Available online at www.sciencedirect.com

SciVerse ScienceDirect

Speech Communication 55 (2013) 432–450

SPEECH COMMUNICATION

www.elsevier.com/locate/specom

Sparse coding with adaptive dictionary learning for underdetermined blind speech separation

Tao Xu [a,*], Wenwu Wang [a], Wei Dai [b,1]

[a] Department of Electronic Engineering, University of Surrey, Guildford GU2 7XH, United Kingdom
[b] Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, United Kingdom

Received 30 April 2012; received in revised form 4 December 2012; accepted 5 December 2012
Available online 4 January 2013

Abstract

A block-based approach coupled with adaptive dictionary learning is presented for underdetermined blind speech separation. The proposed algorithm, derived as a multi-stage method, is established by reformulating the underdetermined blind source separation problem as a sparse coding problem. First, the mixing matrix is estimated in the transform domain by a clustering algorithm. Then a dictionary is learned by an adaptive learning algorithm for which three algorithms have been tested, including the simultaneous codeword optimization (SimCO) technique that we have proposed recently. Using the estimated mixing matrix and the learned dictionary, the sources are recovered from the blocked mixtures by a signal recovery approach. The separated source components from all the blocks are concatenated to reconstruct the whole signal. The block-based operation has the advantage of improving considerably the computational efficiency of the source recovery process without degrading its separation performance. Numerical experiments are provided to show the competitive separation performance of the proposed algorithm, as compared with the state-of-the-art approaches. Using mutual coherence and sparsity index, the performance of a variety of dictionaries that are applied in underdetermined speech separation is compared and analyzed, such as the dictionaries learned from speech mixtures and ground truth speech sources, as well as those predefined by mathematical transforms such as discrete cosine transform (DCT) and short time Fourier transform (STFT).

Keywords: Underdetermined blind speech separation (BSS); Sparse representation; Signal recovery; Adaptive dictionary learning

1. Introduction

Over the past two decades, blind source separation (BSS) has attracted a lot of attention in the signal processing community, owing to its wide range of potential applications, such as in telecommunications, biomedical engineering, and speech enhancement (Hyvärinen et al., 2001; Cichocki and Amari, 2003). BSS aims to estimate the unknown sources from their observations without or with little prior knowledge about the channels through which the sources propagate to the sensors. The instantaneous model of BSS, which is the focus of this paper, can be described as:

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

where $\mathbf{A} \in R^{M\times N}$ is the unknown mixing matrix assumed to be of full row rank, $\mathbf{X} \in R^{M\times T}$ is the observed data matrix whose row vector $\mathbf{x}_i$ is the ith sensor signal having T samples at discrete time instants $t = 1, \ldots, T$, $\mathbf{S} \in R^{N\times T}$ is the unknown source matrix containing N source vectors, and $\mathbf{V} \in R^{M\times T}$ is the noise matrix containing M noise vectors. The objective of BSS is to estimate $\mathbf{S}$ from $\mathbf{X}$, without knowing $\mathbf{A}$ and $\mathbf{V}$. A classical example for BSS is the so called "cocktail party problem", where a number of people are talking simultaneously in a cocktail party, and each one can distinguish the others' speech in this sound mixing environment, but it is difficult for machines to replicate such capabilities.

* Corresponding author. Tel.: +44 1483686039/4025; fax: +44 1483686031.
E-mail addresses: t.xu@surrey.ac.uk (T. Xu), w.wang@surrey.ac.uk (W. Wang), wei.dai1@imperial.ac.uk (W. Dai).
[1] Tel.: +44 2075946333; fax: +44 2075946302.

0167-6393/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.specom.2012.12.003

Many algorithms have been successfully developed for blind source separation, especially for the exactly or over determined cases where the number of mixtures is no smaller than that of the sources. Independent component analysis (ICA) is a well-known family of BSS techniques based on the assumption that the source signals are statistically independent. However, ICA does not work in the underdetermined case, where the number of mixtures is smaller than that of the sources. Although several approaches (Makino et al., 2007) have been developed to address this problem, which are reviewed in next section, it remains an open problem, especially for speech source signals. In this underdetermined case, we propose an approach to improve the separation performance for speech signals by using sparse signal recovery with adaptive dictionary learning.

It is worth noting that the cocktail party problem is often addressed by a convolutive BSS model for which many algorithms have been published such as some recent works (Kowalski et al., 2010; Mandel et al., 2010; Jan et al., 2011; Sawada et al., 2011; Alinaghi et al., 2011). Although the convolutive BSS model is not the focus of this paper, the methods developed here can also be used in convolutive speech separation algorithms, such as many frequency domain BSS algorithms where the convolutive model is transformed into the frequency domain, leading to multiple instantaneous but complex-valued BSS problems to be solved, subject to permutation alignment and scale ambiguity correction, along the frequency channels (Smaragdis, 1998; Parra and Spence, 2000; Wang et al., 2005).

1.1. Underdetermined blind speech separation

Underdetermined blind speech separation is an ill-posed inverse problem, due to the lack of sufficient observations, i.e. the number of unknown speech sources to be separated is greater than the number of observed mixtures. Several approaches have been developed to address this problem, such as the higher order statistics based method in (Comon, 1998), the sparse representations based technique in (Zibulevsky and Pearlmutter, 2001; Bofill and Zibulevsky, 2001), the time–frequency mask based degenerate unmixing estimation technique (DUET) in (Jourjine et al., 2000; Yilmaz and Rickard, 2004), the clustering techniques in (Luo et al., 2006; Araki et al., 2007), the method combining the techniques of ICA and ideal binary mask (IBM) (Pedersen et al., 2008), and matrix and tensor decomposition based methods (Wang, 2007; Wang et al., 2008; Wang and Zou, 2008; Wang et al., 2009; Nion and Lathauwer, 2008; Cichocki et al., 2009; Comon, 2004; Tichavský and Koldovský, 2011).

The authors in (Zibulevsky and Pearlmutter, 2001) proposed a method for the selection of signal priors from a signal dictionary assuming that the sources can be sparsely represented. The sources are then estimated under the maximum a posteriori (MAP) framework. In (Bofill and Zibulevsky, 2001), a two-stage approach was developed which consists of estimating the mixing system by a clustering technique and separating sources by solving a low-dimensional linear programming problem for each of the data points. The DUET method (Jourjine et al., 2000; Yilmaz and Rickard, 2004) attempts to solve the underdetermined speech separation problem using a time–frequency masking technique based on the assumptions of W-disjoint orthogonality (WDO) between the speech signals and the short distance between the microphones. The binary mask based technique combined with a K-means clustering algorithm was presented in (Araki et al., 2007). The methods in (Hulle, 1999; Arberet et al., 2010) are also based on the clustering technique. The majority of the above work is based on sparse signal representation. Good reviews on using sparse component analysis for source separation can be found in (Gribonval and Lesage, 2006; Sudhakar, 2011).

1.2. Sparse coding

The key idea of sparse signal representation is to assume that the sources are sparse, or can be decomposed into the combination of a small number of signal components. By sparse, we mean that most values in the signal or its transformed coefficients are zero, except for a few nonzero values. These signal components are called atoms or codewords, and the collection of all the atoms is referred to as a dictionary. Finding the sparsest representation (i.e. the non-zero coefficients) which best approximates the observation is often an NP-hard problem (Donoho, 2006). A great number of works have shown that a near optimal performance can be achieved by using different relaxations for the sparsity measure, e.g. the ℓ_1 norm (Chen et al., 1999; Wang, 2008), the ℓ_p norm (Daubechies et al., 2010), and the smoothed ℓ_0 norm (Mohimani et al., 2009). Such relaxations have led to a wide range of algorithms for signal reconstruction: the basis pursuit technique based on linear programming (Chen et al., 1999), the greedy algorithm such as matching pursuit (Mallat and Zhang, 1993) and iterative thresholding (Blumensath and Davies, 2008), subspace techniques such as subspace pursuit (Dai and Milenkovic, 2009) and CoSaMP (Needell and Tropp, 2008), and regression shrinkage and selection (LASSO) (Tibshirani, 1996).

1.3. Contribution and organization

In this paper, sparse coding, based on various types of dictionaries (both learned and predefined), is used to solve the problem of underdetermined blind speech separation. In particular, we propose a novel algorithm in which the BSS model is reformulated to a sparse signal recovery model. As a result, any of the state-of-the-art sparse signal recovery algorithms could be incorporated into this model to solve the underdetermined blind speech separation problem, with various separation performance and computational efficiency. Several signal recovery algorithms have been examined in the proposed system. We then extend this approach to a multi-stage method for enhancing the separation performance by incorporating adaptive

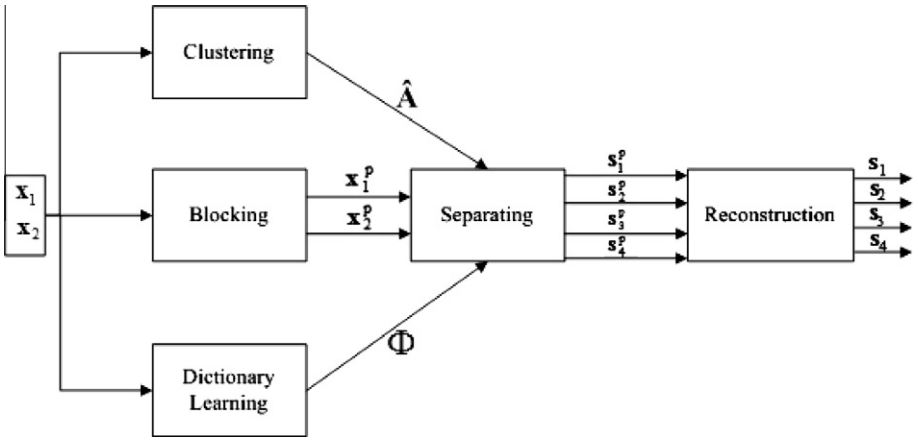

Fig. 1. The flow chart of the proposed system for separating four speech sources from two mixtures.

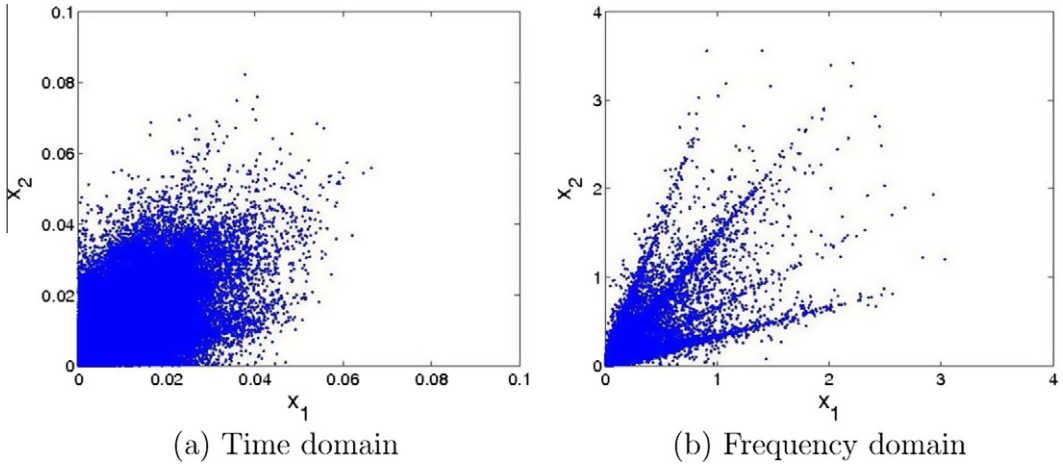

Fig. 2. An example of the scatter plots for two mixtures of four speech sources in the time (a) and frequency (b) domain. Note that, the absolute values of the mixtures and their STFT coefficients are plotted.

dictionary learning algorithms for the signal recovery and incorporating a blocking process to improve its computational efficiency. In other words, the predefined transform traditionally used is replaced by an adaptive transform containing a group of atoms trained from the speech data. Under the adaptive transform, a speech signal can be decomposed as a linear combination of only a few atoms, i.e. it has a sparse representation. This sparse representation not only captures important features from the speech data, but also has the potential to reduce the effects of noise. We will also evaluate the performance of the proposed algorithm systematically and compare it with the state-of-the-art.

The results show that the separation performance obtained by using the adaptive dictionary is more robust in noisy environments as compared with the fixed dictionary obtained by, for example, the discrete cosine transform (DCT). Among the dictionary learning algorithms compared, simultaneous codeword optimization (SimCO) (Dai et al., 2012), which we proposed recently and is used for the first time in an underdetermined speech separation application, offers the best performance as compared with others. To further improve the computational efficiency and enhance the system performance, we employ a block processing stage in the front-end of our system. The proposed algorithm will be compared with the recent methods in the source separation evaluation campaign SiSEC2008 (Vincent et al., 2009) using the same datasets and evaluation approach. Preliminary results of this work were presented in (Xu and Wang, 2009, 2010, 2011).

The remainder of the paper is organized as follows. The proposed multi-stage method with clustering, dictionary learning, blocking, separating and reconstruction stages is presented in Section 2. Experimental results are given in Section 3. Finally, conclusions and future work are summarized in Section 4.

