

AN ADAPTIVE ORTHOGONAL SPARSIFYING TRANSFORM FOR SPEECH SIGNALS

Maria G. Jafari and Mark D. Plumbley

Centre for Digital Music, Department of Electronic Engineering
Queen Mary University of London, London E1 4NS, UK
maria.jafari@elec.qmul.ac.uk

ABSTRACT

In this paper we consider the problem of representing a speech signal with an adaptive transform that captures the main features of the data. The transform is orthogonal by construction, and is found to give a sparse representation of the data being analysed. The orthogonality property implies that evaluation of both the forward and inverse transform involve a simple matrix multiplication. The proposed dictionary learning algorithm is compared to the K singular value decomposition (K-SVD) method, which is found to yield very sparse representations, at the cost of a high approximation error. The proposed algorithm is shown to have a much lower computational complexity than K-SVD, while the resulting signal representation remains relatively sparse.

Index Terms— Approximation methods, Signal analysis, Signal representations, Discrete transforms

I. INTRODUCTION

Flexible signal decompositions have been the subject of ongoing research for many years [1]–[3], with the aim of better characterising local structures in the signal to be analysed. The matching pursuit (MP) algorithm in [1] has been widely used over the years to perform this type of decompositions, offering an adaptive signal representation, but only within a particular pre-defined redundant time-frequency dictionary, such as a dictionary of Gabor functions. Matching pursuit is quite a versatile algorithm, that works also with user-defined dictionaries; however, its main drawback is that it is computationally expensive, so that work has been done to implement a fast version [4], culminating in the release of a matching pursuit toolkit [5].

In this paper, we consider the problem of decomposing a signal $\mathbf{s}(t) \in \mathbb{R}^{t_{\max}}$ into a linear combination of atoms $\mathbf{a}_l(t)$ [3]

$$\mathbf{s}(t) = \sum_{l=1}^L \alpha_l \mathbf{a}_l(t), \quad \forall t \in \{1, \dots, t_{\max}\} \quad (1)$$

where α_l are the expansion coefficients which encode explicit information regarding the properties of the signal $\mathbf{s}(t)$, depending on the choice of dictionary [1]. The set of atoms $\mathbf{a}_l(t)$, $\forall l = 1, \dots, L$, constitutes the dictionary.

The desire to derive compact representations for certain signal processing applications such as signal analysis, coding, or denoising, implies that we seek representations that are sparse, i.e. with most coefficients close to zero. In this case, typically the dictionary is overcomplete, or redundant, and a relatively small number of dictionary elements are taken out of a large dictionary to represent the signal $\mathbf{x}(t)$. Thus, the expansion in (1) is not unique. It is often difficult to determine a relationship between a class of signals and a particular dictionary, especially for natural signals [6]. This has led researchers to look for learned, rather than fixed dictionaries, using techniques such as independent component analysis (ICA), as the underlying learning algorithm [2], [7], [8]. These methods, however, are computationally very expensive, and often require fairly large datasets in order to learn the dictionary bases.

Here, we propose an adaptive transform, for speech signals, that finds dictionary atoms from the available data. The transform is based on a greedy algorithm that selects the frame of data with highest energy as the first atom in the dictionary; the contribution of this atom to all data frames is subsequently subtracted from the speech signal, resulting in one data block being set to zero. The process is then repeated, forcing the transform to be orthogonal, since all the components lying in the direction of a particular vector are set to zero at each iteration.

Our heuristic algorithm results in an orthogonal transform by construction; we believe that the transform is likely to give a reasonably sparse representation for the speech signal, because of the re-arranging of the data into frames, and ensuring that the L1-norm is minimum. There are two main advantages to this algorithm: firstly, the atoms are extracted from the observed data, and therefore they will be directly relevant to the data being analysed; secondly, the fact that the transform is orthogonal, implies that only direct matrix multiplication (or multiplication by its transpose) will be needed to analyse the signal (reconstruct it). Hence, the computational complexity of the proposed transform is lower than that of other sparsifying transforms which, having learned a dictionary, must then use tools such as matching pursuit for reconstruction.

