Published in NeuroComputing (Elsevier), vol. 69, pp. 1458-1466, August 2006.

Sparse ICA via cluster-wise PCA

Massoud Babaie-Zadeh[a,1] and Christian Jutten[b,1]
Ali Mansour[c]

[a]*Advanced Communications Research Institute (ACRI), Electrical Engineering Department, Sharif University of Technology, Tehran, Iran*

[b]*Laboratory of images and signals (CNRS UMR 5083, INPG, UJF), Grenoble, France*

[c]*E3I2, ENSIETA, Brest, France*

Abstract

In this paper, it is shown that Independent Component Analysis (ICA) of sparse signals (sparse ICA) can be seen as a cluster-wise Principal Component Analysis (PCA). Consequently, Sparse ICA may be done by a combination of a clustering algorithm and PCA. For the clustering part, we use, in this paper, an algorithm inspired from K-means. The final algorithm is easy to implement for any number of sources. Experiment results points out the good performance of the method, whose the main restriction is to request an exponential growing of the sample number as the number of sources increases.

1 Introduction

Blind Source Separation (BSS) consists in retrieving unknown statistically independent signals from their mixtures, assuming there is no information either about the original source signals, or about the mixing system (hence the term *Blind*). Let $\mathbf{s}(t) \triangleq (s_1(t), \ldots, s_N(t))^T$ be the vector of unknown source signals (assumed to be zero-mean and statistically independent), and $\mathbf{x}(t) \triangleq (x_1(t), \ldots, x_N(t))^T$ be the vector of observed signals (in this paper, the number of observations and sources are assumed to be equal). Then, for linear instantaneous mixtures $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$, where $\mathbf{A}$ is the $N \times N$ (unknown)

[1] This work has been partially funded by Sharif University of Technology, by French Embassy in Tehran, and by Center for International Research and Collaboration (ISMO).

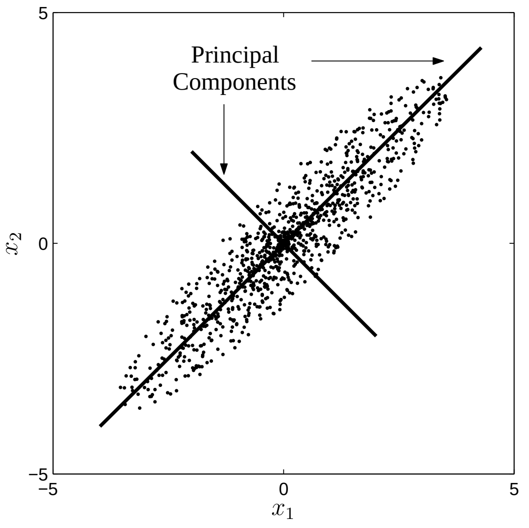

Fig. 1. Principal Components of a set of (two-dimensional) points.

'mixing matrix'. The problem is then to estimate the source vector $\mathbf{s}(t)$ only by knowing the observation vector $\mathbf{x}(t)$.

Since the only information about the source signals is their statistical independence, an idea for retrieving them is to find a 'separating matrix' $\mathbf{B}$ that transforms again the observations into independent signals. In other words, $\mathbf{B}$ is calculated in such a way that the output vector $\mathbf{y} \triangleq \mathbf{Bx}$ has independent components. This approach, which is usually called Independent Component Analysis (ICA), has been shown [1] to retrieve the source signals up to a scale and a permutation indeterminacy (*i.e.* the energies of the sources and their order cannot be restored).

On the other hand, Principal Component Analysis (PCA) is a technique to transform a random vector to another random vector with decorrelated components. Let $\mathbf{R_x} \triangleq E\left\{\mathbf{x}\,\mathbf{x}^T\right\}$ be the correlation matrix of the zero-mean random vector $\mathbf{x}$. Moreover, let $\lambda_i,\ i = 1, \dots, N$ be the eigenvalues of $\mathbf{R_x}$ corresponding to (orthonormal) eigenvectors $\mathbf{e}_i,\ i = 1, \dots, N$. Now, if:

$$\mathbf{B} = \mathbf{E}^T \tag{1}$$

where $\mathbf{E} \triangleq [\mathbf{e}_1, \dots, \mathbf{e}_N]$, then it can be easily verified that the covariance matrix of $\mathbf{y} = \mathbf{Bx}$ is diagonal. More precisely, $\mathbf{R_y} = \mathbf{\Lambda}$, where $\mathbf{R_y}$ is the correlation matrix of $\mathbf{y}$ and $\mathbf{\Lambda} \triangleq \mathrm{diag}(\lambda_1, \dots, \lambda_N)$. In other words, the components of $\mathbf{y}$ (called the principal components of $\mathbf{x}$) are decorrelated, and their variances are $\lambda_i,\ i = 1, \dots, N$. Figure 1 shows the plot of the samples of a random vector $\mathbf{x}$ and its principal components.

