Proceedings of the 2007 IEEE International Conference on Telecommunications and
Malaysia International Conference on Communications, 14-17 May 2007, Penang, Malaysia

Estimating the mixing matrix in Sparse Component Analysis (SCA) based on multidimensional subspace clustering

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

***Abstract*— In this paper we propose a new method for estimating the mixing matrix, A, in the linear model X = AS, for the problem of underdetermined Sparse Component Analysis (SCA). Contrary to most existing algorithms, in the proposed algorithm there may be more than one active source at each instant (i.e. in each column of the source matrix S), and the number of sources is not required to be known in advance. Since in the cases where more than one source is active at each instant, data samples concentrate around multidimensional subspaces, the idea of our method is to first estimate these subspaces and then estimate the mixing matrix from these estimated subspaces.**

I. Introduction

Because of its many applications, the problem of Blind Source Separation (BSS) has been extensively studied in the last twenty years (refer for example to the books [1], [2]. The applications include channel estimation and equalization [3], multimedia signal processing [4] and HDTV system [5]). In the mathematical form, the problem consists in separating a set of mixed signals from their mixtures, where there are no information about the sources or about the mixing system (hence the term "Blind").

The aim of Sparse Component Analysis (SCA) is to solve the BSS problem under the sparsity prior [6], [7], [8], [9], [10]. A sparse signal is a signal whose most samples are nearly zero, and just a few percent take significant values. Consequently, at each 'time' instant, only a few number of sources have significant values (say they are 'active'), and most of them are almost zero (say they are 'inactive').

The problem of SCA can be stated as follows. Consider the linear model:

$$\mathbf{X} = \mathbf{AS} \tag{1}$$

where $\mathbf{A} = [\mathbf{a}_1 \dots \mathbf{a}_n] \in \mathbb{R}^{m\times n}$ is the mixing matrix, $\mathbf{S} = [\mathbf{s}_1 \dots \mathbf{s}_T] \in \mathbb{R}^{n\times T}$ and $\mathbf{X} = [\mathbf{x}_1 \dots \mathbf{x}_T] \in \mathbb{R}^{m\times T}$ are the matrices of n sources and m observed signals. Each column of $\mathbf{S}$ and $\mathbf{X}$ corresponds to an instant of 'time' and T is the number of 'time' samples. Sparsity of source signals implies that in each column of $\mathbf{S}$, there are just a few significant values (active sources) and most of the elements are almost zero (inactive sources). The goal of SCA is then to estimate $\mathbf{A}$ and $\mathbf{S}$, only from $\mathbf{X}$ and the sparsity assumption. In this paper, each column of the mixing matrix, i.e. each $\mathbf{a}_i$, $1 \le i \le n$, is called a *mixing vector*.

Although the word "time" is used in the above paragraphs ('time' samples or instant of 'time'), and will be used in the continuation of this paper, the above model may be in another domain, in which the sparsity assumption holds. To see this, let $\mathcal{T}$ be a linear 'sparsifying' transform (like Short Time Fourier Transform (STFT) or wavelet packet transform for speech signals), and the mixing system is stated as $\mathbf{X} = \mathbf{AS}$ in the time domain. Then, we have $\mathcal{T}\{\mathbf{X}\} = \mathbf{A}\,\mathcal{T}\{\mathbf{S}\}$ in the transformed domain, and because of the sparsity of $\mathcal{T}\{\mathbf{S}\}$, it is in the form of (1).

Generally, more than one source may be active at each instant of time. The number of active sources at each instant is a random variable and its average[1] is denoted by k.

The SCA problem is usually solved in two steps. The first step is the estimation of the mixing matrix ($\mathbf{A}$), and the second step is the recovery of the source signals ($\mathbf{S}$) by knowing the mixing matrix. Note that in the underdetermined case, in which the number of sources exceeds the number of sensors, these two problems are not identical [11]. In this paper, we address only the problem of the estimation of the mixing matrix.

In the field of SCA, two different cases should be distinguished for estimating the mixing matrix: single dominant component and multiple dominant components. In the former, the average number of active sources is less than or approximately equal to one. In the latter, the average number of active sources is greater than one. Up to now, many papers have been addressed to the former case [6], [8], [9], while only few researchers have considered the latter case [12], [13]. In this paper, we focus on the case of multiple dominant components.

In the single dominant component SCA, the observed data in the m-dimensional scatter plot of mixtures concentrate along the directions of n mixing vectors. Similarly, in the multiple

[1]Advance Communications research Institute (ACRI), 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 Sharif University of Technology, by French Embassy in Tehran, and by Center for International Research and Collaboration (ISMO).

