Global Journal of Computer Science and Technology: E
Network, Web & Security
Volume 14 Issue 8 Version 1.0 Year 2014
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Fast Dictionary Learning for Sparse Representations of Speech Signals

By Maria G. Jafari & Mark D. Plumbley
Queen Mary University, United Kingdom

Abstract- For dictionary-based decompositions of certain types, it has been observed that there might be a link between sparsity in the dictionary and sparsity in the decomposition. Sparsity in the dictionary has also been associated with the derivation of fast and efficient dictionary learning algorithms. Therefore, in this paper we present a greedy adaptive dictionary learning algorithm that sets out to find sparse atoms for speech signals. The algorithm learns the dictionary atoms on data frames taken from a speech signal. It iteratively extracts the data frame with minimum sparsity index, and adds this to the dictionary matrix. The contribution of this atom to the data frames is then removed, and the process is repeated. The algorithm is found to yield a sparse signal decomposition, supporting the hypothesis of a link between sparsity in the decomposition and dictionary. The algorithm is applied to the problem of speech representation and speech denoising, and its performance is compared to other existing methods.

Keywords: *adaptive dictionary, dictionary learning, sparse decomposition, sparse dictionary, speech analysis, speech denoising.*

GJCST-E Classification : *I.2.6*

Strictly as per the compliance and regulations of:

Fast Dictionary Learning for Sparse Representations of Speech Signals

Maria G. Jafari [α] & Mark D. Plumbley [σ]

Abstract- For dictionary-based decompositions of certain types, it has been observed that there might be a link between sparsity in the dictionary and sparsity in the decomposition. Sparsity in the dictionary has also been associated with the derivation of fast and efficient dictionary learning algorithms. Therefore, in this paper we present a greedy adaptive dictionary learning algorithm that sets out to find sparse atoms for speech signals. The algorithm learns the dictionary atoms on data frames taken from a speech signal. It iteratively extracts the data frame with minimum sparsity index, and adds this to the dictionary matrix. The contribution of this atom to the data frames is then removed, and the process is repeated. The algorithm is found to yield a sparse signal decomposition, supporting the hypothesis of a link between sparsity in the decomposition and dictionary. The algorithm is applied to the problem of speech representation and speech denoising, and its performance is compared to other existing methods. The method is shown to find dictionary atoms that are sparser than their time-domain waveform, and also to result in a sparser speech representation. In the presence of noise, the algorithm is found to have similar performance to the well established principal component analysis.

Index-terms: adaptive dictionary, dictionary learning, sparse decomposition, sparse dictionary, speech analysis, speech denoising.

I. Introduction

SPARSE signal representations allow the salient information within a signal to be conveyed with only a few elementary components, called atoms. For this reason, they have acquired great popularity over the years, and they have been successfully applied to a variety of problems, including the study of the human sensory system [1]–[3], blind source separation [4]–[6], and signal denoising [7]. Successful application of a sparse decomposition depends on the dictionary used, and whether it matches the signal features [8].

Two main methods have emerged to determine a dictionary within a sparse decomposition: dictionary selection and dictionary learning. Dictionary selection entails choosing a pre-existing dictionary, such as the Fourier basis, wavelet basis or modified discrete cosine basis, or constructing a redundant or overcomplete dictionary by forming a union of bases (for example the Fourier and wavelet bases) so that different properties of the signal can be represented [9]. Dictionary learning, on the other hand, aims at deducing the dictionary from the training data, so that the atoms directly capture the specific features of the signal or set of signals [7]. Dictionary learning methods are often based on an alternating optimization strategy, in which the dictionary is fixed, and a sparse signal decomposition is found; then the dictionary elements are learned, while the signal representation is fixed.

Early dictionary learning methods by Olshausen and Field [2] and Lewicki and Sejnowski [10] were based on a probabilistic model of the observed data. Lewicki and Sejnowski [10] clarify the relation between sparse coding methods and independent component analysis (ICA), while the connection between dictionary learning in sparse coding, and the vector quantization problem was pointed out by Kreutz–Delgado et al. [11]. The authors also proposed finding sparse representations using variants of the focal underdetermined system solver (FOCUSS) [12], and then updating the dictionary based on these representations. Aharon, Elad, and Bruckstein [13] proposed the K-SVD algorithm. It involves a sparse coding stage, based on a pursuit method, followed by an update step, where the dictionary matrix is updated one column at the time, while allowing the expansion coefficients to change [13]. More recently, dictionary learning methods for exact sparse representation based on ℓ_1 minimization [8], [14], and online learning algorithms [15], have been proposed.

