Dictionary Learning for Sparse Representations with Applications to Blind Source Separation

Tao Xu

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey

Centre for Vision, Speech and Signal Processing
Faculty of Engineering and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

February 2013

Abstract

During the past decade, sparse representation has attracted much attention in the signal processing community. It aims to represent a signal as a linear combination of a small number of elementary signals called atoms. These atoms constitute a dictionary so that a signal can be expressed by the multiplication of the dictionary and a sparse coefficients vector. This leads to two main challenges that are studied in the literature, i.e. sparse coding (find the coding coefficients based on a given dictionary) and dictionary design (find an appropriate dictionary to fit the data). Dictionary design is the focus of this thesis.

Traditionally, the signals can be decomposed by the predefined mathematical transform, such as discrete cosine transform (DCT), which forms the so-called analytical approach. In recent years, learning-based methods have been introduced to adapt the dictionary from a set of training data, leading to the technique of dictionary learning. Although this may involve a higher computational complexity, learned dictionaries have the potential to offer improved performance as compared with predefined dictionaries.

Dictionary learning algorithm is often achieved by iteratively executing two operations: sparse approximation and dictionary update. We focus on the dictionary update step, where the dictionary is optimized with a given sparsity pattern. A novel framework is proposed to generalize benchmark mechanisms such as the method of optimal directions (MOD) and K-SVD where an arbitrary set of codewords and the corresponding sparse coefficients are simultaneously updated, hence the term *simultaneous codeword optimization (SimCO)*. Moreover, its extended formulation 'regularized SimCO' mitigates the major bottleneck of dictionary update caused by the singular points. First and second order optimization procedures are designed to solve the primitive and regularized SimCO. In addition, a tree-structured multi-level representation of dictionary based on clustering is used to speed up the optimization process in the sparse coding stage.

This novel dictionary learning algorithm is also applied for solving the underdetermined blind speech separation problem, leading to a multi-stage method, where the separation problem is reformulated as a sparse coding problem, with the dictionary being learned by an adaptive algorithm. Using mutual coherence and sparsity index, the performance of a variety of dictionaries for 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.

Finally, we propose a new method for joint dictionary learning and source separation. Different from the multistage method, the proposed method can simultaneously estimate the mixing matrix, the dictionary and the sources in an alternating and blind manner. The advantages of all the proposed methods are demonstrated over the state-of-the-art methods using extensive numerical tests.

Key words: Blind source separation (BSS), compressed sensing (CS), dictionary learning, sparse representation, sparse coding, numerical optimization, underdetermined blind speech separation, image denoising and separation

Email: t.xu@surrey.ac.uk

WWW: http://www.surrey.ac.uk/feps/

Acknowledgements

First of all, I would like to express my most sincere thanks to my principal supervisor, Dr. Wenwu Wang who has given me his guidance, enthusiasm and encouragement during my PhD research period. My PhD research cannot make such progress without his great support on theoretical development, experimental analysis and academic writing. Most of the advice and education from him will be helpful to my professional career in the future. I would also like to thank my second supervisor Dr. Philip Jackson for his kind behavior and support throughout my PhD studies. Further appreciations will be given to my colleagues in CVSSP including some graduated PhD students and friends. Their help since I joined the University of Surrey makes my PhD route totally different.

In addition, my appreciation goes to Dr. Wei Dai in Imperial College London with whom I collaborated on a joint project which led to important progress in this academic field. His great inspiration in academic research has motivated and impressed me during this fruitful collaboration period. Here thank Dr. Wenwu Wang again for providing this valuable opportunity to me. I also would like to thank the other two PhD students, Guangyu Zhou and Xiaochen Zhao from Dr. Wei Dai's research team. The cooperation with them is very pleasant.

Finally, I want to give special thanks to my parents. They look after me all the time and supply continual spiritual support during my study. Without this care and love, I cannot reach the final PhD stage and finish it successfully.

List of Figures

List of Tables

Contents

Acronyms and Mathematical Symbols

List of Acronyms

Acronym/Abbreviation	Meaning
BSS	Blind Source Separation
CASA	Computational Auditory Scene Analysis
CPP	Cocktail Party Problem
DFT	Discrete Fourier Transform
DCT	Discrete Cosine Transform
MDCT	Modified Discrete Cosine Transform
STFT	Short Time Fourier Transform
IBM	Ideal Binary Mask
ICA	Independent Component Analysis
ML	Maximum Likelihood
MMSE	Minimum Mean Squared Error
NMF	Non-negative Matrix Factorization
SDR	Signal to Distortion Ratio
SIR	Signal to Interference Ratio
SAR	Signal to Artifacts Ratio
SNR	Signal to Noise Ratio
MTD	Mixture-Trained Dictionary
STD	Source-Trained Dictionary
SimCO	Simultaneous Codeword Optimization
GAD	Greedy Adaptive Dictionary
MOD	Method of Optimal
ILS	Iterative Least Squares
RLS	Recursive Least Squares
MP	Matching Pursuit
BP	Basis Pursuit
GP	Gradient Pursuit
SP	Subspace Pursuit
OMP	Orthogonal Matching Pursuit
WDO	W-Disjoint Orthogonality
MAP	Maximum A Posteriori
DUET	Degenerate Unmixing Estimation Technique
L1LS	l1-Regularized Lease Squares
BMMCA	Blind Multichannel Morphological Component Analysis

List of Symbols

M	The number of mixtures
N	The number of sources
T	The length of mixtures and sources
m	The dimension of training signal
n	The number of training signals
d	The number of atoms in dictionary
$\mathbf{Z}$	Mixture matrix
$\mathbf{A}$	Mixing matrix
$\mathbf{S}$	Source matrix
$\mathbf{Y}$	Training data matrix
$\mathbf{D}$	Dictionary
$\mathbf{X}$	Coefficients matrix
$\mathbf{V}$	Noise matrix
$\mathbf{r}$	Residual of the signal
$\mathbf{R}$	Residual matrix
$\mathbf{E}$	Error matrix
Ω	Index set of sparsity pattern
$\mathcal{I}$	Index set of the codewords to be updated
$\mathcal{I}^c$	Index set complementary to $\mathcal{I}$
$\mathcal{U}_{m,1}$	The Stiefel manifold
$\mathcal{G}_{m,1}$	The Grassmann manifold
∇	Gradient of the cost function

∇^2	Hessian of the cost function
$\mathbf{H}$	Newton direction matrix
$\mathbf{\Phi}$	Dictionary for reshaped signals
$\hat{\mathbf{A}}$	Estimated mixing matrix
$\mathbf{b}$	Reshaped mixtures vector
$\mathbf{f}$	Reshaped sources vector
$\mathbf{y}$	Reshaped coefficients vector
$\mathbf{T}$	Training sources matrix
$\mathbf{G}$	Corresponding coefficients matrix to $\mathbf{T}$
$\mathbf{M}$	Reshaped mixing matrix
P	The number of blocks
L	The number of samples in consecutive speech frame
F	The number of samples of speech frame overlap
$\nu(\mathbf{\Phi})$	The mutual coherence of a dictionary

List of Publications

PUBLICATIONS

Journal Articles

T. Xu, W. Wang, and W. Dai, "Joint Blind Source Separation and Adaptive Dictionary Learning with Compound Optimization." *to be submitted.*

T. Xu, W. Wang, and W. Dai, "Sparse Coding with Adaptive Dictionary Learning for Underdetermined Blind Speech Separation." *Speech Communication*, vol. 55, no. 3, pp. 432–450, 2013

W. Dai, T. Xu, and W. Wang, "Simultaneous Codeword Optimisation (SimCO) for Dictionary Update and Learning." *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6340–6353, 2012.

Conference Papers

X. Zhao, T. Xu, G. Zhou, W. Dai, and W. Wang, "Joint Image Separation and Dictionary Learning." in *Proc. 18th International Conference on Digital Signal Processing (DSP 2013)*, Santorini, Greece, 1-3 July 2013, submitted.

W. Dai, T. Xu, and W. Wang, "Dictionary Learning and Update based on Simultaneous Codeword Optimization (SIMCO)." in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012)*, Kyoto, Japan, March 25-30, 2012.

T. Xu, W. Wang, and W. Dai, "Fast Dictionary Learning Algorithm via Codeword Clustering and Hierarchical Sparse Coding." in *Proc. 9th IMA International Conference on Mathematics in Signal Processing (IMA 2012)*, Birmingham, UK, December 17-20, 2012.

W. Dai, T. Xu, and W. Wang, "Simultaneous Codeword Optimization (SimCO) for Dictionary Learning." in *Proc. 49th Annual Allerton Conference on Communication, Control, and Computing (ALLERTON 2011)*, Monticello, Illinois, USA, Sept 28-30, 2011. (Invited Paper)

T. Xu and W. Wang, "Methods for Learning Adaptive Dictionary for Underdetermined Speech Separation." in *Proc. IEEE 21st International Workshop on Machine Learning for Signal Processing (MLSP 2011)*, Beijing, China, Sept 18-21, 2011.

T. Xu and W. Wang, "Learning Dictionary for Underdetermined Blind Speech Separation Based on Compressed Sensing Method." in *Proc. INSPIRE Conference on Information Representation and Estimation (INSPIRE 2010)*, London, United Kingdom, 7th September 2010.

T. Xu and W. Wang, "A Block-based Compressed Sensing Method for Underdetermined Blind Speech Separation Incorporating Binary Mask." in *Proc. 35th International Conference on Acoustics, Speech and Signal Processing (ICASSP)* in Dallas, Texas, United States, 17th April 2010.

T. Xu and W. Wang, "A Compressed Sensing Approach for Underdetermined Blind Audio Source Separation with Sparse Representations." in *Proc. IEEE International Workshop on Statistical Signal Processing (SSP 2009)*, Cardiff, United Kingdom, August 31-Sept 3, 2009.

Chapter 1

Introduction

1.1 Problem Description and Motivation

Sparse signal representation has recently drawn much attention in signal processing and information theory [64, 98, 118, 149]. The key idea of sparse signal representation is to assume that the signals are sparse, or can be decomposed into the combination of signal components with a small number of significant coefficients, which means that most values in the signal or its transformed coefficients are zero, except for a few nonzero values. In sparse representation, the elementary signals are called atoms or codewords and the collection of all the atoms is referred to as a dictionary. The dictionary can be either based on a mathematical model of the data or learned directly from the data. The mathematical models that are often used include discrete cosine transform (DCT), short-time Fourier transform (STFT), Gabor transform, wavelets [136], curvelets [33], contourlets [55] and bandelets [97], which can reveal certain structures of a signal such as sparsity in a different domain. Such dictionaries are relatively easy to obtain and more suitable for generic signals. Other than choosing atoms from a complete dictionary (square), such as DCT, STFT basis, the atoms can be chosen from an overcomplete (redundant) dictionary in which the number of basis vectors (atoms) exceeds the dimensionality of the signal space. The signal can be represented by more than one combination of different atoms based on the overcomplete dictionary. The representation of the signal by an overcomplete dictionary can be sparser and has a greater

flexibility in matching the structure of the signals as well as being more robust in the presence of noise [98]. The overcomplete dictionary can be generated either by the combination of the classical complete dictionaries or by dictionaries learned from training data. This raises a key problem emerging in signal processing society during the recent years: 'dictionary design' [147].

The goal of dictionary learning is to seek an over-complete dictionary from which every training signal can be best approximated by a linear combination of only a few codewords. This task is often achieved by iteratively executing two operations: sparse coding and dictionary update. In the literature, there are two benchmark mechanisms for the update of a dictionary. The first approach, for example the method of optimal directions (MOD) algorithm [61], is characterized by searching for the optimal codewords while fixing the sparse coefficients. In the second approach, represented by the K-SVD method [5], one codeword and the related sparse coefficients are simultaneously updated while all other codewords and coefficients remain unchanged. We propose a novel framework that generalizes the aforementioned two methods. The unique feature of our approach is that one can update an arbitrary set of codewords and the corresponding sparse coefficients simultaneously: when sparse coefficients are fixed, the underlying optimization problem is the same as that in the MOD algorithm; when only one codeword is selected for update, it can be proved that the proposed algorithm is equivalent to the K-SVD method; and more importantly, our method allows to update all codewords and all sparse coefficients simultaneously, hence the term *simultaneously codeword optimization (SimCO)*. Under the proposed framework, we design two algorithms. We prove the convergence and where to converge under certain conditions. Simulations demonstrate that our approach excels the benchmark K-SVD in terms of both learning performance and running speed. The learning process, however, may involve a higher computational complexity, rendering the algorithms to be less practical in computation extensive applications, for example, when dealing with large scale or high-dimensional data. Therefore, we then propose a new method to improve the computational efficiency of the dictionary learning algorithms based on codeword clustering and hierarchical sparse coding, and we apply this method to the SimCO algorithm. The numerical results have shown the advantage of this tree structure sparse coding

operation.

The proposed novel dictionary learning algorithm is then applied in the real applications e.g. blind source separation (BSS), which is also one of the central problems in signal processing. The aim of BSS is to separate the superposition of multiple source signals when the information about the mixing process and sources is unknown or very limited. For example, in acoustic processing, human's auditory perception of sound mixtures results from the vibration of the ear drum by superposition of many different air-pressure signals, emitted from different audio sources at the same time. The brain can separate the sources from the mixtures quite efficiently according to the theory of auditory scene analysis (ASA) [29]. For a machine, however, it is a difficult task to separate the sources from the mixtures, widely known as the machine cocktail party problem [41]. Many approaches have been proposed for this problem including computational auditory scene analysis (CASA) [154] and BSS.

In the simplest mixing models, each mixture consists of different weighted source signals, which is called 'instantaneous case' (spatial-only). However, in the real-world applications, such as in acoustics, the mixing process is more complex. The observed mixtures consist of original source signals, at different time delays and amplitude levels, as well as multipath copies of the sources, distorted by the environment, which is called 'convolutive case' (spatio-temporal). The convolutive BSS model has been investigated in many publications [6,84,93,108,133]. Although the convolutive BSS model is not the focus of this thesis, the methods developed here can also be used in convolutive speech separation algorithms.

In this thesis, we focus on the instantaneous BSS problem. Given M mixtures obtained by mixing from N sources via an unknown $M \times N$ mixing matrix, the purpose of BSS is to estimate the N sources from the M mixtures. When $M = N$, this can be called the even-determined case. When $M > N$, this is called the overdetermined case, and for $M < N$, the underdetermined case. Many algorithms have been successfully developed for the BSS problem, especially for the even or over determined cases. Independent component analysis (ICA) is a well-known family of BSS techniques based on the assumption that the source signals are statistically independent [80]. In the first two

cases, once the mixing matrix $\mathbf{A}$ is estimated, the process of separation can be achieved by solving a linear equation. The underdetermined case is an ill-posed problem and cannot be addressed by an exact inverse operation such as ICA. Solving this problem normally requires making some assumptions (e.g. sparsity) about the sources, and the source estimation is usually achieved by using a sparse representation technique.

Although several approaches [105] have been developed to address this problem, which are reviewed in next chapter, it remains an open problem, especially for speech source signals. Using sparse representation, we propose an approach to improve the separation performance for speech signals based on dictionary learning, where the BSS problem is reformulated to the sparse signal recovery problem. However, the optimization process for source estimation is computationally demanding as the microphone signals of full length are stacked into a single vector, resulting in a large dimension of the measurement matrix, as well as the signal dictionary. Therefore, a block based operation is incorporated into the approach. This multi-stage method has shown good performance compared with the state-of-the-art approaches.

This multi-stage method uses, however, the dictionaries pretrained from the speech data (either clean sources or mixtures). In BSS, clean speech sources are usually not available *a priori*, while training dictionaries from mixtures often render poor and noisy atoms. Therefore, we propose a new BSS method that simultaneously estimates the mixing matrix and sources and learns the dictionaries. We adapt the SimCO optimization framework to this method in order to keep the optimization on a product of Grassmann manifolds, which ensures that the constraints on the column norms of the mixing matrix and dictionaries are satisfied. The numerical experiments show the performance advantages of the proposed method over the benchmark algorithms.

Overall, in this thesis, we investigate theories and techniques based on sparse representations including the foundational methods in dictionary learning, sparse coding and use them to address blind speech and image separation problems.

1.2 Contributions

There are some major contributions of this thesis as described below:

1. A novel optimization framework SimCO is proposed for dictionary learning problem, where an *arbitrary* subset of the codewords is allowed to be updated simultaneously. The proposed framework has the following characteristics.

SimCO generalizes MOD and K-SVD. We show that the MOD algorithm is in fact an inexact Newton's method under the proposed framework while K-SVD can be viewed as a special case of SimCO where only one codeword is selected for update at each iteration. The SimCO framework is general and flexible. Two possible algorithmic implementations are presented: one is based on gradient descent and the other uses a Newton's method.

The proposed optimization framework allows the discovery of the bottleneck of dictionary update. As opposed to traditional formulations, in the SimCO framework, the objective function involves only the dictionary by treating sparse coefficients as a function of the dictionary. In this way, the gradient can be easily computed and analyzed. Surprisingly, against the traditional belief that local minima are the major problem, we empirically discover that singular points are the bottleneck.

Regularized SimCO is introduced to mitigate the singularity problem. To avoid the singularity problem, an additive regularization term is introduced. The resulting objective function is differentiable. Significant improvement in empirical performance is observed. This, from another angle, verifies that singularity is the bottleneck.

A fast version of SimCO is proposed for improving the computational efficiency of dictionary learning algorithms based on codeword clustering and hierarchical sparse coding. Specifically, we develop a tree-structured multi-level representation of dictionary based on clustering, which is used to derive a hierarchical algorithm in the sparse coding stage. The proposed idea is then applied to the original SimCO algorithm resulting in a new algorithm: fast SimCO. Numerical examples are provided to show its computational efficiency and the performance for image denoising.

2. 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, such as basis pursuit (BP) [40], the least squares method L1LS [90], matching pursuit (MP) [107] and orthogonal matching pursuit (OMP) [122, 150], 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 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 techniques. 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 DCT. Among the dictionary learning algorithms compared, SimCO offers the best performance as compared with others [5, 61]. 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 [28] [69, 70] in the source separation evaluation campaigns using the same datasets and evaluation approach.

3. We propose a new joint dictionary learning and source separation method which unifies two stages (dictionary learning and source separation) in the optimization process. The coarsely separated sources are used to learn the dictionaries which are used in turn to refine the source estimate. The refined source is then used to update the dictionary further in the next iteration. The process is alternated until the cost function is optimized. By adding the regularization term we are also able to force the search path away from singular points to achieve improved performance. To show the performance of our

proposed method, synthetic data experiment and real application experiment (blind image separation and denoising) are provided to compare this novel joint method with benchmark methods such as ICA [80], generalized morphological component analysis (GMCA) [27] and blind multichannel morphological component analysis (BMMCA) [1].

1.3 Thesis Structure

In the next chapter, we will give a literature survey for the technical field that we mentioned above. The details of the proposed SimCO and fast SimCO are described in Chapter 3. The multi-stage method for underdetermined BSS is presented in Chapter 4. Finally, the joint dictionary learning and source separation method is presented in Chapter 5, followed by conclusions and future works in Chapter 6.

Chapter 2

Background and Literature Survey

2.1 Sparse Representations

Sparse representations have found successful applications in data interpretation [119, 146], source separation [71,162,168], signal denoising [60,83], coding [92,128,134], classification [77,104,135], recognition [159], impainting [3,39] and many more (see e.g. [16]). Finding the sparse representations of signals is an important issue in signal processing, and it is often referred to as sparse decomposition or sparse approximation. It aims to estimate the coding coefficient vector $\mathbf{x} \in \mathbb{R}^{d\times 1}$ from the data vector $\mathbf{y} \in \mathbb{R}^{m\times 1}$ by solving the following ℓ_0 minimization problem

$$\min \| \mathbf{x} \|_0 \qquad \text{s.t.} \qquad \mathbf{y} = \mathbf{D}\mathbf{x} \tag{2.1}$$

where the dictionary $\mathbf{D} \in \mathbb{R}^{m\times d}$ consists of basis function vectors called atoms. $\| \mathbf{x} \|_0$ is the ℓ_0 norm measuring the sparseness of $\mathbf{x}$ and $\mathbf{D}$ is an overcomplete dictionary matrix. However, the process of finding the representation with the smallest number of atoms from an overcomplete dictionary is a NP-hard problem [66], which is not a good choice in practice. In this case, signals no longer have unique representations and finding the sparsest representation (i.e. the minimum number of non-zero coefficients) is non-trivial. Therefore, the sparse coding algorithms typically fall into two broad classes: convex relaxation and greedy iteration.

For convex relaxation methods, the problem formulation in (2.1) which is non-convex is replaced by some sub-optimal schemes. Many works have already shown that near optimal performance can be achieved by using different relaxations for the sparsity measure, e.g. the ℓ_1 norm [40], the ℓ_p norm [53], and the smoothed ℓ_0 norm [110]. Such relaxations have led to a wide range of algorithms for signal reconstruction, e.g. the basis pursuit technique based on linear programming [40], the least squares method L1LS [90], regression shrinkage and selection (LASSO) [144], and the focal under-determined system solver (FOCUSS) [68].

In contrast, greedy algorithms operate iteratively on the signal measurements, by deriving the locally optimal solution at each stage to reach a global optimization after a number of iterations. Compared to convex relaxation approaches, these algorithms are often more efficient and better for large scale problems [36], such as matching pursuit [107], orthogonal matching pursuit (OMP) [122, 150], iterative thresholding [25], CoSaMP [114] and subspace pursuit [51]. There are four sparse coding methods used in this thesis and we give a brief introduction below for these methods.

Basis Pursuit

It has been shown in [40] that the solution to the ℓ_0 minimization problem is essentially equivalent to the solution of the following ℓ_1 minimization problem

$$\min \| \mathbf{x} \|_1 \qquad \text{s.t.} \qquad \mathbf{y} = \mathbf{D}\mathbf{x}. \tag{2.2}$$

This ℓ_1 norm minimization problem corresponds to the so-called basis pursuit (BP) [40] and can be solved through linear programming. One starts from a solution to the overcomplete representation problem $\mathbf{y} = \mathbf{D}\mathbf{x}^{(0)}$ then iteratively updates the coefficients while keeping $\mathbf{y} = \mathbf{D}\mathbf{x}^{(k)}$, as follows. First, we choose an initial basis matrix $\mathbf{B}$ which is a square matrix having the same rank as $\mathbf{D}$ and consists of the selected columns of $\mathbf{D}$, i.e. the smallest possible complete dictionary. Then, we update the basis by swapping a column of $\mathbf{B}$ with an unselected column in $\mathbf{D}$. When the basis cannot be further improved based on a pre-defined criterion, we reach the optimal solution. Finally, $\mathbf{x}$ can be readily computed by $\mathbf{B}^{-1}\mathbf{y}$.

ℓ_1-Regularized Least Squares Method

A least squares method with ℓ_1 regularization (L1LS) is described in [90] for large scale problems. This interior-point method uses the preconditioned conjugate gradient algorithm to compute the search direction. It solves the following problem:

$$\min \| \mathbf{y} - \mathbf{D}\mathbf{x} \|_2^2 + \lambda \| \mathbf{x} \|_1 \tag{2.3}$$

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 preconditioned conjugate gradients algorithm [54] is applied to update the coefficients. Note that for a "small" λ, this method will become equivalent to basis pursuit (BP).

Matching Pursuit

Apart from the convex relaxation approaches BP and L1LS described above, there are alternative greedy methods for recovering the speech signals, such as matching pursuit (MP) [107]. The basic idea of MP is to represent a signal as a weighted sum of atoms using Equation (2.4) which involves finding the "best matching" projections of multidimensional data onto an over-complete dictionary,

$$\mathbf{y} = \sum_{i=1}^{k} x_i \mathbf{d}_{\gamma_i} + \mathbf{r}^{(k)} \tag{2.4}$$

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

$$\mathbf{r}^{(i+1)} = \mathbf{r}^{(i)} - x_i \mathbf{d}_{\gamma_i} \tag{2.5}$$

where $x_i = < \mathbf{r}^{(i)}, \mathbf{d}_{\gamma_i} >$, and $< , >$ is an inner product operation. Then the process is repeated until the signal is satisfactorily decomposed.

Orthogonal Matching Pursuit

The orthogonal matching pursuit (OMP) [122] 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:

$$\mathbf{y} = \sum_{i=1}^{k} x_i \mathbf{d}_{\gamma_i} + \mathbf{r}^{(k)}, s.t. < \mathbf{r}^{(k)}, \mathbf{d}_{\gamma_i} >= 0 \tag{2.6}$$

As proven in [122] 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.

There are also many other sparse coding methods developed in recent years such as [25, 51], together with our novel tree-structured OMP algorithm [163] to be presented in Chapter 3 of this thesis.

2.2 Compressed Sensing

Compressed sensing (also known as compressive sensing, compressive sampling, or sparse sampling) [34, 56] is a signal processing technique for efficiently acquiring and reconstructing a signal, by finding solutions to underdetermined linear systems. This takes advantage of the signal's sparseness or compressibility in some domain, allowing the entire signal to be determined from reducing the number of measurements necessary to reconstruct a signal. It is based on the principle that the object has some underlying sparse representation, i.e. that it can be described using a small number of non-zero coefficients.

The traditional Shannon-Nyquist sampling theorem states that a certain minimum number of samples is required to perfectly recover a signal, which must sample at least two times faster than the signal bandwidth. In many applications, including digital image and video capture, the data has to be compressed prior to storage or transmission because the Nyquist rate may be too high. However, compressed sensing (CS) techniques can capture and represent compressible signals at a rate significantly below the Nyquist

rate. The fundamental idea behind CS: rather than first sampling at a high rate and then compressing the sampled data, we would like to find ways to directly sense the data in a compressed form, i.e., at a lower sampling rate [17].