2. The proposed multi-stage system based on sparse signal recovery and dictionary learning

For underdetermined BSS, i.e. $M < N$, one has to estimate $\mathbf{A}$ first, then the sources $\mathbf{S}$. However, in this case, even if $\mathbf{A}$ is available, the solution to $\mathbf{S}$ is not unique. To address this problem, we reformulate the BSS model into a sparse

signal recovery model with an adaptive dictionary learned from training data. As a result, the proposed method is a multi-stage procedure. To explain the concept, we omit the noise $\mathbf{V}$ when necessary in the following sections. It is worth noting that the designed algorithm works well for noisy mixtures according to our numerical results.

To demonstrate the proposed multi-stage approach, we typically consider the underdetermined case of $M = 2$ and $N = 4$ in the model (1). However, the approach can be readily extended to other underdetermined cases with various numbers of sources and mixtures. As depicted in Fig. 1, the proposed method using sparse signal recovery and dictionary learning for underdetermined blind speech separation (SDUBSS in short) is composed of the clustering stage, the dictionary learning stage, the blocking stage, the separating stage, and reconstruction stage. Note that, the dictionary learning stage can be replaced by a predefined transform, such as the DCT transform, if a fixed dictionary is applied. The segments of the signals obtained by the blocking stage will be used only in the separating and reconstruction stages, while the clustering and the dictionary learning stages are still performed for the whole signal, and $\widehat{\mathbf{A}}$ and $\mathbf{\Phi}$ obtained in these stages will be shared by all the segments in the separating stage. The details of all the stages are given in the following subsections.

2.1. Estimating the mixing matrix by clustering

In the clustering stage, we use the standard technique as in (Zibulevsky and Pearlmutter, 2001) to estimate the mixing matrix $\mathbf{A}$ by using the K-means clustering algorithm based on the short-time fourier transform (STFT) coefficients of the mixtures. Assuming the sources are sparse, i.e. ideally only one source has nonzero value at each time instant, some lines in the scatter plot of the mixtures can be clearly identified, and the number of lines should be equal to that of the columns of $\mathbf{A}$. For example, when $M = 2$, at any time instant, the point on the scatter plot of $\mathbf{x}_1$ versus $\mathbf{x}_2$ should lie on the line that can be represented by one of the column vectors in $\mathbf{A}$, as there exists only one source in this time instant. The vector of the plotted points is a product of a scalar and one of the column vectors in $\mathbf{A}$. When all the data points are plotted, some lines in the coordinate plane can be clearly identified, and the number of lines should be equal to that of the columns of $\mathbf{A}$. In practice, however, the sparseness assumption is seldom satisfied nicely, due to the observation noise in real data. The lines are usually broadened especially in the time domain, as shown in Fig. 2(a). It has been observed that the audio mixtures become sparser if they are transformed into the frequency domain. As a result, it becomes easier to observe the distributions of the data points in the scatter plot, as shown in Fig. 2(b).

Therefore, to estimate the mixing matrix, we apply the K-means algorithm to the speech data in the frequency domain obtained by the STFT. The algorithm can be summarized in Algorithm 1.

Algorithm 1. K-means algorithm for mixing matrix estimation.

Task: estimate the mixing matrix for the following stages in SDUBSS
Input: $\mathbf{X}$
Output: $\widehat{\mathbf{A}}$.

- Apply the STFT to each mixture signal in $\mathbf{X}$ to obtain a spectrogram of this mixture signal then reshape it to a vector as the coefficient vector in $\tilde{\mathbf{x}}$, i.e. the time–frequency representation of $\mathbf{X}$.
- Normalize the vectors in $\tilde{\mathbf{X}}$ to move all the points to a unit semi-circle for the K-means algorithm to be applied.
- Choose the starting points for the K-means algorithm, and divide $\tilde{\mathbf{x}}$ to four parts (equals to the number of sources) and compute the mean values of each part as the initial centers.
- Run the K-means clustering algorithm to update iteratively the four centers until convergence, and compute the column vectors of the estimated mixing matrix $\widehat{\mathbf{A}}$ as the final cluster centers.

2.2. Separating sources by sparse signal recovery

In the separating stage, with the estimated mixing matrix $\widehat{\mathbf{A}}$, we formulate the underdetermined blind speech separation problem as a sparse signal recovery problem. Eq. (1) can be expanded as:

$$\begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_M \end{pmatrix} = \begin{pmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{M1} & \cdots & a_{MN} \end{pmatrix} \begin{pmatrix} \mathbf{s}_1 \\ \vdots \\ \mathbf{s}_N \end{pmatrix} \tag{2}$$

where $\mathbf{x}_i$ $(i = 1, \ldots, M)$ are the mixtures, $\mathbf{s}_j$ $(j = 1, \ldots, N)$ are the sources, and a_{ij} is the ijth element of the mixing matrix $\mathbf{A}$. We can further write the above equation as follows,

$$\underbrace{\begin{pmatrix} x_1(1) \\ \vdots \\ x_1(T) \\ \vdots \\ \vdots \\ x_M(1) \\ \vdots \\ x_M(T) \end{pmatrix}}_{\mathbf{b}} = \underbrace{\begin{pmatrix} \Lambda_{11} & \cdots & \Lambda_{1N} \\ \vdots & \ddots & \vdots \\ \Lambda_{M1} & \cdots & \Lambda_{MN} \end{pmatrix}}_{\mathbf{M}} \underbrace{\begin{pmatrix} s_1(1) \\ \vdots \\ s_1(T) \\ \vdots \\ \vdots \\ s_N(1) \\ \vdots \\ s_N(T) \end{pmatrix}}_{\mathbf{f}} \tag{3}$$

where T is the length of the signal, $\Lambda_{ij} \in R^{T \times T}$ is a diagonal matrix whose diagonal elements are all equal to a_{ij}. Let $\mathbf{b} = \mathrm{vec}(\mathbf{X}^T)$, $\mathbf{f} = \mathrm{vec}(\mathbf{S}^T)$, where vec is an operator stacking the column vectors of a matrix into a single vector. Eq. (3) can be written in a compact form as:

$$\mathbf{b} = \mathbf{M}\mathbf{f} \tag{4}$$

The above equation can be interpreted as a sparse signal recovery problem in a compressed sensing model, in which $\mathbf{M}$ is the measurement matrix and $\mathbf{b}$ is the compressed vector of samples in $\mathbf{f}$. Therefore, a sparse representation in the transform domain can be employed for $\mathbf{f}$:

$$\mathbf{f} = \mathbf{\Phi}\mathbf{y} \tag{5}$$

where $\mathbf{\Phi}$ is a transform dictionary and $\mathbf{y}$ contains the weighting coefficients in the $\mathbf{\Phi}$ domain. Combining (4) and (5), we have

$$\mathbf{b} = \mathbf{M}\mathbf{\Phi}\mathbf{y} \tag{6}$$

In Eq. (6), if $\mathbf{y}$ is sparse, the signal $\mathbf{f}$ can be recovered from the measurement $\mathbf{b}$ using an optimization process. This indicates that source estimation in the underdetermined problem can be achieved by computing $\mathbf{y}$ in (6) using sparse signal recovery (i.e. sparse coding) methods.

2.2.1. Alternative sparse signal recovery methods

An approach for estimating $\mathbf{f}$ from $\mathbf{b} = \mathbf{M}\mathbf{f} = \mathbf{M}\mathbf{\Phi}\mathbf{y}$ is to solve the following ℓ_0 minimization problem

$$\min \|\mathbf{y}\|_0 \quad \text{s.t.} \quad \mathbf{b} = \mathbf{M}\mathbf{\Phi}\mathbf{y} \tag{7}$$

where $\|\mathbf{y}\|_0$ is the ℓ_0 norm measuring the sparseness of $\mathbf{y}$. The solution to the optimization of the above cost function is an NP-hard problem (Donoho, 2004), which is not a good choice in practice. However, it has been shown in (Chen et al., 1999) that the solution to the ℓ_0 minimization problem is essentially equivalent to the solution of the following ℓ_1 minimization problem

$$\min \|\mathbf{y}\|_1 \quad \text{s.t.} \quad \mathbf{b} = \mathbf{M}\mathbf{\Phi}\mathbf{y} \tag{8}$$

This ℓ_1 norm minimization problem corresponds to the so-called basis pursuit (BP) (Chen et al., 1999) and can be solved through linear programming. One starts from a solution to the overcomplete representation problem $\mathbf{b} = \mathbf{M}\mathbf{\Phi}\mathbf{y}^{(0)}$ then iteratively updates the coefficients while keeping $\mathbf{b} = \mathbf{M}\mathbf{\Phi}\mathbf{y}^{(k)}$, as follows. First, we choose an initial basis matrix $\mathbf{B}$ which is a square matrix having the same rank as $\mathbf{M}\mathbf{\Phi}$ and consists of the selected columns of $\mathbf{M}\mathbf{\Phi}$, i.e. the smallest possible complete dictionary. Then, we update the basis by swapping a column of $\mathbf{B}$ with an unselected column in $\mathbf{M}\mathbf{\Phi}$. When the basis cannot be further improved based on a pre-defined criterion, we reach the optimal solution. Finally, $\mathbf{y}$ can be readily computed by $\mathbf{B}^{-1}\mathbf{b}$. The vector $\mathbf{f}$ which consists of the separated sources can be obtained by simply multiplying the dictionary $\mathbf{\Phi}$ with $\mathbf{y}$ using (5).

Apart from the BP method described above, there are alternative methods for recovering the speech signals, such as matching pursuit (MP) (Mallat and Zhang, 1993). The basic idea of MP is to represent a signal as a weighted sum of atoms using Eq. (9) which involves finding the "best matching" projections of multidimensional data onto an over-complete dictionary,

$$\mathbf{b} = \sum_{i=1}^{k} y_i \mathbf{q}_{\gamma_i} + \mathbf{r}^{(k)} \tag{9}$$

where $\mathbf{r}^{(k)}$ is a residual after k iterations, and $\mathbf{q}_{\gamma_i}$ is the atom of $\mathbf{M}\mathbf{\Phi}$ that has the largest inner product with the residual. At stage i, it identifies the dictionary atom that best correlates with the residual then subtract its contribution as follows,

$$\mathbf{r}^{(i+1)} = \mathbf{r}^{(i)} - y_i \mathbf{q}_{\gamma_i} \tag{10}$$

where $y_i = \langle \mathbf{r}^{(i)}, \mathbf{q}_{\gamma_i} \rangle$, and $\langle , \rangle$ is an inner product operation. Then the process is repeated until the signal is satisfactorily decomposed.

The orthogonal matching pursuit (OMP) (Pati et al., 1993) was developed to improve the MP by projecting the signal vector to the subspace spanned by the atoms selected as in MP via the same method. However, as opposed to MP, OMP maintains full backward orthogonality of the residual at each step when updating the coefficients:

$$\mathbf{b} = \sum_{i=1}^{k} y_i \mathbf{q}_{\gamma_i} + \mathbf{r}^{(k)}, \quad \text{s.t.} \quad \langle \mathbf{r}^{(k)}, \mathbf{q}_{\gamma_i} \rangle = 0 \tag{11}$$

As proven in (Pati et al., 1993) the necessary number of iterations for OMP to converge is no greater than the number of atoms in the dictionary, while MP does not possess this property.

A least squares method L1LS with ℓ_1 regularization is described in (Kim et al., 2007a) for large scale problems. This interior-point method uses the preconditioned conjugate gradient algorithm to compute the search direction. It solves the following problem:

$$\min \|\mathbf{b} - \mathbf{M}\mathbf{\Phi}\mathbf{y}\|_2^2 + \lambda \|\mathbf{y}\|_1 \tag{12}$$

where $\|\cdot\|_2$ denotes the ℓ_2 norm and λ is the regularization parameter. The search direction is computed first, followed by setting the step size using a line search mechanism. After computing the Hessian matrix, the PCG algorithm (Demmel and Heath, 1997) is applied to update the coefficients. Note that for a "small" λ, this method will become equivalent to BP.

The performance of the signal recovery methods discussed above will be studied in Section 3. Based on the reformulation of the speech separation problem to the problem of signal recovery as shown from Eq. (3) to (6), the proposed speech separation algorithm is summarized in Algorithm 2. We would like to note that the above algorithms are just examples of many sparse signal recovery algorithms, which, including the recent method FISTA (Beck and Teboulle, 2009), can all be used for reconstructing the sources.

Algorithm 2. Separating speech sources

Task: separating speech sources from each block based on signal recovery method

Input: $\mathbf{x}_i^p,\ i=1,2,\ p=1,\ldots,P$. ($\mathbf{x}_i^p$ is one of the blocks of mixtures, which are generated from blocking stage to be discussed in Section 2.5), $\widehat{\mathbf{A}}$. $\mathbf{\Phi}$.

Output: $\mathbf{s}_j^p,\ j=1,\ldots,4,\ p=1,\ldots,P$.

Initialization: p=1.