Generally, the methods described above are computationally expensive algorithms that look for a sparse decomposition, for a variety of signal processing applications. In this paper, we are interested in targeting speech signals, and deriving a dictionary learning algorithm that is computationally fast. The algorithm should be able to learn a dictionary from a short speech signal, so that it can potentially be used in real-time processing applications.

a) Motivation

The aim of this work is to find a dictionary learning method that is fast and efficient. Rubinstein et al. have shown that this can be achieved by means of "double sparsity" [16]. Double sparsity refers to seeking a sparse decomposition and a dictionary $\mathbf{D} = \mathbf{AB}$ such that the atoms in $\mathbf{A}$ are sparse over the fixed dictionary $\mathbf{B}$, such as Wavelets or the discrete cosine transform (DCT). Also, in previous results in [17], it was

Author α σ : Department of Electronic Engineering, Queen Mary University of London, London E1 4NS, U.K.
e-mails: maria.jafari@eecs.qmul.ac.uk, mark.plumbley@eecs.qmul.ac.uk.

found that dictionary atoms learned from speech signals with a sparse coding method based on ICA (SC-ICA) [18], are localized in time and frequency. This appears to suggest that for certain types of signals (e.g., speech and music) there might be a link between sparsity in decomposition and sparsity in dictionary.

This is further supported by the success of transforms such as the Wavelet transform whose basis functions are localized, and are well-suited to the analysis of natural signals (audio, images, biomedical signals), often yielding a sparse representation.

Thus, in this paper we propose to learn sparse atoms as in [16], but rather than learning atoms that are sparse over a fixed base dictionary, we directly learn sparse atoms from a speech signal. In order to build a fast transform, the proposed algorithm seeks to learn an orthogonal dictionary from a set of local frames that are obtained by segmenting the speech signal. Over several iterations, the algorithm "grabs" the sparsest data frame, and uses a Gram–Schmidt-like step to orthogonalize the signal away from this frame.

The advantage of this approach is its computational speed and simplicity, and because of the connection that we have observed between sparsity in the dictionary and in the representation, we expect that the signal representation that is obtained with the learned dictionary will be also sparse.

b) Contributions

In this paper, we consider the formulation of our algorithm from the point of view of minimizing the sparsity index on atoms.We seek the sparsity of the dictionary atoms alone rather than of the decomposition, and to the authors' knowledge this perspective has not been considered elsewhere.[1] Further, we propose a stopping rule that automatically selects only a subset of the atoms. This has the potential of making the algorithm even faster, and to aid in denoising applications by using a subset of the atoms within the signal reconstruction.

c) Organization of the Paper

The structure of the paper is as follows: the problem that we seek to address is outlined in Section II, and our sparse adaptive dictionary algorithm is introduced in Section III, along with the stopping rule. Experimental results are presented in Section IV, including the investigation of the sparsity of the atoms and speech representation, and speech denoising. Conclusions are drawn in Section VII.

II. Problem Statement

Given a one-dimensional speech signal $x(t)$, we divide this into overlapping frames $\mathbf{x}_k$, each of L length samples, with an overlap of M samples. Hence, the kth frame $\mathbf{x}_k$ is given by

$$\mathbf{x}_k = [x((k-1)(L-M)+1), \ldots, x(kL-(k-1)M)]^T \quad (1)$$

where $k \in \{1, \ldots, K\}$. Then we construct a new matrix $\mathbf{X} \in \mathbb{R}^{L \times K}$ whose k th column corresponds to the signal block $\mathbf{x}_k$, and whose (l,k) th element is given by

$$[\mathbf{X}]_{l,k} = x(l + (k-1)(L-M)) \quad (2)$$

where $l \in \{1, \ldots, L\}$, and $K > L$.

The task is to learn a dictionary $\mathbf{D}$ consisting of L atoms $\boldsymbol{\psi}^l$, that is $\mathbf{D} = \{\boldsymbol{\psi}^l\}_{l=1}^L$, providing a sparse representation for the signal blocks $\mathbf{x}_k$. We seek a dictionary and a decomposition of $\mathbf{x}_k$, such that [19]

$$\mathbf{x}_k = \sum_{l=1}^{L} \alpha_k^l \boldsymbol{\psi}^l \quad (3)$$

where α_k^l are the expansion coefficients, and

$$||\boldsymbol{\alpha}_k||_0 \ll L. \quad (4)$$

The ℓ_0-norm $||\boldsymbol{\alpha}_k||_0$ counts the number of non–zero entries in the vector $\boldsymbol{\alpha}_k$, and therefore the expression in (4) defines the decomposition as "sparse," if $||\boldsymbol{\alpha}_k||_0$ is small. In the remainder of this paper, we use the definition of sparsity given later in (5).

The dictionary is learned from the newly constructed matrix $\mathbf{X}$. In the case of our algorithm, we begin with a matrix containing K columns, and we extract the first L columns according to the criterion discussed in the next section.

