Rejected in IEEE Signal Processing Letters
(Submission date was on 2 Dec 2006)
(see http://ee.sharif.ir/~SLzero for MATLAB code)

Fast Sparse Representation based on Smoothed ℓ^0 Norm

G. Hosein Mohimani[1], Massoud Babaie-Zadeh[1]* *Member* and Christian Jutten[2] *Member*

***Abstract*— In this paper, a new algorithm for Sparse Component Analysis (SCA) or atomic decomposition on over-complete dictionaries is presented. The algorithm is essentially a method for obtaining sufficiently sparse solutions of underdetermined systems of linear equations. The solution obtained by the proposed algorithm is compared with the minimum ℓ^1-norm solution achieved by Linear Programming (LP). It is experimentally shown that the proposed algorithm is about two to three orders of magnitude faster than the state-of-the-art interior-point LP solvers, while providing the same (or better) accuracy.**

***Index Terms*— sparse component analysis, over-complete atomic decomposition, sparse representation, over-complete signal representation, sparse source separation, blind source separation.**

I. INTRODUCTION

Obtaining sparse solutions of under-determined systems of linear equations is of significant importance in signal processing and statistics. Despite recent theoretical developments [1], [2], [3], the computational cost of the methods has remained as the main restriction, especially for large systems (large number of unknown/equations). In this article, a new approach is proposed which provides a considerable reduction in complexity. To introduce the problem in more details, we will use the context of Sparse Component Analysis (SCA). The discussions, however, may be easily followed in other contexts of application, for example, in finding 'sparse decomposition' of a signal in an over-complete dictionary, which is the goal of the so-called over-complete 'atomic decomposition' [4].

SCA can be viewed as a method to achieve separation of sparse sources. The general Blind Source Separation (BSS) problem is to recover n unknown (statistically independent) sources from m observed mixtures of them, where little or no information is available about the sources (except their statistical independence) and the mixing system. In linear instantaneous (noiseless) model, it is assumed that $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$ in which $\mathbf{x}(t)$ and $\mathbf{s}(t)$ are the $m \times 1$ and $n \times 1$ vectors of sources and mixtures and $\mathbf{A}$ is the $m \times n$ mixing matrix. The goal of BSS is then to find $\mathbf{s}(t)$ only by observing $\mathbf{x}(t)$. The general BSS problem is not easy for the case $n > m$. However, if the sources are sparse (i.e., not a totally blind situation), then the problem can be solved in two steps [3], [2]: first estimating the mixing matrix, and then estimating the sources assuming $\mathbf{A}$ to be known. For sparse sources, the first step is not difficult and may be accomplished by means of clustering [3], [2], [5], [6]. The second step requires finding the sparse solution of an underdetermined system of linear equation (which is the subject of this article): for each instant of time (t), the under-determined system of linear equations $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$ should be solved under the sparsity assumption [3], [2], [7], [8].

In the atomic decomposition viewpoint [9], we have a 'single' signal $s(t)$ whose samples are collected in the $m \times 1$ signal vector $\mathbf{s} = [s(1), \ldots, s(m)]^T$ and we would like to represent it as a linear combination of n, $m \times 1$ signal vectors $\{\boldsymbol{\varphi}_i\}_{i=1}^n$. After [10], the $\boldsymbol{\varphi}_i$'s are called *atoms* and they collectively form a *dictionary* over which the signal is to be decomposed. We may write $\mathbf{s} = \sum_{i=1}^n \alpha_i \boldsymbol{\varphi}_i = \boldsymbol{\Phi}\boldsymbol{\alpha}$, where $\boldsymbol{\Phi} \triangleq [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_n]$ is the $m \times n$ dictionary (matrix) and $\boldsymbol{\alpha} \triangleq (\alpha_1, \ldots, \alpha_n)^T$ is the $n \times 1$ vector of coefficients. A dictionary with $m > n$ is called *overcomplete*. Although, $m = n$ is sufficient to obtain such a decomposition (e.g. Discrete Fourier Transform), using overcomplete dictionaries has a lot of advantages in many diverse applications (refer for example to [4] and the references in it). In all these applications, we are looking for a sparse representation, that is, we would like to use a number of atoms as small as possible to represent the signal. Again, we have the problem of finding sparse solutions of the underdetermined system of linear equations $\boldsymbol{\Phi}\boldsymbol{\alpha} = \mathbf{s}$.