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

[1]In fact, this random variable may have non integer average. In this case, k is the closest integer to this average.

dominant components SCA, the observed data concentrate around k-dimensional subspaces which are spanned by a set of k mixing vectors. The total number of these subspaces is equal to $N_p = \binom{n}{k}$. We call these subspaces *concentration subspaces* throughout this paper.

The basic idea of this paper is to find these k-dimensional concentration subspaces, and then to estimate the mixing vectors using them. This general idea has also been used by Washizawa *et. al.* in [13], but our method archives this goal by another technique which has lower computational cost and lets us to solve the medium scale problems (e.g., $n = 12$ and $m = 6$). In fact, in our method, it is not necessary to find all N_p concentration subspaces. Moreover, if some of these subspaces are found mistakenly, the estimating part of the mixing matrix is in a way robust to these errors.

It should be emphasized that in this paper, k, the average number of active sources, is assumed to be determined a priori. However, a method for estimating k has been proposed in [14].

The paper is organized as follows. In the following section, we will explain the procedure of estimating the concentration subspaces. In Section III, a method for estimating the mixing vectors from the estimated concentration subspaces is developed. In Section IV, the algorithm for estimating the matrix $\mathbf{A}$ is finalized, while Section V presents various computer simulations to justify the algorithm. Finally, Section VI contains some discussions and concludes the paper.

II. Estimating Concentration Subspaces

In this section, we try to estimate k-dimensional concentration subspaces. Each k-dimensional subspace can be represented by an m by k matrix, whose columns form an orthonormal basis for the subspace[2]. In this paper, we do not distinguish between a subspace and its matrix representation.

Let $\mathbf{B} \in \mathbb{R}^{m\times k}$ be the matrix representation of an arbitrary k-dimensional subspace. We define the following function to detect whether $\mathbf{B}$ is a concentration subspace or not:

$$f_\sigma(\mathbf{B}) = \sum_{i=1}^{T} \exp\left(\frac{-d^2(\mathbf{x}_i, \mathbf{B})}{2\sigma^2}\right) \tag{2}$$

where $d(\mathbf{x}_i, \mathbf{B})$ is the distance of $\mathbf{x}_i$ from the subspace represented by $\mathbf{B}$ (the definition of this distance is presented in appendix 1).

For small values of $d(\mathbf{x}_i, \mathbf{B})$ compared to σ, $\exp(-d^2(\mathbf{x}_i, \mathbf{B})/2\sigma^2)$ is about 1 and for large values of $d(\mathbf{x}_i, \mathbf{B})$, it is nearly zero. Therefore, for sufficiently small values of σ, the above function *is approximately equal to the number of data points close to* $\mathbf{B}$. Moreover, if the set of points are concentrated around several different k-dimensional concentration subspaces, f has a local maximum where $\mathbf{B}$ is close to the basis of each of them. These local maxima are very strong if σ is small enough. In fact, their values are approximately equal to the number of data samples which lie in that subspace. Therefore, by maximizing the function f, we actually maximize the number of data points close to $\mathbf{B}$. Now an example is presented for demonstrating this function.

[2]Note that this representation is not unique.

Example 1. Consider the problem for the case $n = 5$, $m = 3$ and $k = 2$. In this case, there are five sources, three mixtures and two sources are active in most instants of time. Therefore, the data concentrate around $\binom{5}{2} = 10$ two-dimensional concentration subspaces. To show the efficiency of the function f, the data points near origin are first omitted (they lie in all concentration subspaces) and the remaining points are projected onto the surface of a unit semi-sphere (by normalizing the data in every sample and forcing sign of the first component to be positive. This operation does not change the subspace of a data sample).

Given an arbitrary two-dimensional subspace, let $\mathbf{B}$ be its matrix representation and (x, y, z) be its normal vector[3] representation in Cartesian coordinate. This vector can be represented by (φ, θ) satisfying:

$$\begin{cases} x = \sin(\varphi)\sin(\theta) \\ y = \cos(\varphi)\sin(\theta) \\ z = \cos(\theta) \end{cases} \tag{3}$$

The function f is plotted versus φ, θ for $0 \le \varphi, \theta \le \pi$ and different values of σ. The results are shown in Fig. 1(a) and Fig. 1(b).

Note that for larger σ, f is smoother, but for smaller one, the peak locations are better distinguishable (all 10 concentration subspaces are separated). Therefore, they form a better representation for the actual concentration subspaces.