We find that many of the basis resulting from the transform exhibit properties suggesting that they represent characteristics of the speech data. We compare the results for our proposed algorithm with the application of the discrete cosine transform (DCT), which is believed to give sparse representations for music and speech [9]. The method is also compared to the K-SVD algorithm, which also adaptively learns a dictionary from the data, under the constraint that the signal representation is sparse. The structure of the paper is as follows: our proposed adaptive sparsifying orthogonal transform is introduced in section II, and K-SVD is described in section III. Their computational complexity is discussed in section IV, while some experimental results are presented in section V, and conclusions are drawn in section VI.

II. ADAPTIVE ORTHOGONAL SPARSIFYING TRANSFORM

The proposed method operates upon a signal block $\mathbf{x}_k(n) \in \mathbb{R}^N$, obtained when overlapping data frames are taken from the observed signal, $x(t) \in \mathbb{R}^{t_{\max}}$, in the usual way, where the discrete time index has been changed from t to n to reflect this, and the overlap is set to T samples. The signal $\mathbf{x}_k(n)$, $k \in \{1, \dots, k_{\max}\}$, is then the k -th column of the newly constructed matrix $\mathbf{X}(n) \in \mathbb{R}^{N \times k_{\max}}$.

We propose an iterative, matching pursuit-like greedy algorithm, that adaptively learns a user-defined dictionary by sequentially extracting columns of the matrix $\mathbf{X}(n)$, based on the following

$$\max_k \frac{\|\mathbf{x}_k(n)\|_2}{\|\mathbf{x}_k(n)\|_1} \quad (2)$$

where $\|\cdot\|_1$ and $\|\cdot\|_2$ denote the L1- and L2-norm respectively. Thus at each iteration, the method reduces the energy of the data by a maximum amount, across all frames, while ensuring that the L1-norm is reduced by a minimum amount. This is achieved by the following steps:

- 1) Ensure that the columns of $\mathbf{X}(n)$ have unit L1-norm

$$\tilde{\mathbf{x}}_k(n) = \frac{\mathbf{x}_k(n)}{\|\mathbf{x}_k(n)\|_1} \quad (3)$$

where $\mathbf{x}_k(n)$ the k -th column of $\mathbf{X}(n)$. This leads to a new data matrix $\tilde{\mathbf{X}}(n)$, whose columns now have unit L1-norm. Note that the superscript $\tilde{\cdot}$, used hereafter, denotes the normalised matrix and its columns.

- 2) Compute the energy of each frame

$$E(k) = \sum_{n=1}^N |\tilde{\mathbf{x}}_k(n)|^2. \quad (4)$$

- 3) Identify the signal block $\tilde{\mathbf{x}}_{\hat{k}}(n)$, with largest energy

$$E_{\max} = \arg \max_{k \in \mathbb{K}} (E(k)) \quad (5)$$

where $\mathbb{K} = \{1, \dots, k_{\max}\}$, the set of all elements of $\tilde{\mathbf{X}}(n)$.

At each iteration $j \in \{1, \dots, N\}$, the signal with highest energy, $\tilde{\mathbf{x}}_{\hat{k}}(n)$, becomes a dictionary element, and we iteratively define a residual matrix $\mathbf{R}^j(n) \in \mathbb{R}^{N \times k_{\max}}$, which decreases by the appropriate amount, determined by the selected atom $\mathbf{a}_j(n)$ and the coefficient of expansion $\alpha_j(k)$.

- 4) Set the j -th dictionary element $\mathbf{a}_j(n)$ to be equal to $\tilde{\mathbf{x}}_{\hat{k}}(n)$

$$\mathbf{a}_j(n) = \tilde{\mathbf{x}}_{\hat{k}}(n). \quad (6)$$

- 5) Evaluate the coefficients of expansion, given by the inner product between each data block $\tilde{\mathbf{x}}_k(n)$, and the atom $\mathbf{a}_j(n)$

$$\alpha_j(k) = \langle \tilde{\mathbf{x}}_k(n), \mathbf{a}_j(n) \rangle. \quad (7)$$

- 6) Compute the new residual, by removing the component along the chosen atom, for each element k in $\mathbb{R}^j(n)$

$$\mathbf{R}^j(n) = \mathbf{R}^{j-1}(n) - \frac{\alpha_j(k)}{\max_{k \in \mathbb{K}} (\alpha_j(k))} \mathbf{a}_j(n). \quad (8)$$

