Fast Sparse Representation Based on Smoothed ℓ^0 Norm

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

[1] Electrical Engineering Department, Advanced Communications Research Institute (ACRI), Sharif University of Technology, Tehran, Iran
[2] GIPSA-lab, Department of Images and Signals, National Polytechnic Institute of Grenoble (INPG), France
gh1985im@yahoo.com, mbzadeh@yahoo.com, Christian.Jutten@inpg.fr

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 orders of magnitude faster than the state-of-the-art ℓ^1-magic, while providing the same (or better) accuracy.

Keywords: sparse component analysis, over-complete atomic decomposition.

1 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 unknowns/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 $n \times 1$

⋆ This work has been partially funded by Sharif University of Technology, by Center for International Research and Collaboration (ISMO) and by Iran NSF (INSF).

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 from $\mathbf{x}(t)$. 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 [3,2,5,6], and then estimating the sources assuming $\mathbf{A}$ to be known [3,2,7,8]. In this paper we only consider the second step.

To obtain the sparsest solution of $\mathbf{As} = \mathbf{x}$, 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 [4,9,10,3] 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 any small amount of noise completely changes the ℓ^0 norm of a vector). Consequently, the researchers look for other approaches to find sparse solution of $\mathbf{As} = \mathbf{x}$ which are tractable. One of the most successful approaches is Basis Pursuit (BP) [11,1,10,3] which finds the minimum ℓ^1 norm (that is, the solution of $\mathbf{As} = \mathbf{x}$ 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 property that for large systems of equations, the minimum ℓ^1 norm solution is also the minimum ℓ^0 norm solution [1,11,12,13]. By utilizing fast LP algorithms, specifically interior-point LP solvers or ℓ^1-magic [14] (which is about one order of magnitude faster), large-scale problems with thousands of sources and mixtures become tractable. However, although this approach is tractable, it is still very time-consuming. Another approach is Matching Pursuit (MP) [15,16,3] 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 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 3. Finally, Section 4 provides some experimental results of our algorithm and its comparison with BP.

2 The Main Idea

The main idea of this article is to approximate the ℓ^0 norm by a smooth (continuous) function, which lets us to use gradient based methods for its minimization and solves also the sensitivity of ℓ^0 norm to noise. 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 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 accuracy and smoothness of the approximation: the smaller σ, the better approximation, and the larger σ, the smoother approximation.

From (6), 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 $\mathcal{S} = \{\mathbf{s} \mid \mathbf{A}\mathbf{s} = \mathbf{x}\}$.

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 values of σ. However, as value of 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 the value of σ changes slowly, 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 1. Equation (6) proposes that $F_\sigma(\cdot)$ can be seen as a measure of 'sparsity' of a vector (especially for small values of σ): 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.

