Underdetermined Blind Source Separation Based on Sparse Representation

Yuanqing Li, Shun-Ichi Amari, *Fellow, IEEE*, Andrzej Cichocki, *Member, IEEE*, Daniel W. C. Ho, *Senior Member, IEEE*, and Shengli Xie, *Senior Member, IEEE*

***Abstract*—This paper discusses underdetermined (i.e., with more sources than sensors) blind source separation (BSS) using a two-stage sparse representation approach. The first challenging task of this approach is to estimate precisely the unknown mixing matrix. In this paper, an algorithm for estimating the mixing matrix that can be viewed as an extension of the DUET and the TIFROM methods is first developed. Standard clustering algorithms (e.g., K-means method) also can be used for estimating the mixing matrix if the sources are sufficiently sparse. Compared with the DUET, the TIFROM methods, and standard clustering algorithms, with the authors' proposed method, a broader class of problems can be solved, because the required key condition on sparsity of the sources can be considerably relaxed. The second task of the two-stage approach is to estimate the source matrix using a standard linear programming algorithm. Another main contribution of the work described in this paper is the development of a recoverability analysis. After extending the results in [7], a necessary and sufficient condition for recoverability of a source vector is obtained. Based on this condition and various types of source sparsity, several probability inequalities and probability estimates for the recoverability issue are established. Finally, simulation results that illustrate the effectiveness of the theoretical results are presented.**

***Index Terms*—Blind source separation (BSS), l^1-norm, probability, recoverability, sparse representation, wavelet packets.**

I. Introduction

SPARSE representation, or sparse coding, of signals has received a great deal of attention in recent years (e.g., [1]–[6]). Sparse representation of signals using large-scale linear programming under given overcomplete bases (e.g., wavelets) was discussed in [1]. Several improved FOCUSS-based algorithms were designed to solve underdetermined linear inverse problems in cases where both the dictionary and the sources were unknown (e.g., [4]). Donoho and Elad discussed optimal sparse representation in general (nonorthogonal) dictionaries via l^1 minimization [7]. An important application of sparse representation is in underdetermined blind source separation (BSS), which is difficult to deal with using a standard independent component analysis (ICA) method. In several references [9], [10], the mixing matrix and sources were estimated using the maximum posterior approach or the maximum-likelihood approach. A variational expectation maximization algorithm for sparse presentation was proposed in [11], which also can be used in underdetermined BSS. A two-stage cluster-then-l^1-optimization approach has been proposed for underdetermined BSS in which the mixing matrix and the sources are estimated separately [8].

Recently, [12] analyzed the two-stage clustering-then-l^1-optimization approach for sparse representation and its application in BSS. In [12], the basis matrix (or mixing matrix in BSS) was estimated using a K-means clustering algorithm. The uniqueness of the l^1-norm solution and its robustness to additive noise were discussed. The equivalence of the l^1-norm solution and the l^0-norm solution was also discussed within the context of a probabilistic framework. Furthermore, These results were then used in a recoverability analysis in BSS.

In this paper, we further extend and discuss the applications of the two-stage sparse representation approach for underdetermined BSS. We consider the following noise-free model and noisy model:

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

$$\mathbf{X} = \mathbf{AS} + \mathbf{V} \tag{2}$$

where the mixing matrix $\mathbf{A} \in R^{n\times m}$ is unknown, and $\mathbf{V} \in R^{n\times K}$ is a Gaussian noise matrix. The matrix $\mathbf{S} = [\mathbf{s}(1),\ldots,\mathbf{s}(K)] \in R^{m\times K}$ is composed of the m unknown sources, and the only observable $\mathbf{X} = [\mathbf{x}(1),\ldots,\mathbf{x}(K)] \in R^{n\times K}$ is a data matrix that has rows containing mixtures of sources. We assume, in general, that the number (m) of sources is unknown. In this paper, we mainly deal with the underdetermined case in which $n < m$. In fact, the algorithms in this paper are also suitable for the case in which $n = m$ without any change.

The task of BSS is to recover the sources using only the observable matrix $\mathbf{X}$. In the two-stage approach solving the problem, the mixing matrix $\mathbf{A}$ is estimated in the first stage, and the source matrix $\mathbf{S}$ is estimated in the second stage.

Manuscript received March 20, 2004; revised March 21, 2005. This work was supported in part by the National Natural Science Foundation of China (no. 60475004, no. 60325310), Guangdong Province Science Foundation for Research Team Program (no. 04205789), and the Excellent Young Teachers Program of MOE, P. R. China when the first author was with RIKEN Brain Science Institute, and with Institute of Automation Science and Engineering, South China University of Technology, Guangzhou, 510640, China. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ercan E. Kuruoglu.

Y. Li is with the Laboratory for Neural Signal Processing, Institute for Infocomm Research, Singapore 119613 (e-mail: yqli2@i2r.a-star.edu.sg).

S.-I. Amari is with the RIKEN Brain Science Institute, Saitama 3510198, Japan (e-mail: amari@brain.riken.jp).

A. Cichocki is with the Laboratory for Advanced Brain Signal Processing, RIKEN Brain Science Institute, Saitama 3510198, Japan, and also with the Department of Electrical Engineering, Warsaw University of Technology, Warsaw 00-661, Poland (e-mail: cia@brain.riken.jp).

D. W. C. Ho is with the Department of Mathematics, City University of Hong Kong, Hong Kong (e-mail: madaniel@cityu.edu.hk).

S. Xie is with the South China University of Technology, Guangzhou, 510640, China (e-mail: adshlxie@scut.edu.cn).

Digital Object Identifier 10.1109/TSP.2005.861743

In this paper, we base our theoretical analysis mainly on a noise-free model (1); the discussion on the influence of noise in (2) can been seen in [12]. However, the noise is still considered in Algorithm 1 for estimating the mixing matrix.

As will be seen in the following sections, the sparsity of source components plays a key role in the two-stage approach. Generally, sources are not sparse in the time domain; however, they can be sparse in the time-frequency (TF) domain if a suitable linear transformation is applied. Thus, if necessary, we can consider the problem in the TF domain rather than in the time domain. To obtain a more sparse TF representation, we apply a discrete wavelet packets transformation to (1) instead of a standard wavelet transformation. After applying the discrete wavelet packets transformation, which is a kind of linear transformation, we can obtain a new transformed model

$$\tilde{\mathbf{X}} = \mathbf{A}\tilde{\mathbf{S}} \tag{3}$$

where each row of $\tilde{\mathbf{X}}$ is the TF representation of the corresponding mixture signal in $\mathbf{X}$. Each row of $\tilde{\mathbf{S}}$ is the TF representation of the corresponding source in $\mathbf{S}$. Note that each row of $\tilde{\mathbf{X}}$, which is composed of the discrete wavelet packets transformation coefficients of a corresponding row of $\mathbf{X}$, is still discrete. We base the one-dimensional arrangement of these coefficients on the structure of the discrete wavelet packets transformation so that the inverse wavelet packets transformation can be used (see, for example, discrete wavelet packets transformation toolbox of Matlab).

How to estimate precisely the mixing matrix $\mathbf{A}$ is a challenging problem. There are several methods for estimating the mixing matrix, e.g., clustering algorithms (e.g., the K-means method), the DUET-type methods developed by Rickard, *et al.* [15], [16], the TIFROM-type methods developed by Deville, *et al.* [17]-[19], etc. However, we found that it is difficult to estimate the mixing matrix precisely with clustering algorithms when the sources are insufficiently sparse. The DUET-type and TIFROM-type methods are based on the ratios of TF transforms of observations. The DUET method can estimate the mixing matrix quite precisely if the W-disjoint orthogonality condition (or approximate W-disjoint orthogonality condition) is satisfied across the whole TF region. If the sources overlap to some degree, there exist some adjacent TF regions where only one source occurs, and the TIFROM method can be used to estimate the mixing matrix. In reality, there exist several factors that may lead to the violation of the sparsity condition required or necessary in the DUET and the TIFROM methods, e.g., noise, a relatively large number of sources, and a relatively high degree of sources' overlapping in the TF domain. Thus, we need to relax the sparsity condition in the two methods. In this paper, we propose a novel algorithm to estimate the mixing matrix, which is similarly based on the ratios of the TF transform; this algorithm can be viewed as an extension of the DUET- and TIFROM-type methods.

Compared with [12], one contribution of the present paper is the development of an algorithm for estimating the mixing matrix $\mathbf{A}$, an approach that requires less stringent conditions on sources and that can be applied to a wider range of data or signals than can the existing DUET- and TIFROM-type methods. Furthermore, our simulation results show that a more precise estimate of the mixing matrix $\mathbf{A}$ can be achieved using the proposed algorithm rather than by using a standard clustering algorithm (e.g., the K-means algorithm), as is typically used.

As in [12], this paper also discusses recoverability from the viewpoint of probability. We extend the results of [7] and [5] and obtain a sufficient and necessary condition for recoverability of a single-source column vector $\mathbf{s}(k)$. Using this sufficient and necessary condition, we establish and prove several probability inequalities on recoverability. These inequalities describe qualitatively the changing tendencies of the probability of recoverability with respect to the number of nonzero entries of the source column vector $\mathbf{s}(k)$, the number of observations, and the number of the sources. Furthermore, when the mixing matrix is given or estimated, several new probability estimation formulas on recoverability of a single-source column vector $\mathbf{s}(k)$ are obtained in three cases: 1) the number of nonzero entries of a source vector is fixed; 2) the probability that an entry of a source vector is equal to zero is fixed; and 3) the source entries are drawn from a Laplacian distribution. These probability estimates can provide new insights into the performance and limitations of the linear programming algorithm when it is used to recover sources.

The remainder of this paper is organized as follows. Section II presents an algorithm for estimating the mixing matrix. Section III focuses on the recoverability analysis when a standard linear programming algorithm is used to estimate the sources. A sufficient and necessary condition and several probability inequalities are obtained. Section IV presents several probability estimates for recoverability of a source column vector in the three different cases mentioned above. Simulation results are presented in Section V. The concluding remarks in Section VI review the approach proposed in this paper and state two open problems for future research.

II. Algorithm for Estimating the Mixing Matrix

In this section, we discuss the identification of the mixing matrix based on the TF model (3) and develop an estimating algorithm.

In the literature, we found that several standard clustering algorithms have been used to find the mixing matrix (e.g., Fuzzy-C clustering algorithm in [9] or K-means method). One essential condition for using these methods is that the sources must be very sparse in the analyzed domain. As with the standard clustering approach, the algorithm presented in this section is also based on source sparseness, but this precondition of sparseness is considerably relaxed.

Using an approach based on the ratios of the TF transform of the observed mixtures, [16] presented a DUET algorithm for estimating channel parameters and sources under so-called W-disjoint orthogonal condition (or approximate W-disjoint orthogonality). The TIFROM algorithm [17]–[19] also uses the ratios of the TF transform of the observed mixtures to estimate the mixing matrix and then to estimate the sources. With the TIFROM algorithm, TF coefficients of sources can overlap; however, there is a necessary condition, i.e., that there exist some adjacent TF regions where only one source occurs. Note

that, in each of these TF regions, the ratios of TF coefficients of the observed signals are equal (or almost equal) to a constant.

The starting point of our estimating algorithm is the same as that of the DUET and TIFROM algorithms, i.e., it is also based on ratios of the TF transforms of observed signals and on the fact there exist many TF points where only one source has a nonzero value. As we understand, the key point of the TIFROM algorithm is to find these adjacent TF regions where only one source has a nonzero value. Similarly, as in the DUET and the TIFROM methods, we first construct a ratio matrix using the wavelet packets transformation coefficients of the observable mixtures in the following algorithm. Next, we detect directly several submatrices of the ratio matrix each of which has almost identical columns. Note that if we plot the values of the entries of a submatrix with identical columns, we can obtain n horizontal lines corresponding to its n rows. The detection of these submatrices can be carried out by determining these horizontal lines. For each of these submatrices, the column indexes (in the ratio matrix) can be disconnected, which correspond to the TF points where only one source has a nonzero value. However, these TF points can be isolated (or discrete) (see Step a2) of Algorithm 1). Compared with the DUET method, the sources can overlap in the TF region to some degree in our algorithm, i.e., the W-disjoint orthogonal condition is not necessary to satisfy. Furthermore, unlike the TIFROM method, with our algorithm, it is unnecessary that there exist some adjacent TF regions where only one source occurs.