There are two important parts of compressed sensing: the sampling strategy and the reconstruction algorithm [17]. In other words, once a signal is known to be sparse in a specific basis, one of the main challenges is to find a set of measurement tools (producing the compressed measurements) and the attendant nonlinear solver that reconstructs the original full signal. The compressible signal is randomly projected to a low dimensional space, using an appropriate sampling matrix, then the nonlinear reconstruction techniques developed for finding sparse representations can be used to decode the signal from the compressed measurements. There are theoretical results analyzing the minimum number of measurements required to produce the original signal when the specific measurement matrices and nonlinear solvers were given [56]. In all cases, the number of compressed measurements in compressed sensing is expected to be much less than the sampled measurements in traditional Nyquist sampling constraints.

One of the key differences between conventional sampling and compressed sensing is the sampling strategy. The measurement matrix in CS must allow the reconstruction of the signal from measurements, in which the problem appears ill-conditioned. It needs to follow the restricted isometry property (RIP) [35] while a related condition, referred to as incoherence [58], is necessary. Both the RIP and incoherence can be achieved with high probability simply by selecting the measurement matrix as an appropriate random matrix. Certain families of matrices can satisfy this, for example, the random Gaussian measurement matrix.

Another key difference between conventional sampling and compressed sensing is that the reconstruction operator is nonlinear [37]. Essentially this selects the significant coefficients for the data in some sparse representation and then calculates a least squares approximation using the associated basis functions. Theoretical study of compressed sensing has been focusing on proving that near optimal performance is possible by using either a convex relaxation that amounts to solving a linear or quadratic program or greedy algorithm that iteratively selects the coefficients one at a time. There exist

nonlinear solvers that can be used to reconstruct the original signal from its compressed measurements on incoherent bases. Reconstruction algorithms span a wide range of techniques that include Greedy [107], Linear Programming [40], Bayesian [85] and Iterative Thresholding [25]. There are plenty of tutorials and reviews [15, 32, 37] by Baraniuk, Candes and Wakin respectively to discuss compressed sensing while more academic papers and applications based on this emerging technique is currently appearing.

The multi-stage approach based on compressed sensing is presented in Chapter 4 of this thesis for solving underdetermined BSS problems. This work is motivated by reformulating the BSS problem to a compressed sensing model. However, the reconstruction algorithm i.e. sparse coding, is the core technique used in this approach. Therefore, we use the term 'sparse signal recovery' instead of 'compressed sensing' in that chapter.

2.3 Dictionary Design and Learning

2.3.1 Predefined and Learned Dictionaries

Designing dictionary is the key issue in sparse coding algorithms because sparse decompositions of a signal highly rely on the degree of fitting between the data and the dictionary. The dictionary can be either based on a mathematical model of the data or learned directly from the data. The mathematical models, that can be used for generating dictionaries include discrete cosine transform (DCT), short-time Fourier transform (STFT), Gabor transform, wavelets [136], curvelets [33], contourlets [55] and bandelets [97]. Such dictionaries are relatively easy to obtain and suitable for generic signals. In learning-based approaches, the dictionaries are adapted from a set of training data [60, 61, 67, 74, 82, 94, 119, 120, 127, 158]. Although this may involve higher computational complexity, learned dictionaries have the potential to offer improved performance as compared with predefined dictionaries, since the atoms are derived to capture the salient information directly from the signals.

One of the early algorithms that adopted such a two-step structure was proposed by Olshausen and Field [119, 120], where a maximum likelihood (ML) learning method was used to sparsely code the natural images upon a redundant dictionary. The sparse

approximation step in the ML algorithm [119] which involves probabilistic inference is computationally expensive. In a similar probabilistic framework, Kreutz-Delgado et al. [94] proposed a maximum a posteriori (MAP) dictionary learning algorithm, where the maximization of the likelihood function as used in [119] is replaced by the maximization of posterior probability that a given signal can be synthesized by a dictionary and the sparse coefficients. Based on the same ML objective function as in [119], Engan [61] developed a more efficient algorithm, called the method of optimal directions (MOD), in which a closed-form solution for the dictionary update has been proposed. This method is one of the earliest methods that implements the concept of sparification process [130]. Several variants of this algorithm, such as the iterative least squares (ILS) method, have also been developed which were summarized in [62]. A recursive least squares (RLS) dictionary learning algorithm was recently presented in [137] where the dictionary is continuously updated as each training vector is being processed, which is different from the ILS dictionary learning method. Aharon, Elad and Bruckstein developed the K-SVD algorithm in [5] by generalizing the K-means algorithm for dictionary learning. This algorithm uses a similar block-relaxation approach to MOD, but updates the dictionary on atom-by-atom basis, without having to compute matrix inversion as required in the original MOD algorithm. The majorization method was proposed by [164] in which the original objective function is substituted by a surrogate function in the each step of optimization process.

In contrast to generic dictionaries described above, learning structure-oriented parametric dictionaries has also been attracted attention in this academic field. For example, a Gammatone generating function has been used by Yaghoobi et al. [165] to learn dictionaries from audio data. In [132], a pyramidal wavelet-like transform was proposed to learn a multiscale structure in the dictionary. Other constraints have also been considered in the learning process to favor the desired structures of the dictionaries, such as the translation-invariant or shift-invariant characteristics of the atoms imposed in [4, 24, 31, 102, 141] and the orthogonality between subspaces enforced in [73], and the de-correlation between the atoms promoted in [86]. An advantage of parametric dictionary lies in its potential of reducing the number of free parameters and thereby leads to more efficient implementation and better convergence of dictionary learning algo-

rithms [130]. Other recent efforts in dictionary learning include the seek of robust and computationally efficient algorithms, such as [83,95,103], and learning dictionaries from multimodal data [38,111]. Comprehensive reviews of dictionary learning algorithms can be found in recent survey papers e.g. [130] [147].

2.3.2 Benchmark Approaches for Dictionary Update

The dictionary learning problem can be formulated as follows: let $\mathbf{Y} \in \mathbb{R}^{m\times n}$ be the training data, where each column of $\mathbf{Y}$ corresponds to one training sample; one is looking for the solution of the following optimization problem

$$\begin{aligned} &\min_{\mathbf{D}\in\mathbb{R}^{m\times d},\, \mathbf{X}\in\mathbb{R}^{d\times n}} \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2, \\ &\text{subject to } \|\mathbf{D}_{:,i}\|_2 = 1,\ \forall 1 \le i \le d. \end{aligned} \tag{2.7}$$

Here, the matrices $\mathbf{D}$ and $\mathbf{X}$ are often referred to as a dictionary and the corresponding coefficients respectively, and $\mathbf{D}_{:,i}$ denotes the ith column (i.e., the ith codeword) of the dictionary $\mathbf{D}$. In practice, it is typical that $m < d < n$, i.e., an over-complete dictionary is considered and the number of training samples is larger than the number of codewords. Generally speaking, the optimization problem (2.7) is ill-posed unless there are extra constraints on $\mathbf{X}$. The most common constraint on $\mathbf{X}$ is that $\mathbf{X}$ is sparse, i.e., the number of nonzero entries in $\mathbf{X}$, compared with the total number of entries, is small.

Dictionary learning algorithms are often established on an optimization process involving the iteration between two stages: sparse approximation and dictionary update. First an initial dictionary is given and a signal is decomposed as a linear combination of only a few atoms from the initial dictionary. Then the atoms of the dictionary are trained with fixed or sometimes unfixed weighting coefficients. After that, the trained dictionary is used to compute the new weighting coefficients. The process is iterated until the most suitable dictionary is obtained eventually.

There are different formulations for dictionary update stage, leading to substantially different algorithms. In the following subsections, we introduce three methods: method of optimal directions (MOD), K-SVD and greedy adaptive dictionary (GAD) algorithms,

Algorithm 1 A typical dictionary learning algorithm

Task: find the best dictionary to represent the data sample matrix $\mathbf{Y}$.
Initialization: Set the initial dictionary $\mathbf{D}^{(1)}$. Set $J = 1$.
Repeat until convergence (use stop rule):

- Sparse coding stage: Fix the dictionary $\mathbf{D}^{(J)}$ and update $\mathbf{X}^{(J)}$ using some sparse coding technique.
- Dictionary update stage: Update $\mathbf{D}^{(J)}$, and $\mathbf{X}^{(J)}$ as appropriate.
- $J = J + 1$.

which will be used as baseline algorithms in our experimental evaluations.

MOD

Method of optimized directions (MOD) can be used to find the dictionary for a finite learning set $\mathbf{Y}$ which is decomposed as a sparse representation with sparseness constraint defined by either ℓ_0-norm (the number of non-zero coefficient is limited) or ℓ_1-norm (the sum of absolute values of coefficients is limited). In MOD, the optimization problem of Equation (2.7) is iteratively solved in three steps. Firstly, we find $\mathbf{X}$ keeping $\mathbf{D}$ fixed, which is a sparse approximation problem. Then $\mathbf{X}$ is fixed to find $\mathbf{D}$ using the least squares solution: $\mathbf{D} = (\mathbf{Y}\mathbf{X}^T)(\mathbf{X}\mathbf{X}^T)^{-1} = \mathbf{B}\mathbf{A}^{-1}$. It is convenient to define the matrices $\mathbf{B} = \mathbf{Y}\mathbf{X}^T$ and $\mathbf{A} = \mathbf{X}\mathbf{X}^T$. Finally, $\mathbf{D}$ is normalized, i.e. scale each column vector (atom) of $\mathbf{D}$ to unit norm. However, the normalization step may be skipped if $\mathbf{D}$ is almost normalized and the ℓ_0-sparseness is used. Sparse coding step is computationally demanding, and with this setup there will be a lot of computational effort between each dictionary update, thus slowing down the learning process. This is especially true with large training sets.

K-SVD

The K-SVD algorithm aims to iteratively find the best dictionary to represent the training signals by approximating the solution to Equation (2.7). 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 training signals in $\mathbf{Y}$ 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 the Equation (2.7) 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 $\mathbf{E}_j$ is computed by (2.8).

$$\mathbf{E}_j = \mathbf{Y} - \sum_{i \neq j} \mathbf{d}_i \mathbf{x}_i^T \tag{2.8}$$

where $\mathbf{d}_i$ is the ith column of the dictionary matrix $\mathbf{D}$, $\mathbf{x}_i^T$ is the ith row of the coefficient matrix $\mathbf{X}$ and $\mathbf{E}_j$ stands for the residual when the jth atom is removed from the dictionary matrix. The SVD decomposition is then applied to $\mathbf{E}_j$ to find alternative $\mathbf{d}_i$ and $\mathbf{x}_i^T$ for approximating $\mathbf{E}_j$ 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.

GAD

The greedy adaptive dictionary (GAD) algorithm [83] learns the dictionary atoms based on an iterative process using the sparsity index defined as follows:

$$\sigma_j = \frac{\| \mathbf{y}_j \|_1}{\| \mathbf{y}_j \|_2} \tag{2.9}$$

where $\| \cdot \|_1$ and $\| \cdot \|_2$ denote the ℓ_1- and ℓ_2-norm respectively and $\mathbf{y}_j$ is the column vector of $\mathbf{Y}$. The sparsity index measures the sparsity of a signal, where the smaller σ_j, the sparser the signal vector $\mathbf{y}_j$.

The GAD algorithm begins with the definition of a residual matrix $\mathbf{R}^k = [\mathbf{r}_1^k, ..., \mathbf{r}_J^k]$, where $\mathbf{r}_j^k$ is a residual column vector corresponding to the j-th column of $\mathbf{R}^k$. This residual matrix changes at each iteration k and is initialized to $\mathbf{Y}$. The dictionary is

then built by selecting the residual vector that has the lowest sparsity index j.

$$\hat{j} = \arg\min_{j} \frac{\| \mathbf{r}_j^k \|_1}{\| \mathbf{r}_j^k \|_2} \tag{2.10}$$

Then $\mathbf{r}_{\hat{j}}^k$ 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.

2.4 Blind Source Separation

2.4.1 What is Blind Source Separation

Source separation arises in a variety of signal processing applications, ranging from speech processing to medical image analysis [105] by making assumptions about the sources. When the information about the mixing process and sources is limited, the problem is called blind source separation (BSS) [84]. One well-known application of blind source separation is for processing acoustic mixtures, which is often referred to as the cocktail party problem [41]; that is the separation of individual voices from a myriad of voices in an uncontrolled acoustic environment such as a cocktail party environment. Sensor observations in a natural environment are confounded by room reverberations and consequently the unmixing process needs to identify a source arriving from multiple directions at different times as one individual source. Generally, BSS techniques depart from this difficult real-world scenario and make less realistic assumptions about the environment in order to make the problem more tractable [105]. There are typically three assumptions that are often made about the environment. The most basic assumption is the instantaneous case, where sources arrive instantly at the sensors but with differing signal intensity. An extension to this assumption, where the arrival delays between sensors are also considered, is known as the anechoic case. The anechoic case can be further extended by considering multiple paths between each source and each sensor which results in the echoic case, sometimes known as convolutive mixing [84].

The mathematic model is:

$$z_j(t) = \sum_{i=1}^{N} \sum_{p=1}^{P} h_{ji}(p) s_i(t-p+1) \qquad (j = 1, ..., M) \tag{2.11}$$

where s_i and z_j are the source and mixture signals respectively, h_{ji} is a P-point room impulse response from source s_i to microphone z_j. Hence the separation problem in the convolutive situation is more complex than in the instantaneous situation. There have been many methods to solve the convolutive blind source separation problem [8,9] [106,157]. Blind source separation techniques are not only applied to acoustic signals but also the decomposition of biomedical data such as electroencephalography (EEG) [109], functional magnetic resonance imaging (fMRI) [30] and magnetoencephalography (MEG) [14,47] as well as real time robot audition [112], digital watermark attacks [48] and financial time series analysis [13].

During nearly two decades, a great number of papers have been published for studying blind source separation problems. The earliest approach traces back to [76] which tried to separate an instantaneous linear even-determined mixture of non-Gaussian independent sources. In this work, Herault and Jutten proposed a solution that used a recurrent artificial neural network to separate the unknown sources, with the crucial assumption being that the underlying signals were independent. This early work led to the pioneering adaptive algorithm of [88]. In [99] unsupervised learning rules have been proposed that maximize the average mutual information between the inputs and outputs of an artificial neural network. Mutual information was the most natural measure of independence and showed that maximizing the non-Gaussianity of the source signals was equivalent to minimizing the mutual information between them. A blind source separation algorithm called Infomax is developed in [20], which is similar in spirit to that of Linsker and uses an elegant stochastic gradient learning rule that was proposed by Amari, Cichocki and Yang in [7]. The idea of non-Gaussianity of sources was used by Hyvarinen in [81] to develop their fast ICA algorithm. As an alternative approach to mutual information based separation method, [65] proposed maximum likelihood estimation, an approach elaborated by Pham, Garrat and Jutten in [126], although the Infomax algorithm and maximum likelihood estimation are essentially equivalent.

The early years of BSS research concentrated on solutions for even-determined and over-determined mixing processes. In all of the approaches of solving the BSS problem, ICA [44, 80] is the most fundamental one, as it provides a robust solution based on an ideal environment (the easiest case, i.e., instantaneous and even-determined case) and assumption of sources (independent, at least decorrelated). However, to solve this problem with realistic conditions, especially in underdetermined case, additional information or constraints (e.g. sparseness) need to be considered. This is still an open problem since there are an infinite number of solutions in an underdetermined system in which the number of known mixed signals is smaller than the number of unknown source signals.

2.4.2 Underdetermined Blind Source Separation and Dictionary Learning based Techniques

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 [45], the clustering techniques in [10, 100], the method combining the techniques of ICA and ideal binary mask (IBM) [123], and matrix and tensor decomposition based methods [115], [43, 46, 145]. The DUET [87, 167] method 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 [10]. The methods in [78] and [11] are also based on the clustering technique. However, the majority of methods to solve underdetermined BSS problem is based on sparse signal representation which is a powerful framework even when the independence assumption is dropped. Good reviews on using sparse component analysis for source separation can be found in [72, 143]. Zibulevsky and Pearlmutter [168] 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) [21] framework.

In [28], 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. There are also some benchmark literatures [71, 98, 151] to present sparsity-based underdetermined BSS method. As discussed in the above sections, having an appropriate dictionary is essential for the sparse representation of a signal. There has been increasing interest emerging in blind source separation methods on using dictionary learning based techniques. For example, in morphological component analysis (MCA) [142], source separation is addressed by decomposing the images into different morphological components in terms of sparsity of each component in a signal dictionary. The MCA has also been extended to multichannel case as multichannel MCA (MMCA) [26] and generalized MCA (GMCA) [26]. In MMCA, each source is assumed to be sparse in a specific transform domain. However, in GMCA, each source can be represented by the linear combination of morphological components and each component has a sparse representation by a specific dictionary. Recently, MMCA is also adapted to Blind Multichannel Morphological Component Analysis (BMMCA) [1] based on learned dictionary for separating mixed images. This method is also motivated by the idea of image denoising using a learned dictionary from corrupted images in [60], which in principle extends the denoising problem to BSS. The BMMCA method is interesting in that the dictionary is directly trained from the mixtures, alleviating the issue of requiring training data, and as a result the algorithm can still be performed in a blind manner. However, the BMMCA method trains multiple dictionaries for different sources, and in each iteration only updates one atom, rendering a potentially ineffective sparse representation of the image sources and a computationally inefficient procedure. Overall, these methods take advantages of both morphological diversity and sparsity, using recent sparse overcomplete or redundant signal representations and dictionary learning techniques.

2.5 Other Related Models/Methods

2.5.1 Non-negative Matrix Factorization

Non-negative matrix factorization (NMF) [96] developed by Lee and Seung in 1999 was inspired by the evidence of parts-based representation in neural network. NMF

decomposes a non-negative matrix $\mathbf{V}$ into the product of two non-negative matrices $\mathbf{W}$ and $\mathbf{H}$ and allows only additive but subtractive operations, in contrast to other matrix factorization methods. The formulations can be described below:

$$\mathbf{V} = \mathbf{WH} \tag{2.12}$$

If we apply NMF to a database of facial images, a linear combination of basis images can be learned from the image database to represent a face. The dimensions of the matrix $\mathbf{W}$ and $\mathbf{H}$ are $m \times d$ and $d \times n$, respectively. The d columns of $\mathbf{W}$ are basis images. Each column of $\mathbf{H}$ consists of the coefficients to represent the face by a linear combination of basis images. In vector quantization (VQ), each column of $\mathbf{H}$ is constrained to be a vector in which one element equals to unity and the other elements equal to zero. In principal components analysis (PCA), the columns of $\mathbf{W}$ are orthonormal and the rows of $\mathbf{H}$ are orthogonal to each other, which relaxes the unitary constraint of VQ. In ICA, the variables are assumed to be statistically independent and non-Gaussian. This is in contrast to NMF which assumes that the variables are non-negative, but makes no further assumptions about their statistical dependencies. Therefore, NMF practically has the potential to separate the correlated sources. Apart from image processing applications, NMF can also be applied to audio signals. In this situation, the music or speech data are usually transformed to non-negative spectrogram which can be used in NMF algorithms, such as in music transcription [140] and audio source separation applications [63, 139, 155, 156].

2.5.2 Analysis Sparse Model

Data model is at the heart of signal processing as well as being fundamental for practical applications such as compression, separation, sampling, inverse problems and other related tasks. Essentially, a model poses a set of mathematical properties that the data is assumed to satisfy. During the past decade, in sparse representations, the synthesis model for representing signals has been extensively studied. Such a model assumes that the signal can be decomposed as a linear combination of a few atoms from a given dictionary. However, there is a counterpart viewpoint called analysis model [131] that

did not receive equal attention. The analysis model relies on a linear operator $\mathbf{\Omega}$ as the analysis dictionary whose rows are analysis atoms. Comparing to the synthesis model $\mathbf{y} = \mathbf{D}\mathbf{x}$ in which the signal $\mathbf{y}$ is the multiplication by the dictionary $\mathbf{D}$ and the sparse coefficients vector $\mathbf{x}$, in the analysis model, the analysis representation vector $\mathbf{x} = \mathbf{\Omega}\mathbf{y}$ is expected to be sparse. There are some emerging algorithms [131] [113] developed recently to learn the analysis dictionary $\mathbf{\Omega}$ from a set of training signals. The objective is to find a suitable dictionary so that the analysis coefficients $\mathbf{\Omega}\mathbf{y}$ are sparse. When the analysis dictionary is a square matrix i.e. invertible, the analysis model is equivalent to the synthesis model as $\mathbf{\Omega}^{-1} = \mathbf{D}$. In this case, the general dictionary learning method e.g. K-SVD can be used to train the analysis dictionary $\mathbf{\Omega}$. However, the more general case for using analysis model is the redundant case in which the number of rows is greater than the number of columns of $\mathbf{\Omega}$, which is similar to the overcomplete dictionary case in the synthetic model. This is a challenging problem, which has begun to attract increasing attention [113, 121, 125, 166] in recent years.

2.6 Summary

In this chapter, we review one of the key problems in signal processing 'modeling the data in sparse representation' and discuss benchmark sparse coding approaches such as basis pursuit (BP), the least squares method L1LS, matching pursuit (MP) and orthogonal matching pursuit (OMP). Moreover, we introduce the basic concept and theory of compressed sensing which is used in this thesis as a signal recovery tool to solve the underdetermined blind source separation problem. Then we investigate the dictionary learning problem and related methods, especially the benchmark algorithms such as the method of optimal directions (MOD) algorithm, K-SVD algorithm and the greedy adaptive dictionary algorithm. The blind source separation (BSS) problem is also reviewed comprehensively including its background, foundations, applications and the state-of-the-art methods. The underdetermined BSS case is surveyed independently and the sparsity based method motivated us to develop the proposed multi-stage BSS algorithm in this thesis. Finally, the related method non-negative matrix factorization (NMF) and the state-of-the-art analysis sparse model are summarized in the last subsection of this chapter.

Chapter 3

Simultaneous Codeword Optimization (SimCO) for Dictionary Learning

3.1 Introduction

In this chapter, similar to MOD and K-SVD methods, we focus on the dictionary update step for generic dictionary learning. A novel optimization framework is proposed, where an *arbitrary* subset of the codewords are allowed to be updated simultaneously, hence the term *simultaneous codeword optimization (SimCO)*. This work was done in collaboration with Dr. Wei Dai from Imperial Collage London and my main contributions include algorithm implementation and experiment design. Specifically, I found an important phenomenon during the experiments e.g. the singular point rather than the local minimum is the key issue for algorithm convergence, which has led us to modify the original SimCO framework to the regularized version. The proposed framework has the following characteristics.

- SimCO generalizes MOD and K-SVD. We show that the MOD algorithm is in fact an inexact Newton's method under the proposed framework while K-SVD can be viewed as a special case of SimCO where only one codeword is selected for update

at each iteration. The SimCO framework is general and flexible. This chapter presents two possible algorithmic implementations: one is based on gradient descent and the other uses a Newton's method.

- The proposed optimization framework allows the discovery of the bottleneck of dictionary update. As opposed to traditional formulations, in the SimCO framework, the objective function involves only the dictionary by treating sparse coefficients as a function of the dictionary. In this way, the gradient can be easily computed and analyzed. Surprisingly, against the traditional belief that local minima are the major problem, we empirically discover that singular points are the bottleneck.

- Regularized SimCO is introduced to mitigate the singularity problem. To avoid the singularity problem, an additive regularization term is introduced. The resulting objective function is differentiable. Significant improvement in empirical performance is observed. This, from another angle, verifies that singularity is the bottleneck.

The remainder of the chapter is organized as follows. Section 3.2 introduces the SimCO optimization framework, with particular emphasis on the motivations for regularized SimCO. Section 3.3 discusses the relation of SimCO to MOD and K-SVD, and the possibility of extending MOD and K-SVD to the regularized versions. Section 3.4 provides necessary preliminaries on manifolds and shows that dictionary update can be cast as an optimization problem on manifolds. The algorithmic details on how to apply the first and second order methods to solve the SimCO optimization problem are presented in Section 3.5. In Section 3.6, we rigorously prove the deep connection between SimCO and K-SVD. Numerical results of SimCO algorithms are presented in Section 3.7. A fast version of SimCO via codeword clustering and hierarchical sparse coding is discussed in Section 3.8. Finally, the chapter is concluded in Section 3.9.

3.2 The Optimization Framework of SimCO

Dictionary learning is a procedure to find an over-complete dictionary that best represents the training signals. As we reviewed in Chapter 2, dictionary learning algorithms

usually consist of two stages: sparse coding and dictionary update (see Algorithm 1). The focus of this chapter is on the dictionary update stage. Instead of directly solving the joint optimization problem in (2.7), we view the sparse coefficients as a function of the dictionary so that the optimization is only over the dictionary. Furthermore, our framework allows one to simultaneously update an arbitrary subset of codewords and the corresponding coefficients. This characteristic gives rise to the term *simultaneous codeword optimization* (*SimCO*).