Repeat:

- Form the measurement vector $\mathbf{b}^p$ by concatenating $\mathbf{x}_1^p$ with $\mathbf{x}_2^p$.
- Multiply $\mathbf{M}$ which is formed from $\widehat{\mathbf{A}}$ (obtained from the clustering stage i.e. the output of Algorithm 1), with the dictionary $\mathbf{\Phi}$ (which can be obtained by dictionary learning stage or a pre-defined DCT transform).
- Use the signal recovery methods such as BP, MP or L1LS to find the sparsest coefficients $\mathbf{y}^p$ from $\mathbf{M\Phi}$ and $\mathbf{b}^p$.
- Compute $\mathbf{f}^p$ according to Eq. (5).
- Compute the source vectors $\mathbf{s}_j^p$ from $\mathbf{f}^p$.
- $p=p+1$.

2.3. The adaptive dictionary learning algorithms

Sparse decompositions of a signal, however, highly rely on the degree of the fit between the data and the dictionary, which leads to another important problem, i.e. the issue of designing dictionary $\mathbf{\Phi}$. According to recent research, two main approaches are usually used: the analytical approach and the learning-based approach. In the first approach, a mathematical model of the data is given in advance so that the dictionary can be generated by fast Fourier transform (FFT), DCT, wavelet transform, etc. The second approach applies machine learning techniques to train the dictionary from a set of data so that its atoms can represent the features of the signal. Dictionary learning methods are often established on an optimization algorithm, in which an initial dictionary is given and a signal is decomposed as a linear combination of the atoms from the initial dictionary and the weighted values are a few non-zero coefficients. The atoms of the dictionary are then trained while the weighting coefficients are fixed. After that, the trained dictionary is used to compute the new weighting coefficients. The process is iterated until the most suitable dictionary is learned eventually (Aharon et al., 2006; Rubinstein et al., 2010; Jafari and Plumbley, 2011; Dai et al., 2012), based on a pre-defined criterion.

In this paper, instead of using a predefined transform such as the DCT to obtain the dictionary matrix $\mathbf{\Phi}$, dictionary learning algorithms will be applied to obtain an adaptive dictionary. The model for dictionary learning is given below.

$$\mathbf{T}=\mathbf{\Phi G} \tag{13}$$

where $\mathbf{T}$ is a matrix in which each column contains one of the training samples, $\mathbf{\Phi}$ is the dictionary obtained from the training process, and $\mathbf{G}$ is a matrix consisting of the sparse coefficient vectors. Three recent dictionary learning approaches, namely the K-SVD (Aharon et al., 2006) algorithm, the greedy adaptive dictionary algorithm (GAD) (Jafari and Plumbley, 2011), and the simultaneous codeword optimization (SimCO) algorithm (Dai et al., 2012), are used to learn the dictionary $\mathbf{\Phi}$ on the signal frames extracted from speech data (either mixtures or speech sources).

2.3.1. K-SVD

The K-SVD algorithm aims to iteratively find the best dictionary to represent the signal sample $\mathbf{t}_j$ ($j=1,\ldots,J$, where J is the number of training signals) by approximating the solution to

$$\min_{\mathbf{\Phi},\mathbf{G}}\|\mathbf{T}-\mathbf{\Phi G}\|_F^2 \quad \text{s.t.} \quad \forall j,\|\mathbf{g}_j\|_0\leqslant T_0 \tag{14}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, $\mathbf{g}_j$ ($j=1,\ldots,J$) is the jth column vector of the sparse coefficient matrix $\mathbf{G}$, and T_0 specifies the maximum number of non-zero coefficients for coding each signal sample. The K-SVD algorithm consists of a sparse-coding step and a dictionary update step. The first step is to compute the sparse coefficient vectors from the sample signals in $\mathbf{T}$ using any sparse-approximation approach such as a pursuit method based on the given dictionary. The second step is updating the atoms which are columns in the dictionary matrix to better fit the signal using the sparse representations obtained in the first step. The dictionary update is carried out for one atom each time, i.e. optimizing Eq. (14) for each atom in turn while keeping the other atoms fixed. These two steps are iteratively repeated until the convergence of the algorithm. The essential part of the dictionary update step is presented here. First, the overall representation error matrix $\mathbf{E}_l$ is computed by (15).

$$\mathbf{E}_l=\mathbf{T}-\sum_{i\neq l}\boldsymbol{\phi}_i\mathbf{g}_i^T \tag{15}$$

where $\boldsymbol{\phi}_i$ is the ith column of the dictionary matrix $\mathbf{\Phi}$, $\mathbf{g}_i^T$ is the ith row of the coefficient matrix $\mathbf{G}$ and $\mathbf{E}_l$ stands for the residual when the lth atom is removed from the dictionary matrix. The SVD decomposition is then applied to $\mathbf{E}_l$ to find alternative $\boldsymbol{\phi}_i$ and $\mathbf{g}_i^T$ for approximating $\mathbf{E}_l$ as the closest rank-1 matrix. When all the atoms in the dictionary have been updated, the learned dictionary is ready for the sparse coding stage in the next iteration.

2.3.2. GAD

The greedy adaptive dictionary (GAD) algorithm (Jafari and Plumbley, 2011) learns the dictionary atoms based on an iterative process using the sparsity index defined as follows:

$$\sigma_j=\frac{\|\mathbf{t}_j\|_1}{\|\mathbf{t}_j\|_2} \tag{16}$$

where $\|\cdot\|_1$ and $\|\cdot\|_2$ denote the ℓ_1- and ℓ_2-norm respectively and $\mathbf{t}_j$ is the column vector of $\mathbf{T}$. The sparsity index

measures the sparsity of a signal, where the smaller σ_j, the sparser the signal vector $\mathbf{t}_j$.

The GAD algorithm begins with the definition of a residual matrix $\mathbf{R}^d = [\mathbf{r}_1^d, \ldots, \mathbf{r}_J^d]$, where $\mathbf{r}_j^d$ is a residual column vector corresponding to the j-th column of $\mathbf{R}^d$. This residual matrix is initialized to $\mathbf{T}$. The dictionary is then built by selecting the residual vector that has the lowest sparsity index j.

$$\hat{j} = \arg\min_j \frac{\|\mathbf{r}_j^d\|_1}{\|\mathbf{r}_j^d\|_2} \tag{17}$$

Then $\mathbf{r}_j^d$ is normalized and added to the dictionary. Finally, the new residual is computed for all the columns. The process is repeated until the number of obtained atoms reaches a pre-determined value.

2.3.3. SimCO

We recently developed a novel dictionary learning algorithm called simultaneous codeword optimization (SimCO) (Dai et al., 2012). The unique feature of our approach is that one can update an arbitrary set of codewords and the corresponding sparse coefficients simultaneously. More specifically, let $\Omega \subset [I] \times [J]$ contain the indices of non-zero entries in $\mathbf{G}$ which is the coefficient matrix produced by the sparse coding stage. $G_{i,j} \neq 0$ for all $(i,j) \in \Omega$ and $G_{i,j} = 0$ for all $(i,j) \notin \Omega$. We refer to this set as the *sparsity pattern* which is often obtained via the sparse coding stage using the algorithms discussed in Section 2.2.1 such as BP, MP and OMP. Given the training data $\mathbf{T}$ and a sparsity pattern Ω, the optimization problem under consideration is

$$\min_{\mathbf{\Phi},\mathbf{G}} \|\mathbf{T} - \mathbf{\Phi}\mathbf{G}\|_{\mathbf{F}}^{\mathbf{2}}, \quad \mathbf{s.t.} \quad \|\mathbf{\Phi}_{:,\mathbf{i}}\|_{\mathbf{2}} = \mathbf{1}, \quad \forall \mathbf{i} \in [\mathbf{I}], \quad \mathbf{and} \quad \mathbf{G}_{\mathbf{i},\mathbf{j}} = \mathbf{0}, \quad \forall (\mathbf{i},\mathbf{j}) \notin \Omega \tag{18}$$

where $\mathbf{\Phi}_{:,i}$ represents the ith column of dictionary $\mathbf{\Phi}$ (This notation is used because it can denote the sub-matrix of $\mathbf{\Phi}$ in the following paragraphs). With slight modification in (18), the connection between our approach and K-SVD becomes clear. Let $\mathcal{I} \subset [I]$ be an index set. Suppose that one is only interested in updating the codewords indexed by $\mathcal{I}$, i.e., $\mathbf{\Phi}_{:,i}$'s with $i \in \mathcal{I}$, and keep other codewords constant. Let $\mathbf{\Phi}_{:,\mathcal{I}}$ denote the sub-matrix of $\mathbf{\Phi}$ formed by the columns of $\mathbf{\Phi}$ indexed by $\mathcal{I}$. Let $\mathbf{G}_{\mathcal{I},:}$ denote the sub-matrix of $\mathbf{G}$ consisting of the rows of $\mathbf{G}$ indexed by $\mathcal{I}$. The optimization problem that only updates $\mathbf{\Phi}_{:,\mathcal{I}}$ and $\mathbf{G}_{\mathcal{I},:}$ is given by

$$\min_{\mathbf{\Phi}_{:,\mathcal{I}},\mathbf{G}_{\mathcal{I},:}} \|\mathbf{T} - \mathbf{\Phi}\mathbf{G}\|_F^2, \quad \text{s.t.} \quad \|\mathbf{\Phi}_{:,i}\|_2 = 1, \ \forall i \in \mathcal{I}, \quad \text{and} \quad G_{i,j} = 0, \quad \forall (i,j) \notin \Omega \tag{19}$$

When the index set $\mathcal{I}$ contains only one entry, the optimization problem (19) is the one that each step of K-SVD dictionary update is designed to solve. When $\mathcal{I} = [I]$, the optimization problems (18) and (19) are identical.

For compositional convenience, we introduce the following notation. Suppose that we are updating the codewords indexed by $\mathcal{I}$, i.e., $\mathbf{\Phi}_{:,\mathcal{I}}$, and the corresponding coefficients, i.e., $\mathbf{G}_{\mathcal{I},:}$. Define

$$\mathbf{T}_r = \mathbf{T} - \mathbf{\Phi}_{:,\mathcal{I}^c}\mathbf{G}_{\mathcal{I}^c,:},$$

where $\mathcal{I}^c$ is the complementary set of $\mathcal{I}$. Clearly,

$$\mathbf{T} - \mathbf{\Phi}\mathbf{G} = \mathbf{T}_r - \mathbf{\Phi}_{:,\mathcal{I}}\mathbf{G}_{\mathcal{I},:}.$$

Define the following function

$$f_{\mathcal{I}}(\mathbf{\Phi}) = \min_{\mathbf{G}_{\mathcal{I},:}:\mathbf{G}_{i,j}=0,\forall (i,j)\notin\Omega} \|\mathbf{T} - \mathbf{\Phi}\mathbf{G}\|_F^2.$$

It is clear that

$$f_{\mathcal{I}}(\mathbf{\Phi}) = \min_{\mathbf{G}_{\mathcal{I},:}:\mathbf{G}_{i,j}=0,\forall (i,j)\notin\Omega} \|\mathbf{T}_r - \mathbf{\Phi}_{:,\mathcal{I}}\mathbf{G}_{\mathcal{I},:}\|_F^2 \tag{20}$$

Hence, the optimization problem (19) can be written as

$$\min_{\mathbf{\Phi}_{:,\mathcal{I}}} f_{\mathcal{I}}(\mathbf{\Phi}) \text{ subject to } \|\mathbf{\Phi}_{:,i}\|_2 = 1, \quad \forall i \in \mathcal{I} \tag{21}$$

Based on the derivation in (Dai et al., 2012), the gradient of $f_{\mathcal{I}}(\mathbf{\Phi})$, with respect to $\mathbf{\Phi}_{:,i}, i \in \mathcal{I}$, can be computed via

$$\begin{aligned} \nabla_{\mathbf{\Phi}_{:,i}} f_{\mathcal{I}}(\mathbf{\Phi}) &= -2(\mathbf{T} - \mathbf{\Phi}\mathbf{G}^*)_{:,\Omega(i,:)}\mathbf{G}_{i,\Omega(i,:)}^{*T} \\ &= -2(\mathbf{T} - \mathbf{\Phi}\mathbf{G}^*)\mathbf{G}_{i,:}^{*T} \end{aligned} \tag{22}$$

Here, $\Omega(i,:)$ gives the columns of $\mathbf{T}$ whose sparse representation involves the codeword $\mathbf{\Phi}_{:,i}$. $\mathbf{G}_{i,:}^{*T}$ is the optimal solution to the least squares problem in (20). Let $\mathbf{h}_i = \nabla_{\mathbf{\Phi}_{:,i}} f_{\mathcal{I}}(\mathbf{\Phi})$ be the gradient vector defined in (22). We define

$$\bar{\mathbf{h}}_i = \mathbf{h}_i - \mathbf{\Phi}_{:,i}\mathbf{\Phi}_{:,i}^T\mathbf{h}_i, \quad \forall i \in \mathcal{I} \tag{23}$$

According to Edelman et al. (1999), $\bar{\mathbf{h}}_i$ is in fact the gradient of f with respect to $\mathbf{\Phi}_{:,i}$ on the Grassmann manifold. The line search path for dictionary update, say $\mathbf{\Phi}(t), \ t \geqslant 0$, is defined as

$$\begin{cases} \mathbf{\Phi}_{:,i}(t) = \mathbf{\Phi}_{:,i} \quad \text{if } i \notin \mathcal{I} \text{ or } \|\bar{\mathbf{h}}_i\|_2 = 0, \\ \mathbf{\Phi}_{:,i}(t) = \mathbf{\Phi}_{:,i}\cos\left(\|\bar{\mathbf{h}}_i\|_2 t\right) - \left(\bar{\mathbf{h}}_i/\|\bar{\mathbf{h}}_i\|_2\right)\sin\left(\|\bar{\mathbf{h}}_i\|_2 t\right) \\ \qquad\qquad \text{if } i \in \mathcal{I} \text{ and } \|\bar{\mathbf{h}}_i\|_2 \neq 0 \end{cases} \tag{24}$$

where $\mathbf{h}_i = \nabla_{\mathbf{\Phi}_{:,i}} f_{\mathcal{I}}(\mathbf{\Phi})$ can be computed via (22).

