencies to result a total of 441 elements. The overcomplete Haar dictionary and the overcomplete DCT dictionary are presented in Fig. 5(b) and (c), respectively.

5) Applications: We used the K-SVD results, denoted here as the *learned dictionary,* for two different applications on images. All tests were performed on one face image which was not included in the training set. The first application is filling in missing pixels: we deleted random pixels in the image and filled their values using the various dictionaries' decomposition. We then tested the compression potential of the learned dictionary decomposition and derived a rate-distortion graph. We hereafter describe those experiments in more detail.

A. Filling In Missing Pixels

We chose one random full face image, which consists of 594 nonoverlapping blocks (none of which were used for training). For each block, the following procedure was conducted for r in the range $\{0.2, 0.9\}$.

1) A fraction r of the pixels in each block, in random locations, were deleted (set to zero).
2) The coefficients of the corrupted block under the learned dictionary, the overcomplete Haar dictionary, and the overcomplete DCT dictionary were found using OMP with an error bound of $\|0.02 \cdot \mathbf{1}\|_2$, where $\mathbf{1} \in R^n$ is a vector of all ones[3] (allowing an error of ± 5 gray-values in 8-bit images). All projections in the OMP algorithm included only the noncorrupted pixels, and for this purpose, the dictionary elements were normalized so that the noncorrupted indexes in each dictionary element have a unit norm. The resulting coefficient vector of the block $\mathbf{B}$ is denoted $\mathbf{x_B}$.
3) The reconstructed block $\tilde{\mathbf{B}}$ was chosen as $\tilde{\mathbf{B}} = \mathbf{D} \cdot \mathbf{x_B}$.
4) The reconstruction error was set to $\sqrt{\|\mathbf{B} - \tilde{\mathbf{B}}\|_F^2 / 64}$ (where 64 is the number of pixels in each block).

The mean reconstruction errors (for all blocks and all corruption rates) were computed and are displayed in Fig. 6. Two corrupted images and their reconstructions are shown in Fig. 7. As can be seen, higher quality recovery is obtained using the learned dictionary.

B. Compression

A compression comparison was conducted between the overcomplete learned dictionary, the overcomplete Haar dictionary, and the overcomplete DCT dictionary (as explained before), all of size 64×441. In addition, we compared to the regular (unitary) DCT dictionary (used by the JPEG algorithm). The resulting rate-distortion graph is presented in Fig. 8. In this compression test, the face image was partitioned (again) into 594 disjoint 8×8 blocks. All blocks were coded in various rates (bits-per-pixel values), and the peak SNR (PSNR) was measured. Let $\mathbf{I}$ be the original image and $\tilde{\mathbf{I}}$ be the coded image, combined by all the coded blocks. We denote e^2 as the mean squared error between $\mathbf{I}$ and $\tilde{\mathbf{I}}$, and

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{1}{e^2}\right). \tag{26}$$

In each test we set an error goal ϵ and fixed the number of bits per coefficient Q. For each such pair of parameters, all blocks were coded in order to achieve the desired error goal, and the coefficients were quantized to the desired number of bits (uniform quantization, using upper and lower bounds for each coefficient in each dictionary based on the training set coefficients). For the

[3]The input image is scaled to the dynamic range [0,1].

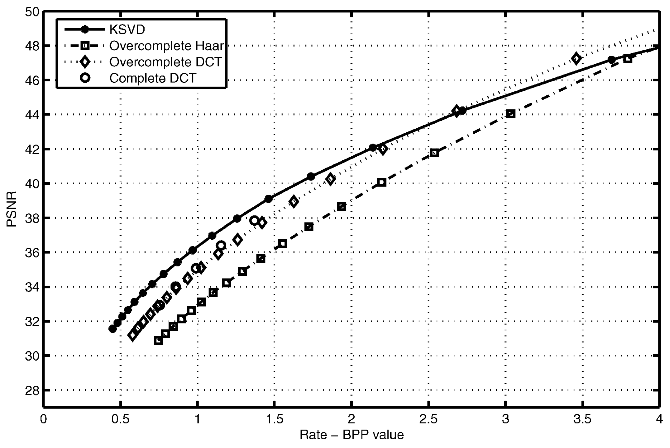


Fig. 8. Compression results: rate-distortion graphs.