[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 $\ell^2$ norm
       solution of $\mathbf{As} = \mathbf{x}$ obtained by pseudo-inverse (see the text).
    2. Choose a suitable decreasing sequence for $\sigma$, $[\sigma_1 \dots \sigma_K]$.
– for $k = 1, \dots, K$:
    1. Let $\sigma = \sigma_k$.
    2. Maximize (approximately) the function $F_\sigma$ 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 $\mu$ is a small positive constant).
            (c) Project $\mathbf{s}$ back onto the feasible set $\mathcal{S}$:
                    $\mathbf{s} \leftarrow \mathbf{s} - \mathbf{A}^T(\mathbf{AA}^T)^{-1}(\mathbf{As} - \mathbf{x})$
    3. Set $\mathbf{v}_k = \mathbf{s}$.
– Final answer is $\mathbf{s} = \mathbf{v}_l$.
```

Fig. 1. The final algorithm (SL0 algorithm)

3 The Algorithm

Based on the main idea of the previous section, the final algorithm (smoothed ℓ^0 or SL0) 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, we may escape from getting trapped in the local maxima, and obtain the sparsest solution.

Remark 2. 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 gradual decrease in value of σ, and the fact that for each value, we do not need the exact maximizer of F_σ. All we need, is to enter a region near the (absolute) maximizer of F_σ for escaping from its local maximizers.

Remark 3. 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 values of σ, smaller values of μ_k should be applied. This is because for smaller values of σ, the function F_σ is 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 4. The algorithm may work by initializing $\mathbf{v}_0$ (initial estimation of the sparse solution) to an arbitrary solution of $\mathbf{As} = \mathbf{x}$. However, the best initial

[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.

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. This is formally stated in the following theorem (refer to appendix for the proof).

Theorem 1. The solution of the problem:

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

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

Remark 5. Having initiated the algorithm with the minimum ℓ^2 norm solution (which corresponds to $\sigma = \infty$), the next value for 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:

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

Consequently, if we take, for example, $\sigma > 4\max_i |s_i|$ for all $1 \le i \le n$, 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 values of s_i, $1 \le i \le n$.

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

Remark 6. 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 energy of the noise. This is because, while σ is in this range, (8) shows that the cost function treats small (noisy) samples as zero (i.e., for which $f_\sigma(s_i) \approx 1$, $1 \le i \le n$). However, below this range, the algorithm tries to 'learn' these noisy values, and moves away from the true answer. Restricting σ to be above 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[3].

4 Experimental Results

In this section, we justify performance of the presented approach and compare it with BP. Sparse sources are artificially created using Mixture of Gaussian (MoG) model[4]:

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

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

[4] The model we have used is also called the Bernoulli-Gaussian model.

Table 1. Progress of SL0, Compared to ℓ^1-magic

itr. #	σ	MSE	SNR (dB)
1	1	$3.75\,e-2$	2.88
2	0.5	$2.19\,e-2$	5.21
3	0.2	$4.28\,e-3$	12.29
4	0.1	$1.67\,e-3$	16.37
5	0.05	$6.18\,e-4$	20.71
6	0.02	$1.91\,e-4$	25.80
7	0.01	$1.87\,e-4$	25.89
algorithm	total time	MSE	SNR (dB)
SL0	0.227 seconds	$2.34\,e-4$	25.67
ℓ^1-magic	20.8 seconds	$4.64\,e-4$	21.95

where p denotes 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 in sources, and the larger values of σ_{off} produces stronger noise. In the simulation σ_{on} is fixed to 1. Each column of the mixing matrix is randomly generated using the normal distribution which is then normalized to unity.

The mixtures are generated using the noisy model $\mathbf{x} = \mathbf{As} + \mathbf{n}$, where $\mathbf{n}$ is an additive white Gaussian noise with variance $\sigma_n \mathbf{I}_m$ ($\mathbf{I}_m$ is the $m \times m$ identity matrix). Note that both σ_{off} and σ_n can be used for modeling the noise and they are both set to 0.01 in the simulation[5].

The values used for the experiment are $n = 1000$, $m = 400$, $p = 0.1$, $\sigma_{\text{off}} = 0.01$, $\sigma_{\text{on}} = 1$, $\sigma_n = 0.01$ and the sequence of σ is fixed to [1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01]. μ is set equal to 2.5. For each value of 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 complexity, and lets us roughly compare SL0 (Smoothed ℓ^0 norm) with BP[6].