In our formulation, we assume that the dictionary matrix $\mathbf{D}$ contains unit ℓ_2-norm columns and the sparsity pattern of $\mathbf{X}$ remains unchanged. Define

$$\mathcal{D} = \left\{ \mathbf{D} \in \mathbb{R}^{m \times d} : \ \|\mathbf{D}_{:,i}\|_2 = 1, \ \forall i \in [d] \right\}, \tag{3.1}$$

where $\|\cdot\|_2$ is the ℓ_2-norm and the set $[d] = \{1, 2, \cdots, d\}$. This is the set of all feasible dictionaries. Represent the *sparsity pattern* of $\mathbf{X}$ by the index set $\Omega \subset [d] \times [n]$ which contains the indices of all the non-zero entries in $\mathbf{X}$: that is, $X_{i,j} \neq 0$ for all $(i,j) \in \Omega$ and $X_{i,j} = 0$ for all $(i,j) \notin \Omega$. Define

$$\mathcal{X}(\Omega) = \left\{ \mathbf{X} \in \mathbb{R}^{d \times n} : \ \mathbf{X}_{i,j} = 0, \ \forall (i,j) \notin \Omega \right\}. \tag{3.2}$$

This is the set of all feasible $\mathbf{X}$ given sparsity pattern Ω. The dictionary update problem is formulated as

$$\inf_{\mathbf{D} \in \mathcal{D}} f(\mathbf{D}) = \inf_{\mathbf{D} \in \mathcal{D}} \underbrace{\inf_{\mathbf{X} \in \mathcal{X}(\Omega)} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2}_{f(\mathbf{D})}. \tag{3.3}$$

To evaluate $f(\mathbf{D})$ for a given $\mathbf{D}$, the least squares problem $\inf_{\mathbf{X} \in \mathcal{X}(\Omega)} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2$ needs to be solved. Denote the optimal $\mathbf{X}$ by $\mathbf{X}(\mathbf{D})$, which can be viewed as a function of $\mathbf{D}$. An update in $\mathbf{D}$ results in an update of $\mathbf{X}(\mathbf{D})$. In other words, both $\mathbf{D}$ and $\mathbf{X}$ are simultaneously updated.

One may notice that the optimal $\mathbf{X}$ that solves the least squares problem above may not be unique. Non-unique solutions happen only when $\mathbf{D}$ is singular, formally defined as follows. For a given sparsity patten Ω, let $\Omega(:,j) = \{i : \ (i,j) \in \Omega\}$. Let $\mathbf{D}_{:,\Omega(:,j)}$ be the sub-matrix of $\mathbf{D}$ containing the columns indexed by $\Omega(:,j)$. A dictionary $\mathbf{D}$ is *singular*

under sparsity patten Ω if there exists $j \in [n]$ such that the columns of $\mathbf{D}_{:,\Omega(:,j)}$ are linearly dependent, i.e., $\mathbf{D}_{:,\Omega(:,j)}$ does not have full column rank. At a singular point, $\mathbf{X}(\mathbf{D})$ is not uniquely defined. This can be solved by arbitrarily choosing one of the multiple solutions as the choice of $\mathbf{X}(\mathbf{D})$ does not affect the value of $f(\mathbf{D})$.

The singularity problem brings several algorithmic problems.

1. Severe performance deterioration in dictionary update. Our empirical experiments (detailed in Section 3.7.1) show that, when the dictionary update procedure fails in finding a globally optimal solution, most likely it converges to a singular point, i.e., an ill-conditioned dictionary.

2. Slow convergence in dictionary update. Let $\lambda_{\min}\left(\mathbf{D}_{:,\Omega(:,j)}\right)$ be the minimum singular value of the matrix $\mathbf{D}_{:,\Omega(:,j)}$. When it is close to zero, the curvature (Hessian) of $f(\mathbf{D})$ is large and the gradient changes significantly in the neighborhood of a singular point. Optimization algorithms typically suffer from a very slow convergence rate.

3. Instability in the subsequent sparse coding stage. When $\lambda_{\min}\left(\mathbf{D}_{:,\Omega(:,j)}\right)$ is close to zero, the solution to the least squares problem $\inf_{\mathbf{X}_{\Omega(:,j),j}} \left\|\mathbf{Y}_{:,j} - \mathbf{D}_{:,\Omega(:,j)}\mathbf{X}_{\Omega(:,j),j}\right\|_F^2$ becomes unstable: small changes in $\mathbf{Y}_{:,j}$ often result in very different least squares solutions $\mathbf{X}^*_{\Omega(:,j),j}$. It is well known that the stability of sparse coding relies on the so called restricted isometry condition (RIP) [35], which requires that the singular values of submatrices of $\mathbf{D}$ center around 1. An ill-conditioned $\mathbf{D}$ violates RIP and hence results in sparse coefficients that are sensitive to noise.

To mitigate the singularity problem, we propose to add a regularization term into the objective function:

$$\inf_{\mathbf{D}\in\mathcal{D}} f_\mu(\mathbf{D}) = \inf_{\mathbf{D}\in\mathcal{D}} \underbrace{\inf_{\mathbf{X}\in\mathcal{X}(\Omega)} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 + \mu\|\mathbf{X}\|_F^2}_{f_\mu(\mathbf{D})}, \tag{3.4}$$

where $\mu > 0$ is a properly chosen constant. Hereafter, we refer to (3.3) and (3.4) as *primitive SimCO* and *regularized SimCO* respectively. Note that when $\mu = 0$, regularized SimCO reduces to primitive SimCO. In practice, one may consider first using

regularized SimCO ($\mu > 0$) to obtain a reasonably good dictionary and then reduce the regularization constant μ to zero to refine the dictionary further. This two-step procedure often results in a well-conditioned dictionary that fits the training data. See the simulation part (Section 3.7) for examples.

The effect of the regularization term is to remove the singular point. Let $\mathbf{Y}_{:,j}$ be the jth column of $\mathbf{Y}$. Let $\mathbf{X}_{\Omega(:,j),j}$ be the sub-vector of $\mathbf{X}_{:,j}$ formed by the entries indexed by $\Omega(:,j)$. Let $m_j = |\Omega(:,j)|$ be the number of non-zeros in the jth column of $\mathbf{X}$. Define

$$\tilde{\mathbf{y}}_j = \begin{bmatrix} \mathbf{Y}_{:,j} \\ \mathbf{0}_{m_j} \end{bmatrix}, \ \mathbf{D}_j = \mathbf{D}_{:,\Omega(:,j)}, \text{ and } \tilde{\mathbf{D}}_j = \begin{bmatrix} \mathbf{D}_{:,\Omega(:,j)} \\ \sqrt{\mu} \cdot \mathbf{I}_{m_j} \end{bmatrix}, \tag{3.5}$$

where $\mathbf{0}_{m_j}$ is the zero vector of length m_j, and $\mathbf{I}_{m_j}$ is the $m_j \times m_j$ identity matrix. Then

$$\begin{aligned} f_\mu(\mathbf{D}) &= \inf_{\mathbf{X} \in \mathcal{X}(\Omega)} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 + \mu \|\mathbf{X}\|_F^2 \\ &= \sum_{j=1}^{n} \inf_{\mathbf{X}_{\Omega(:,j),j}} \left\|\mathbf{Y}_{:,j} - \mathbf{D}_{:,\Omega(:,j)}\mathbf{X}_{\Omega(:,j),j}\right\|_2^2 + \mu \left\|\mathbf{X}_{\Omega(:,j),j}\right\|_2^2 \\ &= \sum_{j=1}^{n} \underbrace{\inf_{\mathbf{X}_{\Omega(:,j),j}} \left\|\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j \mathbf{X}_{\Omega(:,j),j}\right\|_2^2}_{f_{\mu,j}(\mathbf{D}_j), \text{ or equivalently } f_{\mu,j}(\tilde{\mathbf{D}}_j)}. \end{aligned} \tag{3.6}$$

When $\mu > 0$, the matrix $\tilde{\mathbf{D}}_j$ in the jth atomic function $f_{\mu,j}\left(\tilde{\mathbf{D}}_j\right)$ has full column rank. The objective function $f_\mu(\mathbf{D})$ is always continuous and contains no singular points. The algorithmic details for solving the regularized SimCO are presented in Section 3.5.

So far, we have considered only the case where all the codewords and the corresponding nonzero coefficients are simultaneously updated. It is worth noting that SimCO accommodates the case of simultaneously updating an arbitrary subset of codewords and the corresponding nonzero coefficients. More precisely, let $\mathcal{I} \subseteq [d]$ be the index set of the codewords to be updated. That is, only codewords $\mathbf{D}_{:,i}$'s, $i \in \mathcal{I}$, are to be updated while all other codewords $\mathbf{D}_{:,i}$'s, $i \notin \mathcal{I}$, remain constant. Let $\mathbf{D}_{:,\mathcal{I}}$ denote the sub-matrix of $\mathbf{D}$ formed by the columns of $\mathbf{D}$ indexed by $\mathcal{I}$. Let $\mathbf{X}_{\mathcal{I},:}$ denote the sub-matrix of $\mathbf{X}$

consisting of the rows of $\mathbf{X}$ indexed by $\mathcal{I}$. Define

$$\mathbf{Y}_r = \mathbf{Y} - \mathbf{D}_{:,\mathcal{I}^c}\mathbf{X}_{\mathcal{I}^c,:}, \tag{3.7}$$

where $\mathcal{I}^c$ is a set complementary to $\mathcal{I}$. Then $\mathbf{Y} - \mathbf{DX} = \mathbf{Y_r} - \mathbf{D}_{:,\mathcal{I}}\mathbf{X}_{\mathcal{I},:}$. Replacing the $\mathbf{Y}$, $\mathbf{D}$ and $\mathbf{X}$ in (3.3) and (3.4) by $\mathbf{Y}_r$, $\mathbf{D}_{:,\mathcal{I}}$ and $\mathbf{X}_{\mathcal{I},:}$ respectively, the optimization framework developed for the full set $[d]$, i.e., (3.3) and (3.4), can be readily applied to the case $\mathcal{I} \subset [d]$. For this reason, the discussions hereafter will center around the full set $[d]$ case (the subscript $\mathcal{I}$ will be dropped).

Finally, we would like to comment on the column-norm constraint imposed on the dictionary in (3.1). This constraint appears in K-SVD but not in MOD. Theoretically, the performance of a given dictionary is invariant to the column norms: a scaling in columns of $\mathbf{D}$ can be compensated by an inverse scaling in the corresponding rows of $\mathbf{X}$. On the other hand, the constraint on the column norms has certain advantages:

1. A normalized dictionary $\mathbf{D} \in \mathcal{D}$ is required in regularized SimCO. The regularization term $\mu \|\mathbf{X}\|_F^2$ is useful only when the column norms of $\mathbf{D}$ are fixed. Otherwise, the regularized objective function (3.4) can be reduced simply by scaling up the columns of $\mathbf{D}$.

2. A normalized dictionary $\mathbf{D} \in \mathcal{D}$ plays an important role in identifying singular points. As detailed in Section 3.7.1, the gradient of the objective function $f(\mathbf{D})$ is used to distinguish between singular points and local minimizers. Since scaling the columns of the dictionary results in scaling in the gradient (see (3.19) for more details), a normalization is necessary.

3. A normalized dictionary $\mathbf{D} \in \mathcal{D}$ is preferred in the sparse coding stage. Sparse coding algorithms rely heavily on the magnitudes of the coefficients $X_{i,j}$'s, $(i,j) \in [d] \times [n]$, which are affected by the column norms of $\mathbf{D}$. It is a standard practice to normalize the columns of $\mathbf{D}$ before applying sparse coding algorithms.

3.3 Relation to the State of the Art

In this section, we discuss how primitive SimCO is related to two benchmark algorithms MOD and K-SVD. Furthermore, as regularization substantially improves the performance (motivated in Section 3.2 and empirically demonstrated in Section 3.7), we regularize MOD and K-SVD as well. Here, we would like to emphasize that the regularization technique is designed to handle the singularity problem, which is observed via the SimCO framework. We are not aware of regularized versions of MOD and K-SVD in the literature.

In MOD, the dictionary update involves iteratively performing two steps: first fix $\mathbf{D}$ and solve $\mathbf{X}$ for $\inf_{\mathbf{X}\in\mathcal{X}(\Omega)} \|\mathbf{Y}-\mathbf{DX}\|_F^2$; then fix $\mathbf{X}$ and solve $\mathbf{D}$ for $\inf_{\mathbf{D}\in\mathbb{R}^{m\times d}} \|\mathbf{Y}-\mathbf{DX}\|_F^2$. Both steps involve only solving a least squares problem. Denote the dictionaries before and after an iteration by $\mathbf{D}$ and $\mathbf{D}'$ respectively. Then the updated sparse coefficients are $\mathbf{X}(\mathbf{D})$ and the updated dictionary is given by $\mathbf{D}' = \mathbf{Y}\mathbf{X}^{\mathbf{T}}(\mathbf{D})\left(\mathbf{X}(\mathbf{D})\mathbf{X}^{\mathbf{T}}(\mathbf{D})\right)^{-1}$.

MOD can be viewed as an inexact Newton's method to solve the primitive SimCO problem without the column norm constraint. After dropping the column-norm constraint, the optimization problem in SimCO becomes $\inf_{\mathbf{D}\in\mathbb{R}^{m\times d}} f(\mathbf{D})$ where $f(\mathbf{D}) = \inf_{\mathbf{X}\in\mathcal{X}(\Omega)} \|\mathbf{Y}-\mathbf{DX}\|_F^2$. Consider the Newton iteration for dictionary update, where the gradient and the Hessian are given by $\nabla f(\mathbf{D}) = -2(\mathbf{Y}-\mathbf{DX}(\mathbf{D}))\mathbf{X}^T(\mathbf{D})$ and

$$\nabla^2 f(\mathbf{D}) = 2(\nabla\mathbf{D})\mathbf{X}(\mathbf{D})\mathbf{X}^{\mathbf{T}}(\mathbf{D}) \tag{3.8}$$

$$+ 2\mathbf{D}(\nabla\mathbf{X}(\mathbf{D}))\mathbf{X}^{\mathbf{T}}(\mathbf{D}) + 2\mathbf{D}\mathbf{X}(\mathbf{D})(\nabla\mathbf{X}(\mathbf{D}))^{\mathbf{T}} \tag{3.9}$$

respectively (see Section 3.5.2 for more details on how to compute Hessian). Note that the computation of $\nabla\mathbf{X}(\mathbf{D})$ is complicated. To reduce the computational complexity, one may approximate $\nabla^2 f$ by omitting the terms involving $\nabla\mathbf{X}(\mathbf{D})$, i.e., approximate $\nabla^2 f$ by $2(\nabla\mathbf{D})\mathbf{X}(\mathbf{D})\mathbf{X}^{\mathbf{T}}(\mathbf{D})$. Following from this approximation, the objective function at the neighborhood of a given dictionary $\mathbf{D}_0$ can be approximated by $f(\mathbf{D}) \approx \|\mathbf{Y}-\mathbf{DX}(\mathbf{D_0})\|_F^2$. The optimal dictionary with respect to the approximated objective function is then given by $\mathbf{D}' = \mathbf{Y}\mathbf{X}^{\mathbf{T}}(\mathbf{D_0})\left(\mathbf{X}(\mathbf{D_0})\mathbf{X}^{\mathbf{T}}(\mathbf{D_0})\right)^{-1}$, which coincides with the update rule in MOD.

Using a similar approximation for the corresponding Hessian matrix, MOD can be adapted to solve the regularized SimCO problem. We refer to it as *regularized MOD*. It again iteratively performs two steps: first fix $\mathbf{D}$ and solve $\mathbf{X}$ for $\inf_{\mathbf{X}\in\mathcal{X}(\Omega)} \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 + \mu\|\mathbf{X}\|_F^2$; then fix $\mathbf{X}$ and solve $\mathbf{D}$ for $\inf_{\mathbf{D}\in\mathbb{R}^{m\times d}} \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2$. Substantial improvement in empirical performance can be observed in Section 3.7.

The relation between SimCO and K-SVD is straightforward. Consider the SimCO where only one codeword and the corresponding sparse coefficients are updated. The resulting objective function is the same as that of K-SVD. More specifically, suppose that only the ith codeword and the corresponding sparse coefficients are updated. Let $\mathbf{Y}_r = \mathbf{Y} - \mathbf{D}_{:,\{i\}^c}\mathbf{X}_{\{i\}^c,:}$. Then both SimCO and K-SVD aim at solving

$$\inf_{\mathbf{D}_{:,i}:\ \|\mathbf{D}_{:,i}\|_2=1}\ \inf_{\mathbf{X}_{i,\Omega(i,:)}} \left\|(\mathbf{Y}_r)_{:,\Omega(i,:)} - \mathbf{D}_{:,i}\mathbf{X}_{i,\Omega(i,:)}\right\|_2^2. \tag{3.10}$$

In K-SVD, singular value decomposition (SVD) is used to solve the above optimization problem.

Nevertheless, it is not clear how to extend K-SVD to the regularized case. Let $m_i = |\Omega(i,:)|$. With the regularization term, the optimization problem becomes

$$\inf_{\mathbf{D}_{:,i}:\ \|\mathbf{D}_{:,i}\|_2=1}\ \inf_{\mathbf{X}_{i,\Omega(i,:)}\in\mathbb{R}^{m_i}} \left\|(\mathbf{Y}_r)_{:,\Omega(i,:)} - \mathbf{D}_{:,i}\mathbf{X}_{i,\Omega(i,:)}\right\|_2^2 + \mu\left\|\mathbf{X}_{i,\Omega(i,:)}\right\|_2^2 \tag{3.11}$$

$$= \inf_{\mathbf{d}:\ \|\mathbf{d}\|_2=1}\ \inf_{\mathbf{x}\in\mathbb{R}^{m_i}} \left\| \underbrace{\begin{bmatrix} (\mathbf{Y}_r)_{:,\Omega(i,:)} \\ \mathbf{O}_{m_i} \end{bmatrix}}_{\tilde{\mathbf{Y}}_{r,i}} - \underbrace{\begin{bmatrix} \mathbf{d} \\ \sqrt{\mu}\mathbf{1}_{m_i} \end{bmatrix}}_{\tilde{\mathbf{d}}} \mathbf{x}^T \right\|_2^2, \tag{3.12}$$

where $\mathbf{O}_{m_i}$ is the $m_i \times m_i$ zero matrix, $\mathbf{1}_{m_i} \in \mathbb{R}^{m_i}$ is an all-one vector, and we have simplified the notations $\mathbf{D}_{:,i}$ and $\mathbf{X}_{i,\Omega(i,:)}$ to $\mathbf{d}$ and $\mathbf{x}$ respectively. If we apply SVD to $(\mathbf{Y}_r)_{:,\Omega(i,:)}$, the solution is exactly identical to that in the original K-SVD. If we apply SVD to $\tilde{\mathbf{Y}}_{r,i}$, the left singular vectors are not of the form $\tilde{\mathbf{d}}$: the last m_i entries of the left singular vectors are always zero. In either case, in contrast to the original K-SVD, SVD cannot solve the joint optimization problem. In the numerical comparison part of this chapter, we use regularized SimCO with $\mathcal{I} = \{i\} \subset [d]$ to solve this optimiza-

tion problem, and refer to the resulting algorithm as regularized "K-SVD" although it involves no SVD.

3.4 Preliminaries on Manifolds

Our approach for solving the optimization problem in Equation (3.3) and (3.4) relies on the notion of Stiefel and Grassmann manifolds. In particular, the Stiefel manifold $\mathcal{U}_{m,1}$ is defined as $\mathcal{U}_{m,1} = \left\{\mathbf{u} \in \mathbb{R}^m : \mathbf{u}^T\mathbf{u} = \mathbf{1}\right\}$. The Grassmann manifold $\mathcal{G}_{m,1}$ is defined as $\mathcal{G}_{m,1} = \{\text{span}(\mathbf{u}) : \mathbf{u} \in \mathcal{U}_{m,1}\}$. Here, the notations $\mathcal{U}_{m,1}$ and $\mathcal{G}_{m,1}$ follow from the convention in [49, 59]. Note that each element in $\mathcal{U}_{m,1}$ is a unit-norm vector while each element in $\mathcal{G}_{m,1}$ is a one-dimensional subspace in $\mathbb{R}^m$. For any given $\mathbf{u} \in \mathcal{U}_{m,1}$, it can generate a one-dimensional subspace $\mathscr{U} \in \mathcal{G}_{m,1}$. Meanwhile, any given $\mathscr{U} \in \mathcal{G}_{m,1}$ can be generated from different $\mathbf{u} \in \mathcal{U}_{m,1}$: if $\mathscr{U} = \text{span}(\mathbf{u})$, then $\mathscr{U} = \text{span}(-\mathbf{u})$ as well.

With these definitions, the dictionary $\mathbf{D}$ can be interpreted as the Cartesian product of d many Stiefel manifolds $\mathcal{U}_{m,1}$. Each codeword (column) in $\mathbf{D}$ is one element in $\mathcal{U}_{m,1}$. It looks straightforward that optimization over $\mathbf{D}$ is an optimization over the product of Stiefel manifolds.

What is not so obvious is that the optimization is actually over the product of Grassmann manifolds. For any given pair $(\mathbf{D}, \mathbf{X})$, if the signs of $\mathbf{D}_{:,i}$ and $\mathbf{X}_{i,:}$ change simultaneously, the value of the objective function $\|\mathbf{Y} - \mathbf{DX}\|_F^2$ stays the same. Let $\mathbf{D} = [\mathbf{D}_{:,1}, \cdots, \mathbf{D}_{:,i-1}, \mathbf{D}_{:,i}, \mathbf{D}_{:,i+1}, \cdots, \mathbf{D}_{:,d}]$ and $\mathbf{D}' = [\mathbf{D}_{:,1}, \cdots, \mathbf{D}_{:,i-1}, -\mathbf{D}_{:,i}, \mathbf{D}_{:,i+1}, \cdots, \mathbf{D}_{:,d}]$. Then it is straightforward to verify that $f_{[d]}(\mathbf{D}) = f_{[d]}(\mathbf{D}')$. In other words, it does not matter what $\mathbf{D}_{:,i}$ is; what matters is the generated subspace $\text{span}(\mathbf{D}_{:,i})$. As shall become explicit later, this phenomenon has a significant impact on algorithm design and analysis.

It is worth noting that the performance of a given dictionary is invariant to the permutations of the codewords. However, how to effectively address this permutation invariance analytically and algorithmically remains an open problem.

3.5 Implementation Details for SimCO

This section presents the algorithmic details on how to solve the optimization problems in Equation (3.3) and (3.4). As the primitive SimCO is a special case of regularized SimCO where $\mu = 0$, the descriptions below center around regularized SimCO. One of the key properties of SimCO is that the objective function $f_\mu(\mathbf{D})$ only involves the dictionary. To minimize this objective function, derivatives of this function need to be evaluated. First and second order optimization procedures can be implemented. Note that first order methods are often conceptually easier to understand but slower in convergence rate, while second order methods are typically faster in convergence rate but more complicated in the computation of the search direction. In this section, we first outline the proposed algorithms, then give details on the computations of the first and second order derivatives, and finally discuss the line search path that satisfies the column norm constraint.

3.5.1 Outline of Algorithms

A natural choice of first order optimization procedures is the gradient descent line search method. Algorithm 2 summarizes one iteration of the proposed procedure. The computational details of the gradient ∇f_μ and line search path $\mathbf{D}(\mathbf{t})$ are presented in Sections 3.5.2 and 3.5.3 respectively, where t is the step size. For proof-of-concept, we use the method of golden section search[1] (see [129] for a detailed description). The idea is to use the golden ratio to successively narrow the search range of t inside which a local minimum exists. To implement this idea, we design a two-step procedure in Algorithm 2: in the first step (Part A), we increase/decrease the range of t, i.e., $(0, t_4)$, so that it contains a local minimum and the objective function looks unimodal in this range; in the second step (Part B), we use the golden ratio to narrow the range so that we can accurately locate the minimizer. Note that the proposed algorithm is by no means

[1] Algorithm 2 looks more complicated than popular gradient descent methods in standard textbooks, e.g., [117]. We choose this implementation because it mimics the ideal gradient descent with infinitesimal steps more authentically than other optimization methods of which the step size may be so large that local minimizers or singular point may not be seen. In the simulation part, we use Algorithm 2 to catch the singular points.

Algorithm 2 One iteration in a gradient descent line search algorithm.

Input: $\mathbf{Y}$, $\mathbf{D}$, $\mathbf{X}$

Output: $\mathbf{D}'$ and $\mathbf{X}'$.

Parameters: $t_4 > 0$: initial step size. $g_{\min} > 0$: the threshold below which a gradient can be viewed as zero.

Initialization: Let $c = \left(\sqrt{5} - 1\right)/2$.

1. Let $t_1 = 0$. Compute $f_\mu(\mathbf{D})$ and $\nabla f_\mu(\mathbf{D})$. If $\|\nabla f_\mu\|_F \leq g_{\min} \|\mathbf{Y}\|_F^2$, then $\mathbf{D}' = \mathbf{D}$, $\mathbf{X}' = \mathbf{X}$, and quit.

2. Set line search direction $\mathbf{H} = -\nabla \mathbf{f}_\mu$. Let $t_3 = ct_4$ and $t_2 = (1-c)\, t_4$.

Part A: the goal is to find $t_4 > 0$ s.t. $f(\mathbf{D}(t_1)) > f(\mathbf{D}(t_2)) > f(\mathbf{D}(t_3)) \leq f(\mathbf{D}(t_4))$, where $\mathbf{D}(t)$ is defined via (3.25). Iterate the following steps.

3) If $f(\mathbf{D}(t_1)) \leq f(\mathbf{D}(t_2))$, then $t_4 = t_2$, $t_3 = ct_4$ and $t_2 = (1-c)\, t_4$.