To obtain the sparsest solution of $\mathbf{A}\mathbf{s} = \mathbf{x}$, we may search for a solution of it having minimal ℓ^0 norm, i.e., minimum number of nonzero components. It is usually stated in the literature [4], [11], [12], [3] that searching the minimum ℓ^0 norm is an intractable problem as the dimension increases (because it requires a combinatorial search), and it is too sensitive to noise (because every small amount of noise changes completely the ℓ^0 norm of a vector). Consequently, the researchers look for other approaches to find sparse solution of $\mathbf{A}\mathbf{s} = \mathbf{x}$ which are tractable. One of the most successful approaches is Basis Pursuit (BP) [9], [1], [12], [3] which finds the minimum ℓ^1 norm (that is, the solution of $\mathbf{A}\mathbf{s} = \mathbf{x}$ for which $\sum_i |s_i|$ is minimized). Such a solution can be easily found by Linear Programming (LP) methods. The idea of Basis Pursuit is based on the observation that for large systems of equations, the minimum ℓ^1 norm solution is also the minimum ℓ^0 norm solution [1], [9], [13], [14]. By utilizing fast LP algorithms, specifically interior-point LP solvers, large-scale problems with thousands of sources and mixtures become tractable. However, although this approach is tractable, it is still very

[1]Electrical engineering department, Sharif university of technology, Tehran, Iran.

[2]Laboratoire des Images et des Signaux (LIS), Institut National Polytechnique de Grenoble (INPG), France.

This work has been partially funded by French Embassy in Tehran, and by Center for International Research and Collaboration (ISMO).

Author's email addresses are: gh1985im@yahoo.com, mbzadeh@yahoo.com and Christian.Jutten@inpg.fr

slow. Another approach is Matching Pursuit (MP) [10], [15], [3] which is very fast, but is somewhat heuristic and does not provide good estimation of the sources.

In this article, we present a fast method for finding the sparse solution of an under-determined system of linear equations, which is based on minimization of ℓ^0 norm. The main idea is as follows: The problems of using ℓ^0 norm (that is, the need for a combinatorial search for its minimization, and its too large sensibility to noise) are both due to the fact that the ℓ^0 norm of a vector is a *discontinuous* function of that vector. Our idea is then to approximate this *discontinuous* function by a suitable *continuous* one, and minimize it by means of a minimization algorithm for continuous functions (e.g., steepest descent method). We will see that our method performs typically three orders of magnitude faster than BP (based on interior-point LP solvers), while producing solutions with the same or better accuracy than BP.

The paper is organized as follows. The next section introduces a family of Gaussian sparsity norms and discusses their optimization. The algorithm is then stated in Section III. Finally, Section IV provides some experimental results of our algorithm and its comparison with BP.

II. The Main Idea

As stated in the previous section, the main idea is to approximate the ℓ^0 norm by a smooth (continous) function, which will let us to use gradient based methods for its minimization. In this section we introduce a family of smooth approximators of ℓ^0 norm, whose optimization results in a fast algorithm for finding the sparse solution while preserving noise robustness.