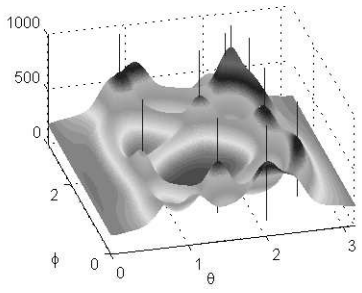
The idea is to maximize the function f *for a small value of* σ, using a maximization method. However, for small σs, many local maxima exist which make this maximization difficult. But even in this case, if we have a good initial guess about the location of the maximum, then by starting from this initial point, the maximization algorithm may easily find the actual maximum. Our idea is then to use the maximum obtained from the maximization of f for a larger σ, as the initial guess for the location of the maximum for smaller σ. This suggests to use a decreasing sequence of σ in order to obtain an accurate estimation.

Up to now, estimating the concentration subspaces is discussed. The presentation of the final algorithm is delayed to Section IV, after introducing the method of estimating the mixing vectors from concentration subspaces in the next section.

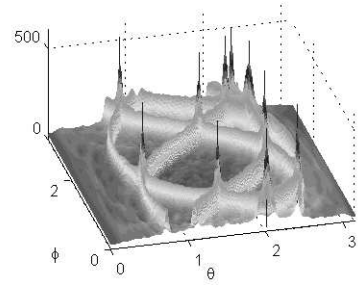
III. Estimating Mixing Vectors

Now suppose that all of the k-dimensional concentration subspaces are estimated and their representation matrices are $\mathbf{B}_i$, $i = 1 \dots N_p$. This section is dedicated to estimating the mixing vectors using these subspaces. To do this, we use an

[3]In this case, $\mathbf{B}$ is a hyperplane in the three-dimensional space. Its normal vector, by definition, is a unit-norm vector which is perpendicular to the hyperplane.



(a)



(b)

Fig. 1. Graph of the function f in the case $n = 5$, $m = 3$ and $k = 2$ for two different values of σ. In (a) $\sigma = 0.1$ and in (b) $\sigma = 0.02$. The exact location of the actual concentration subspaces are indicated by vertical lines. It is observed that by decreasing σ, discrimination increases while a decrease in smoothness is observed.

idea similar to the idea we used in the previous section to find the concentration subspaces.

As mentioned before, every concentration subspace is spanned by a set of k mixing vectors. The number of concentration subspaces which include a certain mixing vector, equals to the number of choices of other $k-1$ mixing vectors from the total $n-1$ ones. Therefore, every mixing vector lies in $\binom{n-1}{k-1}$ of the subspaces. Given an arbitrary vector $\mathbf{v}$ in the m-dimensional space, we define the following function:

$$g_\sigma(\mathbf{v}) = \sum_{i=1}^{N_p} \exp\left(\frac{-d^2(\mathbf{v}, \mathbf{B}_i)}{2\sigma^2}\right) \tag{4}$$

where $d(\mathbf{v}, \mathbf{B}_i)$ is the distance of the vector $\mathbf{v}$ from the ith estimated concentration subspace (refer to appendix 1 for the definition of this distance)[4].

For small values of $d(\mathbf{v}, \mathbf{B}_i)$ compared to σ, $\exp\left(-d^2(\mathbf{v}, \mathbf{B}_i)/2\sigma^2\right)$ is about 1 and for large values of $d(\mathbf{v}, \mathbf{B}_i)$, it is almost zero. Thus, for sufficiently small values of σ, the function g *is approximately equal to the number of subspaces close to* $\mathbf{v}$. Note that the σ used in this formula is

[4]However, as will be explained later, this function is not directly used in the final algorithm.

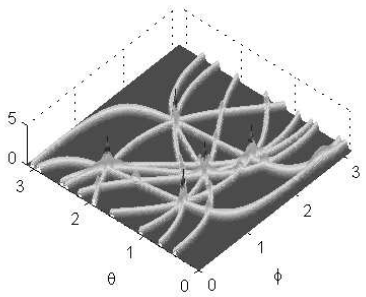


Fig. 2. Graph of the function g in the case $n = 5$, $m = 3$ and $k = 2$ for $\sigma = 0.02$. Distinguished curves represent 10 two-dimensional concentration subspaces and discriminated peaks represent 5 mixing vectors. The exact location of the actual mixing vectors are indicated by added vertical lines. Note that each mixing vector is the intersection of 4 concentration subspaces.

different from the previous one. In order to distinguish the two, the one related to finding subspaces is denoted by $\sigma_{\mathbf{B}}$ and the one related to finding mixing vectors is denoted by $\sigma_{\mathbf{A}}$. The next example explains the behavior of this function.