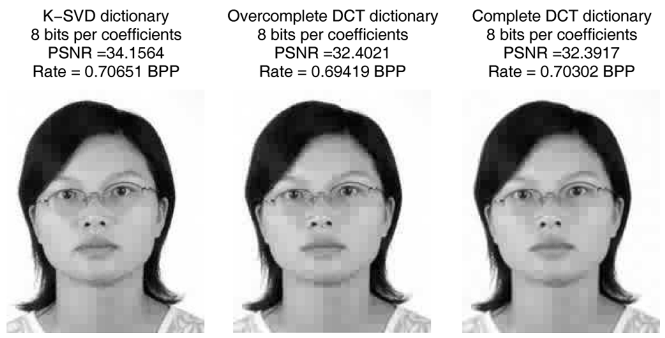


Fig. 9. Sample compression results.

overcomplete dictionaries, we used the OMP coding method. The rate value was defined as

$$R = \frac{a \cdot \sharp\text{Blocks} + \sharp\text{coefs} \cdot (b + Q)}{\sharp\text{pixels}} \tag{27}$$

where the following hold.

- a holds the required number of bits to code the number of coefficients for each block.
- b holds the required number of bits to code the index of the representing atom. Both a and b values were calculated using an entropy coder.
- $\sharp$Blocks is the number of blocks in the image (594).
- $\sharp$coefs is the total number of coefficients required to represent the whole image.
- $\sharp$pixels is the number of pixels in the image ($= 64 \cdot \sharp\text{Blocks}$).

In the unitary DCT dictionary, we picked the coefficients in a zigzag order, as done by JPEG, until the error bound ϵ was reached. Therefore, the index of each atom should not be coded, and the rate was defined by

$$R = \frac{a \cdot \sharp\text{Blocks} + \sharp\text{coefs} \cdot Q}{\sharp\text{pixels}} \tag{28}$$

with the same notation as before.

By sweeping through various values of ϵ and Q, we get per each dictionary several curves in the R-D plane. Fig. 8 presents the best obtained R-D curves for each dictionary. As can be seen, the K-SVD dictionary outperforms all other dictionaries and achieves up to 1 – 2 dB better for bit rates less than 1.5 bits per pixel (where the sparsity model holds true). Samples results are presented in Fig. 9.

VII. Conclusion

In this paper, we addressed the problem of generating and using overcomplete dictionaries. We presented an algorithm—the K-SVD—for training an overcomplete dictionary that best suits a set of given signals. This algorithm is a generalization of the K-means, designed to solve a similar but constrained problem. We have shown that the dictionary found by the K-SVD performs well for both synthetic and real images in applications such as filling in missing pixels and compression and outperforms alternatives such as the nondecimated Haar and overcomplete or unitary DCT.

We believe this kind of dictionary, which nowadays is not being commonly used, can successfully replace popular representation methods both in image enhancement and in compression. Future work is required to enable such a trend. Among the many possible research directions we mention three: i) exploration of the connection between the chosen pursuit method in the K-SVD and the method used later in the application; ii) a study of the effect of introducing weights to the atoms, allowing them to get varying degrees of popularity (in a way we have used this notion when we separated the DC in our experiments); and iii) handling the scalability problem of the K-SVD, when turning to work with larger image patches.

Acknowledgment

The authors thank Prof. K. Kreutz-Delgado and his research team who graciously agreed to share their software for comparison purposes. They also thank the reviewers for their thorough and helpful remarks.

References

[1] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek, "An overview of JPEG-2000," in *Proc. Data Compression Conf.*, 2000, pp. 523–541.
[2] D. L. Donoho and I. M. Johnstone, "Ideal denoising in an orthonormal basis chosen from a library of bases," *Comp. Rend. Acad. Sci.*, ser. A, vol. 319, pp. 1317–1322, 1994.
[3] R. Coifman and D. L. Donoho, "Translation invariant denoising," in *Wavelets and Statistics*. New York: Springer-Verlag, 1995, vol. 103, Lecture Notes in Statistics, pp. 120–150.
[4] E. P. Simoncelli, W. T. Freeman, E. H. Adelsom, and D. J. Heeger, "Shiftable multi-scale transforms," *IEEE Trans. Inf. Theory*, vol. 38, pp. 587–607, 1992.
[5] J. L. Starck, E. J. Candes, and D. L. Donoho, "The curvelet transform for image denoising," *IEEE Trans. Image Process.*, vol. 11, pp. 670–684, 2002.
[6] R. Gastaud and J. L. Starck, "Dynamic range compression : A new method based on wavelet transform," in *Astron. Data Anal. Software Systems Conf.*, Strasbourg, 2003.
[7] J. L. Starck, M. Elad, and D. L. Donoho, "Image decomposition: Separation of texture from piece-wise smooth content," in *SPIE Conf. Signal Image Process.: Wavelet Applicat. Signal Image Process. X, SPIE 48th Annu. Meeting*, San Diego, CA, Aug. 3–8, 2003.
[8] ——, "Image decomposition via the combination of sparse representations and a variational approach," *IEEE Trans. Image Process.*, Feb. 2004.
[9] M. Elad, J. L. Starck, P. Querre, and D. L. Donoho, "Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)," *J. Appl. Comput. Harmon. Anal.*, submitted for publication.