III. Greedy Adaptive Dictionary Algorithm (GAD)

To find a set of sparse dictionary *atoms* we consider the sparsity index ξ [20] for each column $\mathbf{x}_k$, of $\mathbf{X}$, defined as

$$\xi = \frac{||\mathbf{x}||_1}{||\mathbf{x}||_2} \quad (5)$$

where $||\cdot||_1$ and $||\cdot||_2$ denote the ℓ_1-and ℓ_2 -norm, respectively. The sparsity index measures the sparsity of a signal, and is such that the smaller ξ, the sparser the vector $\mathbf{x}$. Our aim is to sequentially extract new atoms from $\mathbf{X}$ to populate the dictionary matrix $\mathbf{D}$, and we do this by finding, at each iteration, the column of $\mathbf{X}$ with minimum sparsity index

$$\min_k \xi_k. \quad (6)$$

Practical implementation of the algorithm begins with the definition of a residual matrix $\mathbf{R}^l = [\mathbf{r}_1^l, \ldots, \mathbf{r}_K^l]$, where $\mathbf{r}_k^l \in \mathbb{R}^K$ is a residual column

[1] The approach proposed in\[16]looks for as parse dictionary over a base dictionary, as well as as parse decomposition, and there for eisquite different to them ethod proposed here.

vector corresponding to the th column of $\mathbf{R}^l$. The residual matrix changes at each iteration l, and is initialized to $\mathbf{X}$. The dictionary is then built by selecting the residual vector $\mathbf{r}_k^l$ that has lowest sparsity index, as indicated in Algorithm 1.

Algorithm 1 Greedy adaptive dictionary (GAD) algorithm

1. Initialize: $l = 0$, $\mathbf{D}^0 = [\,]$ {empty matrix}, $\mathbf{R}^0 = \mathbf{X}$, $\mathcal{I} = \emptyset$
2. **repeat**
3. Find residual column of $\mathbf{R}^l$ with lowest ℓ_1 - to ℓ_2 -norm ratio:
$$k^l = \arg\min_{k \notin \mathcal{I}^l} \{\|\mathbf{r}_k^l\|_1 / \|\mathbf{r}_k^l\|_2\}$$
4. Set the th atom equal to normalized $\hat{\mathbf{r}}_{k^l}^l$:
$$\boldsymbol{\psi}^l = \hat{\mathbf{r}}_{k^l}^l / \|\hat{\mathbf{r}}_{k^l}^l\|_2$$
5. Add to the dictionary:
$$\mathbf{D}^l = [\mathbf{D}^{l-1} \,|\, \boldsymbol{\psi}^l], \quad \mathcal{I}^l = \mathcal{I}^{l-1} \cup \{k^l\}$$
6. Compute the new residual $\mathbf{r}_k^{l+1} = \mathbf{r}_k^l - \boldsymbol{\psi}^l \langle \boldsymbol{\psi}^l, \mathbf{r}_k^l \rangle$ for all columns k
7. **until** "termination" (see Section III-A)

We call our method the greedy adaptive dictionary (GAD) algorithm [21].

Aside from the advantage of producing atoms that are directly relevant to the data, the GAD algorithm results in an orthogonal transform. To see this, consider rewriting the update equation in step 6 in Algorithm 1 as the projection of the current residual $\mathbf{r}_k^l$ onto the atom space, in the style of Matching Pursuit [22], [23]:

$$\mathbf{r}_k^{l+1} = \mathbf{P}_{\boldsymbol{\psi}^l} \mathbf{r}_k^l = \left(\mathbf{I} - \frac{\boldsymbol{\psi}^l \boldsymbol{\psi}^l}{\boldsymbol{\psi}^{l^T} \boldsymbol{\psi}^l} \right) \mathbf{r}_k^l = \mathbf{r}_k^l - \frac{\boldsymbol{\psi}^l \langle \boldsymbol{\psi}^l, \mathbf{r}_k^l \rangle}{\boldsymbol{\psi}^{l^T} \boldsymbol{\psi}^l}. \quad (7)$$

It follows from step 4 in Algorithm 1, that the denominator in the right-hand-side of (7) is equal to 1, and therefore the equation corresponds to the residual update in step 6. Orthogonal dictionaries have the advantage being easily invertible, since if the matrix $\mathbf{B}$ is orthogonal, then $\mathbf{B}\mathbf{B}^T = \mathbf{I}$, and evaluation of the inverse simply requires the use of the matrix transpose.

a) Termination Rules

We consider two possible termination rules:

1. The number of atoms l to be extracted is pre-determined, so that up to L atoms are learned. Then, the termination rule is:
   - Repeat from step 2, until $l = N$, where $N \leq L$.
2. The reconstruction error at the current iteration ϵ^l is defined, and the rule is:
   - Repeat from step 2 until
$$\epsilon^l = \|\hat{x}^l(t) - x(t)\|_2 \leq \sigma \quad (8)$$

Where $\hat{x}^l(t)$ is the approximation of the speech signal $x(t)$, obtained at the l th iteration from $\hat{\mathbf{X}}^l = \mathbf{D}^l (\mathbf{D}^l)^T \mathbf{X}$ by reversing the framing process; $\mathbf{D}^l$ is the dictionary learned so far, as defined in step 5 of Algorithm 1.

IV. Experiments

We compared theGADmethod to PCA [24] and K-SVD [13]. K-SVD was chosen because it learns data-determined dictionaries, and looks for a sparse representation. PCA was chosen because it is a well-established technique, commonly used in speech coding and therefore it sets the benchmark for the speech denoising application.