It is well known that for BSS (or ICA) output independence cannot be simplified as output decorrelation (PCA) [2]. Consequently, PCA cannot be used for solving the ICA problem. However, the goal of this paper is to show that for sparse signals, ICA can be achieved by a *cluster-wise PCA*.

To state the idea more precisely, note first that from (1), each row of $\mathbf{B}$ in PCA is composed of the direction of one of the principal components. We are going to show in this paper that for sparse signals, the ICA matrix can be obtained by a clustering of observation samples, and then to take the direction of the smallest principal component (*i.e.* the principal component with the smallest variance) of each cluster as the rows of $\mathbf{B}$. Developing a clustering algorithm inspired from K-means, we will also obtain an ICA algorithm for sparse signals.

To obtain the above result, we start with the geometrical ICA algorithm [3], and then modify and extend it to sparse signals. Although the development of our approach is started form geometrical interpretations, the final algorithm (see Fig. 7) is completely algebraic. Moreover, contrary to geometrical ICA algorithm, our result and approach are easy to extend for more than two sources.

The paper is organized as follows. Section 2 reviews the geometrical source separation algorithm, and its modification for using it in separating sparse signals. Then, we will see, in Section 3, that how hyper-plane fitting can be used for sparse ICA. After reviewing, in Section 4, the Principal Component Regression (PCR) method for hyper-plane fitting, an approach for fitting N hyper-planes onto a set of data points is proposed in Section 5. Putting all together, the final algorithm is presented in Section 6. Finally, some experimental results are given in Section 7.

2 Geometrical source separation algorithm

2.1 Classical geometric algorithm

The geometrical interpretation of ICA, which results in the geometrical source separation algorithm, has been first introduced in [3]. In this approach (for 2-dimensional case), using source independence *i.e.* $p_{s_1,s_2}(s_1,s_2) = p_{s_1}(s_1)p_{s_2}(s_2)$, where p stands for the Probability Density Function (PDF), one easily sees that, for bounded sources in which there exist A_1 and A_2 such that $p_{s_1}(s_1) = 0$ for $|s_1| > A_1$ and $p_{s_2}(s_2) = 0$ for $|s_2| > A_2$, the support of $p_{s_1,s_2}(s_1,s_2)$ is the rectangular region $\{(s_1,s_2) \ | \ |s_1| \leq A_1, |s_2| \leq A_2\}$. Therefore, for bounded sources, the points (s_1,s_2) will be distributed in a rectangular region (Fig. 2-a). On the other hand, having in mind the scale indeterminacy, the mixing matrix can be assumed to be of the form (*i.e.* normalized with respect to

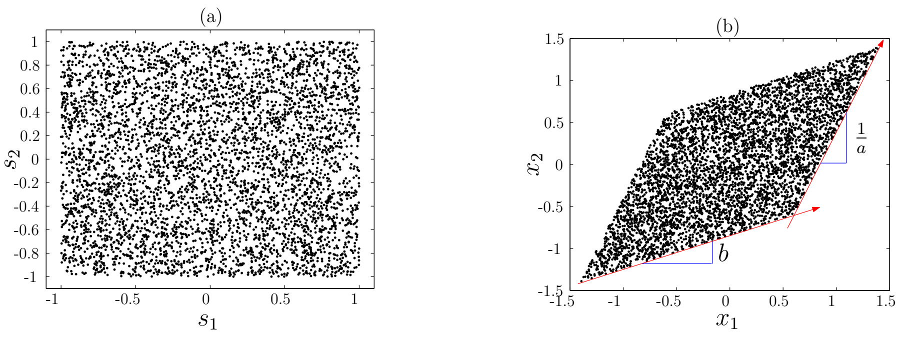

Fig. 2. Distribution of a) source samples, and b) observation samples.

diagonal elements):

$$\mathbf{A} = \begin{pmatrix} 1 & a \\ b & 1 \end{pmatrix} \tag{2}$$

Then, under the transformation $\mathbf{x} = \mathbf{A}\mathbf{s}$, the rectangular region of the s-plane (Fig. 2-a) will be transformed into a parallelogram (Fig. 2-b). It is easy to verify that the slopes of the borders of this parallelogram are $1/a$ and b. Consequently, for estimating the mixing matrix, it is sufficient to plot the observation points (x_1, x_2), which will produce a parallelogram, and then to estimate the slopes of the borders of this parallelogram, which determine a and b and hence the mixing matrix.

2.2 Geometric algorithm for sparse sources

Although the approach of the previous section constitutes a very simple BSS algorithm and provides us a geometrical interpretation of ICA, it has two restrictions: 1) it cannot be easily[2] generalized to separate more than two sources, and 2) it is suitable only for separating sources that allow a good estimation of the borders of the parallelogram (*e.g.* uniform and sinusoidal sources). Indeed, this approach cannot be directly used for separating sparse (like speech and ECG) signals. This is because the PDF of a sparse signal is mostly concentrated about zero, and hence the support of $p_{s_1 s_2}(s_1, s_2)$ is not well filled by the source samples (s_1, s_2) (see Fig. 3 for the case of two speech signals). In other words, for sparse signals, it is practically impossible to find a point on the border of the parallelogram (which would require that both sources have simultaneously high amplitude).