The ℓ^0 norm of $\mathbf{s} = [s_1 \dots s_n]^T$ is defined as the number of non-zero components of $\mathbf{s}$. In other words if we define

$$\nu(s) = \begin{cases} 1 & s \neq 0 \\ 0 & s = 0 \end{cases} \tag{1}$$

then

$$\|\mathbf{s}\|_0 = \sum_{i=1}^{n} \nu(s_i). \tag{2}$$

It is clear that the discontinuities of ℓ^0 norm are caused by the discontinuities of the function ν. If we replace ν by a smooth estimation of it in (2), we obtain a smooth estimation of ℓ^0 norm. This may also provide some robustness to noise.

Different functions may be utilized for this aim. We use zero-mean Gaussian family of functions which seem to be very useful for this application, because of their differentiability. By defining:

$$f_\sigma(s) = \exp(-s^2/2\sigma^2), \tag{3}$$

we have:

$$\lim_{\sigma \to 0} f_\sigma(s) = \begin{cases} 1 & s = 0 \\ 0 & s \neq 0 \end{cases}. \tag{4}$$

Consequently, $\lim_{\sigma \to 0} f_\sigma(s) = 1 - \nu(s)$, and therefore if we define:

$$F_\sigma(\mathbf{s}) = \sum_{i=1}^{n} f_\sigma(s_i), \tag{5}$$

we have:

$$\lim_{\sigma \to 0} F_\sigma(\mathbf{s}) = \sum_{i=1}^{n} (1 - \nu(s_i)) = n - \|\mathbf{s}\|_0. \tag{6}$$

We take then $n - F_\sigma(\mathbf{s})$ as an approximation to $\|\mathbf{s}\|_0$:

$$\|\mathbf{s}\|_0 \approx n - F_\sigma(\mathbf{s}). \tag{7}$$

The value of σ specifies a trade-off between the accuracy and the smoothness of the approximation: the smaller σ, the better approximation, and the larger σ, the smoother approximation.

From (6), the minimization of ℓ^0 norm is equivalent to maximization of F_σ for sufficiently small σ. This maximization should be done on the affine set $\mathcal{S} = \{\mathbf{s} \mid \mathbf{As} = \mathbf{x}\}$.

For small values of σ, F_σ contains a lot of local maxima. Consequently, it is very difficult to directly maximize this function for very small σ's. However, as σ grows, the function becomes smoother and smoother, and for sufficiently large values of σ, as we will show, there is no local maxima (see Theorem 1 of the next section).

Our idea for escaping local maxima is then to decrease the value of σ gradually[1]: for each value of σ we use a steepest ascent algorithm for maximizing F_σ, and *the initial value of this steepest ascent algorithm is the maximizer of F_σ obtained for the previous (larger) value of σ*. Since for each σ, its value is not too different from the previous σ, the steepest ascent algorithm is initialized not far from the actual maximum. Consequently, we hope that it would not be trapped in the local maxima.

Remark. Equation (6) proposes that $F_\sigma(\cdot)$ can be seen as a measure of the 'sparsity' of a vector (especially for small σ's): the sparser $\mathbf{s}$, the larger $F_\sigma(\mathbf{s})$. In this viewpoint, maximizing $F_\sigma(\mathbf{s})$ on a set is equivalent to finding the 'sparsest' element of that set.

III. The Algorithm

The final algorithm based on the main idea of the previous section is given in Fig. 1. As indicated in the algorithm, the final value of previous estimation is used for the initialization of the next steepest ascent. By choosing a slowly decreasing sequence of σ_i, we may escape from getting trapped in the local maxima, and obtain the sparsest solution.

Remark 1. The internal loop (steepest ascent for a fixed σ) is repeated a fixed and small number of times (L). In other words, for increasing the speed, we do not wait for the (internal loop of the) steepest ascent algorithm to converge. This may be justified by the gradual decrease in σ, and the fact that for each σ, we do not need the exact maximizer of F_σ. All we need, is to enter the region near the (absolute) maximizer of F_σ for escaping from its local maximizers.