We used the three algorithms to learn 512 dictionary atoms from a segment of speech lasting 1.25 s. A short data segment was used because this way the algorithm can be used within real-time speech processing applications. The data was taken from the female speech signal "supernova.wav" by "Corsica S," downloaded from The Freesound Project database [25], and downsampled to 16 kHz. We also used the male speech signal "Henry5.mp3" by "acclivity," downloaded from the same database, and downsampled to 16 kHz.

The K-SVD Matlab Toolbox [26] was used to implement the K-SVD algorithm. K-SVD requires the selection of several parameters. We set the number of iterations to 50, as recommended in [13], and the number of nonzero entries in the coefficient update stage to 10, which we found empirically to give

Table1: Comparing The Computational Complexity For The Pca, K-Svd, And Gad Algorithms. The Table Shows The Average Computational Time For Each Algorithm, Obtained Over 100 Trials

Method	Average Computation Time (sec)
PCA	7
K-SVD (Matlab only)	6710
K-SVD (v2)	163
GAD	167

a more accurate, although not as sparse, signal representation than $T_0 = 3$, as used in [13]. The dictionary size was set to 512 and the memory usage to "high."

a) Computational Complexity

In Table I, we report the computational times of the algorithms, when learning a dictionary from speech segment of 1.25s, and averaged over 100 trials. Two versions of the K-SVD were also compared: the original version which is fully based on Matlab M-code, and the second version, which combines M-code with optimized MEX functions written in C. The experiments were conducted on a Quad-Core Intel Xeon Mac at 2.66 GHz, using Matlab Version 7.6.0.324 (R2008a) and under the Mac OS X Version 10.5.8 operating system.

GAD and K-SVD (v2) only need about 2 minutes, and PCA needs as little as 7sec. However, note how the K-SVD version based exclusively on M-code requires around 1 hour and 45 minutes to learn the dictionary. Therefore, we expect that optimizing the code for GAD will lead to even faster computational complexity.

b) Learned Atoms

We begin by visually inspecting some examples of the atoms learned with the three algorithms, and then considering the sparsity of the atoms and signal representation.

Fig. 1(a) shows examples of the overlapping data blocks found in the columns of the matrix $\mathbf{X}$, from which each dictionary is learned, while the remaining plots in the figure show examples of the atoms learned with PCA, K-SVD and GAD. The sparsity index relating to each atom is also given.

The atoms extracted with PCA [Fig. 1(b)] are not localized. Comparing them with Fig. 1(a), they do not appear to be capturing any particular features of the speech signal.

The K-SVD atoms [Fig. 1(c)] exhibit some structure that generally seems to correspond to that of the original data blocks. The atoms obtained with the GAD algorithm are illustrated in Fig. 1(d). Those atoms extracted earlier, shown on the first two lines, are quite similar to the original data, and are also the sparsest atoms, as indicated by the low sparsity index. Atoms extracted later, shown on the last two lines in the figure, capture mostly "noise"-like characteristics, or less meaningful features of the signal.

c) Sparsity of Atoms and Representation

We have seen in Fig. 1 how the GAD algorithm yields atoms that are initially quite sparse and then become more "noise"-