Although for sparse signals the borders of the parallelogram are not visible in Fig. 3, there are two visible "axes", corresponding to lines $s_1 = 0$ and $s_2 = 0$ in the s-plane (throughout the paper, it is assumed that the sources and hence the observations have zero-means). The slopes of these axes, too,

[2] The algorithm becomes very tricky.

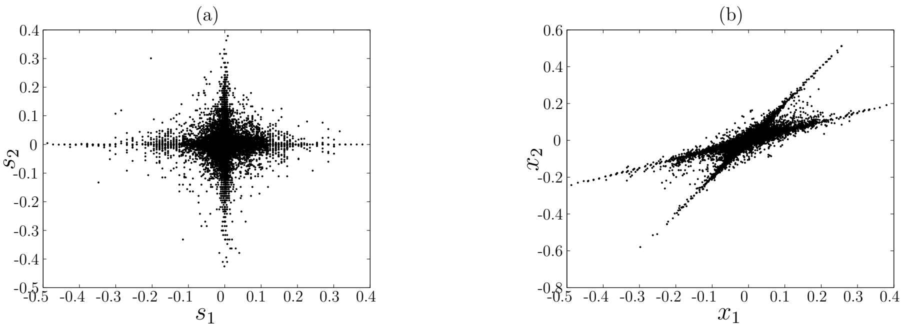

Fig. 3. Distribution of a) two speech samples, and (b) their mixtures.

determine $1/a$ and b in (2). In other words, for sparse signals, instead of finding the borders, we try to find these axes. This idea is used in [4] for separating speech signals by utilizing an "angular" histogram for estimating these axes. In their method, the resolution of the histogram cannot be too fine, since it would require too many data points, and conversely cannot be too coarse, since it would provide a too bad estimation of the mixing matrix. Moreover, their approach cannot be easily generalized to mixtures of more than two source signals.

However, we start here with another idea for finding these axes: 'fitting two straight lines' onto the scatter plot of observations. We will see, in the following sections, that this idea can be easily generalized to more than two sources. Moreover, we will see that this fitting can be done by a cluster-wise PCA, which means that, sparse ICA can be done by a cluster-wise PCA.

3 Sparse ICA by line fitting

3.1 Two dimensional case

As it is explained in the previous section, our main idea is to estimate the slopes of two axes of the scatter plot of observations (Fig. 3-b). These axes correspond to the lines $s_1 = 0$ and $s_2 = 0$ in the scatter plot of sources. The existence of these lines is a result of the sparsity of the source signals. For example, the points with small s_1 and different values for s_2 will form the axis $s_1 = 0$.

However, we do not use (2) as a model for mixing matrix, because it has two restrictions. Firstly, in this model, it is implicitly assumed that the diagonal elements of the actual mixing matrix are not zero, otherwise infinite values for a and b may be encountered (this situation corresponds to vertical axes in the x-plane). Secondly, this approach is not easy to be generalized to higher dimensions.

Instead of starting with mixing matrix (like model (2)), let consider a general "separating matrix" $\mathbf{B} = [b_{ij}]_{2\times 2}$. Under the transformation $\mathbf{y} = \mathbf{B}\mathbf{x}$, one of the axes must be transformed into $y_1 = 0$, and the other into $y_2 = 0$. In other words, for every (x_1, x_2) on the first axis:

$$\begin{pmatrix} 0 \\ y_2 \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \Rightarrow b_{11}x_1 + b_{12}x_2 = 0 \tag{3}$$

The above relation shows that the equation of the first axis in the x-plane is $b_{11}x_1+b_{12}x_2 = 0$. In a similar manner, the second axis will be $b_{21}x_1+b_{22}x_2 = 0$. Consequently, for estimating the separating matrix, the equations of the two axes must be found in the form of $\alpha_1 x_1 + \alpha_2 x_2 = 0$, and then each row of the separating matrix is composed of the coefficients of one of the axes.

It is seen that by this approach, we are not restricted to non-vertical axes (non-zero diagonal elements of the mixing matrix). Moreover, this approach can be directly used in higher dimensions, as stated below.

3.2 Higher dimensions

The approach stated above can be directly generalized to higher dimensions. For example, in the case of 3 sparse sources, the small values of s_1 with different values of s_2 and s_3 will form the plane $s_1 = 0$ in the 3-dimensional scatter plot of sources. Hence, in this 3-dimensional scatter plot, there are 3 visible planes: $s_1 = 0$, $s_2 = 0$ and $s_3 = 0$. These planes will be transformed into three main planes in the scatter plot of observations. With calculations similar to (3), it is seen that each row of the separating matrix is composed of the coefficients of one of these main planes of the form $\alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 = 0$.