Example 2. Here, we demonstrate the function g for the case of example 1, in which $n = 5$, $m = 3$ and $k = 2$. There are $\binom{5}{2} = 10$ concentration subspaces, all of them are presumed to be already estimated. Each mixing vector is close to $\binom{5-1}{2-1} = 4$ of these estimated subspaces.

All of the surface points of the unit semi-sphere are spanned as follows. Each vector with Cartesian coordinate (x, y, z) is transformed to (φ, θ) satisfying (3). The function g is shown versus φ, θ for $0 \leq \varphi, \theta \leq \pi$ in Fig. 2. According to (3), a two-dimensional subspace defined by equation $z = ax + by$ can be represented in this system of coordinates by equation:

$$\cos(\theta) = a\sin(\varphi)\sin(\theta) + b\cos(\varphi)\sin(\theta).$$

Therefore, any arbitrary two-dimensional subspace is transformed into a curve in this figure. In fact, there are exactly 10 curves in the figure which each one represents one of the concentration subspaces.

In this case every mixing vector lies in 4 subspaces. Therefore, the absolute maxima that are located in the intersection of 4 curves represent the mixing vectors and are easily distinguishable in the figure. Note that the values of these peaks are nearly 4, which is equal to the number of concentration subspaces in which they lie.

As observed in the figure, there exist local maxima which make it difficult to design a maximization algorithm for estimating the mixing vectors. In general, each mixing vector lies in $\binom{n-1}{k-1}$ concentration subspaces, but other vectors are close to relatively smaller number of concentration subspaces, say less than q concentration subspaces. Consequently, in order to identify the mixing vectors correctly, we detect the vectors which lie in at least q concentration subspaces.

Note that if q is set to $\binom{n-1}{k-1}$ then all of the concentration subspaces must be estimated accurately. However, the value of

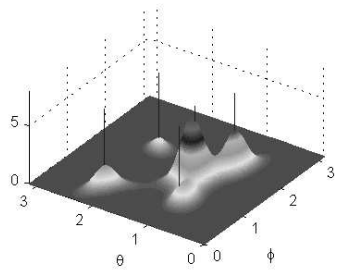


Fig. 3. Graph of the function h in the case $n = 5$, $m = 3$, $k = 2$ and $q = 4$ for $\sigma = 0.15$. The exact location of the actual mixing vectors are indicated by added vertical lines.

q can be set much less in some experiences, and the method accomplishes correctly without requiring all the concentration subspaces to be estimated.

Moreover, in determining the mixing vectors, instead of working with the function g (as defined in (4)), we define the following function to better discriminate between the peaks corresponding to mixing vectors, and to force each detected mixing vector to lie in at least q concentration subspaces:

$$h_\sigma(\mathbf{v}) = \sum_{1 \le i_1 < \cdots < i_q \le N_p} u_\sigma(\mathbf{v}, \mathbf{B}_{i_1}) \cdots u_\sigma(\mathbf{v}, \mathbf{B}_{i_q}) \qquad (5)$$

where

$$u_\sigma(\mathbf{v}, \mathbf{B}) = \exp\left(-d^2(\mathbf{v}, \mathbf{B})/2\sigma^2\right)$$

and $d(\mathbf{v}, \mathbf{B})$ is the distance of vector $\mathbf{v}$ from subspace $\mathbf{B}$. For sufficiently small values of σ, if $\mathbf{v}$ is close to $\mathbf{B}$ then $u_\sigma(\mathbf{v}, \mathbf{B})$ is about 1 and otherwise it is almost zero. Thus,

$$u_\sigma(\mathbf{v}, \mathbf{B}_{i_1}) \cdots u_\sigma(\mathbf{v}, \mathbf{B}_{i_q}) \approx \begin{cases} 1 & \text{if } \mathbf{v} \text{ is close to all } \mathbf{B}_{i_1} \cdots \mathbf{B}_{i_q} \\ 0 & \text{otherwise} \end{cases}$$

This means that $h_\sigma(\mathbf{v})$ is significant if at least one of the summands is significant, i.e. if $\mathbf{v}$ is near to the corresponding subset of q subspaces. Therefore, this function can be utilized for finding mixing vectors. Note that direct computation of (5) is too time-consuming, and should be avoided. Instead, a fast algorithm for computing (5) is presented in appendix 2. The function h is shown for the same case of example 2, with $q = 4$ in Fig. 3.

As it can be observed in the figures, the mixing vectors are more distinguishable and the maximization process is simpler for detecting these maxima. In fact, similar to the function f_σ, value of σ experiences a trade-off between smoothness and discrimination.