Before describing the complete method, we illustrate the basic idea of our algorithm by presenting a simple example of how to estimate the first column of the mixing matrix. We assume that the sources are sparse to some degree in the TF domain such that there exists a sufficiently large number of columns of $\tilde{\mathbf{S}}$ with only one nonzero entry. (In a noisy environment, this means single-entry dominant vectors.) Now we consider those columns of $\tilde{\mathbf{S}}$, of which only the first source has a nonzero value. For instance, we assume that L source column vectors $\tilde{\mathbf{s}}(i_1),\dots,\tilde{\mathbf{s}}(i_L)$ are the columns of $\tilde{\mathbf{S}}$ with only their first entries being nonzero, noting that $i_1,\dots,i_L$ are not generally adjacent. We have

$$\begin{aligned}[\tilde{\mathbf{x}}(i_1),\dots,\tilde{\mathbf{x}}(i_L)] &= \mathbf{A}[\tilde{\mathbf{s}}(i_1),\dots,\tilde{\mathbf{s}}(i_L)] \\ &= [\mathbf{a}_1\tilde{s}_1(i_1),\dots,\mathbf{a}_1\tilde{s}_1(i_L)]. \end{aligned} \tag{4}$$

Taking a $q\in\{1,\dots,n\}$, we calculate the ratio matrix similarly as in DUET and TIFROM algorithms

$$\begin{bmatrix} \frac{\tilde{x}_1(i_1)}{\tilde{x}_q(i_1)} & \cdots & \frac{\tilde{x}_1(i_L)}{\tilde{x}_q(i_L)} \\ \vdots & \vdots & \vdots \\ \frac{\tilde{x}_n(i_1)}{\tilde{x}_q(i_1)} & \cdots & \frac{\tilde{x}_n(i_L)}{\tilde{x}_q(i_L)} \end{bmatrix}. \tag{5}$$

It follows from (4) that

$$\begin{bmatrix} \frac{\tilde{x}_1(i_1)}{\tilde{x}_q(i_1)} & \cdots & \frac{\tilde{x}_1(i_L)}{\tilde{x}_q(i_L)} \\ \vdots & \vdots & \vdots \\ \frac{\tilde{x}_n(i_1)}{\tilde{x}_q(i_1)} & \cdots & \frac{\tilde{x}_n(i_L)}{\tilde{x}_q(i_L)} \end{bmatrix} = \begin{bmatrix} \frac{a_{11}}{a_{q1}} & \cdots & \frac{a_{11}}{a_{q1}} \\ \vdots & \vdots & \vdots \\ \frac{a_{n1}}{a_{q1}} & \cdots & \frac{a_{n1}}{a_{q1}} \end{bmatrix}. \tag{6}$$

Note that although each column of the matrix in (4) is equal to $\mathbf{a}_1$ up to a scale, it is difficult in reality to determine a single column of the matrix especially for noisy data. By using the property of the ratios in (6), it is possible to detect a set of columns of the matrix in (4). Considering the noise, we use the following mean vector as the estimate of $\mathbf{a}_1$:

$$\mathbf{a}_1 = a_{q1}\left[\operatorname{mean}\left(\frac{\tilde{x}_1(i_j)}{\tilde{x}_q(i_j)}\right),\dots,\operatorname{mean}\left(\frac{\tilde{x}_n(i_j)}{\tilde{x}_q(i_j)}\right)\right]^T \tag{7}$$

where $\operatorname{mean}(\tilde{x}_1(i_j))/(\tilde{x}_q(i_j)) = (1/L)\sum_{j=1}^{L}(\tilde{x}_1(i_j))/(\tilde{x}_q(i_j))$. Using (7), we can estimate the column $\mathbf{a}_1$ up to a scale, provided that we detect these identical (or almost identical in reality) columns in (5).

Noting that all columns of the ratio matrix in (6) are identical, if we plot the values of these ratios versus the indexes of these ratios, we can obtain n horizontal lines (see Fig. 2 in Example 1). Each of the n horizontal lines corresponds to a row of the matrix in (6), and the heights of the n horizontal lines represent the entries of one column of the mixing matrix (up to a scale).

Furthermore, the matrix in (6) is a submatrix of the matrix

$$\begin{bmatrix} \frac{\tilde{x}_1(1)}{\tilde{x}_q(1)} & \cdots & \frac{\tilde{x}_1(K)}{\tilde{x}_q(K)} \\ \vdots & \vdots & \vdots \\ \frac{\tilde{x}_n(1)}{\tilde{x}_q(1)} & \cdots & \frac{\tilde{x}_n(K)}{\tilde{x}_q(K)} \end{bmatrix} \tag{8}$$

where we assume that $\tilde{x}_q(k)\neq 0$, for $k=1,\dots K$ (the case that some entries $\tilde{x}_q(k)=0$ will be considered in our algorithm). In the following, we always use $[(\tilde{x}_i(k))/(\tilde{x}_q(k))]_{n\times K}$ to denote the matrix in (8). Thus, if we find a submatrix of $[(\tilde{x}_i(k))/(\tilde{x}_q(k))]_{n\times K}$ such that its n rows form n horizontal lines, then the submatrix has identical columns, which is similar to the matrix in (6). We can calculate a column vector of the mixing matrix using (7). In addition, the column indexes of the columns of the submatrix (in the ratio matrix $[(\tilde{x}_i(k))/(\tilde{x}_q(k))]_{n\times K}$), which correspond to TF points, can be disconnected.

Suppose that $[r_i,R_i]$ is the range (interval) of the entries in the ith row of the ratio matrix $[(\tilde{x}_i(k))/(\tilde{x}_q(k))]_{n\times K}$, i.e., $(\tilde{x}_i(k))/(\tilde{x}_q(k))\in[r_i,R_i]$ for $k=1,\dots,K$. There are n such entry intervals for the ratio matrix. In the following algorithm, we partition differently compared with the way it is done with TIFROM method. That is, we divide the entry intervals into many bins for detecting these horizontal lines mentioned above. This is the key point of our algorithm. In contrast, with the TIFROM method, the partition is performed in the TF regions, and adjacent regions are sought where only one source occurs.

In the following algorithm, the first key point is to construct a ratio matrix (see (9)) using the wavelet packet transform coefficients (similarly as in the DUET and TIFROM methods). The second key point is to detect a submatrix of the ratio matrix which has almost identical columns. Note that such a submatrix corresponds to a single column of the mixing matrix. All entries of the submatrix with almost identical columns can form n horizontal lines corresponding to its n rows. The detection of such a submatrix is mainly achieved by finding the n horizontal lines. The indexes of the columns of the submatrix in the ratio matrix can be arbitrarily disconnected, which correspond to disconnected TF points where only one source has nonzero value. To find the n horizontal lines, we partition the ranges of the entries of two rows of the ratio matrix and obtain many relatively small bins. We found by extensive simulations that if we

detect any two horizontal lines, then the other $n-2$ horizontal lines can be detected simultaneously and automatically. By this method based on partition of ratios, we can first obtain several submatrices of the ratio matrix. The column indexes of these submatrices (in the ratio matrix) correspond to the TF points which can be arbitrarily disconnected. At least one submatrix has almost identical columns. From the several submatrices, we then choose one with the lowest row variance, which is the detected submatrix with almost identical columns.

Now we present the detailed algorithm for estimating the mixing matrix $\mathbf{A}$, and the output estimated matrix is denoted as $\bar{\mathbf{A}}$. Also, let a matrix $\mathbf{E}=[\mathbf{e}_1,\ldots,\mathbf{e}_{N_0}]$ be a matrix to store the estimated columns during intermediate steps. Initially, $\mathbf{E}$ is an empty matrix.

Algorithm 1:

Step 1. Apply a wavelet packets transformation to every row of the matrix $\mathbf{X}\in R^{n\times K}$. A TF representation matrix $\tilde{\mathbf{X}}$ is obtained. The purpose of this step is to produce sparsification.

Step 2. Find a submatrix $\hat{\mathbf{X}}$ of $\tilde{\mathbf{X}}$ such that the norm of its each column is greater than ξ_1, where ξ_1 is a positive constant chosen in advance. (*The purpose of this step is twofold: first, it reduces the computational burden of estimation process, and second, it removes those columns that are disproportionately influenced by noise.*)

Step 3. For $n_1=1$ to n, do the following (Loop 1, including Steps 3.1 and 3.2).

Step 3.1. If the absolute value of an entry of the n_1th row of $\hat{\mathbf{X}}$ is very small, say less than a preset positive constant ξ_2, then we shall remove the corresponding column of $\hat{\mathbf{X}}$ containing the entry. Suppose that there are K_1 columns of $\hat{\mathbf{X}}$ left and denote their indexes as $q_1,\ldots,q_{K_1}$. We construct a new ratio matrix using the left K_1 columns of $\hat{\mathbf{X}}$

$$\tilde{\hat{\mathbf{X}}}=\begin{bmatrix}\frac{\hat{x}_1(q_1)}{\hat{x}_{n_1}(q_1)} & \cdots & \frac{\hat{x}_1(q_{K_1})}{\hat{x}_{n_1}(q_{K_1})}\\ \vdots & \vdots & \vdots\\ \frac{\hat{x}_n(q_1)}{\hat{x}_{n_1}(q_1)} & \cdots & \frac{\hat{x}_n(q_{K_1})}{\hat{x}_{n_1}(q_{K_1})}\end{bmatrix}. \tag{9}$$

Note: *After Step* 3.1, *the matrix* $\tilde{\hat{\mathbf{X}}}$ *contains several submatrices in general with disconnected column indices* (*similar to the matrix in* (6)) *of which the* n *rows can form* n *horizontal lines* (*i.e., each of these submatrices has identical, or almost identical, columns*). *The heights of these horizontal lines correspond to the entries of one column of the mixing matrix up to a scale. The task of the following steps is to find these submatrices with their rows representing horizontal lines*.

Step 3.2. For $n_2=1$ to $n, n_2\neq n_1$, do the following (Loop 2, including Steps 3.2.1, 3.2.2, and 3.2.3).

Step 3.2.1 Find the minimum $\tilde{r}_{n_2}$ and maximum $\tilde{R}_{n_2}$ of $\tilde{\hat{\mathbf{x}}}_{n_2}$, the n_2th row of $\tilde{\hat{\mathbf{X}}}$. Divide the entry range (interval) $[\tilde{r}_{n_2},\tilde{R}_{n_2}]$ of $\tilde{\hat{\mathbf{x}}}_{n_2}$ equally into M_0 subintervals (bins), where M_0 is a chosen large positive integer. Then, divide the matrix $\tilde{\hat{\mathbf{X}}}$ into M_0 submatrices, denoted as $\tilde{\hat{\mathbf{X}}}_1,\ldots,\tilde{\hat{\mathbf{X}}}_{M_0}$, such that all entries of the n_2th row of $\tilde{\hat{\mathbf{X}}}_k$ are in the kth bin, $k=1,\ldots,M_0$.

Note: *In this step, we perform a partition in the range of the entries in the* n_2*th row of the matrix* $\tilde{\hat{\mathbf{X}}}$ *and then perform a corresponding partition to the set of columns of* $\tilde{\hat{\mathbf{X}}}$ *and obtain* M_0 *submatrices. Since* M_0 *is chosen large generally, every bin above is relatively small. Thus, for each of the* M_0 *submatrices, the plot of all entries of the* n_2*th row (in a bin) can be taken as an horizontal line*.

Step 3.2.2. From the submatrix set $\{\tilde{\hat{\mathbf{X}}}_k,k=1,\ldots,M_0\}$, delete those submatrices of which the number of columns is less than J_1, where J_1 is a chosen positive integer. The new set of submatrices is denoted as $\{\tilde{\hat{\mathbf{X}}}_{j_k},k=1,\ldots,N_1\}$.

Note: *The objective of Steps* 3.2.1 *and* 3.2.2 *is to find the horizontal lines mentioned above by performing a partition in the range of entries of one row of* $\tilde{\hat{\mathbf{X}}}$, *then performing a partition to the set of columns the matrix. However, these two steps are not sufficient to detect these horizontal lines; we need to reduce the matrices obtained in Step* 3.2.2 *by more fine partitions. In Example* 1, *a submatrix obtained in Step* 3.2.2 *is shown in the first subplot of* Fig. 2.

Step 3.2.3. For $n_3=1$ to $n, n_3\neq n_1 n_3\neq n_2$, do the following Steps a1, a2, and a3 for every matrix in $\{\tilde{\hat{\mathbf{X}}}_{j_k},k=1,\ldots,N_1\}$.

Step a1. For the matrix $\tilde{\hat{\mathbf{X}}}_{j_k}$, perform a step similar to Step 3.2.1. That is, n_3 and $\tilde{\hat{\mathbf{X}}}_{j_k}$ are used to replace n_2 and $\tilde{\hat{\mathbf{X}}}$ in Step 3.2.1 and then, carrying out Step 3.2.1, we can obtain M_0 submatrices of $\tilde{\hat{\mathbf{X}}}_{j_k}$, denoted as $\tilde{\hat{\mathbf{X}}}_q^{(j_k)}, q=1,\ldots,M_0$.

Step a2. From the submatrix set $\{\tilde{\hat{\mathbf{X}}}_q^{(j_k)}, q=1,\ldots,M_0\}$, deleting those submatrices with their number of columns less than a prefixed positive integer J_2, we obtain a new set of submatrices. From the new submatrix set, choose a matrix, for example, $\tilde{\hat{\mathbf{X}}}_p^{(j_k)}$, such that the sum of variances of its n rows is the smallest. By this step, we find a submatrix with small row variances and its number of columns larger than J_2.