Consequently, for separating the mixtures of N sparse signals from N observed signals, N (hyper-)planes of the form $\alpha_1 x_1 + \cdots + \alpha_N x_N = 0$ must be first "fitted" onto the scatter plot of observations. Then, each row of the separating matrix is the coefficients $(\alpha_1, \ldots, \alpha_N)$ of one of these (hyper-)planes.

4 Fitting a straight line (a hyper-plane) onto a set of points

To use the idea of the previous section in separating two (N) sparse sources, we need a method for fitting two lines (N hyper-planes) onto the scatter plot of observations. In this section, we consider the problem of fitting one line (one hyper-plane) onto a set of points. Then, in the following section, a method for

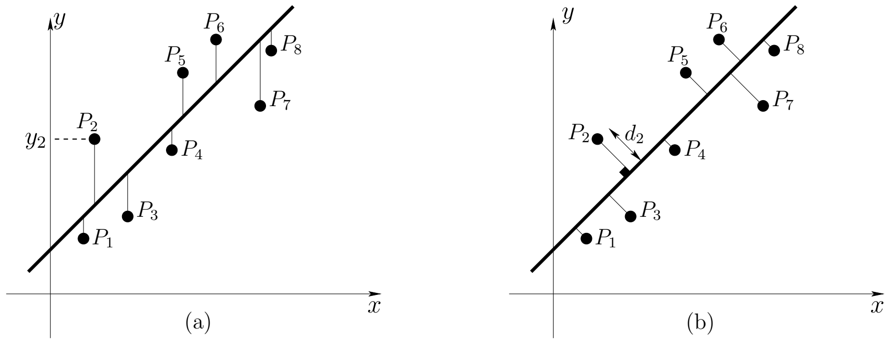

Fig. 4. a) Least squares line fitting, b) Orthogonal line fitting.

fitting two lines (N hyper-planes) will be stated based on the method of this section for fitting one line (one hyper-plane).

The approach presented in this section for line (hyper-plane) fitting has old roots in mathematics [5] and is usually called Principal Component Regression (PCR) [6].

4.1 Two-Dimensional case (line fitting)

Consider the problem of fitting a line onto K data points $(x_i, y_i)^T$, $i = 1 \dots K$. In the traditional least squares method, this is done by finding the line $y = mx+h$ which minimizes $\sum_{i=1}^{K}(y-y_i)^2 = \sum_{i=1}^{K}(mx_i+h-y_i)^2$. This is equivalent to minimizing the "vertical" distances between the line and the data points, as shown in Fig. 4-a. This technique is mainly used in linear regression analysis where there are errors in y_i's, but not in x_i's. Similarly, one could find the line $x = m'y + h'$ which minimizes $\sum_{i=1}^{K}(x - x_i)^2 = \sum_{i=1}^{K}(m'y_i + h' - x_i)^2$ and is equivalent to minimizing the "horizontal" distances between the line and the data points. Of course, changing the model and the criterion will provide different solutions.

Therefore, for fitting a line onto a set of points, a better method consists in minimizing the sum of "orthogonal distances" between the points and the line, as shown in Fig. 4-b. This approach is closer to the geometrical interpretation of 'line fitting', and provides a unique optimal solution at the least square sense.

Moreover, as discussed in the previous sections, we are seeking a line in the form $ax + by = 0$. Consequently, the best fitted line is determined by minimizing $\sum_{i=1}^{K} d_i^2$, where d_i is the orthogonal distance between the i-th point and the line, that is:

$$d_i = \frac{|ax_i + by_i|}{\sqrt{a^2 + b^2}} \tag{4}$$

It must be noted that $ax + by = 0$ is not uniquely determined by a pair (a, b), because (ka, kb) represents the same line. To get a unique solution, the coefficients are normalized such that $a^2 + b^2 = 1$. To summarize, *the line which has the best fit onto the set of points $\{(x_i, y_i), i = 1, \dots, K\}$ is the line $ax + by = 0$ which minimizes the cost function:*

$$\mathcal{C}(a,b) = \sum_{i=1}^{K} (ax_i + by_i)^2 \tag{5}$$

subject to the constraint $a^2 + b^2 = 1$.