Remark 2. Steepest ascent consists of iterations of the form $\mathbf{s} \leftarrow \mathbf{s} + \mu_k \nabla F_\sigma(\mathbf{s})$. Here, the step-size parameters μ_k should be decreasing, i.e., for smaller σ's, smaller μ_k's should be applied. This is because for smaller σ's, the function F_σ is

[1]This idea is similar to simulated annealing, but, here, the sequence of decreasing values is short and easy to define so that the solution is achieved in a few steps, usually less than 10.

- Initialization:
  1) Choose an arbitrary solution from the feasible set $\mathcal{S}$, $\mathbf{v}_0$, e.g., the minimum ℓ^2 norm solution of $\mathbf{As} = \mathbf{x}$ obtained by pseudo-inverse (see the text).
  2) Choose a suitable decreasing sequence for σ, $[\sigma_1 \dots \sigma_K]$.
- for $k = 1, \dots, K$:
  1) Let $\sigma = \sigma_k$.
  2) Maximize (approximately) the function F_σ on the feasible set $\mathcal{S}$ using L iterations of the steepest ascent algorithm (followed by projection onto the feasible set):
     - Initialization: $\mathbf{s} = \mathbf{v}_{k-1}$.
     - for $j = 1 \dots L$ (loop L times):
       a) Let: $\Delta\mathbf{s} = [s_1 \exp(-s_1^2/2\sigma_k^2), \dots, s_n \exp(-s_n^2/2\sigma_k^2)]^T$.
       b) Let $\mathbf{s} \leftarrow \mathbf{s} - \mu\Delta\mathbf{s}$ (where μ is a small positive constant).
       c) Project $\mathbf{s}$ back onto the feasible set $\mathcal{S}$:
     $$\mathbf{s} \leftarrow \mathbf{s} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}(\mathbf{As} - \mathbf{x})$$
  3) Set $\mathbf{v}_k = \mathbf{s}$.
- Final answer is $\mathbf{s} = \mathbf{v}_l$.

Fig. 1. The final algorithm.

more 'fluctuating', and hence smaller step-sizes should be used for its maximization. If we set[2] $\mu_k = \mu\sigma_k^2$, we obtain $\mathbf{s} \leftarrow \mathbf{s} - \mu\Delta\mathbf{s}$ as stated in the algorithm of Fig. 1 (note that $\Delta\mathbf{s} \triangleq -\nabla F_\sigma(\mathbf{s})/\sigma^2$).

Remark 3. The algorithm may work by initializing $\mathbf{v}_0$ (the initial estimation of the sparse solution) to an arbitrary solution of $\mathbf{As} = \mathbf{x}$. However, the best initial value of $\mathbf{v}_0$ is the minimum ℓ^2 norm solution of $\mathbf{As} = \mathbf{x}$, which is given by the pseudo-inverse of $\mathbf{A}$. It is because this solution is the (unique) maximizer of $F_\sigma(\mathbf{s})$ on the feasible set $\mathcal{S}$, where σ tends to infinity[3]. This is formally stated in the following theorem (refer to appendix for the proof).

Theorem 1: The solution of the problem:

$$\textit{Maximize } F_\sigma(\boldsymbol{s}) \textit{ subject to } \boldsymbol{As} = \boldsymbol{x},$$

where $\sigma \to \infty$, is the minimum ℓ^2 norm solution of $\boldsymbol{As} = \boldsymbol{x}$, that is, $\boldsymbol{s} = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1}\boldsymbol{x}$.

Remark 4. Having initiated the algorithm with the minimum ℓ^2 norm solution (which corresponds to $\sigma = \infty$), the next value for the σ (i.e., σ_1) may be chosen about two to four times of the maximum absolute value of the obtained sources ($\max_i |s_i|$). To see the reason, note first that:

$$\exp(-s_i^2/2\sigma^2) = \begin{cases} 1 & , \text{if } |s_i| \ll \sigma \\ 0 & , \text{if } |s_i| \gg \sigma \end{cases}. \qquad (8)$$

Consequently, if we take, for example, $\sigma > 4\max_i |s_i|$, then $\exp(-s_i^2/2\sigma^2) > 0.96 \approx 1$, and comparison with (8) shows that this value of σ acts virtually like infinity for all the s_i's.