4. Else if $f(\mathbf{D}(t_2)) \leq f(\mathbf{D}(t_3))$, then $t_4 = t_3$, $t_3 = t_2$ and $t_2 = (1-c)\, t_4$.

5. Else if $f(\mathbf{D}(t_3)) > f(\mathbf{D}(t_4))$, then $t_2 = t_3$, $t_3 = t_4$ and $t_4 = t_3/c$.

6. Otherwise, quit the iteration.

Part B: the goal is to shrink the interval length $t_4 - t_1$ while keeping $f(\mathbf{D}(t_1)) > f(\mathbf{D}(t_2)) > f(\mathbf{D}(t_3))$. Iterate the following steps until $t_4 - t_1$ is sufficiently small.

7) If $f(\mathbf{D}(t_1)) > f(\mathbf{D}(t_2)) > f(\mathbf{D}(t_3))$, then $t_1 = t_2$, $t_2 = t_3$ and $t_3 = t_1 + c(t_4 - t_1)$.

8. Else $t_4 = t_3$, $t_3 = t_2$ and $t_2 = t_1 + (1-c)(t_4 - t_1)$.

Output: Let $t^* = \underset{t \in \{t_1, t_2, t_3, t_4\}}{\arg\min} f(\mathbf{D}(t))$. Set $\mathbf{D}' = \mathbf{D}(t^*)$ and compute $\mathbf{X}'$ according to (3.18).

optimal. Other ways to do a gradient descent efficiently can be found in [117, Chapter 3].

For second order optimization methods, we choose line search Newton - Conjugate Gradient (LSNCG) method [117, Chapter 7]. (It turns out that the trust region method [117], another popular second order optimization method, is not quite numerically stable under certain conditions.) It is worth noting that LSNCG, which uses the exact Hessian, typically exhibits faster convergence rate than MOD, where an approximate of the Hessian is employed.

Before discussing the details, let us first understand the ideas behind LSNCG. Let $\mathbf{D}$ and $\mathbf{D}'$ be the dictionaries before and after a line search step. In Newton methods,

$\mathbf{D}'-\mathbf{D} = -\left(\nabla^2 \mathbf{f}_\mu\right)^{-1} \nabla \mathbf{f}_\mu$. However, note that the Hessian $\nabla^2 f_\mu$ is an $(m \times d)\times(m \times d)$ matrix with entries $\partial^2 f_\mu / \left(\partial \mathbf{D}_{i,j} \partial \mathbf{D}_{k,\ell}\right)$. Computing it explicitly and taking the inverse are computationally expensive. The main idea of LSNCG method is to use the conjugate gradient method [117, Chapter 5] to avoid explicit computation of the Hessian and its inverse. The steps of LSNCG are based on the concept of directional derivative. Let $\mathbf{H} \in \mathbb{R}^{m\times r}$ be a matrix of the same dimension of $\mathbf{D}$. The directional derivative of f_μ along direction $\mathbf{H}$ is defined as

$$\nabla_{\mathbf{H}} f_\mu(\mathbf{D}) = \lim_{\tau\to 0} \frac{f_\mu(\mathbf{D}+\mathbf{H}\tau) - f_\mu(\mathbf{D})}{\tau}. \tag{3.13}$$

Instead of computing the Hessian, LSNCG only involves directional derivative of the gradient, i.e.,

$$\nabla_{\mathbf{H}} \nabla f_\mu(\mathbf{D}) = \lim_{\tau\to 0} \frac{\nabla f_\mu(\mathbf{D}+\mathbf{H}\tau) - \nabla f_\mu(\mathbf{D})}{\tau}. \tag{3.14}$$

Note that both ∇f_μ and $\nabla_{\mathbf{H}} \nabla f_\mu$ are $m \times d$ matrices[2] and admit closed forms (computation details are given in Section 3.5.2). The computational complexity is greatly reduced. Algorithm 3 summarizes one iteration of the LSNCG procedure for dictionary update, where $\langle \mathbf{A}, \mathbf{B} \rangle$ represents the inner product of matrices $\mathbf{A}$ and $\mathbf{B}$.

3.5.2 Computation of the First and Second Order Derivatives

We now compute ∇f_μ and $\nabla_{\mathbf{H}} \nabla f_\mu$. From the decomposition $f_\mu = \sum_j f_{\mu,j}$ derived in (3.6), it is clear that

$$\nabla f_\mu = \sum_j \nabla f_{\mu,j}. \tag{3.15}$$

For any given $\mathbf{H} \in \mathbb{R}^{m\times d}$, define $\mathbf{H}_j = \mathbf{H}_{:,\Omega(:,j)}$. It can be verified that

$$\nabla_{\mathbf{H}} \nabla f_\mu = \nabla_{\mathbf{H}} \left(\sum_j \nabla f_{\mu,j} \right) \tag{3.16}$$

$$= \sum_j \nabla_{\mathbf{H}} \nabla f_{\mu,j} = \sum_j \nabla_{\mathbf{H}_j} \nabla f_{\mu,j}. \tag{3.17}$$

[2]The entries of $\nabla f_\mu \in \mathbb{R}^{m\times d}$ are $\partial f_\mu / \partial \mathbf{D}_{i,j}$ and those of $\nabla_{\mathbf{H}} \nabla f_\mu$ are $\nabla_{\mathbf{H}} \left(\partial f_\mu / \partial \mathbf{D}_{i,j}\right)$.

Algorithm 3 One iteration in the LSNCG algorithm

Input: $\mathbf{Y}$, $\mathbf{D}$, $\mathbf{X}$

Output: $\mathbf{D}'$ and $\mathbf{X}'$.

Initialization: Set $\mathbf{Z}^{(0)} = \mathbf{0} \in \mathbb{R}^{\mathbf{m}\times\mathbf{d}}$, $\mathbf{R}^{(0)} = \nabla f_\mu$, $\mathbf{H}^{(0)} = -\mathbf{R}^{(0)}$, and $J = 0$. Define tolerance $\epsilon = \min\left(0.5, \sqrt{\|\nabla f_\mu\|_F}\right) \|\nabla f_\mu\|_F$. Define shrink constant $\rho \in (0,1)$.

Part A: the goal is to find the Newton direction $\mathbf{H}$ using conjugate gradient method. Perform the following iterations.

1. If $\langle \nabla f_\mu, \nabla_{\mathbf{H}^{(J)}} \nabla f_\mu \rangle \geq 0$, set $\mathbf{H} = \mathbf{H}^{(\mathbf{J})}$ and quite the iterations.
2. Set $\alpha^{(J)} = \langle \mathbf{R}^{(J)}, \mathbf{R}^{(J)} \rangle / \langle \mathbf{H}^{(J)}, \nabla_{\mathbf{H}^{(J)}} \nabla f_\mu \rangle$, $\mathbf{Z}^{(J+1)} = \mathbf{Z}^{(J)} + \alpha^{(J)} \mathbf{H}^{(J)}$, and $\mathbf{R}^{(J+1)} = \mathbf{R}^{(J)} + \alpha^{(J)} \nabla_{\mathbf{H}^{(J)}} \nabla f_\mu$.
3. If $\left\|\mathbf{R}^{(J+1)}\right\|_F < \epsilon$, set $\mathbf{H} = \mathbf{Z}^{(J+1)}$ and quit the iterations.
4. Set $\beta^{(J+1)} = \langle \mathbf{R}^{(J+1)}, \mathbf{R}^{(J+1)} \rangle / \langle \mathbf{R}^{(J)}, \mathbf{R}^{(J)} \rangle$, $\mathbf{H}^{(J+1)} = -\mathbf{R}^{(J+1)} + \beta^{(J+1)} \mathbf{H}^{(J)}$, and then $J = J + 1$.

Part B: Line search along $\mathbf{H}$.

5) Start with $t = 1$ and repeat setting $t = \rho t$ until $f_\mu(\mathbf{D}(t)) < f_\mu(\mathbf{D})$. Set $t^* = t$, $\mathbf{D}' = \mathbf{D}(\mathbf{t}^*)$ and compute $\mathbf{X}'$ according to (3.18).

As a result, it suffices to compute $\nabla f_{\mu,j}$ and $\nabla_{\mathbf{H}_j} \nabla f_{\mu,j}$ for each atomic function.

The jth atomic function for regularized SimCO, defined in (3.6), is of the form $f_{\mu,j} = \left\|\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j \mathbf{x}_j\left(\tilde{\mathbf{D}}_j\right)\right\|_2^2$ where

$$\mathbf{x}_j\left(\tilde{\mathbf{D}}_j\right) = \tilde{\mathbf{D}}_j^\dagger \tilde{\mathbf{y}}_j = \left(\tilde{\mathbf{D}}_j^T \tilde{\mathbf{D}}_j\right)^{-1} \tilde{\mathbf{D}}_j^T \tilde{\mathbf{y}}_j. \tag{3.18}$$

Again, let $m_j = |\Omega(:,j)|$. Then $\tilde{\mathbf{D}}_j \in \mathbb{R}^{(m+m_j)\times m_j}$. Note that, $f_{\mu,j}$ can be regarded as a function of either $\tilde{\mathbf{D}}_j$ or $\mathbf{D}_j$. We first compute $\nabla f_{\mu,j}\left(\tilde{\mathbf{D}}_j\right) \in \mathbb{R}^{(m+m_j)\times m_j}$, i.e., the gradient of $f_{\mu,j}$ with respect to $\tilde{\mathbf{D}}_j$, and then obtain $\nabla f_{\mu,j}(\mathbf{D}_j) \in \mathbb{R}^{m\times m_j}$, i.e., the gradient of $f_{\mu,j}$ with respect to $\mathbf{D}_j$ from $\nabla f_{\mu,j}\left(\tilde{\mathbf{D}}_j\right)$. The gradient with respect to $\tilde{\mathbf{D}}_j$

is given by[3]

$$\begin{aligned}\nabla f_{\mu,j}(\tilde{\mathbf{D}}_j) &= \left.\frac{\partial f_{\mu,j}}{\partial \tilde{\mathbf{D}}_j}\right|_{\mathbf{x}_j(\tilde{\mathbf{D}}_j)} + \left.\frac{\partial f_{\mu,j}}{\partial \mathbf{x}_j}\right|_{\tilde{\mathbf{D}}_j} \cdot \frac{\partial \mathbf{x}_j}{\partial \tilde{\mathbf{D}}_j} \\ &= -2\left(\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\mathbf{x}_j\right)\mathbf{x}_j^T - 2\tilde{\mathbf{D}}_j^T\left(\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\mathbf{x}_j\right) \cdot \frac{\partial \mathbf{x}_j}{\partial \tilde{\mathbf{D}}_j}.\end{aligned}$$

Note that $\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\mathbf{x}_j = \tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\tilde{\mathbf{D}}_j^{\dagger}\tilde{\mathbf{y}}_j$ is orthogonal to the columns of $\tilde{\mathbf{D}}_j$. One has $\tilde{\mathbf{D}}_j^T\left(\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\mathbf{x}_j\right) = \mathbf{0}$. As a result,

$$\nabla f_{\mu,j}(\tilde{\mathbf{D}}_j) = -2\left(\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\mathbf{x}_j\right)\mathbf{x}_j^T. \tag{3.19}$$

From the definition of $\tilde{\mathbf{D}}_j$ in (3.5), $\mathbf{D}_j$ is a sub-matrix of $\tilde{\mathbf{D}}_j$, and therefore $\nabla f_{\mu,j}(\mathbf{D}_j)$ is also a sub-matrix of $\nabla f_{\mu,j}(\tilde{\mathbf{D}}_j)$, i.e., $\nabla f_{\mu,j}(\mathbf{D}_j) = \left(\nabla f_{\mu,j}(\tilde{\mathbf{D}}_j)\right)_{1:m,1:m_j}$.

A similar procedure is used to compute the second order derivative $\nabla_{\mathbf{H}_j}\nabla f_{\mu}(\mathbf{D}_j)$. For a given $\mathbf{H}_j \in \mathbb{R}^{m\times m_j}$, define $\tilde{\mathbf{H}}_j = \left[\mathbf{H}_j^T, \mathbf{O}_{m_j}\right]^T \in \mathbb{R}^{(m+m_j)\times m_j}$, where $\mathbf{O}_{m_j}$ is the $m_j \times m_j$ zero matrix. By the definition of $\tilde{\mathbf{D}}_j$ and directional derivative, it can be verified that $\nabla_{\mathbf{H}_j}\nabla f_{\mu}(\mathbf{D}_j) = \left(\nabla_{\tilde{\mathbf{H}}_j}\nabla f_{\mu}\left(\tilde{\mathbf{D}}_j\right)\right)_{1:m,1:m_j}$. We compute $\nabla_{\tilde{\mathbf{H}}_j}\nabla f_{\mu}\left(\tilde{\mathbf{D}}_j\right)$ as follows: For any $\tilde{\mathbf{H}}_j$,

$$\begin{aligned}&\frac{1}{2}\nabla_{\tilde{\mathbf{H}}_j}\nabla f_{\mu}\left(\tilde{\mathbf{D}}_j\right) \\ &= \left(\nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j\right)\mathbf{x}_j\mathbf{x}_j^T + \tilde{\mathbf{D}}_j\left(\nabla_{\tilde{\mathbf{H}}_j}\mathbf{x}_j\right)\mathbf{x}_j^T - \left(\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\mathbf{x}_j\right)\left(\nabla_{\tilde{\mathbf{H}}_j}\mathbf{x}_j\right)^T.\end{aligned} \tag{3.20}$$

It is clear that $\nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j = \tilde{\mathbf{H}}_j$. To compute the other term $\nabla_{\tilde{\mathbf{H}}_j}\mathbf{x}_j$, note that $\mathbf{x}_j$ is a function of $\tilde{\mathbf{D}}_j$ involving matrix inversion. To proceed, we use the fact that for any invertible matrix $\mathbf{A}$, it holds $\nabla\mathbf{A}^{-1} = -\mathbf{A}^{-1}(\nabla\mathbf{A})\mathbf{A}^{-1}$ (derived by differentiating both sides of $\mathbf{A}\cdot\mathbf{A}^{-1} = \mathbf{I}$). As a result, one has $\nabla_{\tilde{\mathbf{H}}_j}\mathbf{x}_j = \nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j^{\dagger}\tilde{\mathbf{y}}_j$ where $\nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j^{\dagger}$ is given

[3]Note that the term $\left.\frac{\partial f_{\mu,j}}{\partial \mathbf{x}_j}\right|_{\tilde{\mathbf{D}}_j} \cdot \frac{\partial \mathbf{x}_j}{\partial \tilde{\mathbf{D}}_j}$ is a product of a vector and a tensor defined as $\sum_k \left(\left.\frac{\partial f_{\mu,j}}{\partial (\mathbf{x}_j)_k}\right|_{\tilde{\mathbf{D}}_j}\right) \cdot \frac{\partial (\mathbf{x}_j)_k}{\partial \tilde{\mathbf{D}}_j}$, where $(\mathbf{x}_j)_k$ is the kth element of the vector $\mathbf{x}_j$.

by

$$\begin{aligned}
\nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j^{\dagger} &= \nabla_{\tilde{\mathbf{H}}_j}\left(\left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\tilde{\mathbf{D}}_j^T\right) \\
&= -\left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\nabla_{\tilde{\mathbf{H}}_j}\left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)\left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\tilde{\mathbf{D}}_j^T \\
&\quad + \left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j^T \\
&= -\left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\left(\nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j^T\right)\tilde{\mathbf{D}}_j\tilde{\mathbf{D}}_j^{\dagger} - \tilde{\mathbf{D}}_j^{\dagger}\left(\nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j\right)\tilde{\mathbf{D}}_j^{\dagger} \\
&\quad + \left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\nabla_{\tilde{\mathbf{H}}_j}\tilde{\mathbf{D}}_j^T \\
&= \left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\tilde{\mathbf{H}}_j^T\left(\mathbf{I} - \tilde{\mathbf{D}}_j\tilde{\mathbf{D}}_j^{\dagger}\right) - \tilde{\mathbf{D}}_j^{\dagger}\tilde{\mathbf{H}}_j\tilde{\mathbf{D}}_j^{\dagger}.
\end{aligned}$$

Define $\tilde{\mathbf{y}}_{e,j} = \tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\mathbf{x}_j$. Then,

$$\begin{aligned}
\nabla_{\tilde{\mathbf{H}}_j}\mathbf{x}_j &= \left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\tilde{\mathbf{H}}_j^T\left(\tilde{\mathbf{y}}_j - \tilde{\mathbf{D}}_j\mathbf{x}_j\right) - \tilde{\mathbf{D}}_j^{\dagger}\tilde{\mathbf{H}}_j\mathbf{x}_j \\
&= \left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\tilde{\mathbf{H}}_j^T\tilde{\mathbf{y}}_{e,j} - \left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\tilde{\mathbf{D}}_j^T\tilde{\mathbf{H}}_j\mathbf{x}_j \\
&= \left(\tilde{\mathbf{D}}_j^T\tilde{\mathbf{D}}_j\right)^{-1}\left(\tilde{\mathbf{H}}_j^T\tilde{\mathbf{y}}_{e,j} - \tilde{\mathbf{D}}_j^T\tilde{\mathbf{H}}_j\mathbf{x}_j\right). \qquad (3.21)
\end{aligned}$$

Substitute (3.21) into (3.20). One is able to compute $\nabla_{\tilde{\mathbf{H}}_j}\nabla f_\mu\left(\tilde{\mathbf{D}}_j\right)$, and hence $\nabla_{\mathbf{H}_j}\nabla f_\mu\left(\mathbf{D}_j\right)$, and $\nabla_{\mathbf{H}}\nabla f_\mu$.

3.5.3 Line Search Path

The line search mechanism used in this chapter is significantly different from the standard one due to the column norm constraint in (3.1). In a standard line search algorithm, the kth iteration outputs an updated dictionary $\mathbf{D}^{(k)}$ via

$$\mathbf{D}^{(k)} = \mathbf{D}^{(k-1)} + t \cdot \mathbf{H}, \qquad (3.22)$$

where $t \in \mathbb{R}^+$ is a properly chosen step size and $\mathbf{H} \in \mathbb{R}^{m\times d}$ is the line search direction. Common choices for the search direction include $\mathbf{H} = -\nabla\mathbf{f}_\mu$ for the gradient descent method and $\mathbf{H} = -\left(\nabla^2\mathbf{f}_\mu\right)^{-1}\nabla\mathbf{f}_\mu$ for the Newton method. However, a direct application of (3.22) generally results in a dictionary $\mathbf{D} \notin \mathcal{D}$.

The line search path in this section is restricted to the product of Grassmann manifolds.

This is because, as has been discussed in Section 3.4, the objective function f_μ is indeed a function defined on the product of Grassmann manifolds. On the Grassmann manifold $\mathcal{G}_{m,1}$, the geodesic path plays the same role as the straight line in the Euclidean space: given any two distinct points on $\mathcal{G}_{m,1}$, the shortest path that connects these two points is geodesic [59]. Specifically, let $\mathscr{U} \in \mathcal{G}_{m,1}$ be a one-dimensional subspace and $\mathbf{u} \in \mathcal{U}_{m,1}$ be the corresponding generator matrix (not unique).[4] Consider a search direction $\mathbf{h} \in \mathbb{R}^m$ with $\|\mathbf{h}\|_2 = 1$ and $\mathbf{h}^T\mathbf{u} = \mathbf{0}$. Then the geodesic path starting from $\mathbf{u}$ along the direction $\mathbf{h}$ is given by [59]

$$\mathbf{u}(t) = \mathbf{u} \cdot \cos t + \mathbf{h} \cdot \sin t, \ t \in \mathbb{R}. \tag{3.23}$$

Note that $\mathbf{u}(t) = -\mathbf{u}(t+\pi)$ and hence $\operatorname{span}(\mathbf{u}(t)) = \operatorname{span}(\mathbf{u}(t+\pi))$. In practice, one can restrict the search path within the interval $t \in [0, \pi)$.

For the dictionary update problem at hand, the line search path is defined as follows. Let $\mathbf{H} \in \mathbb{R}^{m\times d}$ be the search direction. ($\mathbf{H} = -\nabla f_\mu$ for the gradient descent method and $\mathbf{H} = -\left(\nabla^2 f_\mu\right)^{-1} \nabla f_\mu$ for the Newton method.) Let $\mathbf{h}_i$ be the ith column of $\mathbf{H}$. Define

$$\bar{\mathbf{h}}_i = \mathbf{h}_i - \mathbf{D}_{:,i}\mathbf{D}_{:,i}^T\mathbf{h}_i, \ \forall i \in \mathcal{I}, \tag{3.24}$$

so that $\bar{\mathbf{h}}_i$ and $\mathbf{D}_{:,i}$ are orthogonal. The line search path for dictionary update, say $\mathbf{D}(\mathbf{t})$, $t \geq 0$, is given by [59]

$$\begin{cases} \mathbf{D}_{:,i}(t) = \mathbf{D}_{:,i} \quad \text{if } \|\bar{\mathbf{h}}_i\|_2 = 0, \\ \mathbf{D}_{:,i}(t) = \mathbf{D}_{:,i}\cos\left(\|\bar{\mathbf{h}}_i\|_2 t\right) + \left(\bar{\mathbf{h}}_i / \|\bar{\mathbf{h}}_i\|_2\right)\sin\left(\|\bar{\mathbf{h}}_i\|_2 t\right) \\ \qquad\qquad \text{if } \|\bar{\mathbf{h}}_i\|_2 \neq 0. \end{cases} \tag{3.25}$$

3.6 Convergence of Primitive SimCO

The focus of this section is on the convergence performance of primitive SimCO when the index set $\mathcal{I}$ contains only one index. The analysis shows deep connections between primitive SimCO and K-SVD. It is clear that the optimization formulations of primitive SimCO and K-SVD are exactly the same when $|\mathcal{I}| = 1$. However, the methods used to solve the optimization problem are quite different: primitive SimCO uses standard

[4] The generator matrix $\mathbf{u}$ is a vector in this case.

optimization methods while K-SVD employs SVD. The question is whether these two different approaches will give the same solution eventually. Theorem 1 of this section shows that a gradient descent finds a global optimum with probability one. Hence, when $|\mathcal{I}| = 1$, primitive SimCO and K-SVD are the same in terms of ultimate learning performance. Note that, even though the general case where $|\mathcal{I}| > 1$ is more interesting, it remains open which point SimCO will converge to in this case.

When $|\mathcal{I}| = 1$, the rank-one matrix approximation problem arises in both primitive SimCO and K-SVD. Formally, let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be a matrix, where $m \geq 1$ and $n \geq 1$ are arbitrary positive integers. Without loss of generality, assume that $m \leq n$. Suppose that the sorted singular values satisfy $\lambda_1 > \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_m$. Define

$$f(\mathbf{u}) = \min_{\mathbf{w}\in\mathbb{R}^n} \left\|\mathbf{A} - \mathbf{u}\mathbf{w}^{\mathbf{T}}\right\|_F^2, \ \forall \mathbf{u} \in \mathcal{U}_{m,1}. \tag{3.26}$$

The rank-one matrix approximation problem can be written as the following optimization problem

$$\min_{\mathbf{u}\in\mathcal{U}_{m,1}} f(\mathbf{u}). \tag{3.27}$$

The performance of gradient descent is analyzed in Theorem 1 for the rank-one matrix approximation problem. To avoid numerical problems that may arise in practical implementations, we consider an ideal gradient descent procedure with infinitesimal step sizes. (Note that true gradient descent requires infinitesimal steps.)

Theorem 1. *Consider a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and its singular value decomposition. Employ the gradient descent procedure with infinitesimal steps to solve (3.26). Suppose the starting point, denoted by $\mathbf{u}_0$, is randomly generated from the uniform distribution on $\mathcal{U}_{m,1}$. Then the gradient descent procedure finds a global minimizer with probability one.*

The proof is detailed in Appendix A.1.

Remark 2. The notion of Grassmann manifold is essential in the proof. The reason is that the global minimizer is unique up to the subspace spanned by $\mathbf{u}$: if $\mathbf{u} \in \mathbb{R}^m$ is a global minimizer, then so is $\mathbf{u}'$ for all $\mathbf{u}'$ such that $\operatorname{span}(\mathbf{u}') = \operatorname{span}(\mathbf{u})$.

Remark 3. According to the authors' knowledge, this is the first result showing that a gradient search on Grassmann manifold solves the rank-one matrix approximation

problem. In the literature, it has been shown that there are multiple stationary points for the rank-one matrix approximation problem [2, Proposition 4.6.2]. Our results show that a gradient descent method will not converge to any other stationary points than global minimizers. More recently, the rank-one decomposition problem where $\lambda_2 = \lambda_3 = \cdots = \lambda_m = 0$ was studied in [49]. Our proof technique is significantly different as the effects of the eigen-spaces corresponding to $\lambda_2, \cdots, \lambda_m$ need to be considered for the rank-one approximation problem.

3.7 Empirical Tests

In this section, we numerically test the proposed primitive and regularized SimCO. In the test of SimCO, all codewords are updated simultaneously, i.e., $\mathcal{I} = [d]$. In Section 3.7.1, we show that MOD[5], K-SVD, and primitive SimCO may result in an ill-conditioned dictionary while regularization can mitigate this problem. Learning performance of synthetic and real data is presented in Sections 3.7.2 and 3.7.3 respectively. A running time comparison of different algorithms is conducted in Section 3.7.4.

It is worth noting that Algorithm 2 (gradient descent) is used for the analysis of singular points in Section 3.7.1 because of the reasons explained in Footnote 1. Algorithm 3 (LSNCG) is employed for synthetic and real data tests in Sections 3.7.2 and 3.7.3 due to its fast convergence rate. Both regularized "K-SVD" and regularized MOD are based on second order optimization methods to ensure a fair comparison.