Table 1 shows the gradual improvement in the output SNR after each iteration, for a typical run of SL0. Moreover, for this run, total time and final SNR have been shown for SL0 and for BP (using ℓ^1-magic). Figure 2 shows the actual source and it's estimations in different iterations for this run of SL0. The experiment is then repeated 100 times (with the same parameters, but for different randomly generated sources and mixing matrices) and the values of SNR (in dB) obtained over these simulations are averaged. For SL0, this averaged SNR was 25.67dB with standard derivation of 1.34dB. For ℓ^1 magic,

[5] Note that, although in the theoretical model only the noiseless case was addressed, because of continuity of the cost functions, the method can work as well in noisy conditions.

[6] Our simulations are performed in MATLAB7 under WinXP using an AMD Athlon sempron 2400+, 1.67GHz processor with 512MB of memory.

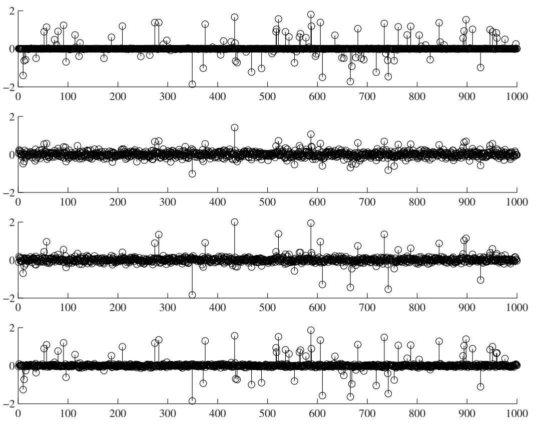

Fig. 2. Evolution of SL0 toward the solution: From top to bottom, first plot corresponds to the actual source, second plot is its estimation at the first level ($\sigma = 1$), third plot is its estimation at the second level ($\sigma = 0.5$), while the last plot is its estimation at third level ($\sigma = 0.2$)

these values were 21.92dB and 1.36dB, respectively. The minimum value of SNR was 20.12dB compared with minimum of 18.51dB for BP.

5 Conclusions

In this article, a fast method for finding sparse solutions of an under-determined system of linear equations was proposed (to be applied in atomic decomposition and SCA). SL0 was based on maximizing a 'smooth' measure of sparsity. SL0 shows to be about two orders of magnitude faster than the ℓ^1-magic, while providing the same (or better) accuracy. The authors conclude that sparse decomposition problem is not computationally as *hard* as suggested by the LP approach.

References

1. Donoho, D.L.: For most large underdetermined systems of linear equations the minimal l^1-norm solution is also the sparsest solution, Tech. Rep. (2004)
2. Bofill, P., Zibulevsky, M.: Underdetermined blind source separation using sparse representations. Signal Processing 81, 2353–2362 (2001)
3. Gribonval, R., Lesage, S.: 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 (2006)
4. Donoho, D.L., Elad, M., Temlyakov, V.: Stable recovery of sparse overcomplete representations in the presence of noise. IEEE Trans. Info. Theory 52(1), 6–18 (2006)
5. Movahedi, F., Mohimani, G.H., Babaie-Zadeh, M., Jutten, C.: Estimating the mixing matrix in sparse component analysis (SCA) based on partial k-dimensional subspace clustering, Neurocomputing (sumitted)

6. Washizawa, Y., Cichocki, A.: on-line k-plane clustering learning algorithm for sparse comopnent analysis. In: Proceedinds of ICASSP'06, Toulouse (France), pp. 681–684 (2006)
7. Li, Y.Q., Cichocki, A., Amari, S.: Analysis of sparse representation and blind source separation. Neural Computation 16(6), 1193–1234 (2004)
8. Zibulevsky, M., Pearlmutter, B.A.: Blind source separation by sparse decomposition in a signal dictionary. Neural Computation 13(4), 863–882 (2001)
9. Georgiev, P.G., Theis, F.J., Cichocki, A.: Blind source separation and sparse component analysis for over-complete mixtures. In: Proceedings of ICASSP'04, Montreal (Canada), May 2004, pp. 493–496 (2004)
10. Li, Y., Cichocki, A., Amari, S.: Sparse component analysis for blind source separation with less sensors than sources. In: ICA2003, pp. 89–94 (2003)
11. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. SIAM Journal on Scientific Computing 20(1), 33–61 (1999)
12. Donoho, D.L., Huo, X.: Uncertainty principles and ideal atomic decomposition. IEEE Trans. Inform. Theory 47(7), 2845–2862 (2001)
13. Elad, M., Bruckstein, A.: A generalized uncertainty principle and sparse representations in pairs of bases. IEEE Trans. Inform. Theory 48(9), 2558–2567 (2002)
14. Candes, E., Romberg, J.: ℓ_1-Magic: Recovery of Sparse Signals via Convex Programming, URL: `www.acm.caltech.edu/l1magic/downloads/l1magic.pdf`
15. Mallat, S., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Trans. on Signal Proc. 41(12), 3397–3415 (1993)
16. Krstulovic, S., Gribonval, R.: MPTK: Matching pursuit made tractable. In: ICASSP'06 (2006)

Appendix: Proof of Theorem 1

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

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

where $\widehat{\boldsymbol{\lambda}}$ 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 existence of local maxima). However, when σ increases to infinity, the system becomes linear and easy to solve:

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

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