4.2 N-Dimensional case (hyper-plane fitting)

In a similar manner, consider the problem of fitting an N-dimensional hyper-plane $\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_N x_N = 0$ onto a set of K data points $\Big\{\mathbf{x}_i = (x_1^{(i)}, x_2^{(i)}, \dots, x_N^{(i)})^T, i = 1, \dots, K\Big\}$. The best hyper-plane is obtained by minimizing $\sum_{i=1}^{K} d_i^2$, where d_i is the distance between the i-th point and the hyper-plane, that is:

$$d_i = \frac{|\alpha_1 x_1^{(i)} + \alpha_2 x_2^{(i)} + \cdots + \alpha_N x_N^{(i)}|}{\sqrt{\alpha_1^2 + \alpha_2^2 + \cdots + \alpha_N^2}} \tag{6}$$

Moreover, to uniquely determine the hyper-plane, we set $\alpha_1^2 + \alpha_2^2 + \cdots + \alpha_N^2 = 1$. In summary, *the hyper-plane which has the best fit onto the set of points $\Big\{\mathbf{x}_i = (x_1^{(i)}, x_2^{(i)}, \dots, x_N^{(i)})^T, i = 1, \dots, K\Big\}$ is the hyper-plane $\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_N x_N = 0$ which minimizes the cost function:*

$$\mathcal{C}(\alpha_1, \dots, \alpha_N) = \sum_{i=1}^{K} \left(\alpha_1 x_1^{(i)} + \cdots + \alpha_N x_N^{(i)}\right)^2 \tag{7}$$

subject to the constraint $\alpha_1^2 + \cdots + \alpha_N^2 = 1$.

4.3 Solution for the N-Dimensional case

The optimum values of $\alpha_1, \dots, \alpha_N$ are obtained by minimizing the cost function $\mathcal{C}(\alpha_1, \dots, \alpha_N)$ in (7) under the constraint $g(\alpha_1, \dots, \alpha_N) = 0$, where $g(\alpha_1, \dots, \alpha_N) \triangleq \alpha_1^2 + \cdots + \alpha_N^2 - 1$. Using Lagrange multipliers, the solution satisfies $\nabla \mathcal{C} = \lambda \nabla g$. After a few algebraic calculations, this equation is written in the matrix form:

$$\mathbf{R}_{\mathbf{x}} \boldsymbol{\alpha} = \frac{\lambda}{K} \boldsymbol{\alpha} \tag{8}$$

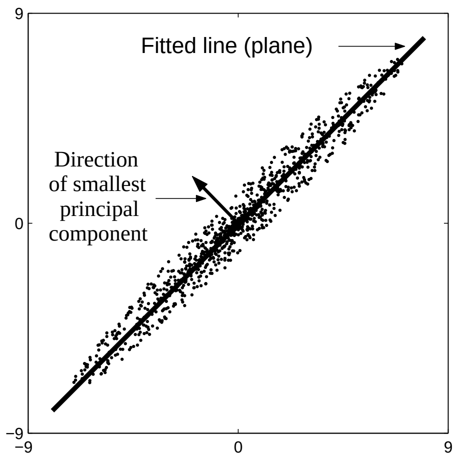

Fig. 5. PCR for two or three dimensional data. If seen as a two dimensional plot, the thick line is the fitted line, if seen as a three dimensional plot it is the fitted plane.

where $\boldsymbol{\alpha} \triangleq (\alpha_1, \dots, \alpha_N)^T$ and $\mathbf{R}_\mathbf{x} \triangleq \frac{1}{K}\sum_{i=1}^{K} \mathbf{x}_i \mathbf{x}_i^T$ is the correlation matrix of data points. Equation (8) shows that λ/K and $\boldsymbol{\alpha}$ are eigenvalue and eigenvector of the correlation matrix $\mathbf{R}_\mathbf{x}$, respectively. Moreover:

$$\mathcal{C} = \sum_{i=1}^{K} \left(\boldsymbol{\alpha}^T \mathbf{x}_i\right)^2 = \sum_{i=1}^{K} \boldsymbol{\alpha}^T \mathbf{x}_i \mathbf{x}_i^T \boldsymbol{\alpha} = K\boldsymbol{\alpha}^T \mathbf{R}_\mathbf{x} \boldsymbol{\alpha} = \lambda \boldsymbol{\alpha}^T \boldsymbol{\alpha} = \lambda$$

and hence for minimizing the cost function, λ must be minimum.

In summary, *the coefficient vector* $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_N)^T$ *of the hyper-plane* $\alpha_1 x_1 + \cdots + \alpha_N x_N = 0$ *which has the best fit onto the set of data points* $\left\{\mathbf{x}_i = (x_1^{(i)}, x_2^{(i)}, \dots, x_N^{(i)})^T, i = 1, \dots, K\right\}$ *is the eigenvector of the correlation matrix* $\mathbf{R}_x$ *which corresponds to its minimum eigenvalue.*

4.4 Relation to PCA

It is interesting to think about the conjunction of the above approach to Principal Component Analysis (PCA), or more precisely Minor Component Analysis (MCA) . Note that $\boldsymbol{\alpha}$ is the vector perpendicular to the plane $\alpha_1 x_1 + \cdots + \alpha_N x_N = 0$, and the solution of the previous section states that the optimum value of this vector is the direction of *minimum principal component* of data points, that is, the direction of minimum spread of data points. This is compatible with our heuristic interpretations of plane (line) fitting (see Fig. 5 for two or three dimensional case). In fact, the above approach for line (hyper-plane) fitting is usually called Principal Component Regression (PCR) [6].