Note: *Steps a1 and a2* (*similar to Steps* 3.2.1 *and* 3.2.2) *reduce the submatrices obtained in Step* 3.2.2. *Specifically, a partition in the entry range of the* n_3*th row of the matrix* $\tilde{\hat{\mathbf{X}}}_{j_k}$ *is performed in Step a1*; *the matrix is then divided into* M_0 *submatrices. For each of the* M_0 *submatrices, the plot of all entries in the* n_3*th row (in a small*

bin) can been considered as a horizontal line. Through these two steps, a submatrix ($\tilde{\tilde{\mathbf{X}}}_p^{(j_k)}$, *as above*) *of* $\tilde{\tilde{\mathbf{X}}}$ *can be obtained, and its* n *rows can form* n *clear horizontal lines as our experience in simulations (i.e., its columns are almost identical). The submatrix is similar to that in* (5). *The column indexes of* $\tilde{\tilde{\mathbf{X}}}_p^{(j_k)}$ *in the ratio matrix* $\tilde{\tilde{\mathbf{X}}}$, *which correspond to TF points, can be disconnected. In Example 1, the second subplot of* Fig. 2 *shows a submatrix* ($\tilde{\tilde{\mathbf{X}}}_p^{(j_k)}$) *obtained in Step a2.*

Step a3. Calculating the mean of all the column vector of the matrix $\tilde{\tilde{\mathbf{X}}}_p^{(j_k)}$ and normalizing the averaged column vector to the unit norm, we obtain an estimated column vector, denoted as $\mathbf{e}_i$, of the mixing matrix $\mathbf{A}$.

Step 4. After carrying out the four loops above, we can obtain a set of estimated columns, denoted as $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_{N_0}]$. Generally, each column of $\mathbf{E}$ is an estimate of one column of the mixing matrix $\mathbf{A}$. Since there are several loops above, one column of $\mathbf{A}$ may be estimated many times; all estimates are stored in $\mathbf{E}$. At last, $\mathbf{E}$ has more columns than $\mathbf{A}$ (e.g., $\mathbf{E}$ has 75 columns in Example 4, while $\mathbf{A}$ has only 8 columns).

In the final step, we need to remove the duplication of column vectors in $\mathbf{E}$. If there exist several columns of $\mathbf{E}$ that are almost parallel to each other (i.e., almost the same or opposite in direction), we should calculate their mean direction vector followed by normalization. Finally, the obtained matrix, denoted as $\bar{\mathbf{A}} \in R^{n \times m_0} (m_0 > m)$ (e.g., $\bar{\mathbf{A}}$ has 21 columns in Example 4), is taken as the estimate of the original mixing matrix $\mathbf{A}$.

End of Algorithm 1.

Remark 1: 1. If the sensor number (n) is equal to 2, then Step 3.2.3, including Steps a1, a2, and a3, is unnecessary; a simplified version of Algorithm 1 still works, provided the sources are sufficiently sparse in the analyzed domain. 2. Note that Step 3.2.1 and Steps a1, a2, and a3 are similar to each other. In Step 3.2.1, the entry range of one row in a ratio matrix was partitioned; in Step a1, the entry range of one row in a submatrix of the ratio matrix was partitioned. It is clear from our simulations (for artificial data, speech signals and EEG signals) that two partitions are sufficient to detect these horizontal lines. In reality, if the sources overlap a lot (i.e., they are not sparse) and twofold partition is insufficient, we need to partition more than two times; that is, partitioning should proceed until n horizontal lines are detected. The implementation of more than two partitions is similar to Step 3.2.1 and Step a1. 3. In reality, because noise exists or because a source column vector lacks strict sparseness (only one nonzero entry), several columns of $\mathbf{E}$ may not be the true estimates of the columns of $\mathbf{A}$. The last estimate $\bar{\mathbf{A}}$ contains a submatrix that is close to $\mathbf{A}$, neglecting nonessential scaling and permutation of columns. From the robustness analysis in [12], it follows that the estimate can be effective for estimating sources (i.e., the second stage of the two-stage approach), provided that the submatrix of $\bar{\mathbf{A}}$ is sufficiently close to the mixing matrix $\mathbf{A}$.

In Algorithm 1, there exist five parameters, which should be set in advance. In principle, their values depend on the data. ξ_1, ξ_2 are related to the amplitude of the entries of the data matrix in the analyzed domain. Let Q_1 denote the maximum of the norms of all columns of the data matrix, and Q_2 denote the maximum of the amplitude of the entries of the data matrix. We set the parameter ξ_1 to reduce noise and computation burden as stated in the algorithm. We can set ξ_1 to be a fraction of Q_1 (e.g., $0.3Q_1$). For implementing division operation, ξ_2 can be set as a fraction Q_2 (e.g., $0.1Q_2$). M_0 is the number of bins, which is set to be 400 in our simulations. J_1 and J_2 are the low bounds of the column number of the selected submatrices. Generally, $J_2 \leq J_1$. The general principle for setting M_0, J_1 and J_2 is to make sure that these horizontal lines can be observed clearly.

It is well known that general clustering methods (e.g., the K-means method) are iterative algorithms without global convergence. For a different number of clusters and a different starting point chosen before iteration, the obtained results are different generally. Thus, the sources should be very sparse when a clustering method is used for estimating the mixing matrix. Algorithm 1 is less sensitive to a lack of source sparsity in the analyzed domain than are general clustering methods (e.g., the K-means method), because it is not an iterative algorithm. Now we present an example to illustrate this assertion.

Example 1: In this example, the source matrix $\mathbf{S} \in R^{5 \times 10\,000}$ has two types of columns. That is, 9000 columns of $\mathbf{S}$ have their entries drawn from a uniform distribution valued in $[-5, 5]$; the other 1000 columns have only one nonzero entry and four zero entries. Furthermore, the 1000 columns are divided into five sets, the ith set of which has 200 columns in which their ith entries have nonzero values (also drawn from the uniform distribution) $(i = 1, \ldots, 5)$. The column indexes of the 1000 columns in the matrix $\mathbf{S}$ are disconnected. More precisely, the column indexes of the 200 columns of the ith set are as follows: $(i-1) \times 10 + 1, (i-1) \times 10 + 51, \ldots, (i-1) \times 10 + 9951, i = 1, \ldots, 5$. The mixtures $\mathbf{X} = \mathbf{AS}$, where the mixing matrix $\mathbf{A} \in R^{3 \times 5}$ is taken randomly followed by normalization as in (10), shown at the bottom of the page.

$$\mathbf{A} = \begin{bmatrix} 0.8412 & -0.0298 & -0.0750 & -0.1735 & 0.7240 \\ -0.5025 & 0.8305 & -0.8294 & 0.3621 & 0.4088 \\ 0.1997 & 0.5563 & 0.5536 & 0.9158 & -0.5556 \end{bmatrix}. \tag{10}$$

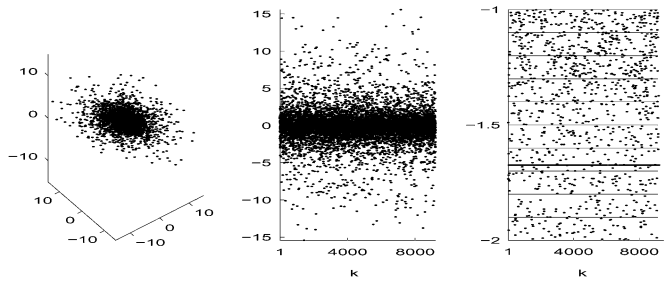

Fig. 1. Left: 3-D scatter plot of the data matrix $\hat{\bar{\mathbf{X}}}$ from Step 3.1 of Algorithm 1, in which one point corresponds to a column value of the matrix. Middle: Plot of all entries in the first row of the matrix. Right: Plot of some entries in the first row of the matrix and some bins (shown by red lines) (see Step 3.2.1) for detecting horizontal lines in Example 1.

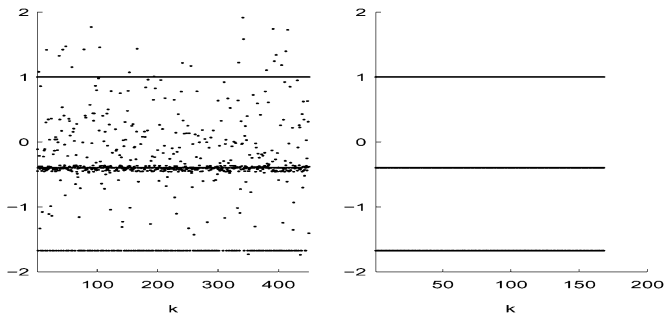

Fig. 2. Left: Plot of all entries of the matrix $(\bar{\tilde{\mathbf{X}}}_{j_k})$ from Step 3.2.2 of Algorithm 1. Right: Plot of all entries of the reduced matrix $\left(\bar{\tilde{\mathbf{X}}}_p^{(j_k)}\right)$ from Step a2 of Algorithm 1 in Example 1.

Now we use Algorithm 1 to estimate the mixing matrix $\mathbf{A}$, based only on the observed mixture matrix $\mathbf{X}$. Noting that the indexes of the 1000 source columns with only one nonzero entry are not disjointed (corresponding to the disconnected time points), the DUET and the TIFROM algorithms thus cannot be used for estimating the mixing matrix in such case.

Considering that the data set is artificial and that noise is absent, we do not need to carry out Steps 1 and 2. By carrying out Step 3, we obtain the set of estimated columns $\mathbf{E}$ as in (11), shown at the bottom of the page. Comparing the columns of $\mathbf{E}$ with those of $\mathbf{A}$, we can see that Algorithm 1 works well, although only 10% of the column vectors of $\mathbf{S}$ are sparse (i.e., only one entry has a nonzero value).

We tried the standard K-means clustering method to estimate the columns of $\mathbf{A}$. However, we did not obtain any satisfactory results. The K-means method does not work here, because there is a very small fraction of sparse source column vectors.

The left subplot of Fig. 1 shows the three-dimensional (3-D) scatter plot of all column values of the matrix $(\hat{\bar{\mathbf{X}}})$ produced in Step 3.1 of Algorithm 1. The middle subplot of Fig. 1 shows the values of all entries in the first row of the matrix. Of course, cursory inspection of these two plots reveals that no cluster is visible. The right subplot of Fig. 1 shows the values of some entries in the first row of the matrix and some bins (see Step 3.2.1). Note that the right subplot shows the portion of the middle subplot from height -2 to height -1. In our algorithm, we mainly use these bins to detect the horizontal lines mentioned above. As we indicated above, the heights of these horizontal lines represent the entries of the mixing matrix.

The left subplot of Fig. 2 shows all entries of the matrix $(\bar{\tilde{\mathbf{X}}}_{j_k})$ produced in Step 3.2.2 of Algorithm 1, in which we can see several line segments interspersed with other points. The heights of these line segments correspond to the entries of the first column of the mixing matrix $\mathbf{A}$ up to a scale. Since we need a more precise estimate, we continue this procedure to further reduce the matrix $\bar{\tilde{\mathbf{X}}}_{j_k}$.

The right subplot of Fig. 2 shows all entries of a matrix $\left(\bar{\tilde{\mathbf{X}}}_p^{(j_k)}\right)$ from Step a2 of Algorithm 1, in which we can see three "clear" line segments. The three line segments are formed by three rows of the matrix $\bar{\tilde{\mathbf{X}}}_p^{(j_k)}$ with their heights corresponding to the three entries of the first column of the mixing matrix $\mathbf{A}$ up to a scale. The normalized mean column vector of the matrix $\bar{\tilde{\mathbf{X}}}_p^{(j_k)}$ is an estimated column of the mixing matrix $\mathbf{A}$ (one of the columns of $\mathbf{E}$).

Finally, we present some information about the five free parameters in Algorithm 1. Because we carried out this simulation in the time domain and in the absence of noise, Steps 1, 2, and parameter ξ_1 are not needed. The other four parameters are $\xi_2 = 0.5$, $M_0 = 400$, and $J_1 = J_2 = 100$.

III. Linear Programming Algorithm for Estimating Source Matrix and Recoverability Analysis

After the mixing matrix $\mathbf{A}$ is estimated, the next step is to recover the sources. This is carried out by using the linear programming algorithm in this paper. However, there is a recoverability problem; that is, rephrased as a question, Under what condition is the solution obtained by linear programming equal to the sources? This problem was preliminarily addressed and discussed in [12]. In this paper, we extend and improve the results in [12] from the viewpoint of probability and obtain several new results.