[2]In fact, we may think about changing the σ in (3) and (5) as looking at the same curve (or surface) at different 'scales', where the scale is proportional to σ^2. For having the same step-sizes of the steepest ascent algorithm in these different scales, μ_k should be proportional to σ^2.

[3]In another point of view, one may think about the minimum ℓ^2 norm solution as a rough estimate of the sparse solution, which will be modified in the future iterations of the algorithm.

TABLE I

PROGRESS OF THE METHOD FOR A PROBLEM WITH $n = 1000$, $m = 400$ AND $p = 0.1$.

itr. #	σ	ΔT	MSE	SNR (dB)
1	1	0.032	$4.75\,e-2$	3.48
2	0.5	0.046	$2.25\,e-2$	6.73
3	0.2	0.047	$5.00\,e-3$	13.29
4	0.1	0.063	$2.00\,e-3$	17.18
5	0.05	0.047	$7.09\,e-4$	21.74
6	0.02	0.031	$1.75\,e-4$	27.80
7	0.01	0.016	$1.28\,e-4$	29.17
algorithm		total time	MSE	SNR (dB)
Our method		0.282 seconds	$1.28\,e-4$	29.17
LP interior		213 seconds	$3.85\,e-4$	24.39

For the next σ_k's ($k \geq 2$), we have used $\sigma_k = \alpha\sigma_{k-1}$, where α is usually between 0.5 and 1.

Remark 5. The final value of σ depends on the noise level. For the noiseless case, it can be decreased arbitrarily to zero (its minimum values is determined by the desired accuracy, and/or machine precision). For the noisy case, it should be terminated about one to two times of the energy of the noise. This is because, while σ is in this range, (8) shows that the cost function treats small (noisy) samples as zeros (i.e., for which $f_\sigma(s_i) \approx 1$). However, below this range, the algorithm tries to 'learn' these noisy values, and moves away from the true answer. Restricting σ_i to be above the energy of the noise, provides the robustness of this approach to noise, which was one of the difficulties of using the exact ℓ^0 norm.

In the simulations of this paper, this noise level was assumed to be known[4].

IV. EXPERIMENTAL RESULTS

In this section, we justify the performance of the presented approach and compare it with BP. Sparse sources are artificially created using the Mixture of Gaussians (MoG) model[5]:

$$s_i \sim p \cdot \mathcal{N}(0, \sigma_{\text{on}}) + (1-p) \cdot \mathcal{N}(0, \sigma_{\text{off}}), \qquad (9)$$

where p denotes the probability of activity of the sources. σ_{on} and σ_{off} are the standard deviations of the sources in active and inactive mode, respectively. In order to have sparse sources, the parameters are required to satisfy the conditions $\sigma_{\text{off}} \ll \sigma_{\text{on}}$ and $p \ll 1$. σ_{off} is to model the noise, and the larger values of σ_{off} produces stronger noise. In the simulation σ_{on} is set to 1. Each column of the mixing matrix is randomly generated using the normal distribution which is then normalized to unity.

To evaluate the decomposition quality, Signal-to-Noise Ratio (SNR) and Mean Square Error (MSE) are estimated as quality measures. SNR (in dB) is defined as $20\log(\|\hat{\mathbf{s}} - \mathbf{s}\|_2/\|\mathbf{s}\|_2)$ and MSE as $(1/n)\|\hat{\mathbf{s}} - \mathbf{s}\|_2^2$, where $\mathbf{s}$ and $\hat{\mathbf{s}}$ denote the actual source and its estimation, respectively.