Finally, a line search procedure over the product space of Grassmann manifolds is proposed in (Dai et al., 2012) to speed up the convergence of the gradient based dictionary learning algorithm.

The dictionary learning algorithm in SDUBSS is summarized in Algorithm 3. Note that, $\mathbf{T}$ can be formed from speech mixtures $\mathbf{X}$ or original speech sources $\mathbf{S}$, while $\mathbf{\Phi}$ needs to be learned from $\mathbf{T}$, as discussed below.

Algorithm 3. Adaptive dictionary learning.

Task: find the best dictionary to represent the training speech data
Input: $\mathbf{T}$.
Output: $\mathbf{\Phi}$.
Initialization: Set the initial dictionary $\mathbf{\Phi}^{(1)}$ and $j = 1$.
Repeat until convergence (use stop rule):
- Sparse coding stage: Fix the dictionary $\mathbf{\Phi}^{(j)}$ and update $\mathbf{G}^{(j)}$ using some sparse coding technique, such as OMP.
- Dictionary update stage: Update $\mathbf{\Phi}^{(j)}$, and $\mathbf{G}^{(j)}$ as appropriate, using e.g. SimCO dictionary update.
- $j = j + 1$.

2.4. Dictionary learning strategies

It is an important practical issue (Xu and Wang, 2011) on how to learn the dictionary from training data. We examine two different training strategies. To this end, we first expand (5) in two mixtures and four sources case as follows.

$$\underbrace{\begin{pmatrix} s_1(1) \\ \vdots \\ s_1(T) \\ s_2(1) \\ \vdots \\ s_2(T) \\ s_3(1) \\ \vdots \\ s_3(T) \\ s_4(1) \\ \vdots \\ s_4(T) \end{pmatrix}}_{\mathbf{f}} = \underbrace{\begin{pmatrix} D_1 & & & \\ & D_2 & & \\ & & D_3 & \\ & & & D_4 \end{pmatrix}}_{\mathbf{\Phi}} \underbrace{\begin{pmatrix} y_1(1) \\ \vdots \\ y_1(T) \\ y_2(1) \\ \vdots \\ y_2(T) \\ y_3(1) \\ \vdots \\ y_3(T) \\ y_4(1) \\ \vdots \\ y_4(T) \end{pmatrix}}_{\mathbf{y}} \tag{25}$$

2.4.1. Source-trained dictionary

The first strategy called the source-trained dictionary (STD) is depicted in Fig. 3 where DL represents dictionary learning which can be achieved by any of the algorithms described in the above section. In this method, for each source, we train a dictionary. Therefore four different dictionaries D_1, D_2, D_3, D_4 are trained from the four original sources respectively. They are then combined to form a single dictionary matrix $\mathbf{\Phi}$ for separating the sources in the following stages. For example, D_1 in Eq. (25) is trained from the source $\mathbf{s}_1$. Firstly, the speech source vector is reshaped to a speech sample matrix which contains consecutive speech frames (each frame has L samples) from the source vector with an overlap of F samples (to ensure a sufficient number of signals in the sample matrix). Therefore, the sample matrix has L rows and $\lfloor (T-L)/(L-F) \rfloor + 1$ columns, where $\lfloor \cdot \rfloor$ rounds the argument to its nearest integer. The dictionary is then computed by one of the three learning algorithms described in Section 2.3 as an $L \times L$ matrix. Finally, the dictionary matrix is arranged in a diagonal form with an overlap of F samples until the $T \times T$ dictionary D_1 is filled in. By using this block diagonal operation we essentially split the signal in small vectors and there will typically be a block boundary issue, i.e. a discontinuity between the joining area of two adjacent blocks, causing undesired artifacts in the coefficients. To avoid this we can multiply the vector by a window function (e.g. the Hamming window), thus smoothing the signal at the boundaries. Some information may be lost because of the windowing, and hence overlapping between blocks is used to eliminate this problem. The other dictionaries D_2, D_3, D_4 can be generated in the same way. The final single dictionary matrix $\mathbf{\Phi}$ is formed by arranging these four dictionaries along the diagonal of $\mathbf{\Phi}$ without overlaps. Ideally, the order of the dictionaries D_i $(i = 1, \ldots, 4)$ should be consistent with the order of sources $\mathbf{s}_i$. According to our experiments, when a mismatch of the orders occurs, the separation performance may be degraded. The reason that this happens could be that the feature of a speech source is better captured by its corresponding dictionary rather than the dictionary obtained from another source.

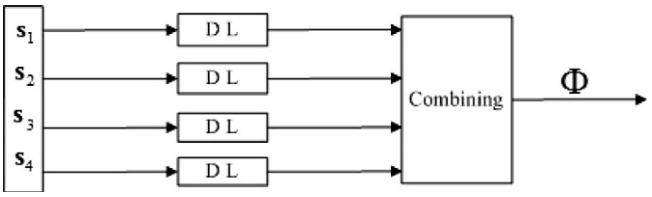

Fig. 3. The flow chart of the STD strategy.

2.4.2. Mixture-trained dictionary

The second strategy, namely the mixture-trained dictionary (MTD), is illustrated in Fig. 4. The two mixtures $\mathbf{x}_i$ $(i = 1, 2)$ used to train the dictionary are segmented with an overlap of F samples (each frame has L samples) to form the sample matrix which has L rows and $(\lfloor (T-L)/(L-F) \rfloor + 1) * 2$ columns. The dictionary is then computed

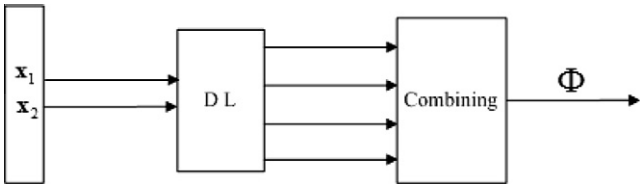

Fig. 4. The flow chart of the MTD strategy.

by the dictionary learning algorithms such as the K-SVD, GAD or SimCO as an $L \times L$ matrix. Finally, the dictionary is arranged in a diagonal form with an overlap of F samples until the $T \times T$ dictionary D_M is filled in. In this method, D_1, D_2, D_3, and D_4 in Eq. (25) are all identical to D_M which is trained from the mixtures by the same methods as used for the first strategy. In comparison, as shown in the experiment section, STD has the best performance among the two different dictionary training strategies. This suggests that the dictionary trained in this way best matches the original speech source. However, this approach requires the sources to be available *a priori* when training the dictionary. Although in BSS, the sources are assumed to be unknown, the STD method shows the performance benchmark that could be achieved by a dictionary learning approach. In MTD, the sources are estimated in a blind manner, as the dictionary is trained directly from the mixtures. Nevertheless it captures the features less accurately from each source as compared with STD. However, MTD will be used in our experiments for fair comparison. It is worth noting that mixture signals can be regarded as noisy signals (corrupted by the interfering signals). Therefore, using mixture signals as training data is reasonable. Similar training methods could be found in (Elad and Aharon, 2006).

2.5. Blocking and reconstruction

According to Eq. (3), the microphone signals of full length are stacked into a single vector. This could result in a large size of measurement matrix for a long speech signal. The optimization process for the source recovery can become computationally demanding. To alleviate this issue, we propose to process the speech mixtures on a block-by-block basis before running the separating stage i.e. split $\mathbf{x}_i$ into $\mathbf{x}_i^p$ where $i = 1, 2,\ p = 1, \ldots, P$. The estimated sources from each block are concatenated to reconstruct the full signals. Therefore, the front-end processing stages (i.e. blocking and reconstruction) will be included in our proposed system to improve its computational efficiency. As shown in Section 3, compared with processing the whole signal, the block-based processing considerably improves the computational efficiency of the algorithm without degrading its separation performance. It is worth noting that we have already applied windowing and overlapping when learning the dictionary atoms. Although the length of the atoms can be different from the block length, the atoms become smoother due to the application of smoothing windows during the dictionary learning process. Using such atoms, the discontinuities between the reconstructed blocks become negligible. Hence, we do not apply any further overlapping when reconstructing the full-length signal. Informal listening tests also confirm that the blocking artefacts are mostly inaudible.

2.6. The whole system

The whole system of the proposed SDUBSS algorithm can be summarized in Algorithm 4. Note that, for comparison purpose, in the dictionary learning stage of SDUBSS, dictionary matrix $\mathbf{\Phi}$ will also be computed from a pre-defined transform such as DCT and/or STFT, as considered in our experiments.

Algorithm 4. CSDUBSS.

Task: Separate the four speech sources from the two speech mixtures
Input: $\mathbf{X}$.
Output: $\mathbf{S}$.

- Clustering stage: obtain the estimated mixing matrix $\widehat{\mathbf{A}}$ from the mixture matrix $\mathbf{X}$ by Algorithm 1.
- Dictionary learning stage: learn the dictionary $\mathbf{\Phi}$ from the mixture matrix $\mathbf{X}$ by Algorithm 3.
- Blocking stage: segment mixtures $\mathbf{x}_i,\ i = 1, 2$ to blocks $\mathbf{x}_i^p,\ p = 1, \ldots, P$.
- Separating stage: separate speech sources $\mathbf{s}_j^p,\ j = 1, \ldots, 4,\ p = 1, \ldots, P$ from each mixture block $\mathbf{x}_i^p,\ i = 1, 2,\ p = 1, \ldots, P$ by Algorithm 2.
- Reconstruction stage: reconstruct the speech source matrix $\mathbf{S}$ including the four whole sources $\mathbf{s}_j,\ j = 1, \ldots, 4$ by concatenating together all the blocks of estimated source components $\mathbf{s}_j^p,\ j = 1, \ldots, 4,\ p = 1, \ldots, P$.

3. Experimental results

3.1. Evaluation dataset and performance metrics

In the first three subsections, we evaluate the proposed algorithm by performing 50 random experiments for each type of comparisons based on the TIMIT database. Twelve speech signals from the TIMIT database are chosen as our signals' pool, from which four signals are randomly selected to be original speech sources in each test. The mixing matrix is randomly generated for each test so that two mixtures are obtained by this mixing matrix and the speech sources randomly picked from the pool. Note that the same random seed is used for all the experiments, which means that each experiment has the same 50 random mixing matrices and 50 groups of random speech sources. All the results in the first three subsections are the average values for the 50 tests. Each speech signal has a duration of 5 s, sampled at 10 kHz. That is, each signal has 50,000 samples.

In the final subsection, another database ('dev2') from the signal separation evaluation campaign (SiSEC2008[2]) and the database from Stereo Audio Source Separation Evaluation Campaign (SASSEC07[3]) are used for making comparison between the proposed algorithm and the state-of-the-art method for this underdetermined blind speech separation task. The original sources are also available for the evaluation with each having a duration of 10 s,

[2] Accessed from http://www.irisa.fr/metiss/SiSEC08/.
[3] Accessed from http://www.irisa.fr/metiss/SASSEC07/.

sampled at 16 kHz. That is, each signal has 160,000 samples.

For the fair comparison of blocking, dictionary learning and separating stages, the true random mixing matrices are used in the first three subsections i.e. the K-means clustering stage is excluded from the proposed multi-stage method. In the last subsection, the full stages of the proposed method are used to compare with the state-of-the-art algorithm.