In this section, we first extend the results of [7] and [5] by obtaining a sufficient and necessary condition for recoverability of a source column vector. Then, we establish and prove several probability inequalities that show the qualitative changing tendencies of probability of recoverability with regard to the sparsity of sources, the number of sources, and the number of sensors.

$$\mathbf{E} = \begin{bmatrix} 0.7246 & 0.8419 & 0.7240 & -0.1733 & -0.0749 & -0.0304 & -0.0304 & -0.0302 \\ 0.4081 & -0.5016 & 0.4088 & 0.3621 & -0.8294 & 0.8305 & 0.8305 & 0.8305 \\ -0.5553 & 0.1991 & -0.5556 & 0.9159 & 0.5536 & 0.5562 & 0.5562 & 0.5561 \end{bmatrix}. \tag{11}$$

Given that the sources are sufficiently sparse in the time domain, the source matrix can be found by solving the following linear programming problem similarly as in [1], [7], [8]:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{K} |s_i(j)|, \quad \text{s.t. } \mathbf{AS} = \mathbf{X} \tag{12}$$

where $s_i(j)(i = 1, \ldots, m, j = 1, \ldots, K)$ are entries of $\mathbf{S}$.

If the sources are not sufficiently sparse in the time domain, we can consider the following linear programming problem in the TF domain:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{K} |\tilde{s}_i(j)|, \quad \text{s.t. } \mathbf{A}\tilde{\mathbf{S}} = \tilde{\mathbf{X}} \tag{13}$$

where $\tilde{s}_i(j)(i = 1, \ldots, m, j = 1, \ldots, K)$ are entries of $\tilde{\mathbf{S}}$. After $\tilde{\mathbf{S}}$ is estimated, $\mathbf{S}$ can be obtained by inverse wavelet packets transformation.

In the previous section, we discussed estimating the mixing matrix, mainly in the TF domain. Hence, to obtain the coherence with the discussion of the previous section, it is better to discuss the recoverability based on the model (13). However, to simplify the notations, we base the discussions outlined in this and the next sections (on recoverability and the probability estimation of recoverability) only on the problem (12). Consequently, we find that the discussions in the two sections are also suitable for the model (13).

Now we analyze the recoverability. Consider the optimization problem

$$(P_1) \min \sum_{i=1}^{m} |s_i|, \quad \text{s.t. } \mathbf{As} = \mathbf{x}^*$$

where $\mathbf{A} \in R^{n \times m}, \mathbf{s} \in R^m, n \leq m$. $\mathbf{s}^* \in R^m, \|\mathbf{s}^*\|_0 = l < n$ (i.e., the number of nonzero entries of $\mathbf{s}^*$ is l), $\mathbf{x}^* = \mathbf{As}^*$. In the following, let $\mathbf{s}_1$ denote a solution of (P_1). Note that $\mathbf{s}^*$ can be considered to be one column $\mathbf{s}(k)$ of $\mathbf{S} \in R^{m \times K}$ in (12).

In [7], Donoho and Elad presented a sufficient condition, that is, less than 50% concentration implies equivalence between the l^1-norm solution and l^0-norm solution (the definition of l^0-norm solution can be seen in [7] and [12]). In the following theorem, we extend the result presented in [5] and [7] into a sufficient and necessary condition.

Let G denote the index set of nonzero entries of $\mathbf{s}^*$, i.e., $G = \{i_1, \ldots, i_l | s^*_{i_j} \neq 0, j = 1, \ldots, l\}$. F denotes the set of all subsets of G except the null set. Note that the cardinality of F is $2^l - 1$.

Theorem 1:

1) $\mathbf{s}^* = \mathbf{s}_1$, if and only if, $\forall I \in F$, when the following optimization problem (14) is solvable, its optimal value is less than $(1/2)$

$$\begin{aligned} &\max \sum_{k \in I} |\delta_k|, \quad \text{s.t.} \\ &\mathbf{A}\boldsymbol{\delta} = 0, \|\boldsymbol{\delta}\|_1 = 1, \\ &\delta_k s^*_k > 0 \quad \text{for } k \in I, \\ &\delta_k s^*_k \leq 0 \quad \text{for } k \in G \setminus I. \end{aligned} \tag{14}$$

2) Specifically, if $\|\mathbf{s}^*\|_0 = 1$, and any two columns of $\mathbf{A}$ are independent, then $\mathbf{s}^* = \mathbf{s}_1$.

In Theorem 1, l^0-norm $\|\mathbf{s}^*\|_0$ represents the number of nonzero entries of $\mathbf{s}^*$. The proof of this theorem is presented in Appendix I. Theorem 1 is fundamental for the subsequent discussion.

Remark 2: For different subset $I \in F$, there are a total $2^l - 1$ optimization problems as (14), of which only some are solvable. The sufficient and necessary condition in Theorem 1 relates only to these solvable optimization problems.

From Theorem 1, we can obtain the following corollary, which has appeared in [13] and [14].

Corollary 1: Given the mixing matrix $\mathbf{A}$, the recoverability of a source column vector $\mathbf{s}^*$ by solving the problem (P_1) does not depend on the magnitudes of its entries but only on the index set and signs of its nonzero entries.

Now we assume that all entries of $\mathbf{A} \in R^{n \times m} (n < m)$ are drawn from a fixed probability distribution (e.g., uniform distribution valued in an interval); all nonzero entries of $\mathbf{s}^* \in R^m$ are also drawn from a fixed probability distribution (e.g., Laplacian distribution). The indexes of nonzero entries of $\mathbf{s}^*$ are also taken randomly, and $\|\mathbf{s}^*\|_0 = l \leq n$. In the following, we will consider the probability of recoverability, which is related to the number of observations (n), the number of sources (m), and the number of the nonzero entries of the source column vector $\mathbf{s}^*$ (l). Let us define the conditional probability

$$P(n, m, l) = P(\mathbf{s}^* = \mathbf{s}_1 / \|\mathbf{s}^*\|_0 = l). \tag{15}$$

We will analyze the conditional probability $P(n, m, l)$ in three different cases: i) n and m are fixed; ii) n and l are fixed; iii) m and l are fixed. Using Theorem 1, we will prove three probability inequalities with regard to recoverability.

Case i) n and m are fixed.

We have the following theorem.

Theorem 2: For fixed n and m, we have $1 = P(n, m, 0) = P(n, m, 1) \geq P(n, m, 2) \geq \cdots \geq P(n, m, n)$.

The Proof of Theorem 2 is presented in Appendix II. The probability inequalities in Theorem 2 describes quantitatively the changing tendency of probability with respect to the number of zero entries of a source vector. That is, the higher sparsity of a source vector is, the higher probability of recoverability is.

Case ii) n and l are fixed.

Suppose that $\mathbf{A}_1 \in R^{n \times m_1}, \mathbf{A}_2 \in R^{n \times m_2} (n < m_1 < m_2), \mathbf{q}^* \in R^{m_1}, \mathbf{r}^* \in R^{m_2}, \mathbf{x}_1 = \mathbf{A}_1\mathbf{q}^*, \mathbf{x}_2 = \mathbf{A}_2\mathbf{r}^*$. Now we consider the following two optimization problems:

$$\min \|\mathbf{q}\|_1, \quad \text{s.t. } \mathbf{A}_1\mathbf{q} = \mathbf{x}_1 \tag{16}$$

$$\min \|\mathbf{r}\|_1, \quad \text{s.t. } \mathbf{A}_2\mathbf{r} = \mathbf{x}_2 \tag{17}$$

and denote $\mathbf{q}_1$ and $\mathbf{r}_1$ the solutions of (16) and (17), respectively.

Theorem 3: Suppose that all entries of $\mathbf{A}_1$ and $\mathbf{A}_2$ in (16) and (17), respectively, are drawn randomly from a distribution, all nonzero entries of $\mathbf{q}^*, \mathbf{r}^*$ are drawn randomly from a distribution, and the indexes of their nonzero entries are also taken randomly, $\|\mathbf{q}^*\|_0 = \|\mathbf{r}^*\|_0 = l \leq n$. Then, we have $P\{\mathbf{q}^* = \mathbf{q}_1\} \geq P\{\mathbf{r}^* = \mathbf{r}_1\}$; that is, $P(n, m_1, l) \geq P(n, m_2, l)$.

The proof is presented in Appendix III. The probability inequality given in Theorem 3 reflects the changing tendency of probability with respect to the number of sources. That is, the larger the number of sources is, the less the probability of recoverability is.

Case iii) m and l are fixed.

Suppose that $\mathbf{A}_3 \in R^{n_1\times m}, \mathbf{A}_4 \in R^{n_2\times m}(n_1 < n_2 < m), \mathbf{s}^* \in R^m, \mathbf{x}_3 = \mathbf{A}_3\mathbf{s}^*, \mathbf{x}_4 = \mathbf{A}_4\mathbf{s}^*$. Consider the following two optimization problems:

$$\min \|\mathbf{t}\|_1, \quad \text{s.t. } \mathbf{A}_3\mathbf{t} = \mathbf{x}_3 \tag{18}$$

$$\min \|\mathbf{w}\|_1, \quad \text{s.t. } \mathbf{A}_4\mathbf{w} = \mathbf{x}_4 \tag{19}$$

and also denote $\mathbf{t}_1$ and $\mathbf{w}_1$ as the solutions of (18) and (19), respectively.

Theorem 4: Suppose that all entries of $\mathbf{A}_3$ in (18) $\mathbf{A}_4$ in (19) are drawn randomly from a distribution, all nonzero entries of $\mathbf{s}^*$ are drawn randomly from a distribution, and the indexes of its nonzero entries are also taken randomly. Then, we have $P\{\mathbf{s}^* = \mathbf{t}_1\} \le P\{\mathbf{s}^* = \mathbf{w}_1\}$; that is, $P(n_1, m, l) \le P(n_2, m, l)$.

The proof is similar to that of Theorem 3, which is presented in Appendix III. The probability inequality in Theorem 4 reflects the changing tendency of probability with respect to the number of sensors. That is, the larger the number of sensors is, the higher the probability of recoverability is.

The qualitative tendencies of change in the probability of recoverability qualitatively shown in the inequalities above were first identified via simulations in [12]. However, the three inequalities of Theorems 2, 3, and 4 were neither stated explicitly nor proven in [12].

IV. Probability Estimation

In the previous section, we presented several probability inequalities showing the qualitative changing tendencies of the probability of recoverability. In this section, we assume that the mixing matrix is given or estimated. We will estimate analytically the probability that a source vector can be recovered by solving a linear programming problem (P_1) with a given or estimated mixing matrix. Three cases are considered: 1) when the number of nonzero entries of the source vector is fixed; 2) when the number of nonzero entries of the source vector is unknown and when the probability is known that every entry of a source vector is equal to zero; 3) when all entries of a source vector are drawn from a Laplacian distribution. The probability estimate of recoverability can give us new insights into the performance and limitation of the linear programming algorithm when it is used to recover sources. For instance, if we know (or estimate) the probability distribution of source data, then we can estimate the probability of recoverability and decide whether the recovered sources are acceptable. Furthermore, if we know (or estimate) the number of sources, we can estimate how many sensors should be available in order for the sources to be recovered with high probability.

First, we present several notations for the index set. Let $G_0 = \{1, \ldots, m\}$, and G denote the index set of nonzero entries of $\mathbf{s}^*$, as in the previous section. Obviously, there are $C_m^j (= (m!/j!(m-j)!))$ index subsets of G_0 with cardinality j. Denote these subsets as $G_k^j, k = 1, \ldots, C_m^j$.

The probability estimate is mainly based on the sufficient and necessary condition of Theorem 1. It is not difficult to find that optimization problem (14) is equivalent to the following optimization problem:

$$\begin{aligned} &\min \sum_{k\in G_0\setminus I} |\delta_k|, \quad \text{s.t.} \\ &\mathbf{A}\boldsymbol{\delta} = \mathbf{0}, \quad \|\boldsymbol{\delta}\|_1 = 1 \\ &\delta_k s_k^* > 0 \quad \text{for } k \in I, \qquad \delta_k s_k^* \le 0 \quad \text{for } k \in G\setminus I. \end{aligned} \tag{20}$$

For $k \in G_0 \setminus G$, define $\delta_k = u_k - v_k$, where $u_k, v_k \ge 0$. Denote $\mathbf{A}_G$ the submatrix composed of all columns of $\mathbf{A}$ with their column indexes being in $G, \mathbf{A}_{G^c} = \mathbf{A} \setminus \mathbf{A}_G$, and denote $\boldsymbol{\delta}_G$ the column vector composed of all entries of $\boldsymbol{\delta}$ with their indexes being in G. Then, (20) can be converted into the following optimization problem:

$$\begin{aligned} &\min \left[\sum_{k\in G\setminus I} |\delta_k| + \sum_{k\in G_0\setminus G} (u_k + v_k)\right], \quad \text{s.t.} \\ &[\mathbf{A}_G, \mathbf{A}_{G^c}, -\mathbf{A}_{G^c}]\left[\boldsymbol{\delta}_G^T, \mathbf{u}^T, \mathbf{v}^T\right]^T = \mathbf{0} \\ &\sum_{k\in G} |\delta_k| + \sum_{k\in G_0\setminus G} (u_k + v_k) = 1 \\ &\delta_k s_k^* > 0, \quad \text{for } k \in I \\ &\delta_k s_k^* \le 0, \quad \text{for } k \in G\setminus I \\ &\mathbf{u}, \mathbf{v} \ge 0. \end{aligned} \tag{21}$$

Furthermore, (21) can be converted into the following optimization problem:

$$\begin{aligned} &\min \left[-\sum_{k\in G\setminus I} \operatorname{sign}(s_k^*)\,\delta_k + \sum_{k\in G_0\setminus G} (u_k + v_k)\right], \quad \text{s.t.} \\ &[\mathbf{A}_G, \mathbf{A}_{G^c}, -\mathbf{A}_{G^c}]\left[\boldsymbol{\delta}_G^T, \mathbf{u}^T, \mathbf{v}^T\right]^T = \mathbf{0} \\ &\sum_{k\in I} \operatorname{sign}(s_k^*)\,\delta_k - \sum_{k\in G\setminus I} \operatorname{sign}(s_k^*)\,\delta_k + \sum_{k\in G_0\setminus G} (u_k + v_k) = 1 \\ &\operatorname{sign}(s_k^*)\,\delta_k > 0, \quad \text{for } k \in I \\ &\operatorname{sign}(s_k^*)\,\delta_k \le 0, \quad \text{for } k \in G\setminus I \\ &\mathbf{u}, \mathbf{v} \ge 0. \end{aligned} \tag{22}$$

Noting that the sign function $\operatorname{sign}(s_k^*)$ is known, thus (22) is a standard linear programming problem.