The term in the denominator of $\frac{\alpha_j(k)}{\max_{k \in \mathbb{K}} (\alpha_j(k))}$ in equation (8), is included to ensure that the coefficient of expansion $\alpha_j(k)$ corresponding to the inner product between the selected atom $\mathbf{a}_j(k)$, and the frame of maximum energy $\tilde{\mathbf{x}}_{\hat{k}}(n)$, is normalised to 1. Then, the corresponding column of the residual matrix $\mathbf{R}^j(n)$ is set to zero, since the whole atom is removed.

This is the step that ensures that the transform is orthogonal. To see this, consider the case when $j = 1$, and let $\mathbf{a}_1(n) = \tilde{\mathbf{x}}_{\hat{k}}(n)$. At the end of the first iteration, $\tilde{\mathbf{x}}_{\hat{k}}(n)|_{k=\hat{k}} = 0$, and the dimension of the space, which initially is N , reduces to $N - 1$. Then, the inner product for the second iteration, when $j = 2$, will be

$$\alpha_2(k) = \begin{cases} \langle \tilde{\mathbf{x}}_k(n), \mathbf{a}_2(n) \rangle, & \forall k \neq \hat{k} \\ \langle \mathbf{a}_1(n), \mathbf{a}_2(n) \rangle = 0, & \text{for } k = \hat{k} \end{cases} \quad (9)$$

that is, the new atom $\mathbf{a}_2(n)$ is orthogonal to $\mathbf{a}_1(n)$. As this is repeated, we find that the first N atoms extracted are mutually orthogonal, and therefore the algorithm is an adaptive orthogonal transform.

Finally, the signal matrix is updated by the residual, and the whole process is repeated:

- 7) Set the data matrix equal to the new residual

$$\tilde{\mathbf{X}}(n) = \mathbf{R}^j(n). \quad (10)$$

- 8) Repeat from step 2.

Once the dictionary has been defined, an approximation of the signal must be calculated, which is typically done using computationally expensive approaches, such as matching pursuit. However, a clear advantage of the proposed algorithm is that, since it results in an orthogonal transform,

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

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

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 approximation the solution of

$$i = 1, 2, \dots, N, \quad \min_{\mathbf{x}_i} \{\|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2\} \quad \text{subject to } \|\mathbf{x}_i\|_0 \leq T_0$$

- *Codebook Update Stage*: For each column $k = 1, 2, \dots, K$ in $\mathbf{D}^{(J-1)}$, update it by
 - Define the group of examples that use this atom, $\omega_k = \{i | 1 \leq i \leq N, \mathbf{x}_i^k(i) \neq 0\}$.
 - 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. 1. A description of the K-SVD algorithm, reproduced from [10]

a representation of the signal can be readily obtained by matrix multiplication, between the dictionary matrix $\mathbf{A} = [\mathbf{a}_1^T(n), \dots, \mathbf{a}_{k_{\max}}^T(n)]$ and the signal, $\mathbf{Y}(n) = \mathbf{A}\mathbf{X}(n)$. Similarly, the inverse transform is evaluated from

$$\mathbf{X}^N(n) = \mathbf{A}^T \mathbf{Y}(n) \quad (11)$$

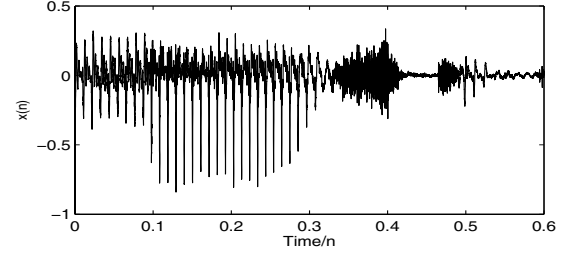
where $\mathbf{X}^N(n)$ is the N term approximation of the signal $\mathbf{X}(n)$.