5 Fitting 2 straight lines (N hyper-planes)

In the previous section, an approach for fitting one hyper-plane onto a set of points was presented. However, as stated in Section 3, for separating N sparse signals (by having N mixtures of them), we need to fit N hyper-planes onto observation points, not to fit just one hyper-plane.

For example, as it is seen in Fig. 3 for the two dimensional case, we need to fit two lines onto the scatter plot of observations for finding the two axes. For doing this, we can first divide the points into two clusters: the points which are closer to the first axis, and the points which are closer to the second axis. Then, a line will be fitted onto the points of each cluster. Note that a point belongs to the first cluster if it is closer to the first axis (*i.e.* its distance to the first axis is smaller than its distance to the second one). Moreover, the axis is fitted onto the points of each cluster in such a manner that the sum of squared distances of the points of that cluster to the axis be minimized. Consequently, the whole process of dividing the points into the two clusters and fitting a line onto the points of each cluster is equivalent to minimizing the following cost function:

$$\mathcal{C} = \sum_{\mathbf{x}_i \in \mathcal{S}_1} d^2(\mathbf{x}_i, l_1) + \sum_{\mathbf{x}_i \in \mathcal{S}_2} d^2(\mathbf{x}_i, l_2) \tag{9}$$

where $\mathcal{S}_j$ is the j-th cluster of points and $d(\mathbf{x}_i, l_j)$ denotes the perpendicular distance of the i-th point from the j-th line. The minimization of the above cost function is done in both dividing the points into two clusters and fitting a line onto the points of each cluster.

In a similar manner, for separating N sparse signals from N observed mixtures of them, we need to divide the observation samples into N clusters, and then to fit a hyper-plane onto the points of each cluster. This is equivalent to minimizing the following cost function:

$$\mathcal{C} = \sum_{\mathbf{x}_i \in \mathcal{S}_1} d^2(\mathbf{x}_i, l_1) + \sum_{\mathbf{x}_i \in \mathcal{S}_2} d^2(\mathbf{x}_i, l_2) + \cdots + \sum_{\mathbf{x}_i \in \mathcal{S}_N} d^2(\mathbf{x}_i, l_N) \tag{10}$$

where $\mathcal{S}_j$ is the j-th cluster of points and $d(\mathbf{x}_i, l_j)$ denotes the perpendicular distance of the i-th point from the j-th hyper-plane. The minimization is done in both fitting the hyper-plane onto the points of each cluster, and dividing the points into the clusters.

```
• Initially distribute the points into clusters $\mathcal{S}_1, \ldots, \mathcal{S}_N$ (e.g. random initialization).
• Loop:
    (1) Fit a line (hyper-plane) onto each set of points $\mathcal{S}_i$ (we call it $l_i$).
    (2) Recalculate the clusters: Let $\mathcal{S}_i$ be the set of all points which are closer to line (hyper-plane) $l_i$ than other lines (hyper-planes), that is:
            $\mathcal{S}_i = \{\mathbf{x} \mid d(\mathbf{x}, l_i) < d(\mathbf{x}, l_j), \forall j \neq i\}$
• Repeat until convergence.
```

Fig. 6. Algorithm of fitting two lines (N hyper-planes) onto a set of points.

5.1 The algorithm of fitting N hyper-planes

The problem is now how to divide the points into clusters and fit the hyper-planes onto each cluster at the same time. In fact, if the hyper-planes were known, the clusters could be easily found: the i-th cluster is composed of the points which are closer to the i-th hyper-plane than any other hyper-plane. On the other hand, if the clusters were known, it was very easy to find the hyper-planes: just use the approach of Section 4 to fit an hyper-plane onto each cluster of points.

However, in our problem, neither the clusters nor the hyper-planes are known in advance. For finding them, we propose here to iterate between these two cases. In other words, having (*e.g.* randomly) divided the points into clusters, fit a hyper-plane onto each cluster; then having the hyper-planes, re-distribute the points into clusters by taking the points closer to the i-th hyper-plane as the i-th cluster; and go on. This idea results in the algorithm of Fig. 6 for fitting N hyper-planes onto a set of points.

It can be seen that the algorithm of Fig. 6 is very similar to (and in fact inspired from) k-means (or Lloyd) algorithm for data clustering [7]. Its difference with respect to k-means is that in k-means, each cluster is mapped onto a point (point $\rightarrow$ point), but in our algorithm each cluster is mapped onto a line or hyper-plane (point $\rightarrow$ line). In the following, this algorithm will be called FITLIN.

5.2 Convergence of the algorithm

One may wander that the algorithm FITLIN converges or not. The following theorem, which is similar to a corresponding theorem for the k-means algorithm [7], insures the convergence FITLIN.