The values used for the experiment are $n = 1000$, $m = 400$, $p = 0.1$, $\sigma_{\text{off}} = 0.01$, $\sigma_{\text{on}} = 1$, and the sequence for σ is fixed

[4]Note that its exact value is not necessary, and in practice a rough estimation is sufficient.

[5]The model we have used is also called the Bernoulli-Gaussian model: each source is 'active' with probability p, and is inactive with probability $1-p$. If it is active, its value is modeled by a zero-mean Gaussian random variable with variance σ_{on}^2; if it is not active, its value is modeled by a zero-mean Gaussian random variable with variance σ_{off}^2, where $\sigma_{\text{off}}^2 \ll \sigma_{\text{on}}^2$.

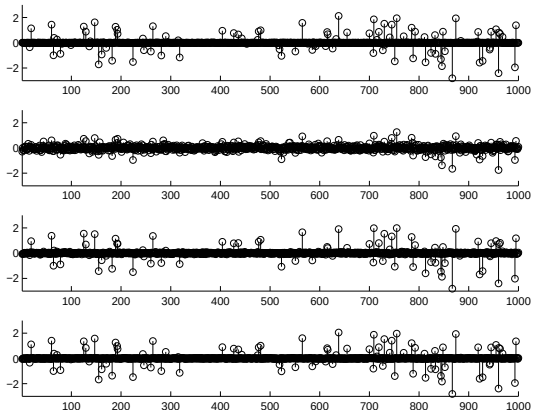

Fig. 2. Evolution of the algorithm toward the solution: $n = 1000$, $m = 400$ and $p = 0.1$. From top to the bottom, first plot corresponds to the actual source, second plot is its estimation at the first level ($\sigma = 1$), third plot is its estimation at the second level ($\sigma = 0.5$), while the last plot is its estimation at third level ($\sigma = 0.2$).

to [1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01]. μ is set equal to 2.5. For each σ the gradient-projection loop (the internal loop) is performed three times (i.e., $L = 3$).

We use the CPU time as a measure of complexity. Although, the CPU time is not an exact measure, it can give us a rough estimation of the complexity, and lets us roughly compare our method with BP. Our simulations were performed in MATLAB7 environment using an AMD Athlon sempron 2400+, 1.67GHz processor with 512MB of memory, and under Microsoft Windows XP operating system.

Table I shows the gradual improvement in the output SNR after each iteration, for a typical run of the algorithm. Moreover, for this run, the total time and final SNR have been shown both for our method and for BP (using interior-point LP solvers). It is seen that our method performs 2 to 3 orders of magnitude faster than BP (based on interior-point LP solvers), while it produces a better SNR. Figure 2 shows the actual source and it's estimations in different iterations for this run of the algorithm.

The experiment was then repeated 100 times (with the same parameters, but for different randomly generated sources and mixing matrix) and the values of SNR's (in dB) obtained over these simulations are averaged. For our algorithm, this averaged SNR was 28.25dB with standard derivation of 2.33dB. For BP, these values were 24.86dB and 1.33dB, respectively. The minimum value of SNR was 18.47dB compared with minimum of 17.73dB for BP, and for 98 out of 100 simulations the SNR was larger than 20dB, compared with 99 for BP.

V. Conclusions

In this article, a fast method for finding sparse solutions of an underdetermined system of linear equations was proposed (to be applied in atomic decomposition and SCA). The method was based on maximizing a 'smooth' measure of sparsity. The overall algorithm was shown to be two to three orders of magnitude faster than the LP interior-point solvers, while providing the same (or better) accuracy. The authors conclude that the sparse decomposition problem is not computationally as *hard* as suggested by the LP approach.