III. K-SVD

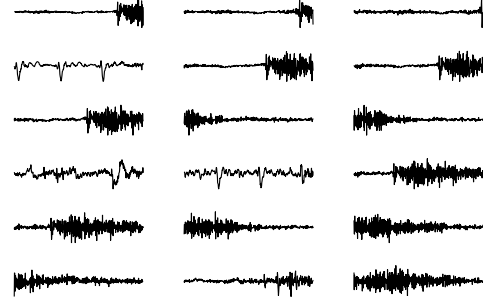
The K-SVD algorithm learns an overcomplete dictionary, under the constraint that the signal representation is sparse, by minimising the following expression [10]:

$$\min_{\mathbf{D}, \mathbf{X}} \{\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2\} \quad \text{subject to } \forall i, \|\mathbf{x}_i\|_0 \leq T_0 \quad (12)$$

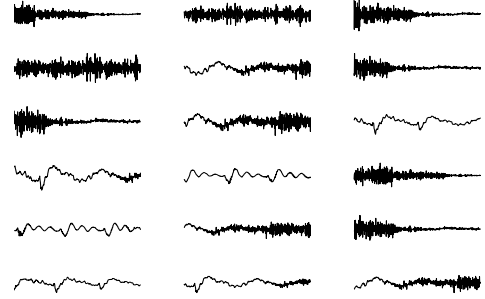
where \mathbf{Y} , \mathbf{X} and \mathbf{D} are the signal to be approximated, the coefficient matrix, and the dictionary matrix, respectively; $\|\cdot\|_F$ is the Frobenius norm, and $\|\cdot\|_0$ is the ℓ^0 norm, counting the nonzero entries of a vector. The algorithm performs dictionary design and signal decomposition simultaneously, by alternatively fixing the dictionary and finding a signal representation, and then updating the dictionary matrix \mathbf{D} one column at the time, while allowing the expansion coefficients to change in this stage [10]. The coefficients update stage can be performed using any approximation pursuit method, such as orthogonal matching pursuit (OMP)



(a) Original speech signal



(b) Examples of atoms learned with the proposed method.



(c) Examples of atoms learned with K-SVD.

Fig. 2. Examples of atoms learned with the (b) adaptive and (c) K-SVD algorithms, from the speech signal shown in the upper plot.

[11], as long as the solution has a fixed and predetermined number of nonzero entries, hence imposing a very strong sparsity constraint. In [10]. The dictionary update stage is based on the singular value decomposition (SVD) of the representation error matrix $\mathbf{Y} - \mathbf{D}\mathbf{X}$, and K SVD computations are performed, each determining a column of the dictionary matrix (K-SVD is described in figure 1).

IV. COMPUTATIONAL COMPLEXITY

The most computationally expensive step in the algorithm described in section II is the evaluation of the coefficients of expansion in equation (7). Evaluation of $\alpha_j(k)$, for a

signal block $\mathbf{x}_k(n)$ of length k_{\max} and an atom $\mathbf{a}_j(n)$ of length N , is $O(Nk_{\max})$ for each iteration $j \in \{1, \dots, N\}$, thus resulting in an overall complexity of $O(N^2k_{\max})$.

The K-SVD algorithm entails performing K (or k_{\max} in the case discussed here) computationally intensive singular value decomposition steps. In general, the complexity of the SVD transform is $O(Nk_{\max}^2)$, and this had to be applied k_{\max} times, each corresponding to a column of the dictionary matrix, thus giving a computational complexity of $O(Nk_{\max}^3)$. In the next section, we will see what these computational times correspond to in real time when performing computer simulations.

V. RESULTS

The algorithm proposed in section II and the K-SVD algorithm were applied to the speech signal in the upper plot of figure 2, which represents a recording from a male speaker, sampled at 16kHz. Some of the atoms obtained with the proposed algorithm from the speech signal, using a frame length of $N = 512$ samples, and an overlap of $T = 500$ samples, are depicted in figure III, while the plots in figure III show some examples of the atoms generated by K-SVD, using the parameters selected in [10]. In both cases, several of the atoms clearly capture characteristics of the speech signal, with many of them being quite localised, while others capturing different properties.