Using the fact that an optimal solution of (22) satisfies $u_k v_k = 0$, for $k \in G_0 \setminus G$, we can prove that the two optimization problems (14) and (22) are equivalent to each other.

For a given or estimated mixing matrix, we consider three cases and estimate the probabilities of recoverability in the following.

Case a) is the number of nonzero entries of a source column vector $\mathbf{s}^*$ is fixed.

We estimate the conditional probability $p_j = P\{\mathbf{s}^* = \mathbf{s}_1/\|\mathbf{s}^*\|_0 = j\}$, where $j = 0, 1, \ldots, m$.

From Theorem 1, we have $p_0 = p_1 = 1$. It is well known that $\mathbf{s}_1$ has at most n nonzero entries; thus $\mathbf{s}^*$ is not equal to $\mathbf{s}_1$ when $n < k \le m$, that is, $p_j = 0$ for $j = n+1, \ldots, m$.

Suppose that $\mathbf{s}^*$ has $j(2 \le j \le n)$ nonzero entries; then, the index set of its nonzero entries is G. Using $\mathbf{s}^*$, we define a sign column vector $\mathbf{t} \in R^m$ in $G, t_k = \text{sign}(s_k^*), k = 1, \ldots, m$. Note that $t_k = 0$ if $s_k^* = 0$.

It follows from Theorem 1 and Corollary 1 that the recoverability of a source vector $\mathbf{s}^*$ is only related to the index set and sign of its nonzero entries, not to the amplitudes of its nonzero entries. Thus, the recoverability of $\mathbf{s}^*$ is equivalent to that of $\mathbf{t}$.

For a given index set G_k^j and a sign column vector $\mathbf{t}$ in G_k^j, if for any nonnull subset $I \in G_k^j$, the optimal value of (14) is less than $(1/2)$, i.e., the optimal value of (22) is greater than $(1/2)$, then $\mathbf{t}$ can be recovered by solving the linear programming problem (P_1).

Noting that there are 2^j sign column vectors in G_k^j, suppose that there are q_k^j sign column vectors that can be recovered. Then, $(q_k^j/2^j)$ is the probability that a source vector $\mathbf{s}^*$, with its nonzero entry index set being G_k^j, can be recovered by solving (P_1). Furthermore, we have

$$p_j = \sum_{k=1}^{C_m^j} \frac{q_k^j}{2^j} \frac{1}{C_m^j} = \frac{1}{2^j C_m^j} \sum_{k=1}^{C_m^j} q_k^j \tag{23}$$

where $j = 2, \ldots, n$.

Remark 3: When the mixing matrix is given or estimated, then the key number q_k^j in (23) can be determined by checking whether the sufficient and necessary condition (14) is satisfied for all sign-column vectors.

Case b) is as follows: For any source vector $\mathbf{s}^* \in R^m$, suppose that the probabilities

$$P(s_k^* = 0) = \alpha, \quad P(s_k^* \ne 0) = 1 - \alpha, \quad k = 1, \ldots, m. \tag{24}$$

We can find that the probability that $\mathbf{s}^*$ has j nonzero entries is $C_m^j (1-\alpha)^j \alpha^{(m-j)}$. Next, we estimate the probability $\text{pro}(\alpha) = P(\mathbf{s}^* = \mathbf{s}_1)$.

Noting that $p_j = P\{\mathbf{s}^* = \mathbf{s}_1 / \|\mathbf{s}^*\|_0 = j\}$, we have

$$\begin{aligned} \text{pro}(\alpha) &= P(\mathbf{s}^* = \mathbf{s}_1) \\ &= P(\mathbf{s}^* = \mathbf{s}_1 / \|\mathbf{s}^*\|_0 = j) P(\|\mathbf{s}^*\|_0 = j) \\ &= \sum_{j=0}^{m} C_m^j (1-\alpha)^j \alpha^{(m-j)} p_j. \end{aligned} \tag{25}$$

Case c) is as follows: All entries of the source vector $\mathbf{s}^*$ are drawn from a Laplacian distribution with probability density function $(\lambda/2)\exp(-\lambda|x|)$. This generally means that $\mathbf{s}^*$ has m nonzero entries.

Using $\mathbf{s}^*$, define a column vector $\bar{\mathbf{s}}^* \in R^m$: $\bar{s}_k^* = 0$ if $|s_k^*| < \epsilon_0, \bar{s}_k^* = s_k^*$, if $|s_k^*| \ge \epsilon_0$, where ϵ_0 is a small positive constant. Consider the optimization problem (P_1) with $\mathbf{x}^*$ replaced by $\bar{\mathbf{x}}^* = \mathbf{A}^* \bar{\mathbf{s}}^*$, and denote its l^1-norm solution $\bar{\mathbf{s}}_1$. The l^1-norm solution of (P_1) corresponding to $\mathbf{s}^*$ is still denoted as $\mathbf{s}_1$.

It follows from the robustness analysis of the l^1-norm solution in [12] that, for a given small positive constant β, if ϵ_0 is sufficiently small, then $\|\mathbf{s}_1 - \bar{\mathbf{s}}_1\|_2 < \beta$, i.e., $\mathbf{s}_1 \approx \bar{\mathbf{s}}_1$. Thus, if an entry s_k^* of $\mathbf{s}^*$ satisfies $|s_k^*| < \epsilon_0$, it can be considered to be a zero entry. Moreover

$$\begin{aligned} \|\mathbf{s}^* - \mathbf{s}_1\|_2 &\le \|\mathbf{s}^* - \bar{\mathbf{s}}^*\|_2 + \|\bar{\mathbf{s}}^* - \bar{\mathbf{s}}_1\|_2 + \|\bar{\mathbf{s}}_1 - \mathbf{s}_1\|_2 \\ &\le \sqrt{m}\epsilon_0 + \beta + \|\bar{\mathbf{s}}^* - \bar{\mathbf{s}}_1\|_2. \end{aligned} \tag{26}$$

Noting that ϵ_0 and β are small, thus if $\bar{\mathbf{s}}^*$ is equal to $\bar{\mathbf{s}}_1$, we can say that $\mathbf{s}^* \approx \mathbf{s}_1$. Hence, we have

$$\text{pro}(\lambda) = P(\|\mathbf{s}^* - \mathbf{s}_1\|_2 \le \beta_0) \approx P(\bar{\mathbf{s}}^* = \bar{\mathbf{s}}_1) \tag{27}$$

where $\beta_0 = \sqrt{m}\epsilon_0 + \beta$.

Remark 4: Because the entries of the source vector $\mathbf{s}^*$ are drawn from a Laplacian distribution, these entries are small but generally not equal to zero. Since we can see that $\mathbf{s}_1$ has at most n nonzero entries; thus, the equality $\mathbf{s}^* = \mathbf{s}_1$ does not hold generally. However, the difference $\|\mathbf{s}^* - \mathbf{s}_1\|_2$ can be very small. In this case, if $\|\mathbf{s}^* - \mathbf{s}_1\|_2 \le \beta_0$, we say that $\mathbf{s}^*$ can be recovered.

In view that all entries of $\mathbf{s}^*$ are drawn from a Laplacian distribution, it can be found that $P(|s_k^*| < \epsilon_0) = 1 - \exp(-\lambda\epsilon_0)$, denoted as α_λ. The probability that $\mathbf{s}^*$ has j entries greater than ϵ_0 is $C_m^j (1-\alpha_\lambda)^j \alpha_\lambda^{(m-j)}$, i.e., $P(\|\bar{\mathbf{s}}^*\|_0 = j) = C_m^j (1-\alpha_\lambda)^j \alpha_\lambda^{(m-j)}$. Similar to (25), we have

$$\begin{aligned} \text{pro}(\lambda) &\approx P(\bar{\mathbf{s}}^* = \bar{\mathbf{s}}_1) \\ &= P(\bar{\mathbf{s}}^* = \bar{\mathbf{s}}_1 / \|\bar{\mathbf{s}}^*\|_0 = j) P(\|\bar{\mathbf{s}}^*\|_0 = j) \\ &= \sum_{j=0}^{m} C_m^j (1-\alpha_\lambda)^j \alpha_\lambda^{(m-j)} p_j. \end{aligned} \tag{28}$$

Obviously, it is reasonable to set a small positive β_0 first, and then find the bound ϵ_0 according to β_0. However, it is very difficult to estimate precisely the bound ϵ_0 using β_0. In the second simulation (Example 2) of Section V, β_0 is set to 0.06, and ϵ_0 is set to 0.023. With these values for the two parameters, the estimated probability in (28) reflects well the true probability of recoverability for Laplacian source vectors.

In this section, based on the necessary and sufficient condition in Theorem 1, several probability estimation formulas have been derived for different cases. These probability estimates reflect the true probability that the 1-norm solution is equal to the source vector.

V. Simulation Examples

The simulation results presented in this section are divided into three categories. Example 2 is concerned with the probability estimation of recoverability of the first two cases presented in the previous section [Cases a) and b)]. Example 3 is concerned with the probability estimation of recoverability in Case c) presented in the previous section. From these simulations, we can see that the probability estimates obtained in this paper reflect well the true probabilities with regard to recoverability. In Example 4, an underdetermined BSS problem is solved using the approach developed in this paper; we also compare Algorithm 1 with the standard K-means clustering algorithm.

Example 2: Suppose that $\mathbf{A} \in R^{7\times 9}$ is given in advance. This example contains two parts in which the recoverability probability estimates (23) and (25) are considered in simulation, respectively.

I) For every index set $G_k^j \subseteq G_0 (k = 1, \ldots, C_m^j, j = 2, \ldots 7)$, we first find the number q_k^j of sign column vectors that satisfy the conditions of Theorem 1 and then we calculate the probabilities $p_2, \ldots, p_7$ using (23). Noting

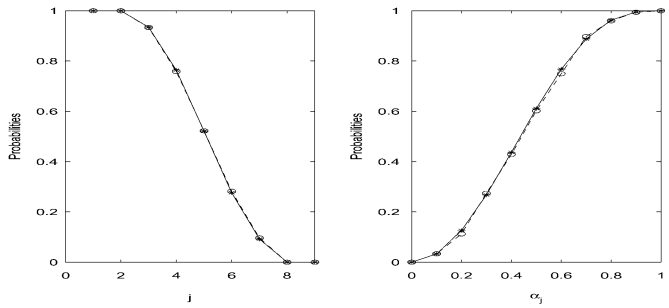

Fig. 3. Curves for estimated probabilities and true probabilities in Example 2. The left subplot is for Case a), while the right subplot is for Case b) of the consideration in Section IV.

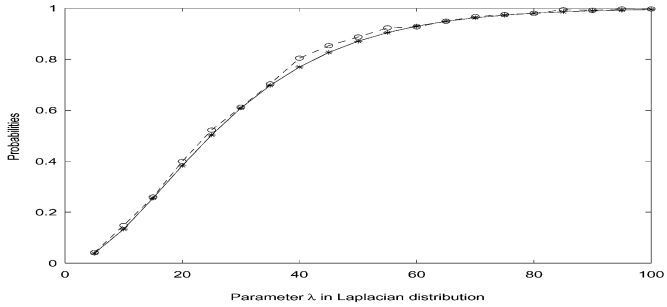

Fig. 4. Curves for estimated probabilities and true probabilities in Example 3.

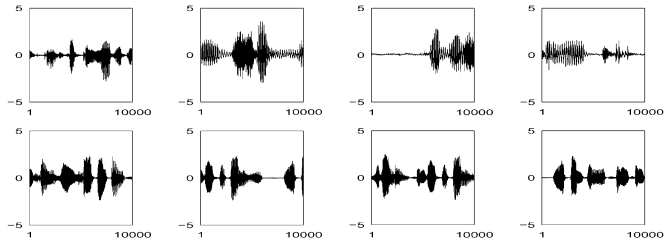

Fig. 5. Eight sources in Example 4.

that $p_1 = 1$ and $p_8 = p_9 = 0$, we thus obtain nine probabilities that are depicted by the "$*$" and the solid curve in the first subplot of Fig. 3.