IV. Final Algorithm of Identifying the Mixing Matrix

As mentioned in previous sections, two decreasing sequences of σ are used in this algorithm. We denote them by $[\sigma_1 \dots \sigma_R]$. However, their lengths and their values can be different.

Usually, the estimation of all the $N_p = \binom{n}{k}$ concentration subspaces is not necessary. It is sufficient to estimate as many concentration subspaces to guarantee the existence of any mixing vectors in at least q of the estimated concentration subspaces. This value is referred to as $N_{\mathbf{B}}$.

1) Remove data samples (samples of the mixture matrix $\mathbf{x}(t)$ $1 \le t \le T$) which are near origin. In these samples, all of the sources are probably inactive. Then normalize every column of $\mathbf{X}$ (normalization simplifies the distance measurement).
2) Choose a suitable value for $N_{\mathbf{B}}$ and a suitable decreasing sequence of $[\sigma_1 \dots \sigma_R]$.
3) For $j = 1 \dots \mathrm{L}_{\mathbf{B}}$
   a) Choose a random starting subspace (an orthonormal m by k matrix $\mathbf{B}_j$).
   b) Set $i = 1$.
   c) Start with $\mathbf{B}_j$ and maximize the function f_{σ_i} using a multi-variate maximization method. Orthonormalize $\mathbf{B}_j$ after each iteration. Update $\mathbf{B}_j$ to the argument that maximizes this function.
   d) If $i < R$ (where R is the number of elements of the sequence $[\sigma_1 \dots \sigma_R]$), increment i and go back to (c).
4) Omit the repeated subspaces.
5) Choose $N_{\mathbf{B}}$ of the obtained subspaces that have the largest value of the function f_{σ_R}.

Fig. 4. The final algorithm for estimating the concentration subspaces.

The idea for estimating the concentration subspaces is to start from randomly different starting points, with the hope of finding different maxima. To achieve this, instead of trying $N_{\mathbf{B}}$ starting points, we use $L_{\mathbf{B}}$ starting points (where $L_{\mathbf{B}}$ is several times greater than $N_{\mathbf{B}}$), and then we take $N_{\mathbf{B}}$ of them which have the greatest f_{σ_R} (note that we are taking advantage that the actual number of concentration subspaces $N_p = \binom{n}{k}$ is large). Using this method, if some of the detected subspaces are false (because of getting trapped in local maxima), they will be ignored, too. The final algorithm for estimating the concentration subspaces is presented in Fig. 4.

Similarly in order to improve the performance of the second part of the algorithm, instead of estimating n mixing vectors, $L_{\mathbf{A}} \gg n$ vectors are estimated, after which the repeated vectors are omitted and actual mixing vectors are extracted via error detection process. The final algorithm for estimating the concentration subspaces is presented in Fig. 5. We do not emphasize on the maximization method. In our simulations, we have used the steepest ascent maximization method which is discussed in appendix 3.

Note that the parameter n (number of sources) is not directly used in the above algorithm. If n is not known, the above algorithm can be equally applied. As will be seen in the experimental results, obtaining more than n (actual number of) mixing vectors is rare, but obtaining less than n vectors happens more frequently, specially where the actual mixing vectors are very close to each other.

V. Experimental Results

In this section, one simulation is presented to show the performance of the algorithm. For this simulation, sparse sources are generated independently and identically distributed (i.i.d)

1) Choose a suitable decreasing sequence of $[\sigma_1 \dots \sigma_R]$.
2) For $j = 1 \dots \mathrm{L}_{\mathbf{A}}$
   a) Choose a random normalized m by 1 vector $\mathbf{v}_j$.
   b) Set $i = 1$.
   c) Start with $\mathbf{v}_j$ and maximize the function h_{σ_i} using any multi-variate maximization method. Normalize $\mathbf{v}_j$ after each iteration. Update $\mathbf{v}_j$ to the argument that maximizes this function.
   d) If $i < R$, increment i and go back to (c).
3) Error detection process: Omit the vectors that are near to less than q of the estimated subspaces.
4) Omit the repeated vectors.

Fig. 5. The final algorithm for estimating the mixing vectors.