3.7.1 Ill-conditioned Dictionaries

In this subsection, we handpick a particular example to show that MOD, K-SVD and primitive SimCO may converge to an ill-conditioned dictionary. In the example, the training samples $\mathbf{Y} \in \mathbb{R}^{16\times 78}$ are computed via $\mathbf{Y} = \mathbf{D}_{\text{true}}\mathbf{X}_{\text{true}}$, where $\mathbf{D}_{\text{true}} \in \mathbb{R}^{16\times 32}$, $\mathbf{X}_{\text{true}} \in \mathbb{R}^{32\times 78}$, and each column of $\mathbf{X}$ contains exactly 4 nonzero components. We

[5]In the tested MOD, the columns in $\mathbf{D}$ are normalized after each dictionary update. This extra step is performed because many sparse coding algorithms require a normalized dictionary. Furthermore, our preliminary simulations (not shown in this chapter) show that the performance of dictionary update could seriously deteriorate if the columns are not normalized.

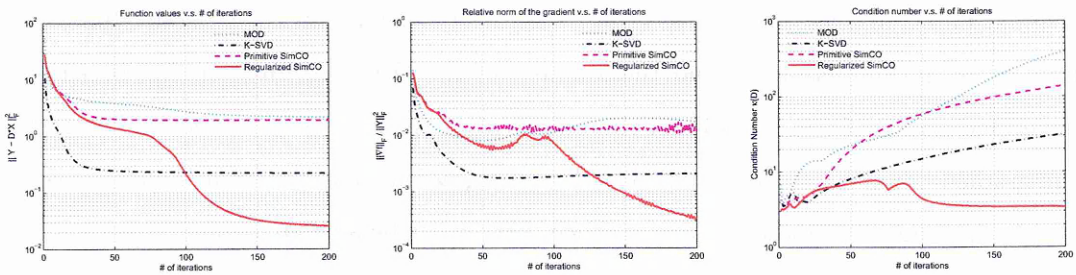


Figure 3.1: Starting with the same point, the convergence behaviors of MOD, K-SVD, primitive SimCO and regularized SimCO are different. In this particular example, only regularized SimCO avoids converging to a singular point.

assume that the sparse coding stage is perfect, i.e., the true sparsity pattern Ω_{true} is available. We start with a particular choice of the initial dictionary $\mathbf{D}_0 \in \mathcal{D}$. The regularization constant μ is set to 0 and 0.01 for primitive and regularized SimCOs respectively. Define the *condition number of a dictionary* $\mathbf{D}$ as

$$\kappa(\mathbf{D}) = \max_{1 \le j \le n} \lambda_{\max}(\mathbf{D}_j) / \lambda_{\min}(\mathbf{D}_j),$$

where $\mathbf{D}_j = \mathbf{D}_{:,\Omega(:,j)}$. The numerical results are presented in Figure 3.1, where $f_{\mu=0}$, $\|\nabla f_{\mu=0}\|_F / \|\mathbf{Y}\|_F$, and $\kappa(\mathbf{D})$ are compared from the left to the right. Note that in this example, $\kappa(\mathbf{D}_{\text{true}}) = 3.39$.

The results in Figure 3.1 show that

1. When the number of iterations exceeds 50, MOD, K-SVD and primitive SimCO stop improving the training performance. Surprisingly, the gradient $\nabla f_{\mu=0}$ in these methods does not converge to zero. This implies that these methods *do not converge to a local minimizer*. A more careful study reveals that these algorithms converge to singular points where $\kappa(\mathbf{D})$ becomes large ($\kappa(\mathbf{D}) > 10$ for MOD, K-SVD, and primitive SimCO).

2. By adding a regularization term and choosing the regularization constant properly, regularized SimCO avoids the convergence to an ill-conditioned dictionary, hence improves the performance.

It is worth noting that the SimCO formulation is crucial for distinguishing between

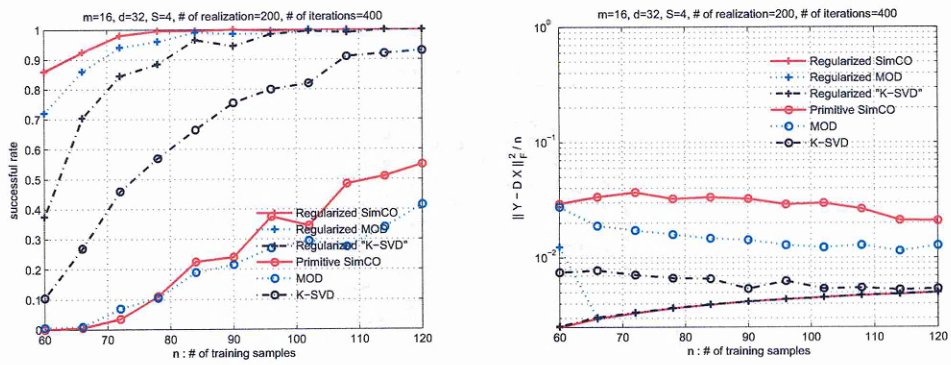
singular points and local minimizers. In the SimCO formulation, the objective function only involves the dictionary, and the gradient of the objective function can be easily computed via (3.15) and (3.19). If the search process converges to a local minimizer, the gradient should converge to zero. When the gradient does not vanish and changes rapidly in a neighborhood, the convergence point must be a singular point.

We also observe that the singular points, rather than the local minima, are the bottleneck. Towards this end, we randomly pick converged dictionaries in MOD, K-SVD, and primitive SimCO (from the case where there is no noise and Ω_{true} is priorly known). Surprisingly, we found that these algorithms either converge to a global minimizer or a singular point. Among the randomly picked converged dictionaries, no local minimizer has been found yet. Furthermore, as we will show in the next subsection, by adding the regularization term and forcing the search path away from singular points, substantial performance improvement can be achieved. All these suggest that singular points tend to be the major obstacle preventing these algorithms from converging to a global minimizer.

3.7.2 Experiments on Synthetic Data

The setting for synthetic data tests is summarized as follows. The training samples are generated via $\mathbf{Y} = \mathbf{D}_{\text{true}}\mathbf{X}_{\text{true}}$. Here, the columns of $\mathbf{D}_{\text{true}}$ are randomly generated from the uniform distribution on the Stiefel manifold $\mathcal{U}_{m,1}$. Each column of $\mathbf{X}_{\text{true}}$ contains exactly S many non-zeros: the position of the non-zeros are uniformly distributed on the set $\binom{[d]}{S} = \{\{i_1, \cdots, i_S\} : 1 \leq i_k \neq i_\ell \leq d\}$; and the values of the non-zeros are standard Gaussian distributed. In the tests, we fix $m = 16$, $d = 32$, and $S = 4$, and change n, i.e., the number of training samples. Generally speaking, the fewer training samples there are, the more challenging the dictionary update is. In our experiments, we intentionally choose the challenging case with small n.

We first focus on the performance of dictionary update by assuming that the true sparsity pattern Ω_{true} is available. In regularized methods, the regularization constant μ is sequentially reduced to zero: the total number of iterations is set to 400; we change μ from $1e-1$ to $1e-2$, $1e-3$, and $1e-4$, for every 100 iterations. Experiments for



(a) Noiseless case.

(b) Noisy case: SNR of training samples is 20 dB. Note that there always exists a floor in the reconstruction error which is proportional to noise.

Figure 3.2: Performance comparison of dictionary update (no sparse coding step).

both noiseless and noisy cases are performed. Note that in the noiseless case, the sparse representation distortion $\|\mathbf{Y} - \mathbf{DX}\|_F^2/n$ can approach zero. It is more indicative to use success rate rather than distortion: a success is claimed when $\|\mathbf{Y} - \mathbf{DX}\|_F^2/n \leq 10^{-7}$ and a failure is claimed otherwise. For the noisy case, there always exists a floor in the representation distortion that is proportional to noise. The normalized distortion $\|\mathbf{Y} - \mathbf{DX}\|_F^2/n$ serves as a good performance measure. The simulation results are presented in Figure 3.2. It is evident that regularization significantly improves the performance and that among all the regularized methods, regularized SimCO is consistently better than others.

Then we evaluate the overall dictionary learning performance by combining the dictionary update and sparse coding stages. For sparse coding, we adopt the OMP algorithm [150] as it has been used for testing the K-SVD method in [5, 60]. The overall dictionary learning procedure is given in Algorithm 1. We refer to the iterations between sparse coding and dictionary learning stages as outer-iterations, and the iterations within the dictionary update stage as inner-iterations. In our tests, the number of outer-iterations is set to 50, and the number of inner-iterations of is set to 1. Furthermore, in regularized SimCO, the regularized constant is set to $\mu = 1e-1$ during the first 30 outer-iterations, and $\mu = 0$ during the rest 20 outer-iterations. The normalized learning performance $\|\mathbf{Y} - \mathbf{DX}\|_F^2/n$ is depicted in Figure 3.3. Again, the average performance

of regularized SimCO is consistently better than that of other methods.

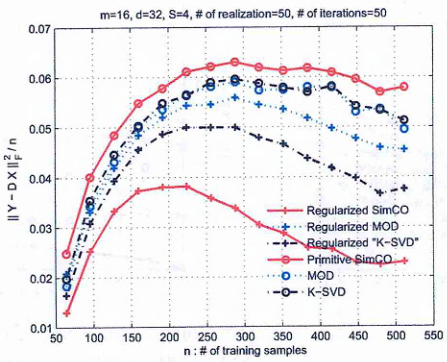


Figure 3.3: Performance comparison of dictionary learning using OMP for sparse coding.

3.7.3 Numerical Results for Image Denoising

As we mentioned in the introduction, dictionary learning methods have many applications. In this subsection, we look at one particular application, i.e., image denoising. Here, an image corrupted by noise was used to train the dictionary: we take 1,000 (significantly less than 65,000 used in [60]) blocks (of size 8×8) of the corrupted image as training samples. The number of codewords in the training dictionary is $d = 256$. For dictionary learning, we iterate the sparse coding and dictionary update stages 10 times. The sparse coding stage is based on the OMP algorithm implemented in [60]. In the dictionary update stage, different algorithms are tested and the number of iterations in dictionary update is set to 50. The regularization constant is set to $\mu = 0.05$. After the whole process of dictionary learning, we use the learned dictionary to reconstruct the image. The reconstruction results are presented in Figure 3.4. The Peak Signal-to-Noise Ratio (PSNR) is used as a measurement of the reconstruction quality. Better quality leads to higher PSNR. It is defined as, $PSNR = 20\log_{10}(\frac{MAX}{\sqrt{MSE}})$, where MAX indicates the maximum possible pixel value of the image. While all dictionary learning methods significantly improves the image PSNR, the largest gain was obtained from regularized SimCO.

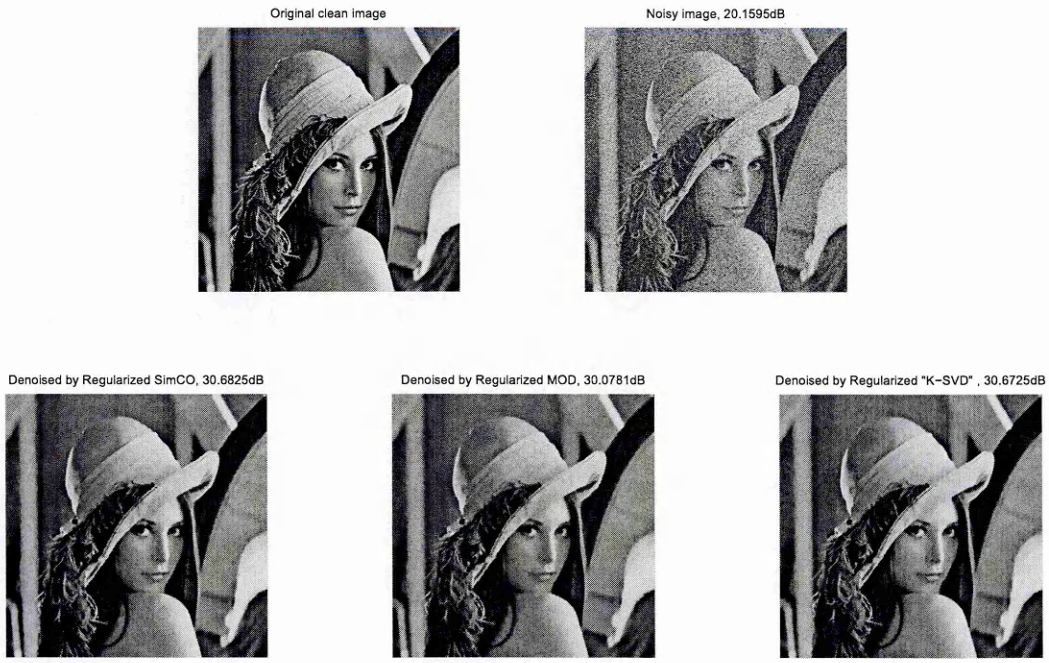


Figure 3.4: Example of the image denoising using dictionary learning. PSNR values in dB are given in sub-figure titles.

3.7.4 Comments on the Running Time

This subsection compares the computational complexity of MOD, K-SVD, and SimCO. As detailed in Sections 3.3 and 3.5, MOD uses an approximation to the Hessian while Algorithm 3 is based on the exact Hessian (without explicitly computed). As a result, the complexity of MOD and SimCO is on the same level: the computational cost for each MOD iteration is less than that for each SimCO iteration, but the number of iterations required for convergence in MOD is larger than that in SimCO. As opposed to MOD and SimCO where all codewords are simultaneously updated, K-SVD updates codewords individually. Despite the fact that a closed form solution can be obtained for each update via SVD, the speed of K-SVD is often slower than MOD and SimCO because of the individual update. The actual running time in practice is compared for different algorithms in Table 3.1. The numerical comparison is consistent with the qualitative analysis above.

Table 3.1: Comparison of running time (in seconds) for dictionary learning. Note that sparse coding step was included in producing Figures 3.3 and 3.4.

	MOD	K-SVD	Prim. SimCO	Reg. MOD	Reg. "K-SVD"	Reg. SimCO
Fig. 3.2(a)	2.4×10^4	2.0×10^5	5.1×10^4	5.9×10^3	2.0×10^5	2.3×10^4
Fig. 3.2(b)	2.3×10^4	1.9×10^5	5.0×10^4	6.3×10^3	8.6×10^4	2.1×10^4
Fig. 3.3	1.5×10^4	3.7×10^4	3.1×10^4	4.2×10^4	9.7×10^4	8.7×10^4
Fig. 3.4	-	-	-	7.50	18.13	14.68

3.8 A Fast Version of SimCO via Codeword Clustering and Hierarchical Sparse Coding

3.8.1 The Proposed Method

The dictionary learning process of SimCO framework is achieved by alternate iterations between sparse coding and dictionary update. The learning process, however, may involve a higher computational complexity, rendering the algorithms to be less practical in computation extensive applications, for example, when dealing with large scale or high-dimensional data. We propose a new method to improve the computational efficiency of dictionary learning algorithms based on codeword clustering and hierarchical sparse coding, based on the original SimCO presented in above sections. The details of the proposed method, i.e. fast SimCO, are given as follows.

When developing the SimCO algorithm, we use the OMP algorithm by Pati in [122] for the sparse coding stage, which aims to solve the optimization problem: Given data matrix $\mathbf{Y}$, find a sparse coefficient matrix $\mathbf{X}$ to minimize $\|\mathbf{Y} - \mathbf{DX}\|_F^2$ for a given overcomplete dictionary $\mathbf{D}$. In OMP, this is achieved by finding the column vector in $\mathbf{D}$ which most closely resembles a residual vector $\mathbf{r}$, which is initialized to $\mathbf{y}$, then adjusted at each iteration to take into account the vectors previously chosen. In the proposed fast SimCO algorithm, we propose to make the following improvements to the SimCO algorithm. First, the dictionary atoms obtained in the dictionary update stage of SimCO are clustered using a K-means algorithm. The cluster centers represent a high-level representation of the dictionary, with the atoms in their neighborhoods representing the low-level dictionary. In the sparse coding stage, the closest centroid from the higher level dictionary to the signal under consideration is found, and their

neighbors are then used to code this signal with a dimension reduced OMP, based on the nearest neighbor search. A tree structure with the multi-level dictionary, obtained by multi-level K-mean clustering, is applied for sparse coding process. We call this new algorithm as the tree-OMP (TOMP) method, summarized in Algorithm 4. The atoms of the dictionary are organized by a tree structure to improve the computational efficiency in the coding stage. In each iteration, the atom that is closest to the residual vector, is selected from the cluster centroids of the atoms in the dictionary, instead of all the atoms in the dictionary, and the coding is performed in the neighborhood of each centroid. Simulations are given in Section 3.8.2 to demonstrate its advantage in computational efficiency.

We have also tested an approximate version of TOMP, called the centralized OMP method (COMP), where only the centroid that is closest to the residual vector is selected in each iteration. The only difference is that the sub-optimization problem in step (4) of TOMP is now replaced by using the centroid directly in COMP. This apparently improves the computational efficiency but may degrade the sparse coding performance, as shown in the next section.

3.8.2 Simulation Results for Fast SimCO

Firstly, we evaluate the proposed fast SimCO algorithm (i.e. using TOMP in the sparse coding stage, but the same dictionary update stage as the original SimCO algorithm) on synthetic data. We compare TOMP with OMP and COMP in the coding stage. As in [52], we refer to the iterations between sparse coding and dictionary learning stages as outer-iterations, and the iterations within the dictionary update stage as inner-iterations. In our tests, all results are averaged over 50 realizations with a random initialization for each realization. The numbers of outer-iterations are set to 20 for all of three algorithms, and in each outer iteration, the numbers of inner-iterations of all the algorithms are set to 1. Furthermore, in dictionary update stage, the regularized constant, as in [52], is set to $\mu = 1e - 1$ during the first 10 outer-iterations, and $\mu = 0$ during the remaining 10 outer-iterations. The average running time by OMP, TOMP and COMP respectively is shown in Figure 3.5. The approximation error

Algorithm 4 TOMP

Task: To find the sparse representation $\mathbf{X}$ from the data sample matrix $\mathbf{Y}$.
Input: The initial $m * d$ dictionary $\mathbf{D}$, the $m * n$ matrix $\mathbf{Y}$, and the sparsity L.
Output: The $d * n$ coefficient matrix $\mathbf{X}$
Initialize: The residual $\mathbf{r}_0 = \mathbf{y}$, the index set Λ is empty and the iteration counter $j = 1$.
Do:

1. Cluster the atoms in dictionary $\mathbf{D}$ by K-means algorithm to obtain the centroids of the atoms $\mathbf{d}_{c_1}, ..., \mathbf{d}_{c_e}$, and e is the number of centers.
2. Find the index of one of the centers c_b that solves the optimization problem $c_b = \arg\max_{i=1,...e} | < \mathbf{r}_{j-1}, \mathbf{d}_{c_i} > |$.
3. Find the index λ_j that solves the sub-optimization problem $\lambda_j = \arg\max_j | < \mathbf{r}_{j-1}, \hat{\mathbf{d}}_{c_b} > |$.
4. Combine the index set and the chosen atom: $\Lambda_j = \Lambda_{j-1} \bigcup \{\lambda_j\}$ and $\mathbf{D}_j = [\mathbf{D}_{j-1}\mathbf{d}_{\lambda_j}]$.
5. Solve the least squares problem to obtain a new coefficient estimate: $\mathbf{x}_j = \arg\max_{\mathbf{x}} \|\mathbf{y} - \mathbf{D}_j\mathbf{x}\|_2$.
6. Calculate the new approximation of the data and the new residual $\hat{\mathbf{y}}_j = \mathbf{D}_j\ \mathbf{x}_j$, $\mathbf{r}_j = \mathbf{y}_j - \hat{\mathbf{y}}_j$.
7. Increase j and return to Step (2) if j is smaller than L. This leads to the search of another best centroid.

$\|\mathbf{Y} - \mathbf{DX}\|_F^2 / n$ versus n are depicted in Figure 3.6.

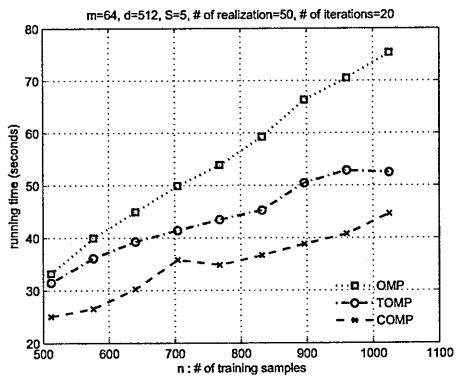


Figure 3.5: Average running time comparison between fast SimCO and the baselines.

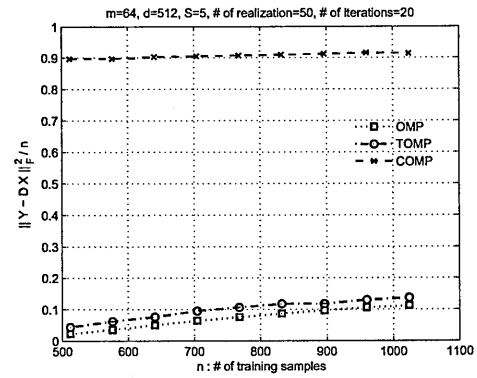


Figure 3.6: Average approximation error comparison between fast SimCO and the baselines.

It can be observed that TOMP improves computational efficiency over OMP, while maintaining a similar reconstruction performance, and COMP, despite being most efficient, gives the worst performance among the three algorithms. Secondly, we test the fast SimCO algorithm for image denoising, and compare it with the original SimCO. All parameters setting of this experiment is the same as in Section 3.7.3. The TOMP instead of OMP algorithm is used in the sparse coding stage in the proposed fast SimCO algorithm which offers a similar PSNR 30.7949 dB for image denoising application but only needs 11.56 seconds, as opposed to 14.68 seconds required by the original SimCO.

3.9 Summary

We have presented a new framework for dictionary update. It allows not only a simultaneous update of all codewords and the corresponding coefficients but also the observation that singular points rather than local minima are the bottleneck for the dictionary update. To mitigate the effects of singularity, regularized SimCO has been proposed. First and second order optimization procedures have been implemented. Numerical experiments verify that regularization substantially improves the performance. A fast SimCO algorithm has also been developed to further improve the computational efficiency of SimCO.

Chapter 4

Multi-stage Underdetermined Blind Speech Separation Based on Sparse Signal Recovery with Learned and Predefined Dictionaries

4.1 Introduction

Over the past two decades, 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 [42, 80]. 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 chapter, can be described as:

$$\mathbf{Z} = \mathbf{AS} + \mathbf{V} \tag{4.1}$$

where $\mathbf{A} \in R^{M \times N}$ is the unknown mixing matrix assumed to be of full row rank, $\mathbf{Z} \in R^{M \times T}$ is the observed data matrix whose row vector $\mathbf{z}_i$ is the ith sensor signal having T samples at discrete time instants $t = 1, ..., T$, $\mathbf{S} \in R^{N \times T}$ 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 $\mathbf{Z}$, 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.

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. 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 [105] have been developed to address this problem, it remains an open problem. 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 [6, 84, 93, 108, 133]. Although the convolutive BSS model is not the focus of this chapter, the methods developed here can also be used in convolutive speech separation algorithms. For example, in many frequency domain BSS algorithms, the convolutive model is transformed into the frequency domain, leading to multiple instantaneous but complex-valued BSS problems to be solved. This is subject to permutation alingnment and scale ambiguity correction, along the frequency channels [138, 157].

In this chapter, 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 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, for example, by the discrete cosine transform (DCT). Among the dictionary learning algorithms compared, SimCO [52], described in Chapter 3, is used for the first time in an underdetermined speech separation application, offering 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 [152] using the same datasets and evaluation approach. Results of this work have been presented in [160–162].

The remainder of the chapter is organized as follows. The proposed multi-stage method with clustering, dictionary learning, blocking, separating and reconstruction stages is presented in Section 4.2. Experimental results are given in Section 4.3. Finally, conclusions and future work are summarized in Section 4.4.

4.2 The Proposed Multi-stage System

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 (4.1). However, the approach can be readily extended to other underdetermined cases with various numbers of sources and mixtures. As depicted in Figure 4.1, 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 $\hat{\mathbf{A}}$ 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.

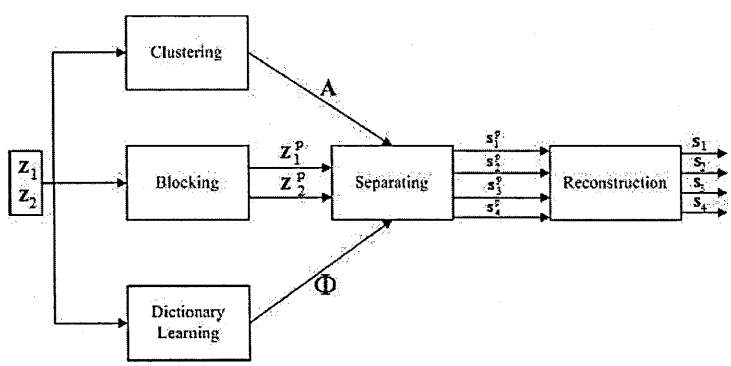
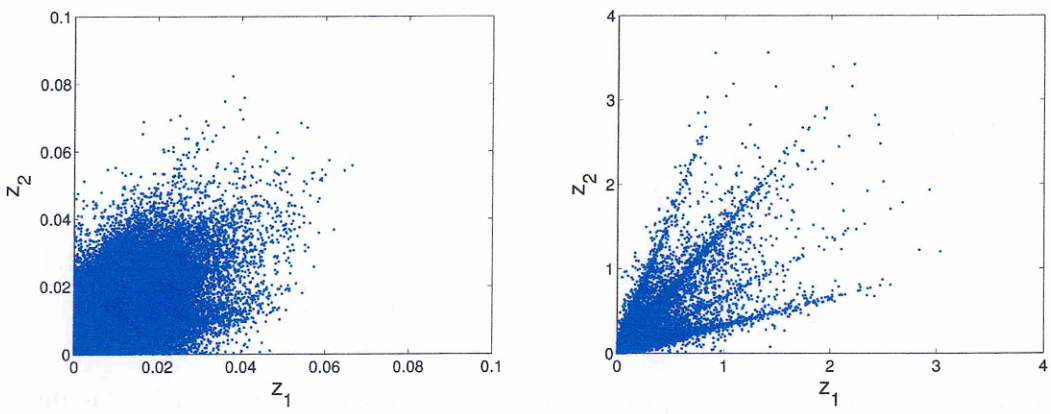


Figure 4.1: The flow chart of the proposed system for separating four speech sources from two mixtures.