Appendix

Proof of Theorem 1: Let $\mathbf{g}(\mathbf{s}) \triangleq \mathbf{As} - \mathbf{x}$, and consider the method of Lagrange multipliers for maximizing $F_\sigma(\mathbf{s})$ subject to the constraints $\mathbf{g}(\mathbf{s}) = \mathbf{0}$. Setting $\nabla F_\sigma(\mathbf{s}) = \nabla(\boldsymbol{\lambda}^T \mathbf{g}(\mathbf{s}))$, where $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$ is the vector collecting the m Lagrange multipliers, along with the constraints $\mathbf{As} = \mathbf{x}$, results in the nonlinear system of $m+n$ equations and $m+n$ unknowns:

$$\begin{cases} \mathbf{As} = \mathbf{x} \\ \mathbf{A}^T \boldsymbol{\lambda}^* = [s_1 \exp(-s_1^2/2\sigma^2) \ldots s_n \exp(-s_n^2/2\sigma^2)]^T \end{cases} \tag{10}$$

where $\boldsymbol{\lambda}^*$ is an $m \times 1$ unknown vector (proportional to $\boldsymbol{\lambda}$). In general, it is not easy to solve this system of nonlinear equations and for small values of σ, the solution is not unique (because of the existence of local maxima). However, when σ increases to infinity, the system becomes linear and easy to solve:

$$\begin{cases} \mathbf{As} = \mathbf{x} \\ \mathbf{A}^T \boldsymbol{\lambda}^* = \mathbf{s} \end{cases} \Rightarrow \mathbf{A}\mathbf{A}^T \boldsymbol{\lambda}^* = \mathbf{x} \Rightarrow \mathbf{s} = \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{x} \tag{11}$$

which is the minimum ℓ^2-norm or the pseudo-inverse solution of $\mathbf{As} = \mathbf{x}$. ■

References

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

[2] P. Bofi ll and M. Zibulevsky, "Underdetermined blind source separation using sparse representations," *Signal Processing*, vol. 81, pp. 2353–2362, 2001.

[3] R. Gribonval and S. Lesage, "A survey of sparse component analysis for blind source separation: principles, perspectives, and new challenges," in *Proceedings of ESANN'06*, April 2006, pp. 323–330.

[4] D. L. Donoho, M. Elad, and V. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Trans. Info. Theory*, vol. 52, no. 1, pp. 6–18, Jan 2006.

[5] F. Movahedi, G. H. Mohimani, M. Babaie-Zadeh, and C. Jutten, "Estimating the mixing matrix in sparse component analysis (SCA) based on partial k-dimensional subspace clustering," *Neurocomputing*, (sumitted).

[6] Yoshikazu Washizawa and Andrzej Cichocki, "on-line k-plane clustering learning algorithm for sparse comopnent analysis," in *Proceedinds of ICASSP'06*, Toulouse (France), 2006, pp. 681–684.

[7] Y.Q. Li, A. Cichocki, and S. Amari, "Analysis of sparse representation and blind source separation," *Neural Computation*, vol. 16, no. 6, pp. 1193–1234, 2004.

[8] M. Zibulevsky and B. A. Pearlmutter, "Blind source separation by sparse decomposition in a signal dictionary," *Neural Computation*, vol. 13, no. 4, pp. 863–882, 2001.

[9] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.

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

[11] P. G. Georgiev, F. J. Theis, and A. Cichocki, "Blind source separation and sparse component analysis for over-complete mixtures," in *Proceedinds of ICASSP'04*, Montreal (Canada), May 2004, pp. 493–496.

[12] Y. Li, A. Cichocki, and S. Amari, "Sparse component analysis for blind source separation with less sensors than sources," in *ICA2003*, 2003, pp. 89–94.

[13] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Trans. Inform. Theory*, vol. 47, no. 7, pp. 2845–2862, Nov. 2001.

[14] M. Elad and A. Bruckstein, "A generalized uncertainty principle and sparse representations in pairs of bases," *IEEE Trans. Inform. Theory*, vol. 48, no. 9, pp. 2558–2567, Sep. 2002.

[15] S. Krstulovic and R. Gribonval, "MPTK: Matching pursuit made tractable," in *ICASSP'06*, 2006.