Theorem 1 *The algorithm FITLIN converges in a finite number of iterations.*

Proof: At each iteration, the cost function (10) cannot be increased. This is because in the first step (fitting hyper-planes onto the clusters) the cost function is either decreased or does not change. In the second step, too, the redistribution of the points in the clusters is done such that it decreases the cost function or does not change it. Moreover, for a finite number of points, there is a finite number of possible clusterings. Consequently, the algorithm must converge in a finite number of iterations.

5.3 Initialization

The proof of Theorem 1 shows that at each iteration of the algorithm FITLIN, the cost function cannot be increased. Consequently, the algorithm may get trapped in a local minimum. This is one of major problems of k-means, too. It depends on the initialization of the algorithm, and become more severe when the dimensionality increases.

In k-means, one approach for escaping local minima is to run the algorithm with several randomly chosen initializations, and then to take the result which produces the minimum cost-function. Here, too, we use the same idea for reducing the probability of getting trapped in a local minimum: run the algorithm FITLIN with several random initializations, and calculate the final cost function (10) after convergence. Then take the answer which results in the smallest final cost function.

6 Final Sparse ICA algorithm

The final separation algorithm is now evident. First, run the algorithm FITLIN. After convergence, there are N lines (hyper-planes) $l_i : \alpha_{i1}x_1+\cdots+\alpha_{iN}x_N = 0$, $i = 1 \dots N$. Then, the i-th row of the separating matrix is $(\alpha_{i1}, \dots, \alpha_{iN})$.

Figure 7 shows the final algorithm of this paper for blind separating sparse sources. Note that, as explained in Section 5.3, to reduce the probability of getting trapped in a local minimum, this algorithm must be run with several

```
• Initialization: randomly distribute the observation samples into
  $N$ clusters $\mathcal{S}_1, \dots, \mathcal{S}_N$.
• Loop:
(1) For $i = 1, \dots, N$:
        · Let $\boldsymbol{\alpha}_i$ be the direction of minimum principal component
          of the points in $\mathcal{S}_i$ (i.e. the eigenvector of the correlation
          matrix of the points in $\mathcal{S}_i$ which corresponds to its min-
          imum eigenvalue).
(2) Calculate the separating matrix:
        $$\mathbf{B} = [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \dots, \boldsymbol{\alpha}_N]^T$$
(3) Recalculate the clusters:
        $$\mathcal{S}_i = \{\mathbf{x} \mid \boldsymbol{\alpha}_i^T \mathbf{x} < \boldsymbol{\alpha}_j^T \mathbf{x}, \forall j \neq i\}$$
• Repeat until convergence.
• Calculate the final cost:
        $$\mathcal{C} = \sum_{\mathbf{x}_i \in S_1} (\boldsymbol{\alpha}_1^T \mathbf{x}_i)^2 + \sum_{\mathbf{x}_i \in S_2} (\boldsymbol{\alpha}_2^T \mathbf{x}_i)^2 + \cdots + \sum_{\mathbf{x}_i \in S_N} (\boldsymbol{\alpha}_N^T \mathbf{x}_i)^2$$
```

Fig. 7. Sparse ICA algorithm based on cluster-wise PCA.

random initializations, and the answer which results in minimum final cost should be taken.

7 Experimental Results

Many simulations have been conducted to separate 2, 3 or 4 sparse sources. In all these simulations, typically less than 30 iterations are needed to achieve separation. The experimental study shows that local minima depends on the initialization of the algorithm and on the number of sources (in our simulations local minima have been never encountered in separating two sources).

Here, the simulation results of 4 typical speech signals as an example of sparse signals are presented. The sparsity of speech signals is because of many low energy (silence and unvoiced) sections in it. The speech signals used in our experiments are sampled at 8KHz. In all the experiments, the diagonal elements of the mixing matrix are 1, while all other elements are 0.5. For each simulation, 10 random initializations are used, and then the matrix which creates minimum cost-function is taken as the answer.

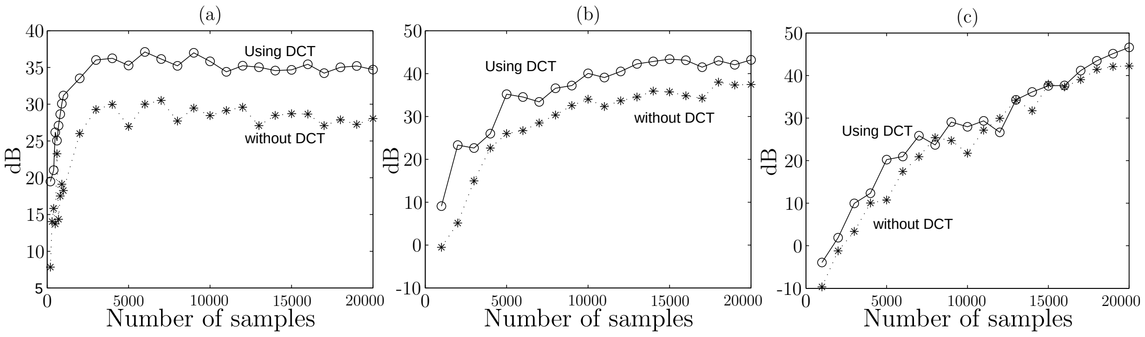

Fig. 8. Separation result in separating N speech signals, a) $N = 2$, b) $N = 3$, c) $N = 4$.