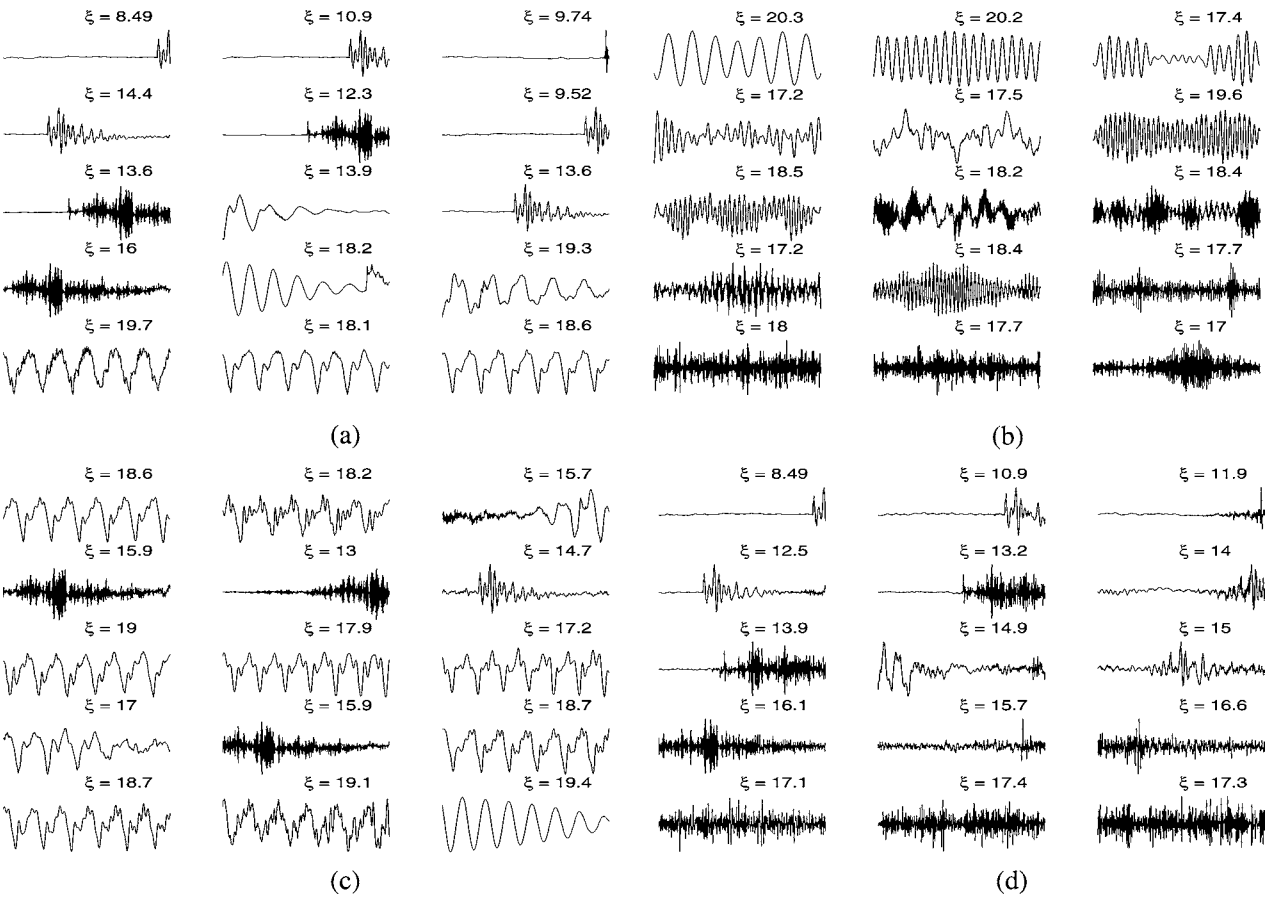

Figure 1: Examples of the frames of the original speech signals, and of the atoms learned with the PCA, K-SVD, and GAD algorithms

like. To investigate this further, 100 segments were taken from the original speech data, each lasting 1.25 s. PCA, K-SVD and GAD were used to learn dictionaries from each segment. The sparsity index ξ_k for each atom was then evaluated, and the average across the 100 trials was taken.

Fig. 2 shows the atom sparsity index for the framed speech data in the columns of $\mathbf{X}$, and for the atoms learned with PCA, K-SVD and GAD. Recall that a sparse atom is characterized by a low sparsity index. The plot shows that the atoms learned by GAD in the beginning are the sparsest, and after around 200 atoms have been extracted, the sparsity index is close to its maximum value. The behavior observed here is in agreement with what was observed in Fig. 1. It also shows that the atoms obtained with the other algorithms are not as sparse as those extracted by GAD. The original data blocks that are considered in the figure correspond to the columns in $\mathbf{X}$ that are extracted by GAD, and therefore they are the sparsest within the speech segment.

The results are shown in the first column of Table II, and they validate our expectations: GAD yields atoms that are sparser than the original signal blocks, and than all the algorithms. However, when we used the termination rule in (8) (shown in Table II as GAD-TR), with $\sigma = 5 \times 10^{-3}$, the average sparsity index for the GAD atoms decreased from 16.2 to 12.6. On average, GAD-TR was found to learn less than 110 atoms, which from Fig. 2 can be seen to correspond to those atoms that are sparsest. The algorithms perform in a similar way on the male speech.

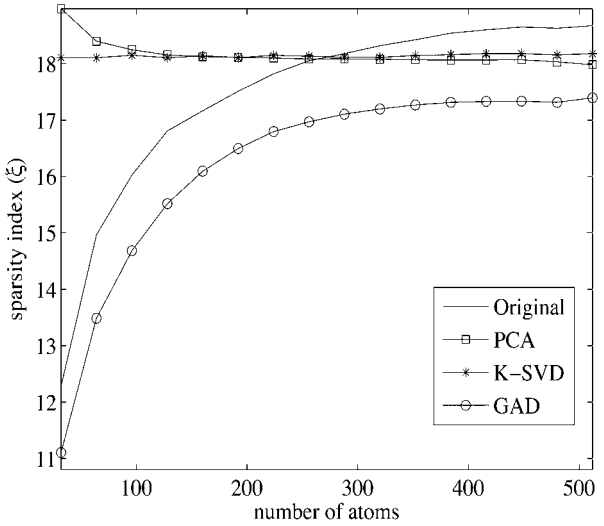

Figure 2 : Sparsity index for atoms GAD, PCA, and K-SVD algorithms, learned from a female speech signal, and averaged over 100 trials

Next, we seek to determine how sparse is the representation obtained with the GAD method. We do this by considering the transform coefficients obtained with all methods, for each block, and across the 100 speech segments taken from the speech signal, each lasting 1.25s. The sparsity index of the transform coefficients is found each time. We then average across the 100 segments and across all blocks to obtain a single