Next, for every $j(j = 1, \ldots, 9)$, we take 3000 source vectors, each of which has exactly j nonzero entries. The j nonzero entries are drawn from a uniform distribution valued in $[-0.5, 0.5]$, and their indexes are also taken randomly. For each source vector, we solve the linear programming problem (P_1) and check whether the l^1-norm solution is equal to the source vector. Suppose that n_j source vectors can be recovered; thus, we obtain the ratio $\bar{p}_j = (n_j/3000)$ that reflects the fraction of recovered source vectors among the 3000 source vectors. All $\bar{p}_j, j = 1, \ldots, 9$ are depicted by the "o" and the dashed curve in the first subplot of Fig. 3. The two curves virtually overlap, which means that the probability p_j calculated by (23) reflects the recoverability of a randomly taken source vector with j nonzero entries.

II) Now we consider the probability estimate (25). For $\alpha_j = (j-1) * 0.1, (j = 1, \ldots, 11)$, we calculate the probabilities $\mathrm{pro}\,(\alpha_j)$ in (25), noting that $p_r(r = 1, \ldots, 9)$ was obtained in the part I).

Next, we define 3000 source vectors as follows:

$$s_q^* = 0, \quad \text{if } \left|\tilde{s}_q^*\right| \le \frac{\alpha_j}{2}$$
$$s_q^* = \tilde{s}_q^*, \quad \text{if } \left|\tilde{s}_q^*\right| > \frac{\alpha_j}{2} \tag{29}$$

where $\tilde{\mathbf{s}}^* \in R^9$ is drawn from a uniform distribution valued in $[-0.5, 0.5]$. From (29), we can see that the probability is α_j that each entry of a source vector is equal to zero.

As in I), for each source vector, we solve the linear programming problem (P_1) and check whether the l^1-norm solution is equal to the source vector. Suppose that m_j source vectors can be recovered; thus, we obtain the ratio $\tilde{\mathrm{pro}}(\alpha_j) = (m_j/3000)$, which reflects the fraction of recovered source vectors among the 3000 source vectors.

In the second subplot of Fig. 3, $\mathrm{pro}\,(\alpha_j), j = 1, \ldots, 11$ are depicted by "$*$" and the solid curve, while $\widetilde{\mathrm{pro}}(\alpha_j), j = 1, \ldots, 11$ are depicted by "o" and the dashed curve. We see here also that the two curves fit very well, virtually overlapping. Thus, if we know the probability that each entry of a source vector is equal to zero, using (25), we can estimate the probability that the source can be recovered by solving (P_1). Conversely, if we set a lower bound of probability that a source vector is recovered, we can obtain the corresponding lower bound of the probability that each entry is equal to zero from (25).

From this example, we can see that the probability estimates (23) and (25) reflect well the true probability that a source vector can be recovered by solving (P_1) in Cases a) and b), respectively.

Example 3: The mixing matrix $\mathbf{A} \in R^{7\times 9}$ is the same as that of Example 2. The aim of this example is to check by simulation whether the probability estimate (28) reflects the recoverability.

First, we set the parameters β_0 and ϵ_0 in Case c) of the previous section to 0.06 and 0.023, respectively. For 20 different Laplacian distributions with parameters $\lambda_j = 5j(j = 1, \ldots, 20)$, we calculate the probabilities $\mathrm{pro}\,(\lambda_j)$ using (28), noting that $\alpha_{\lambda_j} = 1 - \exp(\lambda_j \epsilon_0), p_k(k = 1, \ldots, 9)$ are from the Example 1.

Next, for every $j(j = 1, \ldots, 20), 3000$ source vectors are drawn from a Laplacian distribution with parameter λ_j. For each source vector $\mathbf{s}^*$, the linear programming problem (P_1) is solved. If the l^1-norm solution satisfies $\|\mathbf{s}_1 - \mathbf{s}^*\|_2 < \beta_0$, then we say that $\mathbf{s}^*$ can be recovered. Suppose that there are r_j source vectors recovered, we obtain the ratio $\overline{\mathrm{pro}}(\lambda_j) = (r_j/3000)$.

Fig. 4 shows the curves of $\mathrm{pro}(\lambda_j)$ (solid curve with "$*$") and $\overline{\mathrm{pro}}(\lambda_j)$ (dashed curve with "o"). Here, the two curves also overlap remarkably well. In this example, for $\beta_0 = 0.06$, the selected $\epsilon_0 = 0.023$ is reasonable. From Fig. 4, we can see that as the parameter λ in the Laplacian probability density function increases, the source vector becomes more and more sparse, and thus the probability of recoverability increases.

Example 4: Consider the linear mixing model (1), where source matrix $\mathbf{S}$ is composed of eight speech signals shown in Fig. 5. A 5×8 dimensional mixing matrix $\mathbf{A}$ is selected randomly, and every column of $\mathbf{A}$ is normalized to unit length. Only five mixtures are observable, shown in the subplots of the first row in Fig. 6.

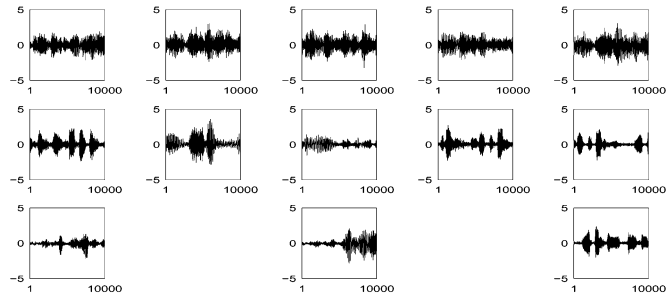

Fig. 6. Blind source separation results of Example 4. First row: Five observable mixtures. Second and third rows: Eight recovered sources.

In this example, we first use Algorithm 1 to estimate the mixing matrix and do the comparison of our algorithm with a standard K-means algorithm. Next, we estimate the sources using the linear programming algorithm.

By Algorithm 1 (where the seven-level Daubechies wavelet packets transformation is used), an estimated 5×21 dimensional mixing matrix is obtained, denoted as $\bar{\mathbf{A}}$. For each column $\mathbf{a}_i$ of $\mathbf{A}$, we choose one column of the 21 column vectors that is the closest to $\mathbf{a}_i$ (if there is a column of the 21 column vectors whose opposite vector is the closest to $\mathbf{a}_i$, we use its opposite vector). Thus, we obtain a submatrix denoted as $\tilde{\mathbf{A}}$. From the difference in (30), shown at the bottom of the page, we can find that $\tilde{\mathbf{A}}$ is very close to the original mixing matrix $\mathbf{A}$.

For comparison, we use the K-means clustering algorithm to estimate the mixing matrix $\mathbf{A}$. The wavelet packets transformation coefficient set obtained in Step 2 of Algorithm 1 is also used. After normalizing all column vectors to the unit norm, and setting the number of clusters to be 21, we use the K-means method to estimate the mixing matrix. From the resulting 21-cluster center vectors, we find a submatrix $\tilde{\tilde{\mathbf{A}}}$, as above, which is close to the mixing matrix $\mathbf{A}$. The difference between $\mathbf{A}$ and $\tilde{\tilde{\mathbf{A}}}$ is presented as follows, (Please see the equation at the bottom of the page.) From (31), shown at the bottom of the page, we can see that the estimates of the first, the fourth, and the fifth column vectors of $\mathbf{A}$ are not good (shown in bold faces). In our experience, those poorly estimated columns reflect poor recovery of corresponding sources. From (30) and (31), we can see that the estimate obtained using our method is better than that obtained using the standard K-means algorithm. in addition, if we use a smaller cluster number when performing the K-means algorithm, the difference becomes greater.

Next, we estimate the source matrix $\mathbf{S}$. Using the estimated 5×21 dimensional mixing matrix, we solve the linear programming problem (13) and obtain the TF matrix $\tilde{\mathbf{S}}$. Applying the inverse wavelet packets transformation to every row of $\tilde{\mathbf{S}}$, we obtain 21 outputs in the time domain. Among these outputs, there are 13 outputs with obviously small amplitudes, which correspond to the columns of $\bar{\mathbf{A}}$, far from the columns of $\mathbf{A}$. By removing the 13 columns of $\bar{\mathbf{A}}$ corresponding to these small outputs, a 5×8 dimensional (reduced) submatrix, still denoted as $\tilde{\mathbf{A}}$, remains. Using the matrix $\tilde{\mathbf{A}}$, we solve the linear programming problem (13) and apply the inverse wavelet packet transformation to estimate the source matrix again. The eight outputs are our recovered sources.

The subplots in the second and third rows of Fig. 6 show the eight estimated signals, from which we can see that all the sources have been recovered successfully.

Now, we consider a noisy model (2) with 27-dB SNR additive Gaussian noise. Using the proposed Algorithm 1, an estimated 5×13 dimensional mixing matrix is obtained, denoted as $\bar{\mathbf{A}}_n$. From $\bar{\mathbf{A}}_n$, we find a submatrix, as above, denoted as $\tilde{\mathbf{A}}_n$. From the difference in (32), shown at the bottom of the next page, we can find that $\tilde{\mathbf{A}}_n$ is very close to the true mixing matrix $\mathbf{A}$, except that one column is poorly estimated (shown in bold faces).

For comparison, we also use the K-means clustering algorithm to estimate the mixing matrix $\mathbf{A}$, where the number of clusters is set to 13. From the obtained 13 cluster-center vectors, we find a submatrix, as above, denoted as $\tilde{\tilde{\mathbf{A}}}_n$, which is close to the mixing matrix $\mathbf{A}$. The difference between $\mathbf{A}$ and $\tilde{\tilde{\mathbf{A}}}_n$ is presented as (33), shown at the bottom of the next page, from which

$$\mathbf{A} - \tilde{\mathbf{A}} = \begin{bmatrix} -0.0671 & 0.0591 & -0.0297 & 0.0240 & 0.0534 & 0.0150 & 0.0279 & 0.0301 \\ -0.0791 & 0.0028 & 0.0091 & -0.0588 & -0.0551 & 0.0142 & -0.0358 & -0.0596 \\ -0.0058 & 0.0060 & 0.0533 & -0.0886 & 0.0361 & 0.0037 & -0.0492 & -0.0054 \\ -0.0363 & -0.0177 & 0.0357 & 0.0755 & -0.0468 & -0.0037 & -0.0208 & -0.0296 \\ -0.0414 & 0.0095 & 0.0011 & -0.0178 & 0.0684 & 0.0032 & -0.0518 & 0.0254 \end{bmatrix}. \tag{30}$$

$$\mathbf{A} - \tilde{\tilde{\mathbf{A}}} = \begin{bmatrix} \mathbf{-0.0346} & -0.0056 & -0.0134 & \mathbf{-0.0320} & \mathbf{-0.0596} & -0.0086 & 0.0067 & 0.0201 \\ \mathbf{0.0568} & 0.0020 & -0.0110 & \mathbf{-0.0539} & \mathbf{0.1188} & -0.0106 & 0.0832 & -0.0121 \\ \mathbf{0.1340} & 0.0061 & 0.0109 & \mathbf{-0.1012} & \mathbf{-0.0861} & 0.0109 & 0.0517 & 0.0691 \\ \mathbf{0.0374} & 0.0077 & 0.0300 & \mathbf{0.1194} & \mathbf{-0.0672} & -0.0144 & 0.0478 & 0.0004 \\ \mathbf{0.0925} & -0.0051 & 0.0024 & \mathbf{0.0230} & \mathbf{-0.0034} & -0.0053 & 0.0263 & 0.0088 \end{bmatrix}. \tag{31}$$

we can see four columns are poorly estimated. From (32) and (33), we see that a better result is obtained using Algorithm 1.

From [12], we can see that the linear programming algorithm for estimating the source matrix is robust to noise to some degree. Using the estimated matrix $\bar{\mathbf{A}}_n$ above, we can solve the linear programming problem (13) and estimate the source matrix as in the noiseless case above. Although we have obtained satisfying estimation of sources under 27-dB SNR additive Gaussian noise in our simulation, we omit the result here due to the limit of pages.

Finally, we try using the TIFROM method to estimate the mixing matrix in the above noisy case. We find that we cannot obtain any satisfactory estimates of these columns of the mixing matrix. We first list the settings of the simulation example in [17]: three sources, including a stereo song and two voices from guitars (which are very sparse in the frequency domain), 60-dB SNR. Comparing these settings and those in our examples, we may conclude that relatively higher noise, larger source number, and wide-band sources lead to the violation of the sparsity condition of the TIFROM method. In addition, the variance operation used in the TIFROM method is not sufficient to detect the TF points precisely where only one source occurs. Thus, the TIFROM method does not work here.

VI. Concluding Remarks

In this paper, blind source separation was discussed using a two-stage sparse-representation approach. As in [8], the first stage estimates the mixing matrix, and the second stage estimates the source matrix. In general, estimating the mixing matrix and source matrix is carried out in the TF domain, since these signals are sparser than those in the time domain alone.