To measure the performance of the algorithm, let $\mathbf{C} \triangleq \mathbf{BA}$ be the global mixing-separating matrix. Then, we define the Signal to Noise Ratio by (assuming no permutation):

$$\mathrm{SNR}_i(\text{in dB}) \triangleq 10 \log_{10} \frac{c_{ii}^2}{\sum_{j \neq i} c_{ij}^2} \tag{11}$$

This criterion shows how much the global matrix $\mathbf{C}$ is close to the identity matrix. For having just one performance index, we take the mean of the SNR's of all outputs: $\mathrm{SNR} = \frac{1}{N} \sum_i \mathrm{SNR}_i$. To justify this, note that for calculating the performance indices, we run the algorithm with 50 different sources, and then for each output the averaged output SNR's are taken over these simulations. Consequently, the averaged SNR's (over 50 experiments) for different outputs are not very different, and taking their mean as the performance criterion seems reasonable.

To virtually create different source signals, each speech signals is shifted randomly in time (more precisely, each speech signal is shifted $128k$ samples, where k is a randomly chosen integer). This results in a completely different source scatter plot, and virtually creates a new set of source signals. Then, for each experiment, the algorithm is run 50 times (with 50 different random shifts), and the averaged SNR is calculated.

Figure 8 shows this averaged SNR's with respect to number of samples, for separating 2, 3 and 4 speech signals. In each simulation, in addition of applying the algorithm on the original observations, we applied them on the Discrete Cosine Transform (DCT) of observations, too. This is because the DCT transform increases the sparsity of speech signals, without affecting the mixing matrix, since the DCT transform is linear.

Figure 8 shows the ability of the algorithm for separating sparse signals, and points out the interest of DCT pre-processing which increases the signal sparsity: sparser the signals, better the separation. This result suggests that any linear transform improving the signal sparsity (but preserving the mixing model, since linear) can be used before the Sparse ICA algorithm for improving its performance.

It is also seen in Fig. 8 that when the number of sources increases, more data samples are required to reach a given separation quality. This is expected, because the algorithm is based on the sparsity of the sources and hyper-plane fitting. For forming the hyper-plane $s_i = 0$ in the s-plane, it is required that the source sample s_i are near zero, while all the other source samples have large values. Denoting $P(|S_i| < u) = p$ is the probability of a source to have a value smaller than u, the probability of the above situation is $p(1-p)^{(N-1)}$, which decreases exponentially with N. Consequently, it is expected that the required number of data samples for achieving a predetermined separation quality grows exponentially with N.

8 Conclusion

In this paper, we showed that sparse ICA can be seen as a cluster-wise PCA (more precisely cluster-wise MCA), and hence it can be done by a combination of a clustering algorithm and PCA. Proposing a clustering algorithm inspired from k-means, we obtained an algorithm for sparse ICA.

Although using a clustering algorithm we proposed a sparse ICA algorithm, it must be noted that the main point of the paper is not the final sparse ICA algorithm, but it is the fact that sparse ICA can be done through a cluster-wise PCA (MCA). Consequently, one may think about other clustering approaches for the clustering part, and obtaining other sparse ICA algorithm. Moreover, the problem of the current algorithm is the existence of local minima. In this paper, this problem was treated using several random initialization. Considering other clustering approaches, or modifying the initialization step of the proposed algorithm is currently under study. Finally, one shows that it is possible to improve the algorithm performance by increasing the signal sparsity: this can been done, for example, by DCT pre-processing for speech signals (as proposed in this paper) or by any other linear pre-processing which preserves the mixing matrix and increases the sparsity.

References

[1] P. Comon, Independent component analysis, a new concept?, Signal Processing 36 (3) (1994) 287–314.

[2] J.-F. Cardoso, Blind signal separation: statistical principles, Proceedings IEEE 9 (1998) 2009–2025.

[3] C. Puntonet, A. Mansour, C. Jutten, A geometrical algorithm for blind separation of sources, in: Actes du XVème Colloque GRETSI 95, Juan-Les-Pins, France, 1995, pp. 273–276.

[4] A. Prieto, B. Prieto, C. G. Puntonet, A. Cañas, P. Martín-Smith, Geometric separation of linear mixtures of sources: Application to speech signals, in: ICA99, Aussois, France, 1999, pp. 295–300.

[5] K. Pearson, On lines and planes of closest fit to systems of points in space, The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science 2 (1901) 559–572.

[6] W. F. Massy, Principal component regression in exploratory statistical research, Journal of American Statistical Association 60 (1965) 234–256.

[7] A. Gersho, R. M. Gray, Vector Quantization and signal compression, Kluwer Academic Publishers, 1992.