Table 2 : Mean Value And Standard Deviation (Std) For The Sparsity Index Of The Atoms And The Signal Representation Obtained With The Pca, K-Svd, And Gad Algorithms Compared To That Of The Original Signal Blocks. The Values For The Original Data Blocks And For Pca, K-Svd, And Gad Were Averaged Across 100 Trials, And 512 Atoms

Method	Mean and Standard Deviation of Sparsity Index							
	Atom Sparsity Index				Representation Sparsity Index			
	Female		Male		Female		Male	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Original	17.5	0.44	17.1	0.48	17.5	0.44	17.1	0.48
PCA	18.2	0.18	18.1	0.23	16.0	0.15	15.5	0.19
K-SVD	18.1	0.04	17.9	0.09	2.1	0.20	2.6	0.18
GAD-FULL	16.2	0.29	16.8	0.29	10.5	0.62	12.3	0.51
GAD-TR	12.6	0.64	14.9	0.70	6.0	0.43	6.9	0.48

figure for each method, and for both the female and male speech signals, as shown in the second column of Table II. This also includes values for the sparsity index for the original signal blocks in $\mathbf{X}$. The lowest representation sparsity index value is obtained with K-SVD, thanks to the strong sparsity constraint imposed by the algorithm on the signal decomposition. This entails limiting the number of nonzero elements in the signal representation to a small number (we use $T_0 = 10$). The signal transformed with the GAD algorithm is sparser than in the time domain, and than the coefficients obtained with PCA when all atoms are used in the signal reconstruction, for both signals. Moreover, the representation becomes even sparser when GAD is used with the termination rule.

Thus, as well as a dictionary whose atoms are sparse GAD leads to a sparse decomposition. This confirms the concepts discussed in Section I-A.

d) Representation Accuracy

The accuracy of the signal approximation given by each algorithm can be assessed with the reconstruction error , as defined in (8), after the dictionary has been learned

$$\epsilon = \|\hat{x}(t) - x(t)\|_2 \qquad (9)$$

Where $x(t)$ is the signal approximation obtained from $\hat{\mathbf{X}} = \mathbf{D}^l(\mathbf{D}^l)^\dagger\mathbf{X}$, and $(\mathbf{D}^l)^\dagger$ is the right pseudo-inverse of $\mathbf{D}^l$. This is plotted in Fig. 3 for each algorithm, as the number of atoms omitted in the signal reconstruction goes from 0 to 462 (or, the total number of atoms used goes from 512 down to 50). K-SVD has a nonzero reconstruction error even when all atoms are included in the signal approximation, because the transform is not complete, and therefore it does not result in an exact reconstruction.

In general, the results show that all algorithms perform quite well when few atoms are omitted in the reconstruction. As more and more atoms are omitted,

the reconstruction error increases. PCA performs best, because the transform arranges the signal components so that most energy is concentrated in a small number of components, corresponding to those extracted earlier. The GAD transform also gives good signal approximations as more atoms are excluded from the reconstruction, although its performance seems to worsen as the number of omitted atoms becomes more than 300 (or less than 200 atoms

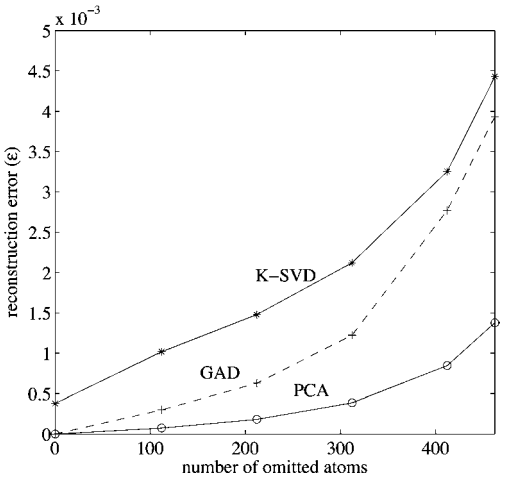

Figure 3: Reconstruction error for the GAD, PCA, and K-SVD algorithms, averaged over 100 trials

are used in the reconstruction). This corresponds to the number, identified in Fig. 2, below which the GAD atoms are sparsest, and above which the sparsity index reaches its maximum value. K-SVD yields signal approximations that suffer most from the reduction in the number of atoms.

The dictionary constructed by GAD separates the coherent components from the incoherent components. The latter can be discarded from the representation to reduce incoherent background noise. This suggests that the GAD algorithm might be suitable for denoising applications. Hence, we will consider this problem in the following section.

V. Application to Speech Denoising

The term denoising refers to the removal of noise from a signal. Sparse transforms have been found to be among the most successful methods for denoising [27], and dictionary learning methods have been used for this application [13].

Table III shows the tolerance of the PCA, K-SVD, and GAD algorithms to a noise level changing from 10 dB to 10 dB, as the number of atoms in the reconstruction is reduced from 512 to 50. This is evaluated with the improvement in signal-to-noise ratio (ISNR):

$$\mathrm{ISNR} = 10\log\frac{E\{(x(t)-x_n(t))^2\}}{E\{x(t)\quad\hat{x}(t))^2\}} \qquad (10)$$

Where $x(t)$ is the original signal, $x_n(t)$ is the observed distorted (noisy) signal, and $\hat{x}(t)$ is the source approximated by the transform. As the signal approximation becomes closer to the original source, ISNR increases.