[10] M. Aharon, M. Elad, and A. M. Bruckstein, K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation Technion—Israel Inst. of Technology, 2005, Tech. Ref..

[11] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *J. Construct. Approx.*, vol. 13, pp. 57–98, 1997.

[12] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, 1993.

[13] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their application to non-linear system identification," *Int. J. Contr.*, vol. 50, no. 5, pp. 1873–96, 1989.

[14] G. Davis, S. Mallat, and Z. Zhang, "Adaptive time-frequency decompositions," *Opt. Eng.*, vol. 33, no. 7, pp. 2183–91, 1994.

[15] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Conf. Rec. 27th Asilomar Conf. Signals, Syst. Comput.*, 1993, vol. 1.

[16] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inf. Theory*, vol. 50, pp. 2231–2242, Oct. 2004.

[17] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, 2001.

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

[19] B. D. Rao and K. Kreutz-Delgado, "An affine scaling methodology for best basis selection," *IEEE Trans. Signal Process.*, vol. 47, pp. 187–200, 1999.

[20] ——, "Deriving algorithms for computing sparse solutions to linear inverse problems," in *Conf. Rec. 31st Asilomar Conf. Signals, Syst. Comput.*, 1998, vol. 1, pp. 955–959.

[21] B. D. Rao, K. Engan, S. F. Cotter, J. Palmer, and K. Kreutz-Delgado, "Subset selection in noise based on diversity measure minimization," *IEEE Trans. Signal Process.*, vol. 51, pp. 760–770, 2003.

[22] M. S. Lewicki and B. A. Olshausen, "A probabilistic framework for the adaptation and comparison of image codes," *J. Opt. Soc. Amer. A: Opt., Image Sci. Vision*, vol. 16, no. 7, pp. 1587–1601, 1999.

[23] B. A. Olshausen and D. J. Field, "Natural image statistics and efficient coding," *Network Comp. Neural Syst.*, vol. 7, no. 2, pp. 333–339, 1996.

[24] B. A. Olshausen and B. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?," *Vision Res.*, vol. 37, pp. 3311–3325, 1997.

[25] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Comp.*, vol. 12, pp. 337–365, 2000.

[26] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Trans. Inf. Theory*, vol. 47, pp. 2845–62, 1999.

[27] M. Elad and A. M. Bruckstein, "A generalized uncertainty principle and sparse representation in pairs of bases," *IEEE Trans. Inf. Theory*, vol. 48, pp. 2558–2567, Sep. 2002.

[28] D. L. Donoho and M. Elad, "Optimally sparse representation in general (non-orthogonal) dictionaries via l_1 minimization," *Proc. Nat. Acad. Sci.*, vol. 100, no. 5, pp. 2197–2202, 2003.

[29] R. Gribonval and M. Nielsen, "Sparse decompositions in unions of bases," *IEEE Trans. Inf. Theory*, vol. 49, pp. 3320–3325, 2003.

[30] J. J. Fuchs, "On sparse representations in arbitrary redundant bases," *IEEE Trans. Inf. Theory*, vol. 50, pp. 1341–1344, 2004.

[31] D. L. Donoho, M. Elad, and V. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," submitted for publication.

[32] J. A. Tropp, "Just relax: Convex programming methods for subset selection and sparse approximation," UT-Austin, 2004, ICES Rep. 04-04.

[33] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal ℓ^1-norm solution is also the sparsest solution," Dept. of Statistics, Stanford Univ., 2004, Tech. Rep. 2004-10.

[34] ——, "For most large underdetermined systems of linear equations, the minimal ℓ^1-norm near-solution approximates the sparsest near-solution," Dept. of Statistics, Stanford Univ., 2004, Tech. Rep. 2004-11.

[35] E. Candes and J. Romberg, "Quantitative robust uncertainty principles and optimally sparse decompositions," to be published.