For objective quality assessment, we use the three performance criteria defined in the BSSEVAL toolbox (Vincent et al., 2006) to evaluate the estimated source signals. These criteria are the signal to distortion ratio (SDR), the source to interference ratio (SIR) and the source to artifacts ratio (SAR), defined respectively as

$$SDR = 10log_{10}\frac{\|s_{target}\|^2}{\|e_{interf} + e_{noise} + e_{artif}\|^2} \tag{26}$$

$$SIR = 10log_{10}\frac{\|s_{target}\|^2}{\|e_{interf}\|^2} \tag{27}$$

$$SAR = 10log_{10}\frac{\|s_{target} + e_{interf} + e_{noise}\|^2}{\|e_{artif}\|^2} \tag{28}$$

where $s_{target}(t)$ is an allowed deformation of the target source $s_i(t)$, $e_{interf}(t)$ is an allowed deformation of the sources which accounts for the interference of the unwanted sources, $e_{noise}(t)$ is an allowed deformation of the perturbation noise (but not the sources), and $e_{artif}(t)$ is an artifact term that may correspond to artifacts of the separation algorithm such as musical noise, etc. Therefore, the estimated source $\hat{s}(t)$ can be decomposed as follows:

$$\hat{s}(t) = s_{target}(t) + e_{interf}(t) + e_{noise}(t) + e_{artif}(t) \tag{29}$$

According to Vincent et al. (2006), both SIR and SAR measure *local* performance. SIR mainly measures how well the algorithm does for the suppression of interfering sources, while SAR measures how much artefact is within the separated (target) source. SDR is a *global* performance index, which may give better assessment to the overall performance of the algorithms under comparison. For this reason, we will focus more on the interpretation of SDR results in subsequent analysis, as opposed to the SIR and SAR results.

3.2. Separation results with fixed dictionary

3.2.1. Comparison of different signal recovery algorithms for separation

One advantage of the proposed system is that any of the state-of-the-art signal recovery techniques can be employed in the separating stage. Therefore, we compare the effect of the different signal recovery algorithms on the separation performance of the proposed system. To this end, we replace the adaptive dictionary by the predefined dictionary obtained by the DCT transform. We use the whole speech signal as a single block, that is, $P = 1$. We then vary the algorithms used in the separating stage for signal recovery. The methods BP, MP and L1LS discussed in Section 2 are used in the experiment. Specifically, we use the following three algorithms in our experiments, i.e. SPGL1 (Spectral Projected Gradient for L1 minimization) (Berg and Friedlander, 2008a,b), the solveMP in SparseLab (Donoho et al., 2005) and L1LS solver for ℓ_1 regularized least squares problem (Kim et al., 2007a,b), which are the typical implementations of the three signal recovery methods. To balance the performance and the running speed of the algorithms, the parameters used in the three algorithms are tuned on the basis of extensive tests. In BP, the optimality tolerance parameter was set to 0.01. In MP, the maximum number of iterations[4] parameter *maxIters* was set to 1000 and the stop condition parameter *lambdastop* was set to zero. In L1LS, the regularization parameter *lambda* was set to 0.01 and the relative target duality gap was set to 1. With this set up, the computational time required by these three algorithms is 1234 s, 8169 s and 4531 s respectively. The separation performance evaluated by SDR, SIR and SAR is shown in Table 1.

Table 1
Average SDR, SIR, SAR (in dB) measured for four estimated speech sources and p-values from the t-test between the methods, where B = BP, M = MP, L = L1LS.

	BP	MP	L1LS	p-value		
				B/M	B/L	M/L
SDR	6.52	0.73	4.88	0.0000	0.0000	0.0000
SIR	9.98	19.07	6.43	0.0000	0.0000	0.0000
SAR	10.65	1.70	12.11	0.0000	0.0000	0.0000

We also perform a paired Student's t-test of the null hypothesis that the results from different methods are significantly different. All the t-tests in this work have been carried out at 5% significance level. If the p-value is greater than 0.05 (i.e. 5% significance level), the difference between the results is statistically insignificant. Otherwise, if the p-value is less than 0.05, the results are statistically significant, which means that the performance difference between these methods is significant. The first letter of these methods is used to represent the p-value between them. For example, in Table 1, B/M is used to denote the p-value obtained by comparing the results from BP and MP respectively. It can be observed that the p-value between different methods in the following tables are almost all smaller than 0.05, suggesting that the performance difference between these methods is statistically significant. The only exceptions are the p-values for K/G in Table 3, and S/F in Table 2, which are both greater than 0.05, suggesting that their difference is

[4] By increasing the number of iterations, the MP algorithm is likely to offer improved results, with however a considerably increased computational cost. Note that the parameters used in our experiments, including those for BP and L1LS, are by no means optimal despite the fact that every attempt has been made in order to find the parameter sets that give the best possible performance for each algorithm under comparison. In practice, however, we have also taken into account the computational complexity of these algorithms to ensure fair comparisons among them.

Table 2
Average SDR, SIR, SAR (in dB) measured for four estimated speech sources by using adaptive dictionary with different learning strategy and compared to using fixed dictionary i.e. DCT, STFT, MDCT. The right four columns present the *p*-values from the *t*-tests between STD and other four methods, respectively MTD, DCT, STFT and MDCT, where S = STD, M = MTD, D = DCT, F = STFT, C = MDCT.

	STD	MTD	DCT	STFT	MDCT	*p*-value			
						S/M	S/D	S/F	S/C
SDR	7.85	5.32	6.87	6.00	5.14	0.0000	0.0001	0.0000	0.0000
SIR	12.43	8.94	10.86	9.37	9.33	0.0000	0.0003	0.0000	0.0000
SAR	10.36	8.80	9.86	10.19	8.58	0.0000	0.0073	0.3507	0.0000

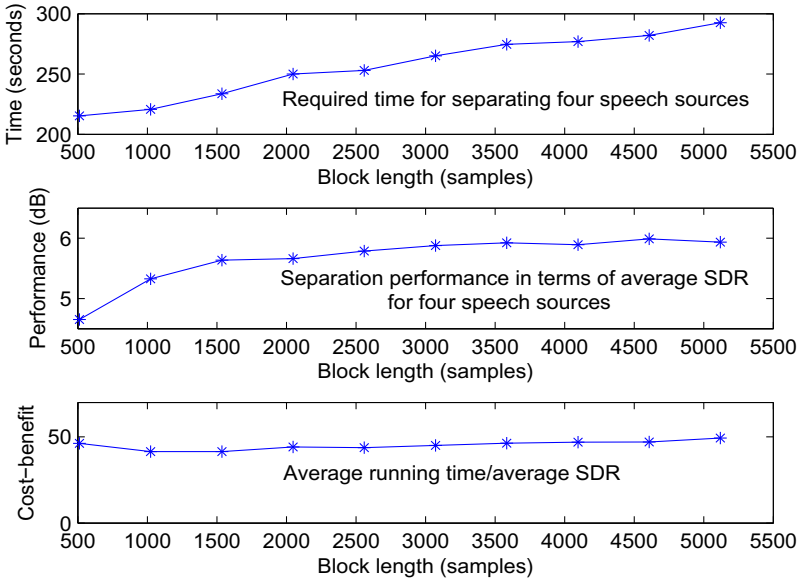

Fig. 5. The effect of different block length on the computational efficiency and separation performance of the proposed algorithm. The cost-benefit (i.e. computing time divided by the output SDR) is also shown.

statistically small. We have also calculated the confidence intervals in each *t*-test performed in Tables 1 and 2 respectively, and the results are shown in Tables 11 and 12 in the Appendix.

It can be seen from this table that BP performs better for sparse signal recovery in our separation task. Therefore, we will use it as the default signal recovery algorithm in the following experiments.

3.2.2. Effect of blocking on system performance

In this section, we perform experiments to evaluate the effect of the block size on the computational efficiency and separation performance of the proposed algorithm. We use the BP algorithm in the separating stage and the DCT transform to obtain the fixed dictionary $\mathbf{\Phi}$. The relation of the computational cost to the block length is shown in the upper subplot of Fig. 5, while the separation performance (measured by the SDR) versus the block length is shown in the middle subplot. Each result on the plots is a value averaged over the four estimated speech sources. From this figure, it can be observed that the algorithm becomes computationally more efficient when reducing the block lengths, with the separation performance getting slightly worse. For example, for the block size equal to 512 samples, it takes only 151 s to run the algorithm, however, the separation performance in terms of average SDR becomes 4.58 dB. For the block size equals to 2048 samples, the algorithm takes 302 s to run which is less efficient as compared to the use of a smaller block size, however it provides an average SDR for up to 5.67 dB. Compared with processing the full-length signal as a single block which takes 1234 s for the algorithm to finish running, using the block size of 2048 samples is reasonably fast. In this case, the block-based algorithm is approximately five times faster than the algorithm without blocking. Fig. 5 suggests that there are only slight changes in the separation performance for a certain range of block lengths. Based on these observations, we will use the block size of 2048 samples in the following experiments. It appears from the upper subplot of Fig. 5 that, as opposed to a very short block length, longer block lengths do not vary the required runtime considerably. This observation can be related back to the motivation for introducing the blocking process in the proposed method. We found that the BP algorithm can take enormous amount of time to converge for a long input signal, and can eventually become computationally prohibitive. When processing the whole signal on a block-by-block basis, this algorithm converges much faster.

Table 3
Average SDR, SIR, SAR (in dB) measured for four estimated speech sources by using the dictionaries learned with different learning algorithms. The right three columns present the p-values from the t-tests between these methods, where S = SimCO, K = K-SVD, and G = GAD.

	SimCO	K-SVD	GAD	p-value		
				S/K	S/G	K/G
SDR	5.32	3.99	2.93	0.0000	0.0000	0.0000
SIR	8.94	6.25	6.19	0.0000	0.0000	0.8079
SAR	8.80	9.35	7.08	0.0001	0.0000	0.0000

As a result, the overall time for processing the whole number of short blocks is still shorter than processing the full-length signal as one long block. With the test results for different block lengths, we attempt to find empirically an appropriate block length around which the pursuit algorithm converges efficiently, and at the same time, the separation performance does not deteriorate. In fact, if the number of iterations in BP is fixed, the blocking process may increase the computational time. In our method, the BP algorithm terminates iterations once a criterion is satisfied (using a threshold). For shorter signals, the pursuit algorithm tends to take much shorter time to find the solution (using a smaller number of iterations). The runtime or the number of iterations taken by the pursuit algorithms to converge varies with respect to several factors including the length of blocks, the nature of the signal, numerical artefacts, and the hardware used for running the tests. Recall that each result in Fig. 5 is an average of 50 random tests, with each taking a different number of iterations to converge. As a result, some longer block lengths do not vary the required runtime considerably.

3.3. Separation performance with adaptive dictionary

3.3.1. Comparison of different strategies for learning the dictionary

From the mixtures, we can recover the four speech sources using the DCT, STFT, MDCT dictionaries as done in sections[5] 3.2.1 and 3.2.2. Alternatively, we can learn the adaptive dictionaries based on the STD and MTD methods. The dictionary learning algorithm applied here is SimCO due to its performance advantage shown in the following section where the setup of the parameters is also given. The average results for 50 random tests are presented in Table 2.

From this table, we can observe that the separation performance using the STD trained dictionary is considerably better than using the fixed dictionary. However, it is not surprising that the MTD method i.e. using the dictionary learned from the mixtures offers lower performance than the STD method. These results suggest that the properly learned dictionaries outperform the pre-defined dictionary in underdetermined speech separation.

[5] The parameters were set to be the same for DCT, STFT, and MDCT, for example, the window (block) lengths were all set to 2048 samples.

Table 4
Performance comparison (measured by SDR in dB) between the learned dictionaries and the predefined dictionary (i.e. DCT) for the noise-free mixtures, noisy mixtures, and the performance degradation (i.e. the difference between the results obtained from the noise-free mixtures and the noisy mixtures).

	DCT	SimCO	K-SVD	GAD
Noise-free mixtures	6.87	5.32	3.95	2.93
Noisy mixtures	5.82	5.31	3.81	2.91
Performance degradation	1.05	0.01	0.14	0.02

3.3.2. Comparison of different dictionary learning algorithms for separation

In this section, we compare the results of using different learning algorithms in the dictionary learning stage of the proposed system. As above, the BP algorithm is used in the separating stage, and the same mixtures are used in these random experiments. The parameters used in the dictionary learning algorithms are also tuned to balance the performance and the running speed of the algorithms. The number of trained atoms was set to 512, the length of each atom i.e. L to 512, the sparsity parameter to 10 and the number of iterations to 30 for both SimCO and K-SVD. In GAD, only the size of the dictionary needs to be set, which is 512. Note that in the first 15 iterations, SimCO was run with the regularization parameter μ set to 0.1, and in the following 15 iterations, to 0. The average results over 50 random tests are given in Table 3. It shows that the separation performance obtained by using SimCO is better than using K-SVD and GAD.