We have seen in a previous section how the proposed method is expected to result in an orthogonal transform. Here, we investigate this by extracting as many atoms as the data affords, that is, we learn $k_{\max} > N$ atoms, and subsequently look at their energy of each one

$$E_a(j) = \sum_{n=0}^N |\mathbf{a}_j(n)|^2. \quad (13)$$

Figure 3 shows the logarithmic plot of the energy of the atoms learned when $k_{\max} = 792$ samples and $N = 512$ samples. It clearly shows that only the first N atoms (indicated by the arrow on the figure) have high energy, while the energy of the following $k_{\max} - N$ atoms are negligibly small. Thus, although we start with an overcomplete dictionary, the algorithm results in a complete set of basis functions.

To determine how sparse the representation obtained with the proposed approach is, we evaluated the sparsity index of the transformed signal blocks $\mathbf{Y}(n)$, and compared this to the transform coefficients obtained with the discrete cosine transform, and the K-SVD algorithm¹, as well as comparing with the sparsity index of the original data. The sparsity index of a signal y is defined as [9] $\xi = \|\mathbf{y}\|_1 / \|\mathbf{y}\|_2$; generally, the lower the sparsity index is, the sparser the

¹The KSVD Matlab Toolbox, downloaded from <http://www.cs.technion.ac.il/~elad/software/>, was used to implement the K-SVD algorithm.

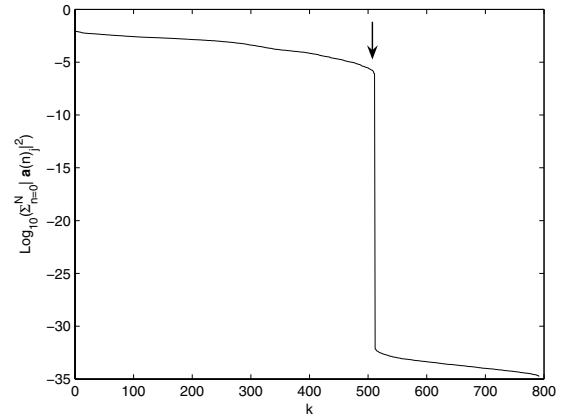


Fig. 3. Logarithmic plot of the energy of the atoms extracted with the proposed adaptive orthogonal transform.

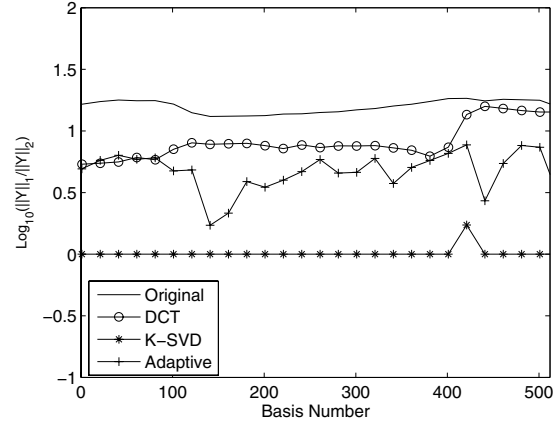


Fig. 4. Logarithmic plot of the energy of the atoms extracted with the proposed adaptive orthogonal transform.

signal y . The discrete cosine transforms was selected because it is believed that speech signals are sparser in this domain [9]. Figure 4 shows a logarithmic plot of the sparsity index for the four cases. The results show that, as expected, the DCT transform gives sparser representations than in time domain. Nonetheless, the proposed transform (denoted as Adaptive in the figure) gives sparser results, which however are not as sparse as those obtained with K-SVD. It should be noted that the latter algorithm imposes a very strong sparsity constraint on the data, by limiting the number of non-zero elements in the representation from OMP to a small number, and in [10] the authors allow only three non-zero entries. A consequence of this is that the K-SVD algorithm yields a signal representation that is not particularly accurate. This can be seen from the approximation error ϵ , obtained when the function f is approximated by \tilde{f} , $\epsilon = \|\tilde{f} - f\|$. The approximation error for the signal reconstructed with

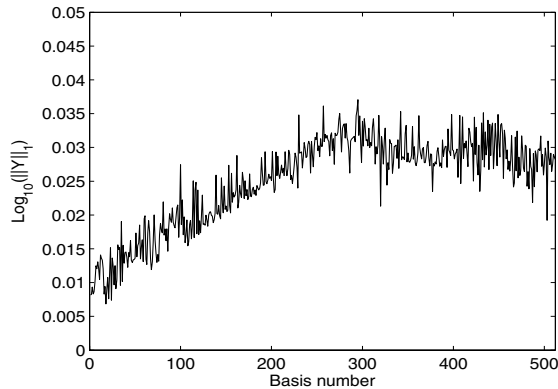


Fig. 5. Logarithmic plot of the L1-norm of the atoms active for a particular frame, ordered as they are extracted.