When all atoms are used in the reconstruction, the complete transforms PCA and GAD, yield an ISNR of 0 dB, while K-SVD gives a nonzero ISNR, since the approximation is not exact. Generally, K-SVD has been shown to perform well for tasks such as image denoising [7], and the results in Table III show that this is also true for speech: the algorithm yields the highest ISNR values across all experiments. For the remaining algorithms, when the noise is low (10 dB), reducing the number of atoms in the reconstruction leads to distortion in the signal approximation. As the level of noise increases, the high ISNR

Table 3 : ISNR For The Gad, Pca, And K-Svd Algorithms. All Isnr Values Are Expressed In Decibels (Db)

Noise Level	Method	Number of Atoms					
		512	400	300	200	100	50
10 dB	PCA	0.00	0.52	1.32	2.61	4.74	5.69
	K-SVD	5.10	6.01	6.83	7.45	7.00	5.85
	GAD	0.00	1.40	2.97	4.71	5.10	2.53
0 dB	PCA	0.00	0.50	1.30	2.69	5.42	8.33
	K-SVD	4.89	5.98	7.10	8.52	10.17	10.97
	GAD	0.00	1.50	3.29	5.65	7.20	7.27
-10 dB	PCA	0.00	0.49	1.28	2.65	5.34	8.28
	K-SVD	4.70	5.75	6.96	8.53	10.64	12.07
	GAD	0.00	1.47	3.27	5.80	8.86	10.21

values for PCA and GAD indicate that there are benefits in reducing the number of atoms used in the signal approximation. It is well-known that PCA can reduce the level of noise present, because it decomposes the space into signal and noise subspaces [28], and the results in Table III show that the performance of GAD is similar.

It should be emphasized that the advantage of using the GAD algorithm over PCA is that methods based on sparse representations do not enforce decorrelation on the data. This results in greater flexibility in adapting the representation to the data, and uncovering previously unobserved structure in the data. Moreover, sparse representations allow the use of powerful and efficient tools for signal analysis.

VI. Discussion

GAD is a computationally fast algorithm that finds atoms that are sparse. It is motivated by the observation that sparsity in the dictionary and sparsity in the decomposition appear to be linked, for certain types of signals. This notion is supported by the results in this paper: whilst looking for sparse atoms and making no assumptions on the decomposition, the GAD algorithm yields a signal decomposition that is sparse.

Although the only sparsity measure considered here is the sparsity index, we have compared the results to other measures of sparsity including the Gini index, which was found to outperform several other sparsity measures [29]. Our experimental results indicated that the performance of the algorithm is not noticeably different. However, we are considering to study this further in future work.

In its present form, the GAD method looks for onsets, and it might be argued that it does not take advantage of all the possible redundancy in the speech signal, by not exploiting the pitch structure of the signal. In future work we are considering searching for sparsity in the frequency domain, and perhaps in prototype waveform domain [30]. On the other hand, in its present form GAD is a general algorithm that can be used with a variety of data because it does not make any assumptions on its characteristics.

Although the GAD algorithm is currently at the theoretical stage, it is a fast method that might in future be used in real practical applications such as speech coding. In this case, like with PCA, this method would require the transmission of the signal adaptive dictionary. Other applications to which we are particularly interested in applying the GAD method include image processing and biomedical signal processing. Biomedical applications typically give rise to large data sets, for instance, in microarray experiments the expression values of thousands of genes are generated. Therefore, in this case the algorithm would have to be extended to deal with large data sets. We are also considering the application of this approach to the problem of source separation.

VII. Conclusion

In this paper, we have presented a greedy adaptive dictionary learning algorithm, that finds new dictionary elements that are sparse. The algorithm constructs a signal-adaptive orthogonal dictionary, whose atoms encode local properties of the signal. The algorithm has been shown to yield sparse atoms and a sparse signal representation. Its performance was compared to that of PCA and K-SVD methods, and it was found to give good signal approximations, even as the number of atoms in the reconstructions decreases considerably.

It results in better signal reconstruction than K-SVD and it has good tolerance to noise and does not exhibit distortion when noise reduction is performed at low noise levels.

VIII. Acknowledgment

The authors would like to thank the anonymous reviewers for their input that helped improving the quality of the presentation of this work.