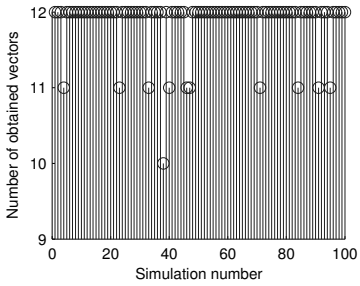


Fig. 6. Efficiency of the overall algorithm for all simulations in the case $n = 12$, $m = 6$, $k = 2$ and $T = 3900$ for 100 different simulations. Number of obtained vectors in each simulation is shown. In some cases fewer than 12 vectors are obtained(7).

by the sum of Gaussians model [11]:

$$s_i \sim \; p\,\mathcal{N}(0, \sigma_{\mathrm{on}}) + (1-p)\,\mathcal{N}(0, \sigma_{\mathrm{off}}) \tag{6}$$

where p is the probability of activity of the sources (and hence $k \approx np$). σ_{on} and σ_{off} are the standard deviations of the sources in active and inactive modes, respectively. In order to have sparse sources, the conditions $\sigma_{\mathrm{on}} \gg \sigma_{\mathrm{off}}$ and $p \ll 1$ should be applied. σ_{off} is to model a white noise.

100 simulations are performed for the case $n = 12$, $m = 6$, $k = 2$ ($p = 0.167$), $T = 3900$, $\sigma_{\mathrm{off}} = 0.01$ and $\sigma_{\mathrm{on}} = 1$. All the mixing matrices are generated randomly and each column of them is normalized. Parameters were chosen as follows: $q = 4$, $N_{\mathbf{B}} = 58$, $\mathrm{L}_{\mathbf{A}} = 240$, $\mathrm{L}_{\mathbf{B}} = 10N_{\mathbf{B}} = 580$, $\sigma_{\mathbf{B}} = [.15, .075, .037, .018]$ and $\sigma_{\mathbf{A}} = [.1, .05, .025, .0125]$.

In the use of the algorithm of Fig. 4, two subspaces are detected identical if their distance (presented in appendix 1) is less than 0.1. In the use of the algorithm of Fig. 5, a vector is considered to lie in a subspace if its distance (presented in appendix 1) from that subspace is less than 0.03. And two vectors are called identical if their angle is less than 6 degrees. This criterion forces any two mixing vectors to have a minimum angle of 6 degrees. In this simulation, all the mixing matrices which did not obey this criteria were omitted.

Figure 6 shows the number of vectors obtained by the algorithm in all simulations. Note that in none of these simulations more than 12 vectors are obtained. However, in 11 of them less than 12 vectors are estimated. In all cases, the obtained vectors are compared with the mixing vectors. For the cases in which 12 vectors were obtained, the criterion

$$\mathcal{E} = \min_{\mathbf{P}\in\mathcal{P}} \|\mathbf{A} - \hat{\mathbf{A}}\mathbf{P}\|_2 \tag{7}$$

is used, where $\mathcal{P}$ is the set of all permutation matrices (this is the same criterion used in [13]). The maximum, the minimum, the mean and the standard deviation of the error obtained by this criterion were 0.0111, 0.0054, 0.0083 and 0.0012, respectively[5]. This shows that obtained vectors were very close to the actual ones.

[5]In the cases where $n_0 < n$ vectors are obtained, the formula

$$\mathcal{E} = \min_{\mathbf{P}\in\mathcal{X}} \|\mathbf{A}\mathbf{P} - \hat{\mathbf{A}}\|_2 \tag{8}$$

is used where $\mathcal{X}$ is the set of all n_0 by n full-rank matrices in which all elements are zero except an element equal to one in each row.

Generally, the method may not succeed in estimating all of the mixing vectors. This error occurs if that mixing vector is not close to at least q of the estimated subspaces. This error may be generated because of error in subspace estimation process. In the 11 cases where less than 12 vectors were estimated, this error has occurred. The negligible errors obtained in the experiment imply that the estimation of mixing vectors has been accurate. The strict error detection process in Fig. 5, usually prevents detection of any false vector.

VI. Discussion and Conclusion

Most existing algorithms assume single dominant case at each instant for estimating the mixing matrix in SCA. Moreover, the number of sources is assumed to be known in most of them. In this paper, we presented a method which omits these restrictions. On the contrary, in our simulations, we assumed that the averaged number of active sources, k, is known in advance. However, some methods exist that estimate k [14].

At our best knowledge, all existing SCA methods are unable to estimate mixing matrix in large and even medium scales (e.g., the case used in the simulations), for the multiple dominant case [13]. However, our method solves the problem at least in medium scale cases. The reason is that there is no necessity to estimate all concentration subspaces. Moreover, the algorithm is in a way robust to the errors in estimation of these concentration subspaces.

Unfortunately, just like other SCA methods, our method suffers from exponential growth in computation cost. The reason is that in order to estimate the concentration subspaces, the number of data samples should be proportional to N_p. Therefore, it is burdensome to solve the problem in the large scales. The large scale case still remains an open problem.