We presented first an algorithm for estimating the mixing matrix, which can be seen as an extension of the DUET and the TIROFM algorithms but requires less stringent condition them. We then confirmed the validity of the algorithm by performing several simulations and by comparing these results with those obtained using a standard clustering algorithm, i.e., the K-means clustering method. The sparseness condition can be considerably relaxed when using our Algorithm 1.

After the mixing matrix was estimated, we were also able to recover the source matrix using a standard linear programming algorithm. By extending the result in [5] and [7], we obtained a necessary and sufficient condition under which a source column vector can be recovered by solving a linear programming problem. Based on the established condition, several probability inequalities with regard to recoverability were established and proved easily. From these inequalities, we confirmed theoretically several intuitively obvious properties. That is, if the sparseness of sources increases, or the number of observations increases (the number of sources is fixed), then the probability of recoverability increases. If the number of sources increases (the number of sensors and the number of nonzero entries of the source vector are fixed), then the probability of recoverability will decrease. Furthermore, for a given or estimated mixing matrix, several theoretical probability estimates of recoverability were obtained in several different cases, e.g., the case in which the source entries were drawn from a Laplacian distribution. Simulation results showed that our theoretical probability estimates fit the true values very well. Finally, a blind speech source separation example was presented to illustrate the proposed approach. We would like to point out that, when using the two-step approach for blind source separation, it is not necessary to know (or estimate precisely) the number of sources in advance (see [12]).

There still exists two challenging open problems: 1) if the source column vector is given or known, all entries of the mixing matrix $\mathbf{A}$ are drawn from a probability distribution; then how does one estimate theoretically the probability of recoverability $P\{\mathbf{s}^* = \mathbf{s}_1\}$; and 2) more generally, how does one estimate the probability $P(n, m, l)$ in (15) when all entries of the unknown mixing matrix and source column vector are drawn from a probability distribution.

Appendix I
Proof of Theorem 1

1. *Necessity*: Suppose that $\mathbf{s}^* = \mathbf{s}_1$; that is, $\|\mathbf{s}^*\|_1$ is the optimal value of (P_1).

$$\mathbf{A} - \tilde{\mathbf{A}}_n = \begin{bmatrix} 0.0019 & 0.0361 & -0.0461 & \mathbf{0.0047} & 0.0146 & 0.0426 & 0.0480 & -0.0323 \\ 0.0369 & 0.0049 & -0.0919 & \mathbf{-0.1080} & 0.0085 & -0.0150 & -0.0154 & 0.0299 \\ 0.0476 & 0.0041 & 0.0018 & \mathbf{-0.1441} & -0.0396 & 0.0469 & -0.0378 & -0.0437 \\ -0.0439 & -0.0156 & 0.0505 & \mathbf{0.1553} & -0.0439 & -0.0001 & -0.0118 & -0.0187 \\ 0.0157 & 0.0113 & -0.0127 & \mathbf{-0.0307} & 0.0563 & 0.0584 & -0.0057 & 0.0632 \end{bmatrix}. \tag{32}$$

$$\mathbf{A} - \tilde{\tilde{\mathbf{A}}}_n = \begin{bmatrix} \mathbf{-0.0652} & -0.0549 & \mathbf{-0.0196} & \mathbf{0.0308} & -0.0415 & -0.0646 & \mathbf{0.0539} & 0.0194 \\ \mathbf{-0.1740} & 0.0038 & \mathbf{-0.1323} & \mathbf{-0.2262} & 0.0211 & 0.0231 & \mathbf{0.1258} & -0.0427 \\ \mathbf{-0.0667} & -0.0120 & \mathbf{-0.0368} & \mathbf{-0.1518} & -0.0062 & -0.0359 & \mathbf{0.0742} & 0.0998 \\ \mathbf{-0.0583} & 0.0135 & \mathbf{-0.0413} & \mathbf{0.1546} & -0.0302 & -0.0674 & \mathbf{0.0533} & 0.0197 \\ \mathbf{-0.1117} & -0.0060 & \mathbf{-0.0544} & \mathbf{-0.0527} & -0.0039 & -0.0553 & \mathbf{0.0721} & 0.0142 \end{bmatrix}. \tag{33}$$

For a subset $I \in F$, when (14) is solvable, there is at least a flexible solution. For any flexible solution $\boldsymbol{\delta}$ of (14), it can be checked that $\mathbf{s}^* + t\boldsymbol{\delta}$ is a solution of the constraint equation of (P_1), where $t < 0$ with sufficiently small absolute value. We have

$$\begin{aligned}
&\|\mathbf{s}^* + t\boldsymbol{\delta}\|_1 \\
&\quad = \sum_{k\in I}|s_k^* + t\delta_k| + \sum_{k\in G\setminus I}|s_k^* + t\delta_k| + \sum_{k\in G^c}|t\delta_k| \\
&\quad = \sum_{k\in I}|s_k^*| - |t|\sum_{k\in I}|\delta_k| + \sum_{k\in G\setminus I}|s_k^*| \\
&\qquad + |t|\sum_{k\in G\setminus I}|\delta_k| + |t|\sum_{k\in G^c}|\delta_k| \\
&\quad = \|\mathbf{s}^*\|_1 + |t|\left(\sum_{k\in I^c}|\delta_k| - \sum_{k\in I}|\delta_k|\right) > \|\mathbf{s}^*\|_1. \quad (34)
\end{aligned}$$

Thus

$$\sum_{k\in I^c}|\delta_k| - \sum_{k\in I}|\delta_k| > 0. \quad (35)$$

It follows from $\|\boldsymbol{\delta}\|_1 = 1$ and (35) that $\sum_{k\in I}|\delta_k| < (1/2)$.

The necessity is proved.

Sufficiency: Suppose that $\mathbf{s}$ is a solution of the constraint equation in (P_1); then, $\mathbf{s}$ can be rewritten as

$$\mathbf{s} = \mathbf{s}^* + t^*\boldsymbol{\delta} \quad (36)$$

where $\boldsymbol{\delta} = (\mathbf{s}^* - \mathbf{s})/(\|\mathbf{s}^* - \mathbf{s}\|_1), t^* = -\|\mathbf{s}^* - \mathbf{s}\|_1$.

Now we define an index set $I \in F : I = \{k | k = i_1, \ldots, i_l, \operatorname{sign}(s_k^*) = \operatorname{sign}(\delta_k)\}$. It can be checked easily that for the defined index set $I, \boldsymbol{\delta}$ is a flexible solution of (14). From the condition of the theorem, we have $\sum_{k\in I}|\delta_k| < (1/2)$. Furthermore

$$\begin{aligned}
&\|\mathbf{s}^* + t\boldsymbol{\delta}\|_1 \\
&\quad = \sum_{k\in I}|s_k^* + t\delta_k| + \sum_{k\in G\setminus I}|s_k^* + t\delta_k| + \sum_{k\in G^c}|t\delta_k| \\
&\quad \geq \sum_{k\in I}|s_k^*| - |t|\sum_{k\in I}|\delta_k| + \sum_{k\in G\setminus I}|s_k^*| \\
&\qquad + |t|\sum_{k\in G\setminus I}|\delta_k| + |t|\sum_{k\in G^c}|\delta_k| \\
&\quad = \|\mathbf{s}^*\|_1 + |t|\left(\sum_{k\in I^c}|\delta_k| - \sum_{k\in I}|\delta_k|\right) \geq \|\mathbf{s}^*\|_1. \quad (37)
\end{aligned}$$

Note that the equality in the last step of (37) holds when $t = 0$ (i.e., $\mathbf{s} = \mathbf{s}^*$).

Thus, for any solution $\mathbf{s}$ of the constraint equation in (P_1), we have

$$\|\mathbf{s}\|_1 \geq \|\mathbf{s}^*\|_1 \quad (38)$$

and the equality holds only when $\mathbf{s}$ is equal to $\mathbf{s}^*$. Thus, $\mathbf{s}_1 = \mathbf{s}^*$. The sufficiency is proved.

2. The second and third conclusions in this theorem are obvious. Theorem 1 is proved.

APPENDIX II
PROOF OF THEOREM 2

From Theorem 1, it follows that $P(n, m, 0) = P(n, m, 1) = 1$.

Now we prove the inequalities. Suppose that $\mathbf{u}, \mathbf{v} \in R^m, \|\mathbf{u}\|_0 = l_1 < \|\mathbf{v}\|_0 = l_2$ and that all nonzero entries of $\mathbf{u}, \mathbf{v}$ are taken randomly. Set $\mathbf{x}^{(u)} = \mathbf{A}\mathbf{u}$, $\mathbf{x}^{(v)} = \mathbf{A}\mathbf{v}$. Consider the following two optimization problems:

$$\min \|\mathbf{s}\|_1 \quad \text{s.t. } \mathbf{A}\mathbf{s} = \mathbf{x}^{(u)} \quad (39)$$

$$\min \|\mathbf{s}\|_1 \quad \text{s.t. } \mathbf{A}\mathbf{s} = \mathbf{x}^{(v)} \quad (40)$$

and denote $\mathbf{s}_1^{(u)}$ and $\mathbf{s}_1^{(v)}$ the two solutions of (39) and (40), respectively.

From the definition of p_i, we have $P\{\mathbf{u} = \mathbf{s}_1^{(u)}\} = P(n, m, l_1), P\{\mathbf{v} = \mathbf{s}_1^{(v)}\} = P(n, m, l_2)$.

Denote G_v the index set of nonzero entries of $\mathbf{v}$. Take a subset I of G_v such that the cardinality $\sharp(I) = l_1$ and define a column vector $\bar{\mathbf{u}} \in R^m$ as follows:

$$\bar{u}_k = v_k, \quad \text{if } k \in I, \quad u_k = 0, \quad \text{if } k \notin I. \quad (41)$$

Consider the optimization problem

$$\min \|\mathbf{s}\|_1 \quad \text{s.t. } \mathbf{A}\mathbf{s} = \mathbf{x}^{(\bar{u})} \quad (42)$$

where $\mathbf{x}^{(\bar{u})} = \mathbf{A}\bar{\mathbf{u}}$. The solution of (42) is denoted as $\mathbf{s}_1^{(\bar{u})}$.

Since all nonzero entries of $\bar{\mathbf{u}}$ are from the nonzero entries of $\mathbf{v}$ taken randomly, $P\{\bar{\mathbf{u}} = \mathbf{s}_1^{(\bar{u})}\} = P(n, m, l_1)$.

Now suppose that $\mathbf{v} = \mathbf{s}_1^{(v)}$. It follows from Theorem 1 that, $\forall I_1 \subseteq G_v$, when the following optimization problem (43) is solvable, its optimal value is less than $(1/2)$

$$\begin{aligned}
&\max \sum_{k\in I_1}|\delta_k|, \quad \text{s.t.} \\
&\mathbf{A}\boldsymbol{\delta} = 0, \quad \|\boldsymbol{\delta}\|_1 = 1 \\
&\delta_k v_k > 0 \quad \text{for } k \in I_1, \quad \delta_k v_k \leq 0 \quad \text{for } k \in G_v \setminus I_1. \quad (43)
\end{aligned}$$

$\forall I_2 \subseteq I$, which is the index set of nonzero entries of $\bar{\mathbf{u}}$, consider the following optimization problem:

$$\begin{aligned}
&\max \sum_{k\in I_2}|\delta_k|, \quad \text{s.t.} \\
&\mathbf{A}\boldsymbol{\delta} = 0, \quad \|\boldsymbol{\delta}\|_1 = 1 \\
&\delta_k \bar{u}_k > 0 \quad \text{for } k \in I_2 \\
&\delta_k \bar{u}_k \leq 0 \quad \text{for } k \in I \setminus I_2. \quad (44)
\end{aligned}$$

Define an index set $\tilde{I}_2 \subseteq G_v$ such that it contains the indexes of all nonzero entries of $\mathbf{v}$ each of which has the same sign as a corresponding nonzero entry of δ. In view that I is also a subset of G_v and that all nonzero entries of $\bar{\mathbf{u}}$ are taken from $\mathbf{v}$, a flexible solution δ of (44) for the index set I_2 is also a flexible solution of (43) with the index set $\tilde{I}_2$ substituting for I_1. Furthermore, from that $I, \tilde{I}_2 \subseteq G_v$, and that $\bar{u}_k = v_k$ when $k \in I$, it is not difficult to find that $I_2 \subseteq \tilde{I}_2$. Hence, $\sum_{k\in I_2}|\delta_k| \leq \sum_{k\in\tilde{I}_2}|\delta_k| < (1/2)$, where the second inequality is from the assumption of $\mathbf{v} = \mathbf{s}_1^{(v)}$. From Theorem 1, we have $\bar{\mathbf{u}} = \mathbf{s}_1^{(\bar{u})}$. We can conclude that the equality $\mathbf{v} = \mathbf{s}_1^{(v)}$ implies $\bar{\mathbf{u}} = \mathbf{s}_1^{(\bar{u})}$.