4.2.1 Estimating the Mixing Matrix by Clustering

In the clustering stage, we use the standard technique as in [168] to estimate the mixing matrix $\mathbf{A}$ by using the K-means clustering algorithm based on the 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 $\mathbf{z}_1$ versus $\mathbf{z}_2$ 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, due to the observation noise in real data. The lines are usually broadened especially in the time domain, as shown in Figure 4.2(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 Figure 4.2(b).



(a) Time domain (b) Frequency domain

Figure 4.2: 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.

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 5.

4.2.2 Separating Sources by Sparse Signal Recovery

In the separating stage, with the estimated mixing matrix $\hat{\mathbf{A}}$, we formulate the underdetermined blind speech separation problem as a sparse signal recovery problem.

Algorithm 5 K-means algorithm for mixing matrix estimation

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

- Apply the STFT to each mixture signal in $\mathbf{Z}$ to obtain a spectrogram of this mixture signal then reshape it to a vector as the coefficient vector in $\tilde{\mathbf{Z}}$, i.e. the time-frequency representation of $\mathbf{Z}$.
- Normalize the vectors in $\tilde{\mathbf{Z}}$ 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 $\tilde{\mathbf{Z}}$ to four parts (equals to the number of sources) and compute the mean values of each part as the initial centres.
- Run the K-means clustering algorithm to update iteratively the four centres until convergence, and compute the column vectors of the estimated mixing matrix $\hat{\mathbf{A}}$ as the final cluster centres.

Equation (4.1) can be expanded as:

$$\begin{pmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_M \end{pmatrix} = \begin{pmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{M1} & \cdots & a_{MN} \end{pmatrix} \begin{pmatrix} \mathbf{s}_1 \\ \vdots \\ \mathbf{s}_N \end{pmatrix} \tag{4.2}$$

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

$$\underbrace{\begin{pmatrix} z_1(1) \\ \vdots \\ z_1(T) \\ \vdots \\ \vdots \\ z_M(1) \\ \vdots \\ z_M(T) \end{pmatrix}}_{\mathbf{b}} = \underbrace{\begin{pmatrix} \Lambda_{11} & \cdots & \Lambda_{1N} \\ \vdots & \ddots & \vdots \\ \Lambda_{M1} & \cdots & \Lambda_{MN} \end{pmatrix}}_{\mathbf{M}} \underbrace{\begin{pmatrix} s_1(1) \\ \vdots \\ s_1(T) \\ \vdots \\ \vdots \\ s_N(1) \\ \vdots \\ s_N(T) \end{pmatrix}}_{\mathbf{f}} \tag{4.3}$$

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 $\mathbf{b} = \text{vec}(\mathbf{Z}^T)$, $\mathbf{f} = \text{vec}(\mathbf{S}^T)$, where vec is an operator stacking the column vectors of a matrix into a single vector. Equation (4.3) can be written in a compact form as:

$$\mathbf{b} = \mathbf{M}\mathbf{f} \tag{4.4}$$

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 $\mathbf{f}$:

$$\mathbf{f} = \mathbf{\Phi}\mathbf{y} \tag{4.5}$$

where $\mathbf{\Phi}$ is a transform dictionary and $\mathbf{y}$ contains the weighting coefficients in the $\mathbf{\Phi}$ domain. It is noted that the transform dictionary $\mathbf{\Phi}$ is different from the transform dictionary $\mathbf{D}$ in chapter 3, because $\mathbf{\Phi}$ is for decomposing the reformulated vector $\mathbf{f}$ which is concatenated by N source signals. Combining (4.4) and (4.5), we have

$$\mathbf{b} = \mathbf{M}\mathbf{\Phi}\mathbf{y} \tag{4.6}$$

In equation (4.6), 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 (4.6) using sparse signal recovery (i.e. sparse coding) methods in Section 2.1.

The performance of these signal recovery methods will be studied in Section 4.3. Based on the reformulation of the speech separation problem to the problem of signal recovery as shown from Equation (4.3) to (4.6), the proposed speech separation algorithm is summarized in Algorithm 6. We would like to note that basis pursuit (BP), matching pursuit (MP), the least squares method L1LS used in Algorithm 6 are just examples of many sparse signal recovery algorithms, which, including the recent method FISTA [19], can all be used for reconstructing the sources.

Algorithm 6 Separating speech sources

Task: separating speech sources from each block based on signal recovery method
Input: $\mathbf{z}_i^p$, $i = 1, 2$, $p = 1, ..., P$. ($\mathbf{z}_i^p$ is one of the blocks of mixtures, which are generated from blocking stage to be discussed in Section 4.2.5), $\hat{\mathbf{A}}$. $\mathbf{\Phi}$.
Output: $\mathbf{s}_j^p$, $j = 1, ..., 4$, $p = 1, ..., P$.
Initialization: p=1.
Repeat:

- Form the measurement vector $\mathbf{b}^p$ by concatenating $\mathbf{z}_1^p$ with $\mathbf{z}_2^p$.
- Multiply $\mathbf{M}$ which is formed from $\hat{\mathbf{A}}$ (obtained from the clustering stage i.e. the output of Algorithm 5), 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 equation (4.5).
- Compute the source vectors $\mathbf{s}_j^p$ from $\mathbf{f}^p$.
- $p = p + 1$.

4.2.3 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 $\mathbf{\Phi}$. As discussed in chapter 3, 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.

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

$$\mathbf{T} = \mathbf{DG} \tag{4.7}$$

where $\mathbf{T}$ is a matrix in which each column contains one of the training samples, $\mathbf{D}$

Algorithm 7 Adaptive dictionary learning

Task: find the best dictionary to represent the training speech data
Input: $\mathbf{T}$.
Output: $\mathbf{D}$.
Initialization: Set the initial dictionary $\mathbf{D}^{(1)}$ and $j = 1$.
Repeat until convergence (use stop rule):

- Sparse coding stage: Fix the dictionary $\mathbf{D}^{(j)}$ and update $\mathbf{G}^{(j)}$ using some sparse coding technique, such as OMP.
- Dictionary update stage: Update $\mathbf{D}^{(j)}$, and $\mathbf{G}^{(j)}$ as appropriate, using e.g. SimCO dictionary update.
- $j = j + 1$.

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 [5] algorithm, the GAD [83], and the SimCO algorithm [52], are used to learn the dictionary $\mathbf{D}$ on the signal frames extracted from speech data (either mixtures or speech sources).

The dictionary learning algorithm in SDUBSS is summarized in Algorithm 7. Note that, $\mathbf{T}$ can be formed from speech mixtures $\mathbf{Z}$ or original speech sources $\mathbf{S}$, while $\mathbf{D}$ needs to be learned from $\mathbf{T}$. Then $\mathbf{D}$s constitute the transform dictionary $\mathbf{\Phi}$ by different manner as discussed below.

4.2.4 Dictionary Learning Strategies

It is an important practical issue [162] on how to train the dictionary from training data. We examine two different training strategies. To this end, we first expand Equation (4.5)

into two mixtures and four sources case as follows.

$$\underbrace{\begin{pmatrix} s_1(1) \\ \vdots \\ s_1(T) \\ s_2(1) \\ \vdots \\ s_2(T) \\ s_3(1) \\ \vdots \\ s_3(T) \\ s_4(1) \\ \vdots \\ s_4(T) \end{pmatrix}}_{\mathbf{f}} = \underbrace{\begin{pmatrix} D_1 & & & \\ & D_2 & & \\ & & D_3 & \\ & & & D_4 \end{pmatrix}}_{\Phi} \underbrace{\begin{pmatrix} y_1(1) \\ \vdots \\ y_1(T) \\ y_2(1) \\ \vdots \\ y_2(T) \\ y_3(1) \\ \vdots \\ y_3(T) \\ y_4(1) \\ \vdots \\ y_4(T) \end{pmatrix}}_{\mathbf{y}} \tag{4.8}$$

Source-trained Dictionary

The first strategy called the source-trained dictionary (STD) is depicted in Figure 4.3 where DL represents dictionary learning which can be achieved by any of the algorithms described in Section 2.3.2. 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 Φ for separating the sources in the following stages. For example, D_1 in Equation (4.8) is trained from the source $\mathbf{y}_1$. 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 $\lfloor . \rfloor$ rounds the argument to its nearest integer. The dictionary is then computed by one of the three learning algorithms described in Section 2.3.2 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

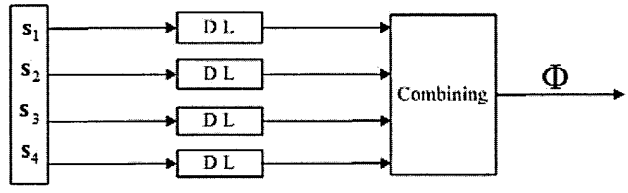


Figure 4.3: The flow chart of the STD strategy.

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 $D_i (i = 1, ..., 4)$ should be consistent with the order of sources $\mathbf{y}_i$. 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.

Mixture-trained Dictionary

The second strategy, namely the mixture-trained dictionary (MTD), is illustrated in Figure 4.4. The two mixtures $\mathbf{z}_i (i = 1, 2)$ 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 $(\lfloor (T - L)/(L - F) \rfloor + 1) \times 2$ columns. The dictionary is then computed 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 Equation (4.8) 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

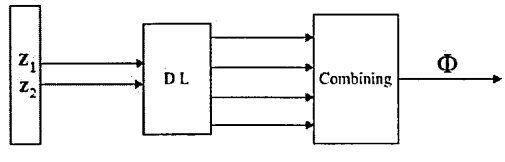


Figure 4.4: The flow chart of the MTD strategy.

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 [60].

4.2.5 Blocking and Reconstruction

According to Equation (4.3), 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 $\mathbf{z}_i$ into $\mathbf{z}_i^p$ where $i = 1, 2$, $p = 1, ..., P$. 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 4.3, 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

Algorithm 8 SDUBSS

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

- Clustering stage: obtain the estimated mixing matrix $\hat{\mathbf{A}}$ from the mixture matrix $\mathbf{Z}$ by Algorithm 5.
- Dictionary learning stage: learn the dictionary $\mathbf{\Phi}$ from the mixture matrix $\mathbf{Z}$ by Algorithm 7.
- Blocking stage: segment mixtures $\mathbf{z}_i$, $i = 1, 2$ to blocks $\mathbf{z}_i^p$, $p = 1, ..., P$.
- Separating stage: separate speech sources $\mathbf{s}_j^p$, $j = 1, ..., 4$, $p = 1, ..., P$ from each mixture block $\mathbf{z}_i^p$, $i = 1, 2$, $p = 1, ..., P$ by Algorithm 6.
- Reconstruction stage: reconstruct the speech source matrix $\mathbf{S}$ including the four whole sources $\mathbf{s}_j$, $j = 1, ..., 4$ by concatenating together all the blocks of estimated source components $\mathbf{s}_j^p$, $j = 1, ..., 4$, $p = 1, ..., P$.

are mostly inaudible.

4.2.6 The Whole System

The whole system of the proposed SDUBSS algorithm can be summarized in Algorithm 8. 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.

4.3 Experimental Results

4.3.1 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 Acoustic-Phonetic Continuous Speech Corpus database TIMIT. 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 seconds, sampled at 10 kHz. That is, each signal has 50000 samples.

In the final subsection, another database ('dev2') from the signal separation evaluation campaign (SiSEC 2008[1]) and the database from Stereo Audio Source Separation Evaluation Campaign (SASSEC07[2]) are used for making comparison between the proposed algorithm and the state-of-the-art methods [28, 69, 70] for this underdetermined blind speech separation task. The original sources are also available for the evaluation with each having a duration of 10 seconds, sampled at 16 kHz. That is, each signal has 160000 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 [153] to evaluate the estimated source signals. These criteria are the signal to distortion ratio (SDR), the signal to interference ratio (SIR) and the signal to artifacts ratio (SAR), defined respectively as

$$SDR = 10log_{10}\frac{\| s_{target} \|^2}{\| e_{interf} + e_{noise} + e_{artif} \|^2} \tag{4.9}$$

$$SIR = 10log_{10}\frac{\| s_{target} \|^2}{\| e_{interf} \|^2} \tag{4.10}$$

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

$$SAR = 10log_{10}\frac{\| s_{target} + e_{interf} + e_{noise} \|^2}{\| e_{artif} \|^2} \tag{4.11}$$

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:

$$\hat{s}(t) = s_{target}(t) + e_{interf}(t) + e_{noise}(t) + e_{artif}(t) \tag{4.12}$$

According to [153], 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.

4.3.2 Separation Results with Fixed Dictionary

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 (which can obtain the best result among the fixed dictionaries). 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 2.1 are used in the experiment. Specifically, we use the following three algorithms in our experiments, i.e. SPGL1 (Spectral Projected Gradient for L1 minimization) [22,23], the solveMP in SparseLab [57] and L1LS solver

for ℓ_1 regularized least squares problem [90, 91], 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 optimized on the basis of extensive tests. In BP, the optimality tolerance parameter was set to 0.01. In MP, the maximum number of iterations[3] 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 seconds, 8169 seconds and 4531 seconds respectively. The separation performance evaluated by SDR, SIR and SAR is shown in Table 4.1.

We also perform a paired Student'y 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 4.1, 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 4.3, and S/F in Table 4.2, which are both greater than 0.05, suggesting that their difference is statistically small. We have also calculated the confidence intervals in each t-test performed in Tables 4.1, 4.2 and 4.3 respectively, and the results are shown in Tables B.1, B.2 and B.3 in the Appendix.

It can be seen from Table 4.1 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

[3]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.

	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

Table 4.1: 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.

following experiments.

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 Figure 4.5, 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 seconds 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 seconds 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 seconds 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 5 times faster than the algorithm without blocking. Figure 4.5 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 Figure 4.5 that, as opposed to a very short block length, longer block lengths do not vary the

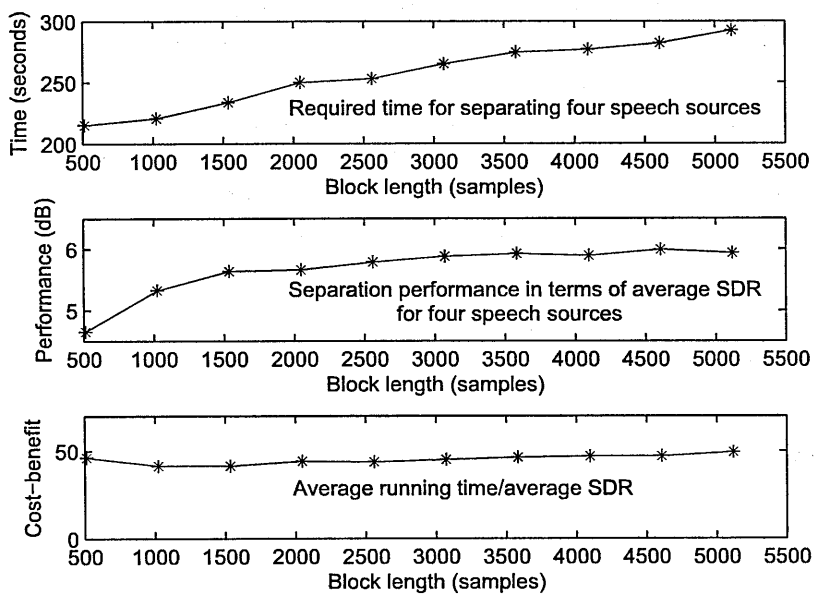


Figure 4.5: The effect of different block length on the computational efficiency and separation performance of the proposed algorithm. The cost-benefit (i.e. computing time divided by the output SDR) is also shown.

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. 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 Figure 4.5 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.

4.3.3 Separation Performance with Adaptive Dictionary

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 presented in Sections[4] 4.3.2 and 4.3.2. Alternatively, we can train the adaptive dictionaries based on the STD and MTD methods. The dictionary learning algorithm applied here is SimCO due to its performance advantages 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 4.2.

	learned dictionary		fixed dictionary			p-value			
	STD	MTD	DCT	STFT	MDCT	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

Table 4.2: 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.

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.

[4]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.

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 optimized 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 4.3. It shows that the separation performance obtained by using SimCO is better than using K-SVD and 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

Table 4.3: 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.

4.3.4 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 4.4 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 indicates that the learned dictionary tends to be more robust than a fixed dictionary for the separation of noisy speech 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

Table 4.4: 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).

4.3.5 Comparison with the State-of-the-art Method

In this section, the proposed algorithm is compared with two related methods, namely, [69,70] developed by Gowreesunker and Tewfik and [28] proposed by Bofill and Zibulevsky. First, we compare our algorithm with [69, 70], which also uses adaptive dictionary for speech separation. The results in [69,70] have been reported in the evaluation campaign SiSEC 2008. 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 (BP) 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 SiSEC 2008 'Under-determined speech and music mixtures development 2' database.

In the beginning of the experiment, we used two instantaneous mixtures which were

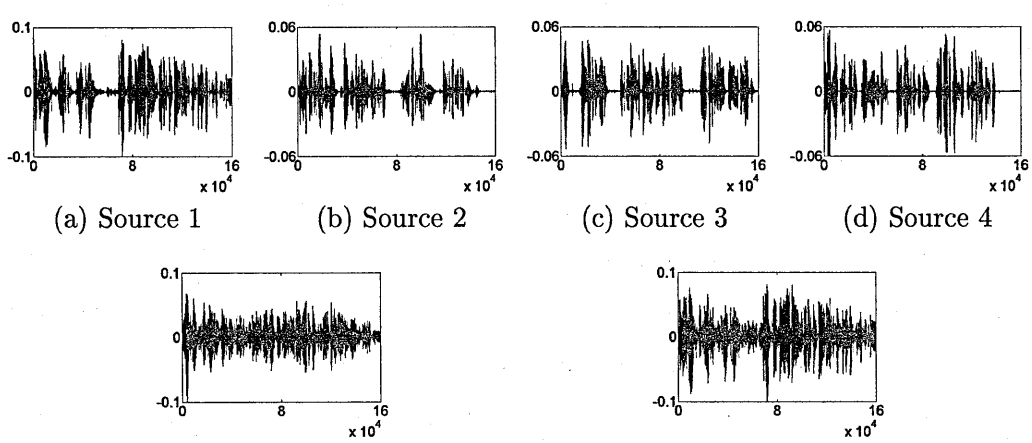
obtained by mixing four male speech sources with the following mixing matrix.

$$\mathbf{A} = \begin{pmatrix} 0.3338 & 0.6495 & 0.8241 & 0.9397 \\ 0.9426 & 0.7604 & 0.5664 & 0.3420 \end{pmatrix} \tag{4.13}$$

The $\hat{\mathbf{A}}$ obtained from the two instantaneous mixtures by the clustering algorithm (i.e. clustering stage of the proposed algorithm) is shown in Equation (4.14). We see that the estimated mixing matrix $\hat{\mathbf{A}}$ is reasonably close to the true mixing matrix $\mathbf{A}$ except the permutation ambiguity.

$$\hat{\mathbf{A}} = \begin{pmatrix} 0.6312 & 0.9368 & 0.8132 & 0.3532 \\ 0.7756 & 0.3499 & 0.5820 & 0.9355 \end{pmatrix} \tag{4.14}$$

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 Figure 4.6, the separation results by the proposed system are shown in Figure 4.7, where the adaptive dictionary was learned by SimCO in the dictionary learning stage. It can be observed that the estimated sources in Figure 4.7 are very similar to the original sources in Figure 4.6.

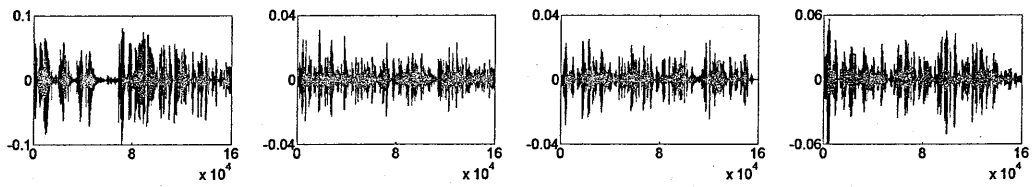


(a) Source 1 (b) Source 2 (c) Source 3 (d) Source 4

(e) Mixture 1 (f) Mixture 2

Figure 4.6: 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 Figure 4.7.

The average performance of these four separated sources measured by SDR, SIR, SAR



(a) Estimation 1 (b) Estimation 2 (c) Estimation 3 (d) Estimation 4

Figure 4.7: The four estimated male speech sources.

is shown in Table 4.5, 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).

	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 4.5: 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.

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 seconds while the compared method needs 1200 seconds to separate the speech sources.

We have also tested these methods on four female speech signals in the SiSEC 2008 evaluation campaign, using the exactly same parameters as those for male speech tests. Table 4.6 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.

	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

Table 4.6: 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.

To compare our method with another benchmark method by [28], we use the dataset in the evaluation campaign SASSEC07, which can be regarded as an earlier version of SiSEC 2008. The reason for this choice is that the results of the algorithm by Bofill and Zibulevsky were reported in SASSEC07, but not in SiSEC 2008. 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'. 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 implemented 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 [28]. The maxima of the potential function are considered to be the estimated directions of the basis vectors [28]. The average performance measured by SDR, SIR, SAR from these four separated sources is shown in Table 4.7, where the results for using predefined dictionary STFT are also included for comparison.

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

The results of another test based on four female speech signals are shown in Table 4.8.

	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

Table 4.8: Average SDR, SIR, SAR (in dB) measured for four estimated female speech sources.

It is observed from Tables 4.5 and 4.6 that, the reference method [69, 70] performs better in terms of SIR and worse in terms of SAR than our proposed method. We

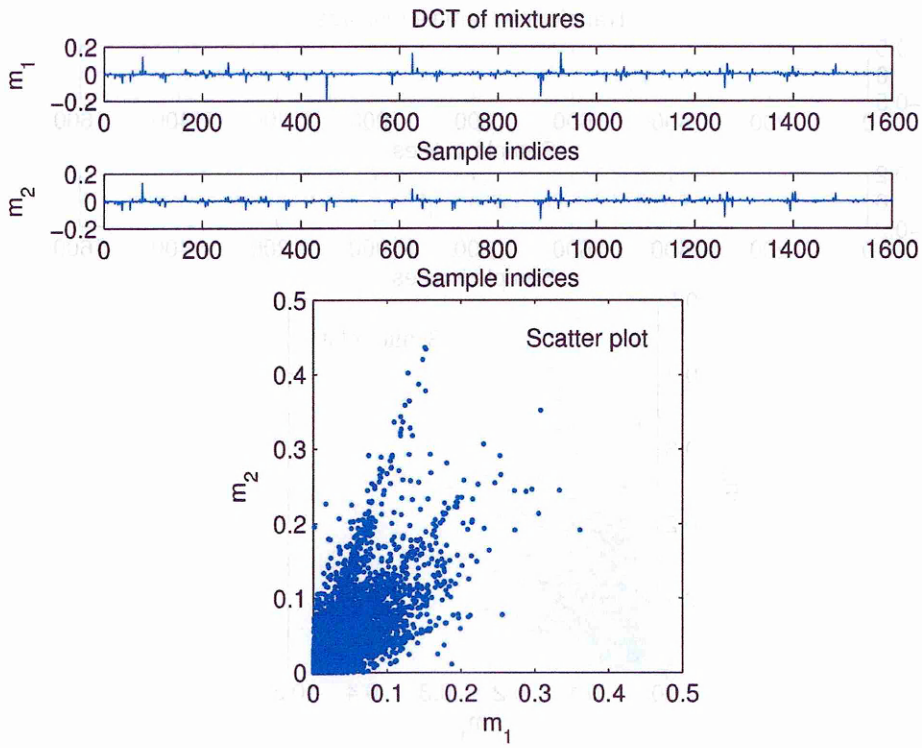


Figure 4.10: DCT coding coefficients (down sampled at rate 100:1) and their scatter plot.

ence. For example, comparing Tables 4.5 and 4.6, 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 Equation (2.9) 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 to Table 4.9) and their coding coefficients are shown in Table 4.10. 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 by Jafari and Plumbley in [83].