3.4. Separation in noisy case

In this section, we examine the performance of adaptive dictionaries for noisy mixtures. To this end, we add white Gaussian noise to the speech mixtures with a SNR = 20 dB. The dictionary is trained from the noisy mixtures and the sources are separated from the same noisy mixtures using the proposed algorithm. The set up of the proposed system is identical to the one for the noise-free case. Table 4 shows the results measured by SDR for the noise-free mixtures, noisy mixtures, and the difference between them (i.e. the performance degradation). From this table, it can be observed that using a fixed dictionary, the performance degradation based on 50 random tests from noise-free to noisy environment is considerably greater than that using an adaptive dictionary. This

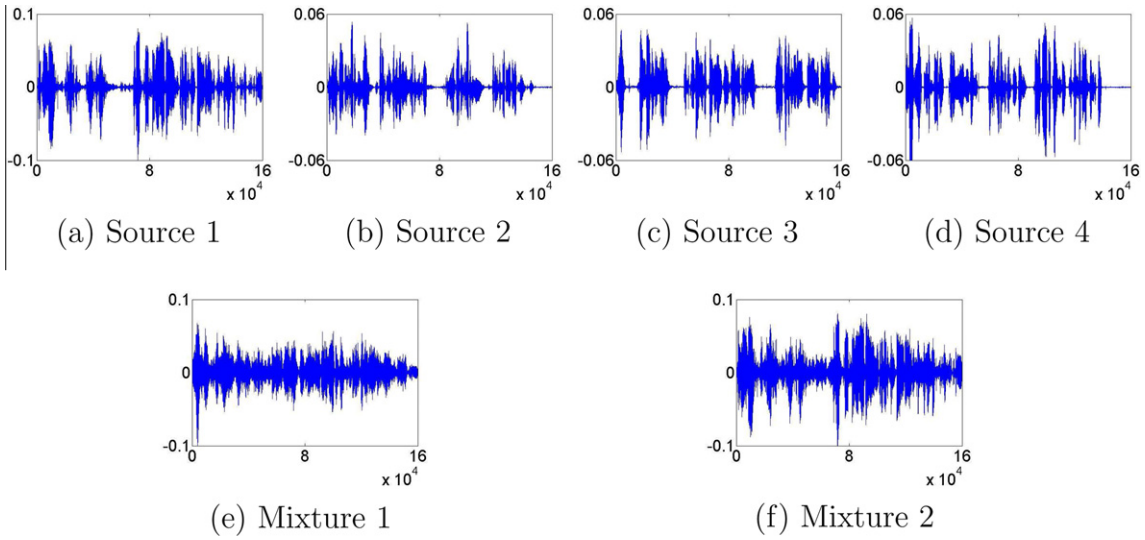

Fig. 6. The four male speech sources (a), (b), (c), (d) and the two mixtures (e), (f) used in the experiment. The horizontal and vertical axis are the sample indices and amplitude respectively, same for those in Fig. 7.

indicates that the learned dictionary tends to be more robust than a fixed dictionary for the separation of noisy speech mixtures.

3.5. Comparison with the state-of-the-art method

In this section, the proposed algorithm is compared with two related methods, namely, Gowreesunker and Tewfik (2009, 2008) and Bofill and Zibulevsky (2001). First, we compare our algorithm with (Gowreesunker and Tewfik, 2009, 2008), which also uses adaptive dictionary for speech separation. Their results have been reported in the evaluation campaign SiSEC2008. In this method, the mixing matrix is estimated using peak picking on a threshold histogram and separation using coefficient space partitioning with a K-SVD trained dictionary. In our proposed method, the techniques used in different stages are specified based on the above experimental results. The mixing matrix is estimated by K-means clustering in the clustering stage. The basis pursuit is used in the separating stage for signal recovery. The dictionary update algorithm SimCO and the training strategy MTD (for 'blind' separation) is used in the dictionary learning stage. All the speech mixtures are processed by the blocking stage and the reconstruction stage to obtain the final separation performance. The test data used are the four male speech signals from SiSEC2008 'Under-determined speech and music mixtures development 2' database.

In the beginning of the experiment, we used two instantaneous mixtures which were obtained by mixing four male speech sources with the following mixing matrix.

$$\mathbf{A} = \begin{pmatrix} 0.3338 & 0.6495 & 0.8241 & 0.9397 \\ 0.9426 & 0.7604 & 0.5664 & 0.3420 \end{pmatrix} \tag{30}$$

The $\widehat{\mathbf{A}}$ obtained from the two instantaneous mixtures by the clustering algorithm (i.e. clustering stage of the proposed algorithm) is shown in Eq. (31). We see that the estimated mixing matrix $\widehat{\mathbf{A}}$ is reasonably close to the true mixing matrix $\mathbf{A}$ except the permutation ambiguity.

$$\widehat{\mathbf{A}} = \begin{pmatrix} 0.6312 & 0.9368 & 0.8132 & 0.3532 \\ 0.7756 & 0.3499 & 0.5820 & 0.9355 \end{pmatrix} \tag{31}$$

Based on the estimated mixing matrix, we can then recover the four speech sources using the remaining stages of the proposed system. For the mixtures shown in Fig. 6, the separation result by the proposed system is shown in Fig. 7, where the adaptive dictionary was learned by SimCO in the dictionary learning stage. It can be observed that the estimated sources in Fig. 7 are very similar to the original sources in Fig. 6.

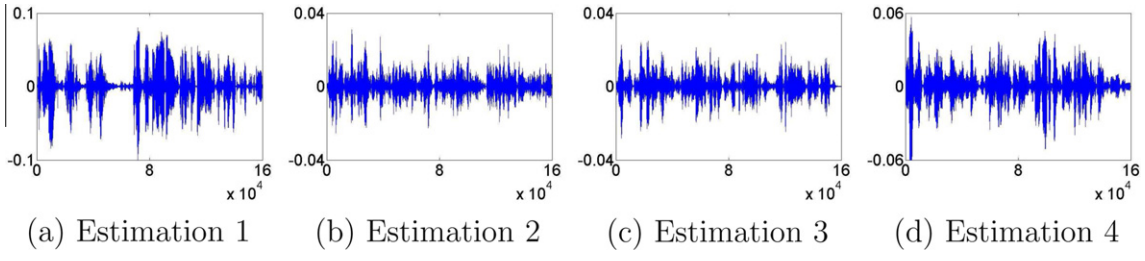

Fig. 7. The four estimated male speech sources.

Table 5
Average SDR, SIR, SAR (in dB) measured for four estimated male speech sources obtained by the proposed method (with the learned dictionary), the method due to Gowreesunker and Tewfik, and the proposed method with the STFT dictionary.

	Proposed method	Gowreesunker and Tewfik	STFT method
SDR	4.38	2.73	4.77
SIR	7.53	8.15	7.99
SAR	9.02	5.93	9.23

Table 6
Average SDR, SIR, SAR (in dB) measured for four estimated female speech sources obtained by the proposed method (with the learned dictionary), the method due to Gowreesunker and Tewfik, and the proposed method with the STFT dictionary.

	Proposed method	Gowreesunker and Tewfik	STFT method
SDR	4.04	3.80	4.51
SIR	6.19	8.58	6.86
SAR	9.73	6.60	9.78

The average performance of these four separated sources measured by SDR, SIR, SAR is shown in Table 5, where the results are compared between the proposed method, the method by Gowreesunker and Tewfik, and the proposed method without using dictionary learning (i.e. using the STFT basis instead).

We can see that, using the proposed method, there is an approximately 2 dB improvement over the method by Gowreesunker and Tewfik. For this task, the proposed method takes 868 s while the compared method needs 1200 s to separate the speech sources.

We have also tested these methods on four female speech signals in the SiSEC2008 evaluation campaign, using the exactly same parameters as those for male speech tests. Table 6 shows the performance of the compared methods measured by SDR, SIR, and SAR from these four separated sources. Again, the proposed method offers consistently better performance than these baseline methods.

To compare our method with another benchmark method by (Bofill and Zibulevsky, 2001), we use the dataset in the evaluation campaign SASSEC07, which can be regarded as an earlier version of SiSEC2008. The reason for this choice is that the results of the algorithm by Bofill and Zibulevsky were reported in SASSEC07, but not in SiSEC2008. Using the data from the campaign SASSEC07 thus enables us to compare the results of our algorithm with those of Bofill and Zibulevsky's method. Specifically, we used the signals from the Instantaneous Mixtures in the 'Development data'.

Table 7
Average SDR, SIR, SAR (in dB) measured for four estimated male speech sources.

	Proposed method	Bofill and Zibulevsky	STFT
SDR	6.15	3.33	6.40
SIR	7.36	7.65	7.75
SAR	9.76	8.50	10.08

Table 8
Average SDR, SIR, SAR (in dB) measured for four estimated female speech sources.

	Proposed method	Bofill and Zibulevsky	STFT
SDR	5.52	4.30	5.72
SIR	6.06	8.90	6.64
SAR	10.54	9.20	10.73

The algorithm by Bofill and Zibulevsky has been used as a benchmark for performance comparison in many papers on underdetermined source separation. Even though this method does not use an adaptive dictionary, it is one of the early papers that implement the idea of sparse coding for underdetermined source separation. In this approach, the mixing matrix is estimated by maximizing a potential function which is defined as the sum of the individual contributions from each angular direction of all the possible directions along the circle of unit length (Bofill and Zibulevsky, 2001). The maxima of the potential function are considered to be the estimated directions of the basis vectors (Bofill and Zibulevsky, 2001). The average performance measured by SDR, SIR, SAR from these four separated sources is shown in Table 7, where the results for using predefined dictionary STFT are also included for comparison.

The results of another test based on four female speech signals are shown in Table 8.

It is observed from Tables 5 and 6 that, the reference method (Gowreesunker and Tewfik, 2009, 2008) performs better in terms of SIR and worse in terms of SAR than our proposed method. We believe such an effect is mainly caused by the different ways of processing taken in these two methods, instead of by parameter tuning. In an ideal situation, one would expect an algorithm to be able to suppress the interfering sources as much as possible, without introducing artefacts to the separated source (i.e. the source of interest). In practice, however, many algorithms may introduce processing artefacts to the source of interest when suppressing the interfering sources. Such artefacts may be introduced by the separation algorithm due to, for example, time–frequency masking (causing e.g. musical noise), filtering operations, or deformations that are not allowed (Vincent et al., 2006). Such a difference (i.e. a higher SIR, but a lower SAR) can also be observed from the results of the algorithms reported in the SiSEC2008 evaluation campaign (Vincent et al., 2009).

The reference method (Gowreesunker and Tewfik, 2009, 2008) uses a coefficient space partitioning technique for source recovery and separation. Such an approximation is likely to introduce additional artefacts despite its ability in suppressing the interfering sources. This may well be the situation that is observed here, i.e. giving a higher SIR but a lower SAR, in comparison to our proposed method. As shown in (Vincent et al., 2006, 2009), using SDR may give better overall performance assessment to the algorithms under comparison, as SDR is a *global* performance index (Vincent et al., 2006).

Table 9
Mutual coherence of the dictionaries learned from the female and male speech mixtures using SimCO, as compared with the DCT and STFT dictionaries. Note that, the DCT and STFT atoms (bases) are pre-defined, hence they are kept the same for female and male speech in this example.

	SimCO (female)	SimCO (male)	DCT	STFT
Coherence	0.16	0.28	5.26e−16	1.37e−16

Table 10
The average sparsity indices (and their standard deviations) of all the atoms and the coding coefficients from the learned dictionaries (different for female and male speech mixtures) and the predefined dictionaries (DCT and STFT, fixed for male and female speech mixtures).

	SimCO (female)	SimCO (male)	DCT	STFT
Atoms	20.37 (0.15)	20.36 (0.17)	20.38 (0.14)	20.34 (0.32)
Coefficients	8.29 (0.24)	8.45 (0.25)	8.28 (0.25)	8.72 (0.25)

3.6. Additional performance analysis

Recent progresses suggest that the performance of dictionary learning algorithms is highly dependant on the level of sparsity (of atoms and/or coding coefficients) achieved in sparse approximation and the mutual coherence between the atoms (Tropp, 2004; Gribonval and Vandergheynst, 2006; Dai et al., 2012; Mailhe et al., 2012; Barchiesi and Plumbley, 2012; Plumbley et al., 2010). To gain a deeper understanding about the results obtained in above sections, we have performed additional experiments and numerical analysis from the following two aspects: namely sparse coding effect and mutual atom coherence (either learned, or predefined). We perform the analysis based on the SiSEC2008 campaign data as already used in Section 3.5. The mixtures both have a length of 160,000 samples, and the length of each dictionary atom was set to 512.

As shown earlier (e.g. in Tables 2–5), using the dictionaries learned from the 'ground truth' speech sources offers significantly better separation performance than using the predefined dictionary consisting of DCT basis functions, while the dictionaries learned from speech mixtures tend to perform worse than the DCT dictionary. The difference in separation performance due to the use of these dictionaries can be well explained by the difference in their mutual coherences. The mutual coherence of a dictionary, i.e. $v(\mathbf{\Phi})$, can be defined as the maximum absolute inner product between any two different atoms,

$$v(\mathbf{\Phi}) = \max_{i \neq j} |\langle \phi_i, \phi_j \rangle| \tag{32}$$

Using the same speech data as for Tables 5 and 6, we show in Table 9 the mutual coherence results of the learned dictionaries and the predefined dictionaries (DCT and STFT). It is not surprising that the coherence values of the DCT and STFT dictionary are very small, as their bases are orthogonal to each other. In contrast, the learned dictionary (from either female or male speech) has a much greater coherence value. According to Tropp (2004), only if the mutual coherence of the dictionary is low, the sparse coding algorithm OMP, which is used in our SimCO algorithm, will guarantee to obtain the right support, i.e. the selection of the atoms, for sparse signal recovery. As a consequence, the dictionaries learned from speech mixtures may produce worse results than the predefined dictionaries (DCT and STFT), due to the higher coherence. As such, increasing efforts are devoted to the learning of incoherent dictionaries from data (Gribonval and Vandergheynst, 2006; Mailhe et al., 2012; Barchiesi and Plumbley, 2012).