References Références Referencias

1. P. Földiák, "Forming sparse representations by local anti-Hebbian learning," *Biolog. Cybern.,* vol. 64, pp. 165–170, 1990.
2. B. Olshausen and D. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature,* vol. 381, pp. 607–609, 1996.
3. T. Shan and L. Jiao, "New evidences for sparse coding strategy employed in visual neurons: From the image processing and nonlinear approximation viewpoint," in *Proc. Eur. Symp. Artif. Neural Netw. (ESANN),* 2005, pp. 441–446.
4. M. Zibulevsky and B. A. Pearlmutter, "Blind source separation by sparse decomposition in a signal dictionary," *Neural Comput.,* vol. 13, no. 4, pp. 863–882, 2001.
5. Ö. Yilmaz and S. Rickard, "Blind separation of speech mixtures via time-frequency masking," *IEEE Trans. Signal Process.,* vol. 52, no. 7, pp. 1830–1847, Jul. 2004.
6. R. Gribonval, "Sparse decomposition of stereo signals with matching pursuit and application to blind separation of more than two sources from a stereo mixture," in Proc. *IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP),* 2002, vol. 3, pp. 3057–3060.
7. M. Elad and M. Aharon, "Image denoising via sparse redundant representations over learned dictionaries," *IEEE Trans. Image Process.,* vol. 15, no. 12, pp. 3736–3745, Dec. 2006.
8. R. Gribonval and K. Schnass, "Some recovery conditions for basis learning by L1-minimization," in *Proc. Int. Symp. Commun., Control, Signal Process. (ISCCSP),* 2008, pp. 768–733.
9. L. Rebollo-Neira, "Dictionary redundancy elimination," *IEE Proc.—Vis., Image, Signal Process.,* vol. 151, pp. 31–34, 2004.
10. M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Comput.,* vol. 12, pp. 337–365, 2000.
11. K. Kreutz-Delgado, J. Murray, D. Rao, K. Engan, T. Lee, and T. Sejnowski, "Dictionary learning

algorithms for sparse representations," *Neural Comput.,* vol. 15, pp. 349–396, 2003.
12. I. Gorodnitsky, J. George, and B. Rao, "Neuromagnetic source imaging with FOCUSS: A recursive weighted minimum norm algorithm," *J. Electroencephalography Clinical Neurophysiol.,* vol. 95, pp. 231–251,1995.
13. M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representations," *IEEE Trans. Signal Process.,* vol. 54, no. 11, pp. 4311–4322, Nov. 2006.
14. M. D. Plumbley, "Dictionary learning for L1-exact sparse coding," in *Proc. Int. Conf. Ind. Compon. Anal. Signal Separat. (ICA),* 2007, pp. 406–413.
15. J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. Int. Conf. Mach. Learn. (ICML),* 2009, vol. 382, pp. 689–696.
16. 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, Mar. 2010.
17. M. G. Jafari, E. Vincent, S. A. Abdallah, M. D. Plumbley, and M. E. Davies, "An adaptive stereo basis method for convolutive blind audio source separation," *Neurocomputing,* vol. 71, pp. 2087–2097, 2008.
18. S. A. Abdallah and M. D. Plumbley, "If edges are the independent components of natural images, what are the independent components of natural sounds?," in *Proc. Int. Conf. Ind. Compon. Anal. Blind Signal Separat. (ICA),* 2001, pp. 534–539.
19. M. Goodwin and M. Vetterli, "Matching pursuit and atomic signal models based on recursive filter banks," *IEEE Trans. Signal Process.,* vol. 47, no. 7, pp. 1890–1902, Jul. 1999.
20. V. Tan and C. Févotte, "A study of the effect of source sparsity for various transforms on blind audio source separation performance," in *Proc. Workshop Signal Process. With Adapt. Sparse Structured Represent. (SPARS),* 2005.
21. M. Jafari and M. Plumbley, "An adaptive orthogonal sparsifying transform for speech signals," in *Proc. Int. Symp. Commun., Control, Signal Process. (ISCCSP),* 2008, pp. 786–790.
22. S. Mallat and Z. Zhang, "Matching pursuit with time-frequency dictionaries," *IEEE Trans. Signal Process.,* vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
23. S. Mallat, G. Davis, and Z. Zhang, "Adaptive time-frequency decompositions," *SPIE J. Opt. Eng.,* vol. 33, pp. 2183–2191, 1994.
24. S. Haykin, *Neural Networks: A Comprehensive Foundation,* 2nd ed. Upper Saddle River, NJ: Prentice Hall, 1999.
25. "The freesound project. A collaborative database of creative commons licensed sounds," [Online]. Available: www.freesound.org
26. R. Rubinstein, [Online]. Available: www.cs.technion.ac.il/~ronrubin/software.html
27. S. Valiollahzadeh, H. Firouzi, M. Babaie-Zadeh, and C. Jutten, "Image denoising using sparse representations," in *Proc. Int. Conf. Ind. Compon. Anal. Signal Separat. (ICA),* 2009, pp. 557–564.
28. A. Hyvaerinen, J. Karhunen, and E. Oja, *Independent Component Analysis.* New York: Wiley, 2000.
29. N. Hurley and S. Rickard, "Comparing measures of sparsity," *IEEE Trans. Inf. Theory,* vol. 55, no. 10, pp. 4723–4741, Oct. 2009.
30. W. B. Kleijn, "Encoding speech using prototype waveforms," *IEEE Trans. Speech Audio Process.,* vol. 4, no. 4, pp. 386–399, Oct. 1993.