Thus, $P\{\mathbf{v} = \mathbf{s}_1^{(v)}\} \leq P\{\bar{\mathbf{u}} = \mathbf{s}_1^{(\bar{u})}\}$, i.e., $P(n, m, l_1) \geq P(n, m, l_2)$ for $l_1 < l_2$. Theorem 2 is proved.

APPENDIX III
PROOF OF THEOREM 3

From $\{1,\ldots,m_2\}$, choose an index set $\{i_1,\ldots,i_{m_1}\}$ that contains all indexes of nonzero entries of $\mathbf{r}^*$. Selecting the m_1 columns of $\mathbf{A}_2$ with indexes $i_1,\ldots,i_{m_1}$, we obtain a $n\times m_1$ dimensional matrix denoted as $\bar{\mathbf{A}}_1$. Also selecting the m_1 entries of $\mathbf{r}^*$ with indexes $i_1,\ldots,i_{m_1}$, we obtain an m_1 dimensional column vector denoted as $\bar{\mathbf{q}}^*$.

Consider the following optimization problem:

$$\min \|\mathbf{q}\|_1 \quad \text{s.t. } \bar{\mathbf{A}}_1\mathbf{q}=\bar{\mathbf{x}}_1 \tag{45}$$

where $\bar{\mathbf{x}}_1=\bar{\mathbf{A}}_1\bar{\mathbf{q}}^*$, and the solution of (45) is denoted as $\bar{\mathbf{q}}_1$.

In view that $\bar{\mathbf{A}}_1$ is a $n\times m_1$-dimensional submatrix of $\mathbf{A}_2$, which is taken randomly from a distribution, and that all nonzero entries of $\bar{\mathbf{q}}^*$ are the same as those of $\mathbf{r}^*$, also taken randomly, we have $P\{\bar{\mathbf{q}}^*=\bar{\mathbf{q}}_1\}=P\{\mathbf{q}^*=\mathbf{q}_1\}$.

Now suppose that $\mathbf{r}^*=\mathbf{r}_1$, and denote G_{r^*} the index set of all nonzero entries of $\mathbf{r}^*$. It follows from Theorem 1 that, $\forall I_3\subseteq G_{r^*}$, when the following optimization problem (46) is solvable, its the optimal value is less than $(1/2)$

$$\begin{gathered}\max\sum_{k\in I_3}|\delta_k|,\quad \text{s.t.}\\ \mathbf{A}_2\boldsymbol{\delta}=0,\quad \|\boldsymbol{\delta}\|_1=1\\ \delta_k r_k^*>0\quad \text{for } k\in I_3,\quad \delta_k r_k^*\le 0\quad \text{for } k\in G_{r^*}\setminus I_3.\end{gathered}\tag{46}$$

For the index I_3 above, consider the following optimization problem:

$$\begin{gathered}\max\sum_{k\in I_3}|\bar{\delta}_k|,\quad \text{s.t.}\\ \bar{\mathbf{A}}_1\bar{\boldsymbol{\delta}}=0,\quad \|\bar{\boldsymbol{\delta}}\|_1=1\\ \bar{\delta}_k\bar{q}_k^*>0\quad \text{for } k\in I_3,\quad \bar{\delta}_k\bar{q}_k^*\le 0\quad \text{for } k\in G_{r^*}\setminus I_3.\end{gathered}\tag{47}$$

When (47) is solvable, it is not difficult to find that a flexible solution $\bar{\delta}$ of (47) can become a flexible solution of (46) by adding m_2-m_1 zero entries with corresponding indices. Thus, the optimal value of (46) is greater than or equal to that of (47), and the optimal value of (47) is less than $(1/2)$, given the assumption of $\mathbf{r}^*=\mathbf{r}_1$. From Theorem 1, we have $\bar{\mathbf{q}}^*=\bar{\mathbf{q}}_1$. In conclusion, the equality $\mathbf{r}^*=\mathbf{r}_1$ implies that $\bar{\mathbf{q}}^*=\bar{\mathbf{q}}_1$. Thus, $P\{\mathbf{r}^*=\mathbf{r}_1\}\le P\{\bar{\mathbf{q}}^*=\bar{\mathbf{q}}_1\}=P\{\mathbf{q}^*=\mathbf{q}_1\}$. Theorem 3 is proved.

REFERENCES

[1] S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 33–61, 1998.

[2] B. A. Olshausen, P. Sallee, and M. S. Lewicki, "Learning sparse image codes using a wavelet pyramid architecture," *Advances in Neural Information Processing Systems 13*, pp. 887–893, 2001.

[3] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Comput.*, vol. 12, no. 2, pp. 337–365, 2000.

[4] K. K. Delgado, J. F. Murray, B. D. Rao, K. Engan, T. W. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural Comput.*, vol. 15, pp. 349–396, 2003.

[5] R. Gribonval and M. Nielsen, "Sparse representations in unions of bases," *IEEE Trans. Inform. Theory*, vol. 49, no. 12, pp. 3320–3325, Dec. 2003.

[6] J. A. Tropp, A. C. Gilbert, S. Muthukrishnan, and M. J. Strauss, "Improved sparse approximation over quasi-incoherent dictionaries," presented at the 2003 IEEE Int. Conf. Image Processing, Barcelona, Spain, Sep. 2003.

[7] D. L. Donoho and M. Elad, "Maximal sparsity representation via l^1 minimization," in *Proc. National Academy Science*, vol. 100, 2003, pp. 2197–2202.

[8] P. Bofill and M. Zibulevsky, "Underdetermined blind source separation using sparse representations," *Signal Process.*, vol. 81, no. 11, pp. 2353–2362, 2001.

[9] M. Zibulevsky and B. A. Pearlmutter, "Blind Source Separation by Sparse Decomposition," *Neural Comput.*, vol. 13, no. 4, pp. 863–882, 2001.

[10] T. W. Lee, M. S. Lewicki, M. Girolami, and T. J. Sejnowski, "Blind source separation of more sources than mixtures using overcomplete representations," *IEEE Signal Process. Lett.*, vol. 6, no. 4, pp. 87–90, Apr. 1999.

[11] M. Girolami, "A variational method for learning sparse and overcomplete representations," *Neural Comput.*, vol. 13, no. 11, pp. 2517–1532, 2001.

[12] Y. Q. Li, A. Cichocki, and S. Amari, "Analysis of sparse representation and blind source separation," *Neural Comput.*, vol. 16, pp. 1193–1234, 2004.

[13] J. J. Fuchs, "On sparse representations in arbitrary redundant bases," *IEEE Trans. Inform. Theory*, vol. 50, no. 6, pp. 1341–1344, Jun. 2004.

[14] D. M. Malioutov, M. Cetin, and A. S. Willsky, "Optimal sparse representation in general overcomplete bases," presented at the IEEE Int. Conf. Acoustics Speech, Signal Processing (ICASSP), May 17–21, 2004.

[15] A. Jourjine, S. Rickard, and O. Yilmaz, "Blind separation of disjoint orthogonal signals: Demixing N sources from 2 mixtures," in *Proc. 2000 IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP),*, vol. 5, Istanbul, Turkey, 2000, pp. 2985–2988.

[16] O. Yilmaz and S. Rickard, "Blind separation of speech mixtures via time-frequency masking," *IEEE Trans. Signal Process.*, vol. 52, no. 7, pp. 1830–1847, Jul. 2004.

[17] F. Abrard, Y. Deville, and P. White, "From blind source separation to blind source cancellation in the underdetermined case: A new approach based on time-frequency analysis," in *Proc. 3rd Int. Conf. Independent Component Analysis Signal Separation (ICA)*, San Diego, CA, 2001, pp. 734–739.

[18] F. Abrard and Y. Deville, "Blind separation of dependent sources using the time-frequency ratio of mixtures approach," presented at the 7th Int. Symp. Signal Processing Applications (ISSPA), Paris, France, Jul., 1–4 2003.

[19] ——, "A time- frequency blind signal separation method applicable to underdetermined mixtures of dependent sources," *Signal Process.*, vol. 85, no. 7, pp. 1389–1403, Jul. 2005.

Yuanqing Li was born in Hunan Province, China, in 1966. He received the B.S. degree in applied mathematics from Wuhan University, Wuhan, China, in 1988, the M.S. degree in applied mathematics from South China Normal University, Guangzhou, China, in 1994, and the Ph.D. degree in control theory and applications from South China University of Technology, Guangzhou, China, in 1997.

Since 1997, he has been with South China University of Technology, where he became a Full Professor in 2004. He spent a few years at the Laboratory for Advanced Brain Signal Processing, RIKEN Brain Science Institute, Saitama, Japan, as a Researcher. Presently, he is an Associate Lead Scientist in the Institute of Infocomm Research, Singapore. He is the author or coauthor of more than 50 scientific papers in journals and conference proceedings. His research interests include automatic control, blind signal processing, neural networks, brain–computer interface, neural coding, and neural decoding.

Shun-Ichi Amari (F'94) was born in Tokyo, Japan, on January 3, 1936. He received the Dr.Eng. degree in mathematical engineering from the Graduate School of the University of Tokyo, Tokyo, Japan, in 1963.

He worked as an Associate Professor at Kyushu University and the University of Tokyo, and then as a Full Professor at the University of Tokyo, where he is now Professor-Emeritus. He currently serves as the Director of the RIKEN Brain Science Institute. He has been engaged in research in wide areas of mathematical engineering, such as topological network theory, differential geometry of continuum mechanics, pattern recognition, and information sciences. In particular, he has devoted himself to mathematical foundations of neural network theory, including statistical neurodynamics, dynamical theory of neural fields, associative memory, self-organization, and general learning theory. Another main subject of his research is information geometry initiated by himself, which applies modern differential geometry to statistical inference, information theory, control theory, stochastic reasoning, and neural networks, providing a new powerful method of information sciences and probability theory.

Dr. Amari is President of Institute of Electronics, Information and Communication Engineers (IEICE) of Japan and past President of the International Neural Networks Society. He received the Emanuel A. Piore Award and the Neural Networks Pioneer Award from the IEEE, the Japan Academy Award, the C&C award, among many others.

Andrzej Cichocki (M'96) was born in Poland. He received the M.Sc. degree (with honors), the Ph.D. degree, and the Habilitate Doctorate (Dr.Sc.) degree, all in electrical engineering, from the Warsaw University of Technology, Warsaw, Poland, in 1972, 1975, and 1982, respectively.

Since 1972, he has been with the Institute of Theory of Electrical Engineering and Electrical Measurements at the Warsaw University of Technology, where he became a Full Professor in 1991. He spent a few years as an Alexander Humboldt Research Fellow and a Guest Professor at University Erlangen-Nuernberg, Germany, working in the areas of very large scale integration of electronic circuits, artificial neural networks, and optimization. He conducted and realized several successful research projects. From 1996 to 1999, he was working as a Team Leader of the Laboratory for Artificial Brain Systems, at the Frontier Research Program RIKEN (Japan), in the Brain Information Processing Group, and since 1999 he has been head of the Laboratory for Advanced Brain Signal Processing in the RIKEN Brain Science Institute. He is the coauthor of three books: *Adaptive Blind Signal and Image Processing* (New York: Wiley, 2003) with Prof. S.-I. Amari), *MOS Switched-Capacitor and Continuous-Time Integrated Circuits and Systems* (New York: Springer, 1989), and *Neural Networks for Optimization and Signal Processing* (New York: Wiley and Teubner Verlag, 1993–1994), both with Prof. R. Unbehauen). Two of his books have been translated to Chinese and other languages. He is also coauthor of more than one 150 papers. His current research interests include biomedical signal and image processing (especially blind signal/image processing), neural networks and their applications, learning theory and robust algorithms, generalization and extensions of independent and principal component analysis, optimization problems and nonlinear circuits, and systems theory and their applications.

Dr. Cichocki is a member of several international Scientific Committees and has been the Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS since January 1998.

Daniel W. C. Ho (SM'03) received the B.Sc., M.Sc., and Ph.D. degrees in mathematics (first-class) from the University of Salford, U.K., in 1980, 1982, and 1986, respectively.

From 1985 to 1988, he was a Research Fellow in the Industrial Control Unit, University of Strathclyde, Glasgow, U.K. In 1989, he joined the Department of Mathematics, City University of Hong Kong, where he is currently an Associate Professor. His research interests include H-infinity control theory, robust pole assignment problems, adaptive neural wavelet identification, and nonlinear control theory.

Dr. Ho is now serving as an Associate Editor for the *Asian Journal of Control*.

Shengli Xie (M'01–SM'02) was born in Hubei Province, China, in 1958. He received the M.S. degree in mathematics from Central China Normal University, Wuhan, China, in 1992 and the Ph.D. degree in control theory and applications from South China University of Technology, Guangzhou, China, in 1997.

He is presently a Full Professor with the South China University of Technology and a vice head of the Institute of Automation and Radio Engineering. His research interests include automatic control and blind signal processing. He is the author or coauthor of two books and more than 70 scientific papers in journals and conference proceedings.