[36] K. Engan, S. O. Aase, and J. H. Husøy, "Multi-frame compression: Theory and design," *EURASIP Signal Process.*, vol. 80, no. 10, pp. 2121–2140, 2000.

[37] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural Comp.*, vol. 15, no. 2, pp. 349–396, 2003.

[38] J. A. Tropp, "Topics in sparse approximation," Ph.D. dissertation, Univ. of Texas at Austin, Austin, 2004.

[39] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer Academic, 1991.

[40] K. Engan, S. O. Aase, and J. H. Hakon-Husoy, "Method of optimal directions for frame design," in *IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1999, vol. 5, pp. 2443–2446.

[41] K. Engan, B. D. Rao, and K. Kreutz-Delgado, "Frame design using focuss with method of optimal directions (mod)," in *Norwegian Signal Process. Symp.*, 1999, vol. 65-69.

[42] J. F. Murray and K. Kreutz-Delgado, "An improved focuss-based learning algorithm for solving sparse linear inverse problems," in *IEEE Int. Conf. Signals, Syst. Comput.*, 2001, vol. 4119-53.

[43] A. J. Bell and T. J. Sejnowski, "An information maximisation approach to blind separation and blind deconvolution," *Neural Comp.*, vol. 7, no. 6, pp. 1129–1159, 1996.

[44] K. Kreutz-Delgado and B. D. Rao, "FOCUSS-based dictionary learning algorithms," in *Wavelet Applications in Signal and Image Process. VIII*, 2000, vol. 4119-53.

[45] S. Lesage, R. Gribonval, F. Bimbot, and L. Benaroya, "Learning unions of orthonormal bases with thresholded singular value decomposition," in *IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2005.

[46] S. Sardy, A. G. Bruce, and P. Tseng, "Block coordinate relaxation methods for nonparametric signal denoising with wavelet dictionaries," *J. Comp. Graph. Statist.*, vol. 9, pp. 361–379, 2000.

[47] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.*, ser. B, vol. 39, no. 1, pp. 1–38, 1977.

Michal Aharon received the B.Sc. and M.Sc. degrees from the Department of Computer Science, The Technion—Israel Institute of Technology, Haifa, in 2001 and 2004, respectively, where she is now pursuing the Ph.D. degree.

During her studies, she was worked at Intel and IBM. From 1997 to 1999, she served in Israeli Military Intelligence.

Ms. Aharon received the Guttwirth and Neeman fellowships.

Michael Elad received the B.Sc., M.Sc., and D.Sc. degrees from the Department of Electrical Engineering, The Technion—Israel Institute of Technology, Haifa, in 1986, 1988, and 1997, respectively.

From 1988 to 1993, he served in the Israeli Air Force. From 1997 to 2000, he was with Hewlett-Packard Laboratories as an RD Engineer. From 2000 to 2001, he headed the research division at Jigami Corporation, Israel. During 2001 to 2003, he was a Research Associate with the Computer Science Department, Stanford University (SCCM program). Since September 2003, he has been with the Department of Computer Science, the Technion, as an Assistant Professor. He works in the field of signal and image processing, specializing in particular on inverse problems, sparse representations, and overcomplete transforms.

Prof. Elad received the Technion's Best Lecturer Award four times (1999, 2000, 2004, and 2005). He also received the Guttwirth and Wolf fellowships.

Alfred Bruckstein received the B.Sc. degree (with honors) and the M.Sc. degree in electrical engineering from The Technion—Israel Institute of Technology, Haifa, in 1977 and 1980, respectively, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1984.

Since 1985, he has been a Faculty Member of the Computer Science Department, The Technion, where he currently is a full Professor, holding the Ollendorff Chair. During the summers from 1986 to 1995 and from 1998 to 2000, he was a Visiting Scientist at Bell Laboratories. He is a member of the editorial boards of *Pattern Recognition, Imaging Systems and Technology, Circuits Systems,* and *Signal Processing*. He was a member of program committees of 20 conferences. His research interests are in image and signal processing, computer vision, computer graphics, pattern recognition, robotics—especially ant robotics—applied geometry, estimation theory and inverse scattering, and neuronal encoding process modeling.

Prof. Bruckstein is a member of SIAM, AMS, and MAA. He received the Rothschild Fellowship for Ph.D. Studies at Stanford, the Taub Award, a Theeman grant for a scientific tour of Australian universities, and the Hershel Rich Technion Innovation Award twice.