DCT and for the proposed method was found to be $\epsilon = 3.03 \times 10^{-14}$, and $\epsilon = 16.02$ respectively, while K-SVD gave an error of 260.17. Moreover, the speed of learning of K-SVD is quite low, as discussed in section IV. In our simulations for the experiment described above, conducted on a Pentium IV at 3.4GHz, using Matlab Version 7.0.4 (R14SP2), and under the Microsoft Windows XP operating system, the computation time for the proposed algorithm was about 12 seconds, while the K-SVD algorithm required just over 26 minutes (1611 seconds).

Finally, figure 5 plots the contribution of each atom for a particular data frame, in the order in which they are extracted by the algorithm. The fluctuations in the plot are due The function is not monotonic increasing because the algorithm that we use is a greedy algorithm which makes the best choice at each iteration. Nonetheless, the trend of the curve indicates that more atoms are active as the algorithm proceeds, and interestingly, after about 300 steps, most information available has been learned.

VI. CONCLUSIONS

In this paper we have presented an adaptive orthogonal transform, which we have shown to result in sparse representations for speech signals. The algorithm constructs a data-specific complete dictionary, whose atoms clearly encode local properties of the signal. The algorithm has been compared to the K-SVD algorithm, both in terms of performance, and speed of learning. It was found that the representation obtained with the latter is sparser, but the reconstructed signal is of poor quality, while the speed of convergence is much higher with the proposed adaptive method. Since it is based on a complete dictionary, the algorithm has also the advantage of requiring a simple matrix multiplications for the forward and inverse transform.

VII. REFERENCES

- [1] S. Mallat and Z. Zhang, "Matching pursuit with time-frequency dictionaries," *IEEE Trans. on Signal Processing*, vol. 41, pp. 3397–3415, 1993.
- [2] M. Lewicki and T. Sejnowski, "Learning overcomplete representations," *Neural computation*, vol. 12, pp. 337–365, 2000.
- [3] M. Goodwin and M. Vetterli, "Matching pursuit and atomic signal models based on recursive filter banks," *IEEE Trans. on Signal Processing*, vol. 47, pp. 1890–1902, 1999.
- [4] R. Gribonval, "Fast matching pursuit with a multiscale dictionary of gaussian chirps," *IEEE Trans. on Signal Processing*, vol. 49, pp. 994–1001, 2001.
- [5] S. Krstulovic and R. Gribonval, "Mrtk: Matching pursuit made tractable," in *Proc. ICASSP*, 2006, vol. 3, pp. 14–19.
- [6] S. Lesage P. Jost, P. Vandergheynst and R. Gribonval, "Learning redundant dictionaries with translation invariance property: the motif algorithm," in *Proc. SPARS*, 2005.
- [7] S. A. Abdallah and M. D. Plumbley, "Application of geometric dependency analysis to the separation of convolved mixtures," in *Proc. of the International Conference on Independent Component Analysis and Blind Signal Separation (ICA 2004), Granada, Spain*, 2004, pp. 22–24.
- [8] M. G. Jafari, S. A. Abdallah, M. D. Plumbley, and M. E. Davies, "Sparse coding for convolutive blind audio separation," in *Proc. ICA*. 2006, pp. 132–139, Springer-Verlag, Berlin.
- [9] V. Tan and C Févotte, "A study of the effect of source sparsity for various transforms on blind audio source separation performance," in *Proc. SPARS*, 2005.
- [10] Michal Aharon, Michael Elad, and Alfred Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representations," *IEEE Trans. on Signal Processing*, vol. 54, pp. 4311–4322, 2006.
- [11] G. Davis, *Adaptive nonlinear approximations*, Ph.D. thesis, New York University, 1994.