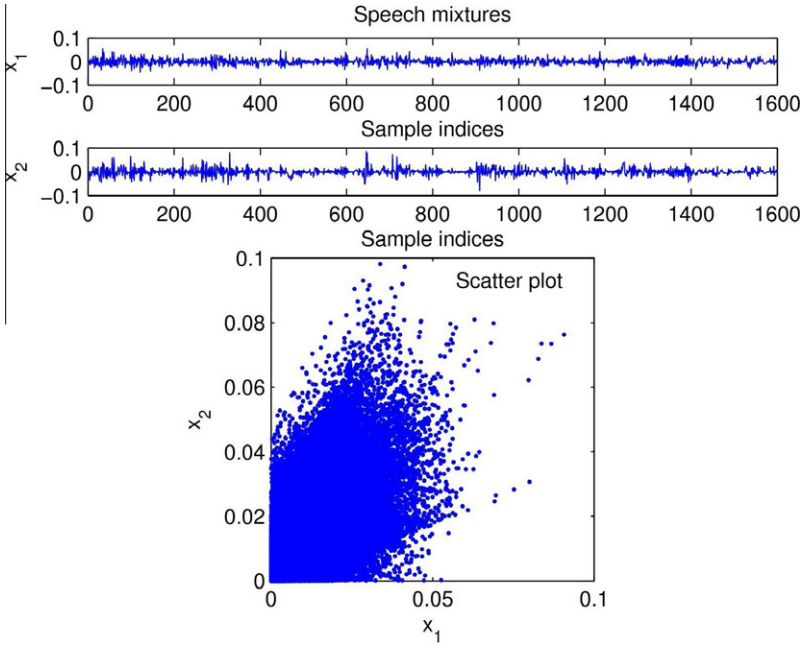

Fig. 8. Time domain original female speech mixtures (down sampled at rate 100:1) and their scatter plot.

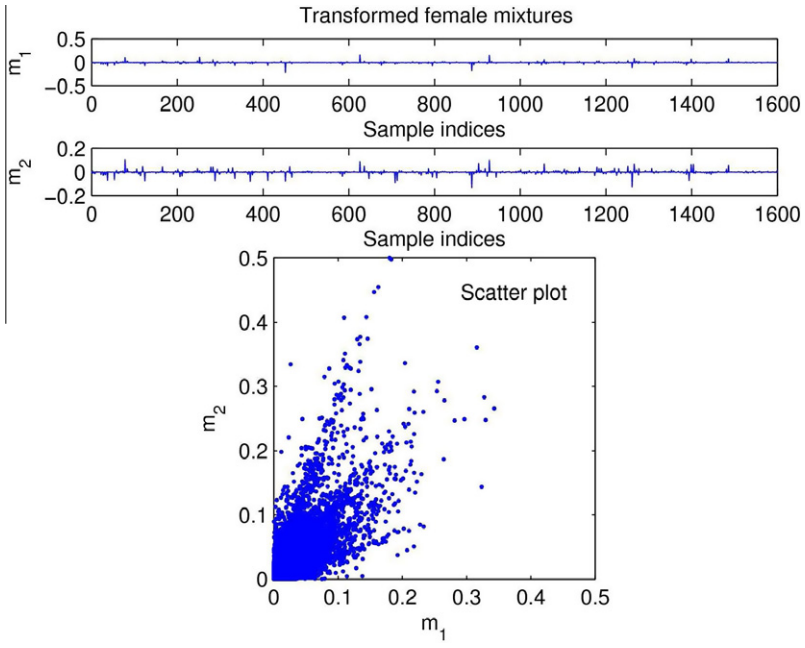

Fig. 9. Coding coefficients obtained using the dictionary learned by SimCO (down sampled at rate 100:1) and their scatter plot.

Incoherency constraints could be incorporated to our separation system in order to further improve the separation performance, which we leave to our future work.

Apart from the mutual coherence, sparsity also contributes to the performance difference. For example, comparing Tables 5 and 6, we found that the coherence of the dictionary learned from the female speech is a little smaller than that from the male speech (despite the difference being small), however, the proposed algorithm tends to give higher SDR results for male speech. Such a single test may not be sufficient to draw an explicit link between the performance variation and the separation results. It is however interesting and useful to compare the sparsity level of the learned dictionaries and the predefined dictionaries, together with their coding coefficients (i.e. sparse approximation results). This can be assessed by checking the sparsity index as defined in Eq. (16) and the joint scatter plots of the coding coefficients. The sparsity index measures the sparsity of an atom. The smaller the sparsity index, the sparser the measured atom. The average sparsity indices of the atoms (corresponding

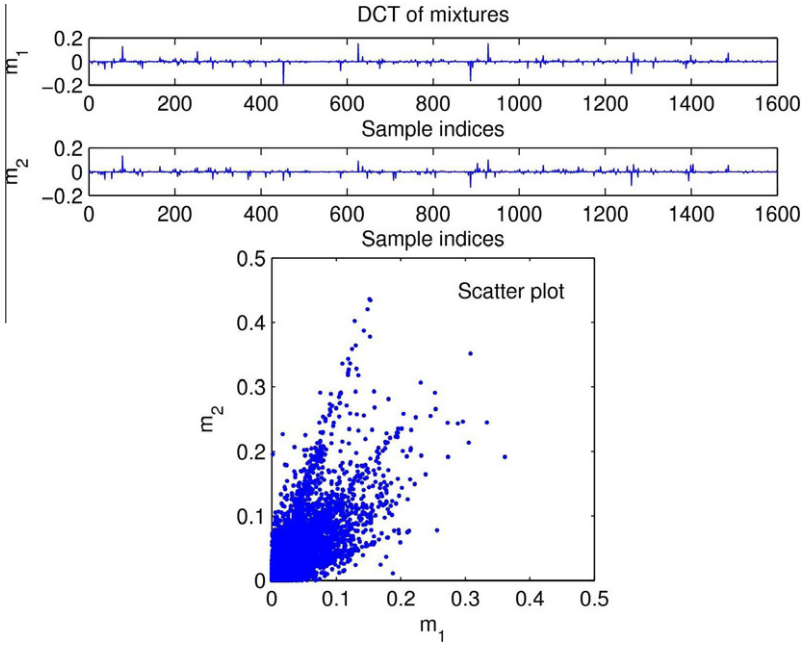

Fig. 10. DCT coding coefficients (down sampled at rate 100:1) and their scatter plot.

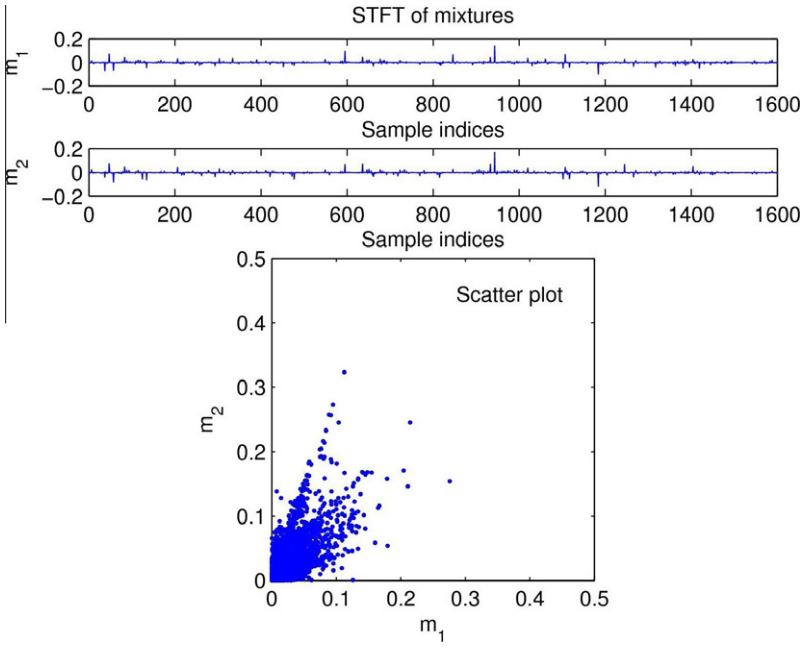

Fig. 11. STFT coding coefficients (down sampled at rate 100:1) and their scatter plot.

Table 11
The confidence intervals corresponding to the *p*-values in Table 1 obtained from the *t*-test between the methods, where B = BP, M = MP, and L = L1LS.

	Confidence intervals		
	B/M	B/L	M/L
SDR	(4.71, 6.85)	(1.24, 2.02)	(−5.14, −3.15)
SIR	(−10.63, −7.57)	(3.00, 4.08)	(11.20, 14.07)
SAR	(8.48, 9.44)	(−1.71, −1.20)	(−10.82, −10.01)

Table 12
The confidence intervals corresponding to the *p*-values in Table 2 obtained from the *t*-tests between the STD and other four methods, respectively MTD, DCT, STFT and MDCT, where S = STD, M = MTD, D = DCT, F = STFT, and C = MDCT.

	Confidence intervals			
	S/M	S/D	S/F	S/C
SDR	(2.08, 2.98)	(0.52, 1.45)	(1.34, 2.35)	(2.20, 3.22)
SIR	(2.69, 4.29)	(0.75, 2.39)	(2.25, 3.87)	(2.27, 3.93)
SAR	(1.20, 1.91)	(0.14, 0.87)	(−0.19, 0.53)	(1.42, 2.15)

Table 13
The confidence intervals corresponding to the *p*-values in Table 3 obtained from the *t*-tests between the methods of SimCO, K-SVD and GAD, where S = SimCO, K = K-SVD, and G = GAD.

	Confidence intervals		
	S/K	S/G	K/G
SDR	(0.95, 1.71)	(2.02, 2.75)	(0.71, 1.41)
SIR	(2.05, 3.31)	(2.15, 3.35)	(−0.47, 0.60)
SAR	(−0.82, −0.28)	(1.45, 1.99)	(2.04, 2.51)

to Table 9) and and their coding coefficients are shown in Table 10. It can be observed that the dictionary with a lower average sparsity index of the atoms tends to produce higher SDR performance. This coincides with the result observed in (Jafari and Plumbley, 2011).

To obtain the coding coefficients, the signal was first divided into frames with each having a length of 512 samples. The DCT coding coefficients were then calculated for these frames, with an analyzing length identical to the length of the segments. The coefficients of each frame were then concatenated to form a vector of coefficients with an equal length to the original mixtures in the time domain. The STFT coefficients are obtained in the same way. The coding coefficients based on the learned dictionary are obtained by multiplying each frame of signals with the dictionary matrix obtained by the SimCO algorithm, which are then concatenated in the same way as the DCT and STFT coefficients. For the convenience of visualization, we show the downsampled versions of the speech mixtures, and their coding coefficients, with a sampling rate of 100:1, resulting in a length of 1600 samples along the horizontal axis. We also show the joint scatter plots of the original female speech mixtures (Fig. 8), and their coding coefficients using dictionaries learned by SimCO (Fig. 9), or predefined transforms such as DCT (Fig. 10) and and STFT (Fig. 11). Note that the joint scatter plots for male speech are omitted here. From these figures, it can be observed that the learned dictionary provides a similar sparsity pattern to those in DCT and STFT. This implies that the learned dictionary offers an alternative to DCT and STFT for coding speech signals. Similar effects have been observed from the male speech mixtures.

Inspecting the mutual coherence and average sparsity index of a dictionary provides some useful clues for interpreting the performance of a dictionary for the task of separation. However, the performance of using these

dictionaries can be dependant on the nature of the data, as well as the objective of the signal processing task. In some cases, it may be beneficial to promote the statistical dependency between the atoms, as shown in a recent paper (Peleg et al., 2012).

4. Conclusions

We have presented a multi-stage system for underdetermined blind speech separation using block-based sparse coding with adaptive dictionary learning. Numerical experiments have shown the competitive separation performance by the proposed method, when compared with the recent underdetermined BSS approaches reported in the recent source separation evaluation campaign. The proposed method builds a new framework for underdetermined BSS, and offers great potential to accommodate the sparse signal recovery and adaptive dictionary learning algorithms to the source separation problems. This study has also shown the benefit of using learned dictionaries for underdetermined BSS, and the advantage of using the block-based processing to improve the computational efficiency of the signal recovery algorithms. Moreover, the framework of the proposed method provides a friendly structure to test the performance of other dictionary learning and signal recovery algorithms in source separation applications in the future.

Acknowledgements

We thank the Associate Editor Dr. Bin Ma and the anonymous reviewers for their helpful comments for improving our paper, and Dr. Mark Barnard for proofreading the manuscript. This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) of the UK (Grant Nos. EP/H050000/1 and EP/H012842/1), the Centre for Vision Speech and Signal Processing (CVSSP), and the China Scholarship Council (CSC), and in part by the MOD University Defence Research Centre (UDRC) in Signal Processing.