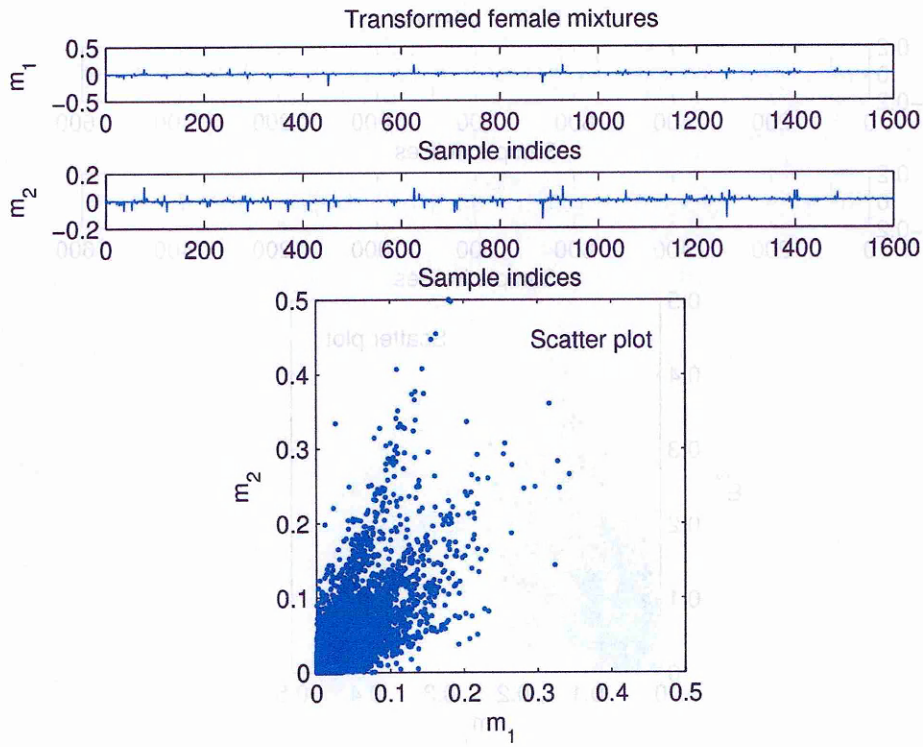


Figure 4.9: Coding coefficients obtained using the dictionary learned by SimCO (down sampled at rate 100:1) and their scatter plot.

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 [12], 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 [18, 75, 101]. 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 differ-

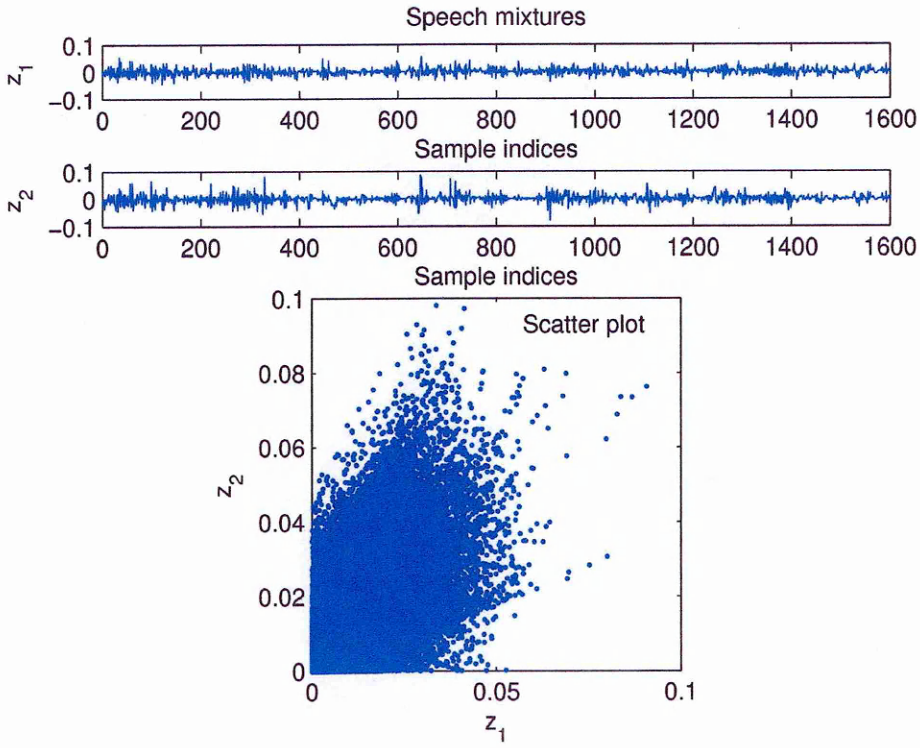


Figure 4.8: Time domain original female speech mixtures (down sampled at rate 100:1) and their scatter plot.

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. $\nu(\mathbf{\Phi})$, can be defined as the maximum absolute inner product between any two different atoms,

$$\nu(\mathbf{\Phi}) = \max_{i \neq j} |\langle \phi_i, \phi_j \rangle|. \tag{4.15}$$

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

Table 4.9: 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.

Using the same speech data as for Tables 4.5 and 4.6, we show in Table 4.9 the mutual

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 [153]. 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 SiSEC 2008 evaluation campaign [152].

The reference method [69, 70] used 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 [153] and [152], using SDR may give better overall performance assessment to the algorithms under comparison, as SDR is a *global* performance index [153].

4.3.6 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 [12, 18, 52, 75, 101]. 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 SiSEC 2008 campaign data as already used in Section 4.3.5. The mixtures both have a length of 160000 samples, and the length of each dictionary atom was set to 512.

As shown earlier (e.g. in Tables 4.2, 4.5 and 4.6), using the dictionaries learned from the 'ground truth' speech sources offers significantly better separation performance than

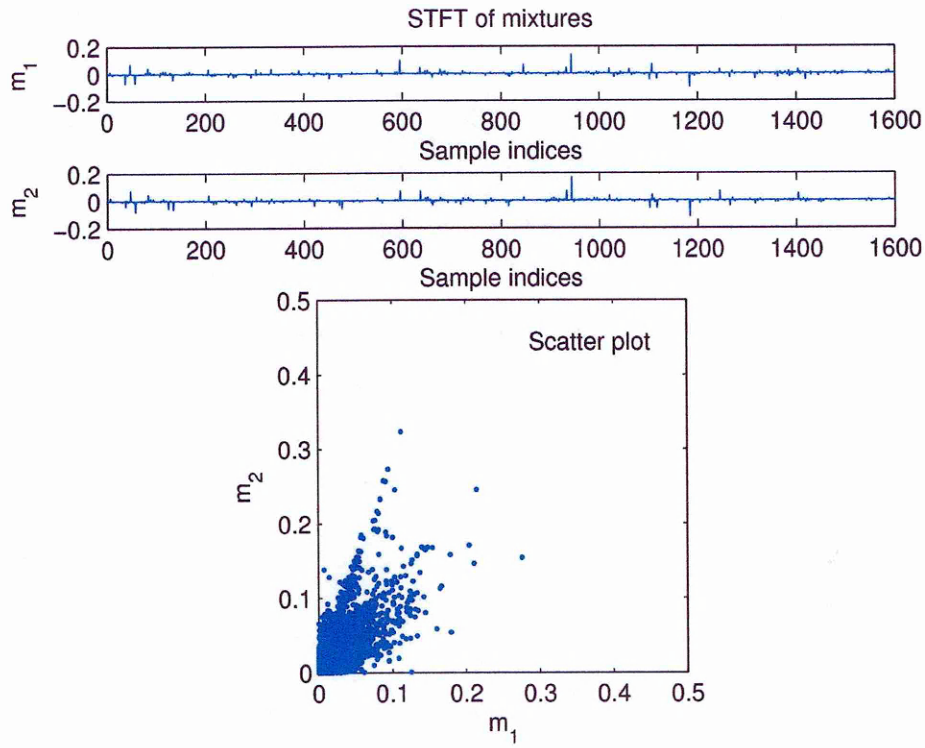


Figure 4.11: STFT coding coefficients (down sampled at rate 100:1) and their scatter plot.

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

	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)

Table 4.10: 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).

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 (Figure 4.8), and their coding coefficients using dictionaries learned by SimCO (Figure 4.9), or predefined transforms such as DCT (Figure 4.10) and STFT (Figure 4.11). 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 [124].

4.4 Summary

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[5], when compared with the baseline 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

[5]Sound demonstrations may be available at http://personal.ee.surrey.ac.uk/Personal/W.Wang/demondata.html

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.

Chapter 5

Joint Blind Source Separation and Adaptive Dictionary Learning

5.1 Introduction

In Chapter 4, the multi-stage method takes advantage of dictionary learning to improve the separation performance and denoising ability. In this chapter, we perform joint dictionary learning and source separation under the same optimization framework as presented in Chapter 3. Therefore, the mixing matrix and dictionary matrix are solved simultaneously in an alternating manner by optimizing a compound cost function. The benefit of the proposed method is in unifying the stages in Chapter 4 to reduce the computational complexity and significantly improve the algorithm performance. This work is in collaboration with Dr. Wei Dai, Mr. Xiaochen Zhao, Mr. Guangyu Zhou from Imperial Collage London and my main contributions include developing the idea, problem model, algorithm implementation and experiment design. I also successfully applied this method to blind image separation application and compared it with other benchmark approaches to show the performance advantage of the proposed method.

Blind image separation is an interesting problem in signal processing applications. To address this problem, several approaches have been proposed in the literature, including, for example, the Bayesian approaches based on Markov random field model

(MRF) [89, 148], and morphological component analysis (MCA) [142] based on sparse representations, as briefly reviewed in Chapter 2.

Inspired by BMMCA [1] which is an extended algorithm of MCA, we propose a new method which not only addresses the limitations of BMMCA but also has some interesting new properties (discussed below). The implementation is based on the SimCO framework. Numerical experiments for blind image separation show the advantages of the proposed method over the ICA, GMCA and BMMCA methods.

The major differences of our proposed algorithm from the existing methods include:

- The BMMCA [1] method uses multiple dictionaries. We assume, however, that there is only one dictionary under which different sources have sparse representations. This can make the algorithm computationally more efficient. The motivation is that when the dictionary redundancy is large enough, using one dictionary to sparsely represent image sources will reach almost the same performance as using multiple dictionaries.

- In our joint optimization framework, we adapt the SimCO optimization method proposed in Chapter 3 to unify the two stages in the separation process: dictionary matrix learning and mixing matrix estimation. Both stages consist of simultaneously updating two variables. The advantage of unifying the two stages is that, in practice, the same algorithmic framework and codes can be used for both stages, thus significantly reducing the computational effort.

- Another important reason to adapt the SimCO framework is to avoid the possible ill-convergence problem existing in the traditional dictionary learning methods, e.g., K-SVD and MOD. It was observed that singular points, rather than the local minima tend to be the major obstacle preventing the algorithm from converging to a global minimizer. By adding the regularization term we are able to force the search path away from singular points to achieve improved performance.

In the remainder of this chapter we will introduce some previous methods related to our proposed algorithm. Then we present the framework, algorithmic details, and the advantages of the proposed method. The comparisons among the proposed algorithm,

and benchmark methods ICA, GMCA and BMMCA are analyzed and demonstrated in the simulation section.

5.2 Related Work

In this section, we briefly introduce the two mainstream image source separation methods, i.e. ICA and MMCA. To better motivate the technique that we will use, we also introduce an image denoising method based on the dictionary learning framework. Such an algorithm is the prototype of a part of our problem formulation. The BMMCA algorithm is included in this section since a similar image denoising model is adapted in our work. All these image source models will be later used as baseline methods for performance comparison against our proposed algorithm.

5.2.1 Independent Component Analysis

As reviewed in Chapter 2, ICA is a benchmark method to find the independent components (latent variables or sources) from signal mixtures by maximizing the statistical independence of the estimated components. Such components capture the essential structure of the data and can be used in source separation and feature extraction. ICA method usually estimates the independent sources by minimizing mutual information or maximizing non-Gaussianity (measured through kurtosis or negentropy).

5.2.2 Image Denoising via Dictionary Learning

Elad and Aharon [60] proposed a local sparsity based method for the image denoising problem. Consider an image which can be sparsely represented. If it is corrupted by additive noise with a known power σ, it is possible to enhance the image by using dictionary learning algorithms. Assume a clean image $\mathbf{s}$ is corrupted by additive Gaussian noise $\mathbf{v}$ and gives the measurement

$$\mathbf{z} = \mathbf{s} + \mathbf{v}. \tag{5.1}$$

The measurement $\mathbf{z}$ can be denoised by solving

$$\min_{\mathbf{s},\mathbf{D},\mathbf{X}} \lambda \|\mathbf{z}-\mathbf{s}\|_F^2 + \|\mathcal{R}\mathbf{s}-\mathbf{D}\mathbf{X}\|_F^2 + \mu \|\mathbf{X}\|_0 . \tag{5.2}$$

Operator $\mathcal{R}$ takes the overlapped patches from the estimated image $\mathbf{s}$ in order to provide enough training samples for dictionary learning. The notation $\|\cdot\|_0$ stands for ℓ_0-pseudo norm which calculates the number of non-zero elements of coding coefficient matrix $\mathbf{X}$. Parameter λ is determined by the noise power σ. Matrix $\mathbf{D}$ is the dictionary containing normalized columns (termed as codewords), which is usually initialized as an over-complete DCT dictionary. Each patch of $\mathbf{s}$ can be obtained by combining very few columns of $\mathbf{D}$.

5.2.3 Multichannel MCA for Blind Source Separation

Assume that an image source $\mathcal{S}_i$ of size $\sqrt{N} \times \sqrt{N}$ can be represented as a vector $\mathbf{s}_i \in \mathbb{R}^{1\times N}$. What is known in our formulation includes the observation $\mathbf{Z} \in \mathbb{R}^{r\times N}$, the number of sources s and the noise power σ. $\mathbf{Z}$ contains the mixtures of multiple sources $\mathbf{S} = \left[\mathbf{s}_1^T, \mathbf{s}_2^T, ...\mathbf{s}_s^T\right]^T$ obtained by a column normalized mixing matrix $\mathbf{A} \in \mathbb{R}^{r\times s}$ in the presence of a zero mean additive Gaussian noise $\mathbf{V}$.

$$\mathbf{Z} = \mathbf{A}\mathbf{S} + \mathbf{V}. \tag{5.3}$$

In MMCA, each source $\mathbf{s}_i$ is assumed to be sparsely represented by different orthogonal dictionaries $\mathbf{D}_i$s which are known a priori or trained from the sources, such that $\mathbf{s}_i = \mathbf{D}_i\mathbf{x}_i$ and hence $\mathbf{x}_i$s are sparse. MMCA [26] aims to find the best estimates $\hat{\mathbf{A}}$ and $\hat{\mathbf{S}}$, given the observation $\mathbf{Z}$ and the dictionaries $\mathbf{D}_i$s, written in the following Lagrangian form:

$$\min_{\mathbf{A},\mathbf{S}} \lambda \|\mathbf{Z}-\mathbf{A}\mathbf{S}\|_F^2 + \sum_{i=1}^{n} \left\|\mathbf{s}_i\mathbf{D}_i^T\right\|_0 . \tag{5.4}$$

Here $\lambda > 0$ is a weighting parameter determined by the noise power σ. MMCA offers a way of separating the sources under the sparse representation framework when the dictionaries for each source are known a priori. In practical applications, however, the true dictionaries are usually difficult to obtain.

Without the known dictionaries in advance, the BMMCA [1] algorithm trains dictionaries from the observed mixture $\mathbf{Z}$. By using the same hierarchical scheme as MMCA, the separation model in BMMCA is split into a few rank-1 approximation problems, where each problem targets one particular source

$$\min_{\mathbf{A}_{:,i},\mathbf{s}_i,\mathbf{D}_i,\mathbf{X}_i} \lambda \left\|\mathbf{E}_i - \mathbf{A}_{:,i}\mathbf{s}_i\right\|_F^2 + \left\|\mathbf{D}_i\mathbf{X}_i - \mathcal{R}\mathbf{s}_i\right\|_2^2 + \mu \left\|\mathbf{X}_i\right\|_0 \tag{5.5}$$

where $\mathbf{D}_i$s are the trained dictionaries for representing sources $\mathbf{s}_i$s, and $\mathbf{E}_i$ is the residual which can be written as

$$\mathbf{E}_i = \mathbf{Z} - \sum_{j \neq i} \mathbf{A}_{:,j}\mathbf{s}_j. \tag{5.6}$$

Both MMCA and BMMCA use multiple dictionaries, one for each source. The difference is that, the dictionaries in BMMCA are adaptively trained by using the K-SVD algorithm. Such trained dictionaries are able to better approximate the source sparsely.

5.3 Proposed Optimization Formulation

In this section, we propose our blind separation scheme. Different from the current benchmark image separation algorithms, in our separation model we train only one dictionary instead of multiple dictionaries. Another difference is that we adapt regularized SimCO framework to unify the two stages of the algorithm, i.e., both mixture learning and dictionary learning stage can be updated by using the same optimization framework. Such an algorithm design significantly reduces the computing efforts. More importantly, it provides a principled way to better estimate the mixing matrix and avoids the singularity problem in dictionary update.

Denote $\mathbf{s}_i \in \mathbb{R}^{1\times N}$ as the vectorized image of $\mathcal{S}_i \in \mathbb{R}^{\sqrt{N}\times\sqrt{N}}$. Denote a binary matrix $\mathbf{P}_k \in \mathbb{R}^{N\times n}$. The product $\mathbf{s}_i \cdot \mathbf{P}_k$ is to vectorize the kth patch of size $\sqrt{n} \times \sqrt{n}$ taken from image $\mathcal{S}_i$. Denote $\mathbf{S} = \left[\mathbf{s}_1^T, \mathbf{s}_2^T, \ldots, \mathbf{s}_s^T\right]^T$. Denote a reversable operation $\mathcal{P}\mathbf{S} = (\mathbf{1}_K \otimes \mathbf{S}) \cdot \mathrm{diag}\left(\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_K\right) \in \mathbb{R}^{s\times Kn}$, where $\mathbf{1}_k \in \mathbb{R}^{1\times k}$ is an all-one vector and K is the number of patches taken from each image. We estimate $\mathcal{P}\mathbf{S}$ by using only one dictionary $\mathbf{D} \in \mathbb{R}^{m\times d}$ and present our proposed optimization formulation for the BSS

problem

$$\min_{\mathbf{A},\mathbf{S},\mathbf{D},\mathbf{X}} \lambda \left\|\mathbf{Z} - \mathbf{AS}\right\|_F^2 + \left\|\mathcal{P}^{-1}\left(\mathbf{DX}\right) - \mathbf{S}^T\right\|_F^2, \tag{5.7}$$

where the coefficients $\mathbf{X}$ is sparse. The matrix $\mathbf{Z}$ denotes the observations generated from equation (5.3). $\mathcal{P}^{-1}$ is the reverse operator of $\mathcal{P}$, which recovers the estimated patches $\mathbf{Dx}_i$ to the estimated vectorized sources. Note that in our model, we found that using one dictionary to sparsely represent all the sources will get almost the same performance as using multiple dictionaries when the dictionary redundancy $\frac{d}{m}$ is large enough. As a result we propose to train only one dictionary for all the sources. One most obvious advantage is that the computational cost is irrelevant to the number of sources, i.e., compared to using multiple dictionaries, the proposed algorithm offers a significant efficiency improvement when the number of sources increases.

To find the solution of the above problem, we propose a joint optimization algorithm to iteratively update the following two stages until a minimizer is found. Note that in each stage there are two items to be updated simultaneously.

- Dictionary learning stage

$$\min_{\mathbf{D},\mathbf{X}} \left\|\mathbf{DX} - \left(\mathcal{P}\mathbf{S}\right)^T\right\|_F^2, \tag{5.8}$$

- Mixture learning stage

$$\min_{\mathbf{A},\mathbf{S}} \lambda \left\|\mathbf{Z} - \mathbf{AS}\right\|_F^2 + \left\|\mathcal{P}^{-1}\left(\mathbf{DX}\right) - \mathbf{S}^T\right\|_F^2. \tag{5.9}$$

During the optimization, we further constrain the updated $\mathbf{A}$ and $\mathbf{D}$ to be column normalized. The explanation is given in subsection 5.4.3. With these constraints, we do not require the pre-knowledge about the scaling matrix in front of the true mixing matrix [27], as otherwise required in MMCA and GMCA algorithms. In addition, as will be shown in our simulations, such an optimization method can result in more accurate estimation of the mixing matrix.

5.4 Algorithmic Details

5.4.1 Dictionary Learning Stage

Our model provides an alternating joint update between $\{\mathbf{D}, \mathbf{X}\}$ and $\{\mathbf{A}, \mathbf{S}\}$. In real data testing, we found that the regularized SimCO gives better performance than the other dictionary learning algorithms (such as K-SVD and MOD) when applied to the image denoising problem. For this reason, SimCO is applied in our dictionary learning stage. The SimCO framework is designed to simultaneously train the dictionary and the sparse coefficients. It works iteratively with sparse coding algorithm which solves the following least squares problem under a fixed dictionary

$$\min_{\mathbf{X}} \|\mathbf{X}\|_0 \text{ s.t. } \left\|\mathbf{DX} - (\mathcal{P}\mathbf{S})^T\right\|_F \leq \epsilon, \tag{5.10}$$

where ϵ is an error bound being proportional to the noise standard deviation. In each iteration, sparse coding algorithm outputs the best sparse coefficients $\mathbf{X}$. Then by fixing the sparse pattern $\Omega = \{(i,j) : \mathbf{X}(i,j) \neq 0\}$ which contains the positions of non-zero elements in $\mathbf{X}$, SimCO simultaneously update $\mathbf{D}$ and $\mathbf{X}$ via first or second order numerical optimization methods (Gradient Descent, Newton Conjugate Gradient, etc. [116]). The convergence analysis of SimCO shows that the failure of finding a global minimizer is probably due to the singularity of the updated dictionary. For this reason, in practice a regularized term $\|\mathbf{X}\|_F^2$ is added in the SimCO formulation to improve its learning performance. Following the convention we denote the feasible set for the dictionary

$$\mathcal{U}_{p,q} = \left\{\mathbf{U} \in \mathbb{R}^{p\times q} : \|\mathbf{U}_{:,i}\|_2 = 1, \forall i \in [q]\right\}. \tag{5.11}$$

where $\|\cdot\|_2$ is the ℓ_2-norm, and $[q] = \{1, 2, \ldots, q\}$ is an integer set. The regularized SimCO, as well as our dictionary update stage, is formulated as

$$\min_{\mathbf{D}\in\mathcal{U}_{m,d}} \min_{\mathbf{X}\in\Omega} \left\|\mathbf{DX} - (\mathcal{P}\mathbf{S})^T\right\|_F^2 + \mu\|\mathbf{X}\|_F^2, \tag{5.12}$$

$$= \min_{\mathbf{D}\in\mathcal{U}_{m,d}} \underbrace{\min_{\mathbf{X}\in\Omega} \left\|\begin{bmatrix}\mathbf{D}\\ \sqrt{\mu}\mathbf{I}\end{bmatrix}\mathbf{X} - \begin{bmatrix}(\mathcal{P}\mathbf{S})^T\\ \mathbf{0}\end{bmatrix}\right\|_F^2}_{f(\mathbf{D})}, \tag{5.13}$$

where $\mu > 0$ is a regularization parameter. Via equation (5.13) we are able to find a succinct expression of the gradient of $f(\mathbf{D})$ thereby a more sufficient algorithm can be obtained.

5.4.2 Mixture Learning Stage

In this stage, we simultaneously update the mixing matrix $\mathbf{A}$ and the sources $\mathbf{S}$. Consider formulation (5.9), it is comprised by a summation of two terms both with form $\|\mathbf{C}_1 - \mathbf{C}_2\mathbf{S}\|_F^2$ as equation (5.12). Therefore the similar method in SimCO can also be used in solving this problem. Referring to 5.13, denote

$$\tilde{\mathbf{Z}} = \begin{bmatrix} \sqrt{\lambda}\mathbf{Z} \\ \mathcal{P}^{\dagger}(\mathbf{DX}) \end{bmatrix}, \tilde{\mathbf{A}} = \begin{bmatrix} \sqrt{\lambda}\mathbf{A} \\ \mathbf{I} \end{bmatrix}, \tag{5.14}$$

The problem formulation of the mixture learning stage (5.10) can therefore be written as

$$\min_{\mathbf{A}\in\mathcal{U}_{r,s}} \underbrace{\min_{\mathbf{S}} \left\|\tilde{\mathbf{Z}} - \tilde{\mathbf{A}}\mathbf{S}\right\|_F^2}_{f(\mathbf{A})}. \tag{5.15}$$

To achieve a simultaneous update of $\mathbf{A}$ and $\mathbf{S}$, we find the general optimal expression of $\mathbf{S}^*$ about $\mathbf{A}$, and then update $\mathbf{A}$ in (5.15) with the substitution of $\mathbf{S}^*$.

$$\mathbf{S}^* = \tilde{\mathbf{A}}^{\dagger}\tilde{\mathbf{Z}}, \forall \mathbf{A} \in \mathcal{U}_{r,s} \tag{5.16}$$

where $\tilde{\mathbf{A}}^{\dagger}$ is the pseudo-inverse of $\tilde{\mathbf{A}}$. Therefore we are able to use the same method as the one in dictionary update stage to compute the gradient of $f(\mathbf{A})$.

$$\begin{aligned} \nabla_{\mathbf{A}} f &= \frac{\partial f}{\partial \mathbf{A}}\Big|_{\mathbf{S}^*} + \frac{\partial f}{\partial \mathbf{S}}\Big|_{\mathbf{S}^*} \cdot \frac{\partial \mathbf{S}}{\partial \mathbf{A}} \\ &= \frac{\partial f}{\partial \mathbf{A}}\Big|_{\mathbf{S}^*} + \mathbf{0} \cdot \frac{\partial \mathbf{S}}{\partial \mathbf{A}} \\ &= -2\left(\tilde{\mathbf{Z}} - \tilde{\mathbf{A}}\mathbf{S}^*\right)\mathbf{S}^{*T}. \end{aligned} \tag{5.17}$$

5.4.3 Advantage of Optimization on Manifolds

For MMCA and GMCA, they assume that the scaling matrix of the mixing matrix $\mathbf{A}$ is known a priori and the dictionaries $\mathbf{D}_i$s are obtained from the original sources. However, for BSS algorithms, these conditions are unknown, therefore a column normalization of the mixing matrix and the dictionaries becomes indispensable constraints. In fact both BMMCA and our proposed method have added them into mixture learning and dictionary learning stage. Here we give a briefly explanation. Consider the noiseless case, suppose $\{\mathbf{A} \in \mathcal{U}_{r,s}, \mathbf{S}, \mathbf{D} \in \mathcal{U}_{m,d}, \mathbf{X}\}$ to be the optimal solution of the BSS problem

$$\min_{\mathbf{A}\in\mathcal{U}_{r,s},\mathbf{S},\mathbf{D}\in\mathcal{U}_{m,d},\mathbf{X}} \lambda \left\|\mathbf{Z} - \mathbf{AS}\right\|_F^2 + \left\|\mathcal{P}^{-1a}(\mathbf{DX}) - \mathbf{S}^T\right\|_F^2. \tag{5.18}$$

Since there is no noise, the minimization of (5.18) should be *zero*. Now if the constraints $\mathbf{A} \in \mathcal{U}_{r,s}$ and $\mathbf{D} \in \mathcal{U}_{m,d}$ are removed, then for any scaling values $\{c_1, c_2\}$, the solution $\{c_1\mathbf{A}, c_1^{-1}\mathbf{S}, c_2\mathbf{D}, c_1^{-1}c_2^{-1}\mathbf{X}\}$ also gives a *zero* solution to (5.18). The same analysis applies to the BMMCA problem. As a result, with the constraints we are actually able to guarantee that there are only finite solutions in the solution set.