As mentioned before, the presented method uses a decreasing sequence of σ for estimating the concentration subspaces. The first and largest σ in the sequence is an essential factor in the quality. A suitable starting σ depends on many factors such as n, m and k. If chosen too large, it may cause mixed peaks and if too small many local maxima exist.

The performance of our method depends on several factors, such as the condition number of mixing matrix, the number of samples, the number of observations, the average number of active sources, the sequences of $\sigma_{\mathbf{A}}$ and $\sigma_{\mathbf{B}}$. A proper estimation of the these sequences is an essential factor in the performance. An optimum choice of the parameters is an open problem which is currently being studied in our group.

Appendix 1

In the algorithm it is required to calculate the distance between a subspace and a vector or two subspaces. Let $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_k]$ be the matrix representation of a k-dimensional subspace of the m-dimensional space, that is, $\{\mathbf{b}_1, \dots, \mathbf{b}_k\}$ is an orthonormal basis for this subspace. Let $\mathbf{v}$ be a unit-norm m-dimensional vector. Then, as a measure of the distance between subspace $\mathbf{B}$ and vector $\mathbf{v}$, we use:

$$d(\mathbf{v}, \mathbf{B}) = \sqrt{1 - [(\mathbf{v} \cdot \mathbf{b}_1)^2 + \dots + (\mathbf{v} \cdot \mathbf{b}_k)^2]} \tag{9}$$

where $\mathbf{v} \cdot \mathbf{b}_i$ represents the dot product of $\mathbf{b}_i$ and $\mathbf{v}$.

Now let $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_k]$ and $\hat{\mathbf{B}} = [\hat{\mathbf{b}}_1 \dots \hat{\mathbf{b}}_k]$ be two k-dimensional subspaces of m-dimensional space represented in orthonormal form. Then, to detect if these two subspaces are identical (step 4 of the algorithm of Fig. 4), we use (refereed as the 'distance' of the two subspaces within the paper):

$$d(\mathbf{B}, \hat{\mathbf{B}}) = \sqrt{d^2(\mathbf{b}_1, \hat{\mathbf{B}}) + \dots + d^2(\mathbf{b}_k, \hat{\mathbf{B}})}$$

where $d(\mathbf{b}_i, \hat{\mathbf{B}})$ is the distance between vector and subspace, stated above.

Appendix 2

In (5) it is necessary to calculate an expression of the form

$$\sum_{1 \le i_1 < \dots < i_q \le N} a_{i_1} \dots a_{i_q}$$

where $N = N_p$ and $a_j = u_\sigma(\mathbf{v}, \mathbf{B}_j), 1 \le j \le N_p$. If directly computed, it requires a cost computation of order $\binom{N}{q}$. However, by implementing a recursive algorithm, cost computation can be decreased to the order of Nq.

By defining:

$$sum_q(a_1 \dots a_N) = \sum_{1 \le i_1 < \dots < i_q \le N} a_{i_1} \dots a_{i_q}$$

We have

$$sum_q(a_1 \dots a_N) = sum_q(a_1 \dots a_{N-1}) + a_N \cdot sum_{q-1}(a_1 \dots a_{N-1})$$

Using this formula the recursive algorithm can be designed.

Appendix 3

As mentioned in Section 4, this method is based on the maximization of two functions, f_σ and h_σ. This appendix is dedicated to development of the steepest ascent methods for their maximization. Note that results of the previous appendices are used in developing this section.

The following equation can be utilized to compute gradient in each iteration of the steepest ascent maximization for function f_σ:

$$\frac{\partial f_\sigma}{\partial \mathbf{b}_j} = \frac{1}{\sigma^2} \sum_{t=1}^{T} \mathbf{x}_t (\mathbf{x}_t \cdot \mathbf{b}_j) \exp(-d^2(\mathbf{x}_t, \mathbf{B})/2\sigma^2) \qquad 1 \le j \le k \tag{10}$$

where $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_k]$. Each iteration of this algorithm composed of:

- Set $\mathbf{b}_j \leftarrow \mathbf{b}_j + \mu(\partial f_\sigma / \partial \mathbf{b}_j)/T$ for $1 \le j \le k$ using (10).
- Orthonormalize $\mathbf{B}$[6].

In our simulation, we have chosen step size of the algorithm (μ) proportional to σ^2, to have smaller step-sizes for more complicated functions (which is the case for smaller σs).