Appendix. Tables 11–13 show the confidence intervals that correspond to the *p*-values in Tables 1–3 respectively. The *p*-values and the confidence intervals were obtained from the *t*-test on the results of the compared methods.

References

Aharon, M., Elad, M., Bruckstein, A., 2006. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans. Signal Process. 54 (11), 4311–4322.

Araki, S., Sawada, H., Makino, S., 2007. K-means based underdetermined blind speech separation. In: Blind Speech Separation. Springer, Netherlands, pp. 243–270.

Arberet, S., Gribonval, R., Bimbot, F., 2010. A robust method to count and locate audio sources in a multichannel underdetermined mixture. IEEE Trans. Signal Process. 58 (1), 121–133.

Alinaghi, A., Wang, W., Jackson, P., 2011. Integrating binaural cues and blind source separation method for separating reverberant speech mixtures. In: Proc. IEEE Internat. Conf. Acoustics, Speech and Signal Processing, pp. 209–212.

Barchiesi, D., Plumbley, M.D., 2012. Learning incoherent dictionaries for sparse approximation using iterative projections and rotations. In: Proc. ICML Workshop on Sparsity, Dictionaries and Projections in Machine Learning and Signal Processing, Edinburgh, Scotland.

Beck, A., Teboulle, M., 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. 2 (1), 183–202.

Berg, E.V., Friedlander, M.P., 2008a. Probing the pareto frontier for basis pursuit solutions. SIAM J. Sci. Comput. 31 (2), 890–912.

Berg, E.V., Friedlander, M.P., 2008b. Spgl1: Spectral projected gradient for l1 minimization. <http://www.cs.ubc.ca/labs/scl/spgl1/>

Blumensath, T., Davies, M.E., 2008. Gradient pursuits. IEEE Trans. Signal Process. 56 (6), 2370–2382.

Bofill, P., Zibulevsky, M., 2001. Underdetermined blind source separation using sparse representations. Signal Process. 81 (11), 2353–2362.

Chen, S.S., Donoho, D.L., Saunders, M.A., 1999. Atomic decomposition by basis pursuit. SIAM J. Sci. Comput. 20 (1), 33–61.

Cichocki, A., Amari, S., 2003. Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications. John Wiley.

Cichocki, A., Zdunek, R., Phan, A.H., Amari, S., 2009. Nonnegative Matrix and Tensor Factorizations – Applications to Exploratory Multi-way Data Analysis and Blind Source Separation. Wiley.

Comon, P., 1998. Blind channel identification and extraction of more sources than sensors. In: Advanced Signal Processing Algorithms, Architectures, and Implementations VIII, Vol. 3461, pp. 2–13.

Comon, P., 2004. Blind identification and source separation in 2×3 under-determined mixtures. IEEE Trans. Signal Process. 52 (1), 11–22.

Dai, W., Milenkovic, O., 2009. Subspace pursuit for compressive sensing signal reconstruction. IEEE Trans. Inform. Theory 55, 2230–2249.

Dai, W., Xu, T., Wang, W., 2012. Simultaneous codeword optimization (SIMCO) for dictionary update and learning. IEEE Trans. Signal Process. 60 (12), 6340–6353.

Daubechies, I., DeVore, R., Fornasier, M., Gunturk, S., 2010. Iteratively re-weighted least squares minimization for sparse recovery. Commun. Pure Appl. Math. 63 (1), 1–38.

Demmel, J.W., Heath, M.T., 1997. Applied numerical linear algebra. Society for Industrial and Applied Mathematics. SIAM.

Donoho, D.L., 2004. For most large underdetermined systems of linear equations the minimal l_1-norm solution is also the sparsest solution. Commun. Pure Appl. Math. 59, 797–829.

Donoho, D.L., 2006. Compressed sensing. IEEE Trans. Inform. Theory 52 (4), 1289–1306.

Donoho, D.L., Drori, I., Stodden, V., Tsaig, Y., 2005. Sparselab. <http://sparselab.stanford.edu/>.

Edelman, A., Arias, T.A., Smith, S.T., 1999. The geometry of algorithms with orthogonality constraints. SIAM J. Matrix Anal. Appl. 20 (2), 303–353.

Elad, M., Aharon, M., 2006. Image denoising via sparse and redundant representations over learned dictionaries. IEEE Trans. Image Process. 15 (12), 3736–3745.

Gowreesunker, B.V., Tewfik, A.H., 2008. Blind source separation using monochannel overcomplete dictionaries. In: Proc. IEEE Internat. Conf. Acoustics, Speech and Signal Processing, pp. 33–36.

Gowreesunker, B.V., Tewfik, A.H., 2009. A novel subspace clustering method for dictionary design. In: Proc. Internat. Conf. on Independent Component Analysis and Signal Separation, pp. 34–41.

Gribonval, R., Lesage, S., 2006. A survey of sparse component analysis for blind source separation: principles, perspectives, and new challenges. In: European Symposium on Artificial Neural Networks, pp. 323–330.

Gribonval, R., Vandergheynst, P., 2006. On the exponential convergence of matching pursuit in quasi-incoherent dictionaries. IEEE Trans. Inform. Theory 52 (1), 255–261.

Hulle, M.V., 1999. Clustering approach to square and non-square blind source separation. In: IEEE Workshop on Neural Networks for Signal Processing, 1999, pp. 315–323.

Hyvärinen, A., Karhunen, J., Oja, E., 2001. Independent Component Analysis. Wiley-Interscience.

Jafari, M.G., Plumbley, M.D., 2011. Fast dictionary learning for sparse representations of speech signals. J. Sel. Top. Signal Process. 5 (5), 1025–1031.

Jan, T., Wang, W., Wang, D.L., 2011. A multistage approach to blind separation of convolutive speech mixtures. Speech Commun. 53, 524–539.

Jourjine, A., Rickard, S., Yilmaz, O., 2000. Blind separation of disjoint orthogonal signals: demixing n sources from 2 mixtures. In: Proc. IEEE Internat. Conf. on Acoustics, Speech and Signal Processing, pp. 2985–2988.

Kim, S.-J., Koh, K., Lustig, M., Boyd, S., Gorinevsky, D., 2007a. An interior-point method for large-scale l1-regularized least squares. IEEE J. Select. Top. Signal Process. 1 (4), 606–617.

Kim, S.-J., Koh, K., Lustig, M., Boyd, S., Gorinevsky, D., 2007b. L1ls: l1-regularized least squares problem solver. <http://www.stanford.edu/boyd>.

Kowalski, M., Vincent, E., Gribonval, R., 2010. Beyond the narrowband approximation: wideband convex methods for under-determined reverberant audio source separation. IEEE Trans. Signal Process. 18 (7), 1818–1829.

Luo, Y., Wang, W., Chambers, J.A., Lambotharan, S., Proudler, I.K., 2006. Exploitation of source nonstationarity in underdetermined blind source separation with advanced clustering techniques. IEEE Trans. Signal Process. 54 (6), 2198–2212.

Makino, S., Lee, T.W., Sawada, H., 2007. Blind Speech Separation. Springer.

Mailhe, B., Barchiesi, D., Plumbley, M.D., 2012. INK-SVD: Learning incoherent dictionaries for sparse representations. In: Proc. IEEE Internat. Conf. on Acoustics, Speech and Signal Processing, Kyoto, Japan, pp. 3573–3576.

Mallat, S.G., Zhang, Z., 1993. Matching pursuits with time–frequency dictionaries. IEEE Trans. Signal Process. 41 (12), 3397–3415.

Mandel, M.I., Weiss, R.J., Ellis, D.P.W., 2010. Model based expectation-maximization source separation and localization dictionaries. IEEE Trans. Audio Speech Language Process. 18 (2), 382–394.

Mohimani, H., Babaie-Zadeh, M., Jutten, C., 2009. A fast approach for overcomplete sparse decomposition based on smoothed $\ell_0$norm. IEEE Trans. Signal Process. 57 (1), 289–301.

Needell, D., Tropp, J.A., 2008. Iterative signal recovery from incomplete and inaccurate samples. Appl. Comput. Harmon. Anal. 26, 301–321.

Nion, D., Lathauwer, L.D., 2008. An enhanced line search scheme for complex-valued tensor decompositions. Application in DS-CDMA. Signal Process. 88 (3), 749–755.

Parra, L., Spence, C., 2000. Convolutive blind source separation of nonstationary sources. IEEE Trans. Speech Audio Process. 8 (3), 320–327.

Pati, Y.C., Rezaiifar, R., Rezaiifar, Y.C.P.R., Krishnaprasad, P.S., 1993. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In: Proc. 27th Annual Asilomar Conf. on Signals, Systems, and Computers, pp. 40–44.

Pedersen, M.S., Wang, D., Larsen, J., Kjems, U., 2008. Two-microphone separation of speech mixtures. IEEE Trans. Neural Network 19 (3), 475–492.

Peleg, T., Eldar, Y.C., Elad, M., 2012. Exploiting statistical dependencies in sparse representations for signal recovery. IEEE Trans. Signal Process. 60 (5), 2286–2303.

Plumbley, M.D., Blumensath, T., Daudet, L., Gribonval, R., Davies, M.E., 2010. Sparse representations in audio and music from coding to source separation. Proc. IEEE 98 (6), 995–1005.

Rubinstein, R., Member, S., Zibulevsky, M., Elad, M., 2010. Double sparsity: learning sparse dictionaries for sparse signal approximation. IEEE Trans. Signal Process. 58 (3), 1558–1564.

Sawada, H., Araki, S., Makino, S., 2011. Underdetermined convolutive blind source separation via frequency bin-wise clustering and permutation alignment. IEEE Trans. Audio Speech Language Process. 19 (3), 516–527.

Smaragdis, P., 1998. Blind separation of convolved mixtures in the frequency domain. Neurocomputing 22, 21–34.

Sudhakar, P., 2011. Sparse Models and Convex Optimisation for Convolutive Blind Source Separation. Ph.D. Thesis, University of Rennes 1, France.

Tichavský, P., Koldovský, Z., 2011. Weight adjusted tensor method for blind separation of underdetermined mixtures of nonstationary sources. IEEE Trans. Signal Process. 59 (3), 1037–1047.

Tibshirani, R., 1996. Regression shrinkage and selection via the lasso. IEEE Trans. Audio Speech Language Process. 58 (1), 267–288.

Tropp, J.A., 2004. Greed is good: algorithmic results for sparse approximation. IEEE Trans. Inform. Theory 50 (10), 2231–2242.

Vincent, E., Araki, S., Bofill, P., 2009. The 2008 signal separation evaluation campaign: a community-based approach to large-scale evaluation. In: Proc. Internat. Conf. on Independent Component Analysis and Blind Signal Separation, pp. 734–741.

Vincent, E., Gribonval, R., Févotte, C., 2006. Performance measurement in blind audio source separation. IEEE Trans. Audio, Speech Language Process. 14 (4), 1462–1469.

Wang, W., Sanei, S., Chambers, J.A., 2005. Penalty function based joint diagonalization approach for convolutive blind separation of non-stationary sources. IEEE Trans. Signal Process. 53 (5), 1654–1669.

Wang, W., 2007. Squared euclidean distance based convolutive non-negative matrix factorization with multiplicative learning rules for audio pattern separation. In: Proc. 7th IEEE Internat. Symposium on Signal Processing and Information Technology, pp. 347–352.

Wang, W., Zou, X., 2008. Non-negative matrix factorization based on projected nonlinear conjugate gradient algorithm. In: Proc. ICA Research Network International Workshop, pp. 5–8.

Wang, W., 2008. Convolutive non-negative sparse coding. In: Proc. Internat. Joint Conf. Neural Network, pp. 3681–3684.

Wang, W., Luo, Y., Chambers, J.A., Sanei, S., 2008. Note onset detection via non-negative factorization of magnitude spectrum. EURASIP J. Adv. Signal Process 2008, Article ID 231367. http://dx.doi.org/10.1155/2008/231367.

Wang, W., Cichocki, A., Chambers, J.A., 2009. A multiplicative algorithm for convolutive non-negative matrix factorization based on squared euclidean distance. IEEE Trans. Signal Process. 57 (7), 2858–2864.

Xu, T., Wang, W., 2009. A compressed sensing approach for underdetermined blind audio source separation with sparse representation . In: Proc. IEEE Internat. Workshop on Statistical Signal Processing, pp. 493–496.

Xu, T., Wang, W., 2010. A block-based compressed sensing method for underdetermined blind speech separation incorporating binary mask. In: Proc. IEEE Internat. Conf. on Acoustics, Speech and Signal Processing, pp. 2022–2025.

Xu, T., Wang, W., 2011. Methods for learning adaptive dictionary for underdetermined speech separation. In: Proc. IEEE 21st Internat. Workshop on Machine Learning for, Signal Processing, pp. 1–6.

Yilmaz, O., Rickard, S., 2004. Blind separation of speech mixtures via time–frequency masking. IEEE Trans. Signal Process. 52, 1830–1847.

Zibulevsky, M., Pearlmutter, B.A., 2001. Blind source separation by sparse decomposition. Neural Comput. 13, 863–882.