For second part of the algorithm, the following equations can be utilized to compute gradient in each iteration of the steepest ascent maximization for function h_σ:

$$\frac{\partial h_\sigma}{\partial \mathbf{v}} = \sum_{l=1}^{N} \frac{\partial}{\partial \mathbf{v}} \{\exp(-d^2(\mathbf{v}, \mathbf{B}_{i_l})/2\sigma^2)\} \left(\sum_{\substack{1 \le i_1 < \dots < i_{q-1} \le N_p \\ l \notin \{i_1 \dots i_{q-1}\}}} \prod_{j=1}^{q-1} \exp(-d^2(\mathbf{v}, \mathbf{B}_{i_j})/2\sigma^2) \right) \tag{11}$$

[6]Orthonormalizing the matrix $\mathbf{B}$ means projecting it on the sphere $\mathbf{B}^T\mathbf{B} = \mathbf{I}$. In fact the matrix $\mathbf{B}^T\mathbf{B}$ is symmetric and it's square root can be easily computed. To do the projection it is sufficient to right multiply the matrix $\mathbf{B}$ by the inverse of this square root.

where

$$\frac{\partial}{\partial \mathbf{v}} \{\exp(-d^2(\mathbf{v}, \mathbf{B})/2\sigma^2)\} = \frac{1}{\sigma^2} \left(\sum_{i=1}^{k} \mathbf{b}_i (\mathbf{b}_i \cdot \mathbf{v}) \right) \exp(-d^2(\mathbf{v}, \mathbf{B})/2\sigma^2) \tag{12}$$

To calculate (11), the method introduced in appendix 2 can be applied. In fact:

$$\sum_{\substack{1 \le i_1 < \dots < i_{q-1} \le N_p \\ l \notin \{i_1 \dots i_{q-1}\}}} \prod_{j=1}^{q-1} \exp(-d^2(\mathbf{v}, \mathbf{B}_{i_j})/2\sigma^2) = sum_{q-1}\{\exp(-d^2(\mathbf{v}, \mathbf{B}_i)/2\sigma^2) | 1 \le i \le N, i \ne l\} \tag{13}$$

Similar to the previous maximization, a suitable step μ proportional to σ^2 has been chosen and the following steps should be performed in each iteration.

- Set $\mathbf{v} \leftarrow \mathbf{v} + \mu(\partial h_\sigma / \partial \mathbf{v}) / \|\partial h_\sigma / \partial \mathbf{v}\|_2$ using (11) and (12).
- Normalize $\mathbf{v}$ by setting $\mathbf{v} \leftarrow \mathbf{v}/\|\mathbf{v}\|_2$.

In the experimental results, in the first maximization μ is set to $10^4\sigma^2$ and in the second, μ is set to $100\sigma^2$.

REFERENCES

[1] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, John Wiley & Sons, 2001.

[2] A. Cichocki and S. Amari, *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications*, John Wiley and sons, 2002.

[3] J. Rinas and K.D. Kammeyer, "Mimo measurements of communication signals and application of blind source separation," in *Proceedings of the 3rd IEEE International Symposium on Signal Processing and Information Technology (ISSPIT 2003)*, pp. 94–97.

[4] Shane F. Cotter and Bhaskar D. Rao, "Sparse channel estimation via matching pursuit with application to equalization," *IEEE Transactions on Communications*, vol. 50, pp. 374–377, March 2002.

[5] W.F. Schreiber, "Advanced television systems for terrestrial broadcasting: Some problems and some proposed solutions," *Proceedings of the IEEE*, vol. 83, pp. 958–981, June 1995.

[6] 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.

[7] 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.

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

[9] 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.

[10] M. Zibulevsky, B.A. Pearlmutter, P. Bofill, and P. Kisilev, *Independent Component Analysis: Principles and Practice, chapter Blind Source Separation by Sparse Decomposition*, Cambridge, 2001.

[11] A. A. Amini, M. Babaie-Zadeh, and Ch. Jutten, "A fast method for sparse component analysis based on iterative detection-projection," in *Proceedings of Twenty sixth International Workshop on Bayesian Inference and MaximumEntropy Methods in Science and Engineering (MaxEnt)*, 2006.

[12] P. G. Georgiev, F. J. Theis, and A. Cichocki, "Sparse component analysis and blind source separation of underdetermined mixtures," *IEEE Transactions of Neural Networks*, vol. 16, no. 4, pp. 992–996, July 2005.

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

[14] N. Noorshams, M. Babaie-Zadeh, and C. Jutten, "Estimating the mixing matrix in sparse component analysis based on converting a multiple dominant to a single dominant problem," in *ICA'07*, (submitted).