Furthermore, for BMMCA the summation of rank-1 approximation problems in mixture learning stage [1] can be finally transformed to

$$\begin{aligned}\sum_i \min_{\mathbf{A}_{:,i}\in\mathcal{U}_{r,1},\mathbf{s}_i} &\left\| \begin{bmatrix} \sqrt{\lambda}\mathbf{E}_i \\ \mathbf{D}_i\mathbf{X}_i \end{bmatrix} - \begin{bmatrix} \sqrt{\lambda}\mathbf{A}_{:,i} \\ \mathcal{R} \end{bmatrix} \mathbf{s}_i \right\|_F^2 \\ &= \sum_i \min_{\mathbf{A}_{:,i}\in\mathcal{U}_{r,1},\mathbf{s}_i} \left\|\tilde{\mathbf{E}}_i - \tilde{\mathbf{A}}_{:,i}\mathbf{s}_i\right\|_F^2 \end{aligned} \tag{5.19}$$

Notice that usually $\min_{\mathbf{A}} \left\|\tilde{\mathbf{E}}_i - \tilde{\mathbf{A}}_{:,i}\mathbf{s}_i\right\|_F^2$ and $\min_{\mathbf{A}\in\mathcal{U}_{r,1}} \left\|\tilde{\mathbf{E}}_i - \tilde{\mathbf{A}}_{:,i}\mathbf{s}_i\right\|_F^2$ do not give a same solution. One fact is, the scheme (used in BMMCA) that splits $\min_{\mathbf{A}} \left\|\tilde{\mathbf{E}}_i - \tilde{\mathbf{A}}_{:,i}\mathbf{s}_i\right\|_F^2$ and column normalization on obtained $\mathbf{A}$ into two separate steps does not give the optimal solution of $\min_{\mathbf{A}\in\mathcal{U}_{r,1}} \left\|\tilde{\mathbf{E}}_i - \tilde{\mathbf{A}}_{:,i}\mathbf{s}_i\right\|_F^2$. This largely affects the accuracy of the mixing matrix estimation and decreases the convergence speed. In contrast, our proposed model numerically updates $\mathbf{A}$ and keeps the update staying on $\mathcal{U}_{r,s}$. After enough iterations, we are able to find an $\mathbf{A} \in \mathcal{U}_{r,s}$ sufficiently close to the solution of $\min_{\mathbf{A}\in\mathcal{U}_{r,s}} \left\|\tilde{\mathbf{Z}} - \tilde{\mathbf{A}}\mathbf{S}\right\|_F^2$.

The mixing matrix error tests in the next section will show the advantage of our method for estimating the mixing matrix $\mathbf{A}$.

5.4.4 Line Search Path

The gradient descent method is used during the update of both dictionary update stage and mixture learning stage. Notice that both dictionaries $\mathbf{D}$ and mixing matrix $\mathbf{A}$ are constrained to have unit column norms. Common selection of an updated direction probably results in an updated $\mathbf{D} \notin \mathcal{U}_{m,d}$ or $\mathbf{A} \notin \mathcal{U}_{r,s}$. As discussed in last subsection, such results tremendously increase the solution set of the optimal dictionaries and the mixing matrix which may cause the failure of the algorithm. Thereafter we refer to [49, 59] and restrict the line search path to the product of Grassmann manifolds. To reach this, we define a projection operator $\bar{(\cdot)}$. Let $\mathbf{u} \in \mathcal{U}_{m,1}$.

$$\bar{\mathbf{h}} = \left(\mathbf{I} - \mathbf{u}\mathbf{u}^T\right)\mathbf{h}, \tag{5.20}$$

so that $\bar{\mathbf{h}}$ and $\mathbf{u}$ are orthogonal. Here $\mathbf{u}$ can be a codeword in dictionary $\mathbf{D}$ or a column of mixing matrix $\mathbf{A}$. Vector $\mathbf{h}$ represents $\nabla_{\mathbf{A}_{:,i}} f$, $i \in [s]$ in the mixture learning stage and $\nabla_{\mathbf{D}_{:,j}} f$, $j \in [d]$ in the dictionary learning stage. For a given non-zero direction $\bar{\mathbf{h}}$ and a step size $t \in \mathbb{R}$, $\mathbf{u}$ is updated as

$$\mathbf{u}(t) = \mathbf{u} \cdot \cos\left(\|\bar{\mathbf{h}}\|_2 t\right) + \frac{\bar{\mathbf{h}}}{\|\bar{\mathbf{h}}\|_2} \cdot \sin\left(\|\bar{\mathbf{h}}\|_2 t\right). \tag{5.21}$$

5.5 Simulations

In the simulations, two source images were mixed together using a 4×2 full rank random column normalized mixing matrix $\mathbf{A}$. Normally, the patch size depends on the size of the sources. We chose 8×8 patches from the source images with size $N = 128 \times 128$. The overlap percentage of the patches was fixed to 50% for our proposed algorithm in both noise and noiseless cases. In order to pursuit a good recovery result, it is better to keep the overlap percentage at a high level. For the dictionary learning stage, we would

Algorithm 9 Proposed Joint BSS and Dictionary Learning Algorithm.

Input: Observations $\mathbf{Z}$, patch size n, number of dictionary codewords d, regularization parameters λ and μ, and total number of iterations l_{max}.

Output: Dictionary $\mathbf{D}$, sparse coefficients $\mathbf{X}$, separated images $\mathbf{S}$, and estimated mixing matrix $\mathbf{A}$.

1. Set $\mathbf{D}$ to over-complete DCT dictionaries.
2. Set a random column-normalized matrix $\mathbf{A}$.
3. Compute $\mathbf{S} = \mathbf{A}^{\dagger}\mathbf{Z}$.
4. **For** $k = 1, 2, \ldots, l_{max}$ **repeat** (6) – (10).
5. $\mathbf{X} \leftarrow \underset{\mathbf{X}}{\arg\min} \left\| \mathbf{D}\mathbf{X} - (\mathcal{R}\mathbf{S})^T \right\|_F^2$.
6. $\mathbf{D}, \mathbf{X} \leftarrow \underset{\mathbf{D} \in \mathcal{U}_{m,d}, \mathbf{X} \in \Omega}{\arg\min} \left\| \mathbf{D}\mathbf{X} - (\mathcal{R}\mathbf{S})^T \right\|_F^2 + \mu \left\| \mathbf{X} \right\|_F^2$.
7. Let $\tilde{\mathbf{Z}} = \begin{bmatrix} \sqrt{\lambda}\mathbf{Z}^T & \mathbf{R}^T \end{bmatrix}^T$, $\tilde{\mathbf{A}} = \begin{bmatrix} \sqrt{\lambda}\mathbf{A}^T & \mathbf{I} \end{bmatrix}^T$.
8. Compute$\mathbf{S} = \tilde{\mathbf{A}}^{\dagger}\tilde{\mathbf{Z}}$.
9. $\mathbf{A} \leftarrow \underset{\mathbf{A} \in \mathcal{U}_{r,s}}{\arg\min} \left\| \tilde{\mathbf{Z}} - \tilde{\mathbf{A}}\mathbf{S} \right\|_F^2$.
10. **end**

like to emphasize again that only one dictionary was generated to sparsely represent all source images. The number of atoms of the dictionary was $d = 256$. We will not discuss the optimal dictionary redundancy factor d/m as it is beyond the scope of this chapter. Refer to [52], the parameter μ of the penalty term was fixed to 0.05. For the mixture learning stage, the constant parameter λ depends on specific noise level of the observations. For the proposed algorithm, the total number of iterations l_{max} was fixed to 50. Each iteration consists of one implementation of dictionary learning and five implementations of mixture learning. Consequently, all above parameters were fixed in the experiments, except for the parameter λ.

Table 5.1: Achieved MSEs of the algorithms in noiseless case.

	FastICA	GMCA	BMMCA	Proposed method
Lena	8.7489	4.3780	3.2631	**3.1346**
Boat	18.9269	**6.3662**	12.5973	6.6555

For the first experiment, we selected two classic images, *Lena* and *Boat* as the source images, which are shown in Fig. 5.2 (a). We compared our proposed algorithm with other benchmark algorithms: FastICA[1] [79], GMCA[2] [27] and BMMCA [1]. For the noiseless case, we calculate the Mean Square Errors (MSEs) to compare the reconstruction performance of the candidate algorithms. The lower the MSE, the better the reconstruction performance. MSE is given in $MSE = (1/N)\,\|\chi - \tilde{\chi}\|_F^2$, where χ is the source image and $\tilde{\chi}$ is the reconstructed image. For the BMMCA algorithm, we set the total number of iterations to be 500 and the overlap percentage was 50%. Table 5.1 illustrates the results of four tested algorithms. GMCA and our proposed algorithm had similar results for boat. However, for Lena, ours is better than GMCA and BMMCA. The results of FastICA is not as good as those three algorithms. For the noise case, we also tested those four algorithms. In this case, we added Gaussian white noise with σ equaling to 10 to the four mixtures, which is shown in Fig. 5.1. The Peak Signal-to-Noise Ratio (PSNR) is used as a measurement of the reconstruction quality. Better quality leads to higher PSNR. It is defined as, $PSNR = 20\log_{10}(\frac{MAX}{\sqrt{MSE}})$, where MAX indicates the maximum possible pixel value of the image. For a uint-8 image, the MAX equals to 255.

[1]Available at: http://research.ics.aalto.fi/ica/fastica/index.shtml
[2]Available at: http://md.cosmostat.org/Generalized_MCA.html

Figure 5.1: Four noisy mixtures with Gaussian noise ($\sigma = 10$).

Figure 5.2: (a) Original Images. Separated images using (b) GMCA, (c) FastICA, (d) BMMCA, and (e) the proposed method.

The separation results are shown in Fig. 5.2 (b)-(e). For the BMMCA algorithm, 200 iterations were set as the stopping criterion and full overlapped patches were selected, which increased the computational complexity. All algorithms successfully separated the noise mixtures. However, FastICA algorithm fails to denoise and GMCA blurred the images. The results of BMMCA are smooth but lost lots of image details. Our proposed algorithm offer significant performance improvement in both separation and denoising, *e.g.* *Lena*'s facial details are the most legible among the four. Moreover, it is mentioned that the overlap percentage of the patches of our proposed algorithm was fixed to 50%. It is sufficient to keep the overlap percentage of patches low. A high overlap percentage will give even better separation results.

Figure 5.3: Separating related human face images by proposed method

Another experiment is performed for separating two human face images 'Lena' and 'Barbara', and the results are shown in Fig. 5.3. We can see that the proposed method also performs well for separating the images that have similar textures.

It is also worth mentioning about the learned dictionary from the mixtures. After applying the SimCO algorithm, the trained dictionary (as shown in Fig. 5.4) looks like the initialization in which an over-complete DCT dictionary was used. Similar dictionaries are also trained for image denoising solved by SimCO. This is quite different from the ones trained via K-SVD algorithm [60]. However dictionaries trained via SimCO can represent images with the same sparsity level as the ones trained via K-SVD and also reach very similar performance. We are doing more detailed analysis on this problem. Note that an over-complete DCT dictionary can already sparsely represent images, therefore the trained dictionary from our proposed algorithm is a reasonable solution.

At last, we compared the performance of all the methods in different noise levels. We

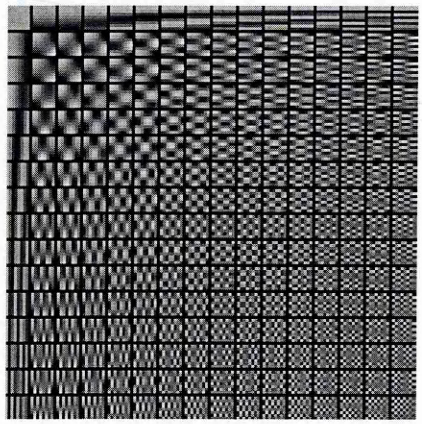


Figure 5.4: Dictionary trained from the proposed algorithm.

use the mixing matrix error as the measurement of the performance. The *mixing matrix error* is defined as $E_{\mathbf{A}} = \left\|\mathbf{A} - \hat{\mathbf{A}}\right\|_F^2$, where $\hat{\mathbf{A}}$ is the approximated column normalized and reformulated mixing matrix. In this experiment, the noise level, which is also the noise standard deviation, varies from 2 to 20. The resulted curves are shown in Fig. 5.5. The performance of GMCA is better than that of FastICA. The curve for BMMCA is not available as the setting for the parameters are sophisticated and varies in different noise levels. It is hard to obtain a good result for BMMCA. Our proposed algorithm outperforms the compared algorithms at all the tested noise levels.

5.6 Summary

We have presented a new method for jointly learning dictionary and separating sources. By unifying the two stages of BSS approach in Chapter 4, the joint optimization framework improves both the algorithm performance and computational efficiency. Only one dictionary instead of multiple dictionaries is learned from the data. Moreover, the proposed method can avoid the possible ill-convergence problem and obtain the improved

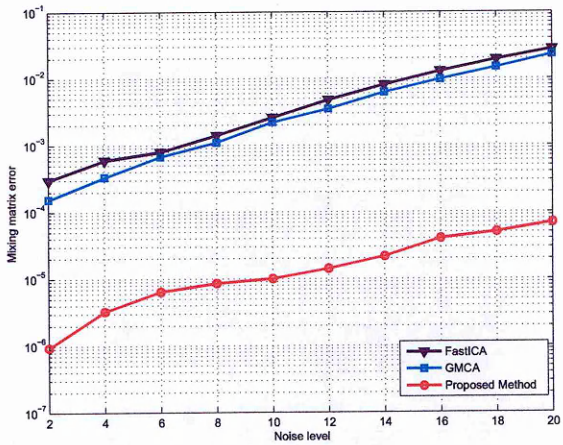


Figure 5.5: The performance of the tested algorithms at different noise levels.

separation performance.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, one of the challenging problems in signal processing i.e. 'dictionary learning' has been studied. We have proposed a new framework where codewords and their corresponding coefficients are allowed to be updated simultaneously. The proposed method differs significantly from the existing algorithms in the literature. With this framework, we have developed two algorithms using the first and second order optimization procedures and proved their performance theoretically. Furthermore, we observed that singular points rather than local minima are the bottleneck for dictionary update. To mitigate the effects of singularity, regularized SimCO has been proposed and numerical experiments verify that regularization substantially improves the performance. We have also provided numerical results to show its advantages over the K-SVD and MOD algorithms, based on the evaluations of the learning performance and the running speed.

Moreover, we extends the SimCO algorithm. First, a tree structure with a multi-level dictionary was obtained by the multi-level K-mean clustering. Then, in the sparse coding stage, the closest centroid from the higher level dictionary to the signal under consideration was found, and their neighbors were then used to code this signal with a dimension reduced OMP, based on the nearest neighbor search. This leads to the tree-OMP (TOMP) method, offering improved computational efficiency. We have

also 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 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.

Finally, we have presented a novel method for blind source separation, where a compound optimization has been employed to perform dictionary learning and source separation simultaneously. The simultaneous recovery for both mixing matrix and dictionary has an advantage over the standard methods in which the dictionary is often pre-trained from data. The remarkable performance improvement of the proposed method compared with other benchmark methods for blind image separation has also been shown by numerical experiments.

6.2 Future Work

We analytically show that benchmark algorithms, including MOD, K-SVD and primitive SimCO, cannot always guarantee to converge to a global minimum, which is caused by the ill-conditioned dictionaries during the optimization process. The key behind the failure is the singularity in the objective function. One can use a weighted technique based on the SimCO optimization framework, where weighting coefficients can be applied to the atomic functions so that singular points are avoided in the optimization process. Using such a method in the dictionary learning stage of the underdetermined BSS algorithm, we may obtain further improvements for source separation from noiseless and noisy mixtures.

Moreover, the joint BSS and dictionary learning algorithm is currently used for overdetermined BSS problem. However, the structure and optimization process make this algorithm potentially useful in more challenging underdetermined case for both blind speech and image separation applications.

We have presented a multi-stage method and a novel simultaneous BSS method for underdetermined blind speech separation in instantaneous case. Numerical experiments have shown the improved separation performance and computational efficiency by the proposed method, as compared with recent underdetermined BSS approaches. The proposed method offers potentials to accommodate new adaptive dictionary learning algorithms for obtaining a better dictionary, and for addressing covolutive source separation problems.

Appendix A

A.1 Proof of Theorem 1

The following notations are repeatedly used in the proofs. Consider the singular value decomposition $\mathbf{A} = \sum_{i=1}^{m} \lambda_i \mathbf{u}_{\mathbf{A},i} \mathbf{v}_{\mathbf{A},i}^T$, where $\lambda_1 > \lambda_2 \geq \cdots \geq \lambda_m \geq 0$ are the singular values, and $\mathbf{u}_{\mathbf{A},i}$ and $\mathbf{v}_{\mathbf{A},i}$ are the left and right singular vectors corresponding to λ_i respectively. It is clear that the objective function $f(\mathbf{u}) = \inf_{\mathbf{w}\in\mathbb{R}^n} \left\|\mathbf{A} - \mathbf{u}\mathbf{w}^T\right\|_F^2$ has two global minimizers $\pm\mathbf{u}_{\mathbf{A},1}$. For a given $\mathbf{u} \in \mathcal{U}_{m,1}$, the angle between $\mathbf{u}$ and the closest global minimizer is defined as

$$\theta = \cos^{-1} \left|\langle \mathbf{u}, \mathbf{u}_{\mathbf{A},1} \rangle\right| .$$

The crux of the proof is that along the gradient descent path, the angle θ is monotonically decreasing. Suppose that the starting angle is less than $\pi/2$. Then the only stationary points are when the angle θ is zero. Hence, the gradient descent search converges to a global minimizer. The probability one part comes from that the starting angle equals to $\pi/2$ with probability zero.

To formalize the idea, it is assumed that the starting point $\mathbf{u}_0 \in \mathcal{U}_{m,1}$ is randomly generated from the uniform distribution on the Stiefel manifold. Define a set $\mathcal{B} \subset \mathcal{U}_{m,1}$ to describe the set of "bad" starting points. It is defined by

$$\mathcal{B} = \left\{\mathbf{u} \in \mathcal{U}_{m,1} : \ \mathbf{u}^T \mathbf{u}_{\mathbf{A},1} = 0\right\},$$

which contains all unit vectors that are orthogonal to $\mathbf{u}_{\mathbf{A},1}$. According to [50], under

the uniform measure on $\mathcal{U}_{m,1}$, the measure of the set $\mathcal{B}$ is zero. As a result, the starting point $\mathbf{u}_0 \notin \mathcal{B}$ with probability one. The reason that we refer to $\mathcal{B}$ as the set of "bad" starting points is explained by the following lemma.

Lemma 4. *Starting from any $\mathbf{u}_0 \in \mathcal{B}$, a gradient descent path stays in the set $\mathcal{B}$.*

Lemma 5. *This lemma can be proved by computing the gradient of f at a $\mathbf{u} \in \mathcal{B}$. Let $\mathbf{w}_\mathbf{u} \in \mathbb{R}^n$ be the optimal solution of the least squares problem in $f(\mathbf{u}) = \inf_{\mathbf{w}\in\mathbb{R}^n} \left\|\mathbf{A} - \mathbf{u}\mathbf{w}^T\right\|_F^2$. It can be verified that $\mathbf{w}_\mathbf{u} = \mathbf{A}^T\mathbf{u}$ and $\nabla f = -2\left(\mathbf{A} - \mathbf{u}\mathbf{w}_\mathbf{u}^T\right)\mathbf{w}_\mathbf{u}$. It is clear that*

$$\begin{aligned}\nabla f &= -2\left(\mathbf{A} - \mathbf{u}\mathbf{w}_\mathbf{u}^T\right)\mathbf{w}_\mathbf{u} = -2\left(\mathbf{A} - \mathbf{u}\mathbf{u}^T\mathbf{A}\right)\mathbf{A}^T\mathbf{u} \\ &= -2\sum_i \lambda_i^2 \mathbf{u}_{\mathbf{A},i}\mathbf{u}_{\mathbf{A},i}^T\mathbf{u} + 2\mathbf{u}\left(\mathbf{u}^T\mathbf{A}\mathbf{A}^T\mathbf{u}\right)\end{aligned}$$

When $\mathbf{u}_0 \in \mathcal{B}$, it holds that $\langle \mathbf{u}_0, \mathbf{u}_{\mathbf{A},1}\rangle = 0$ and $\langle \nabla f(\mathbf{u}_0), \mathbf{u}_{\mathbf{A},1}\rangle = 0$. Since both $\mathbf{u}_0$ and the gradient descent direction are orthogonal to $\mathbf{u}_{\mathbf{A},1}$, the gradient descent path starting from $\mathbf{u}_0 \in \mathcal{B}$ stays in $\mathcal{B}$.

Now consider a starting points $\mathbf{u}_0 \notin \mathcal{B}$. We shall show that the angle θ is monotonically decreasing along the gradient descent path. Towards this end, the notions of directional derivative play an important role. View θ as a function of $\mathbf{u} \in \mathcal{U}_{m,1}$. The directional derivative of θ at $\mathbf{u} \in \mathcal{U}_{m,1}$ along a direction vector $\mathbf{h} \in \mathbb{R}^m$, denoted by $\nabla_\mathbf{h}\theta \in \mathbb{R}$, is defined as

$$\nabla_\mathbf{h}\theta = \lim_{\epsilon\to 0}\frac{\theta(\mathbf{u} + \epsilon\mathbf{h}) - \theta(\mathbf{u})}{\epsilon}.$$

Note the relationship between the directional derivative and the gradient given by $\nabla_\mathbf{h}\theta = \langle \nabla\theta, \mathbf{h}\rangle$. With this definition, the following lemma plays the central role in establishing Theorem 1.

Lemma 6. *Consider a $\mathbf{u} \in \mathcal{U}_{m,1}$ such that $\theta(\mathbf{u}) := \cos^{-1}(|\langle \mathbf{u}, \mathbf{u}_{\mathbf{A},1}\rangle|) \in (0, \pi/2)$. Let $\mathbf{h}_f = -\nabla f(\mathbf{u})$ be the gradient of the objective function f at $\mathbf{u}$. Then it holds $\nabla_{\mathbf{h}_f}\theta < 0$.*

The proof of this lemma is detailed in Appendix A.2.

The implications of this lemma are twofold. First, it implies that $\mathbf{h}_f = -\nabla f \neq \mathbf{0}$ for all $\mathbf{u}$ such that $\theta(\mathbf{u}) \in (0, \pi/2)$. Hence, the only possible stationary points in $\mathcal{U}_{m,1}\backslash\mathcal{B}$ are

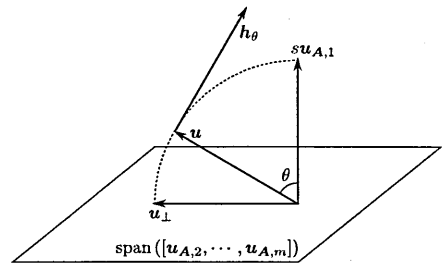


Figure A.1: Illustration of $\mathbf{u}$, $\mathbf{u}_{\mathbf{A},1}$, $\mathbf{h}_\theta$ and $\mathbf{u}_\perp$.

$\mathbf{u}_{\mathbf{A},1}$ and $-\mathbf{u}_{\mathbf{A},1}$. Second, starting from $\mathbf{u}_0 \in \mathcal{B}$, the angle θ decreases along the gradient descent path. As a result, a gradient descent path will not enter $\mathcal{B}$. It will converge to $\mathbf{u}_{\mathbf{A},1}$ or $-\mathbf{u}_{\mathbf{A},1}$. Theorem 1 is therefore proved.

A.2 Proof of Lemma 6

This appendix is devoted to prove Lemma 6, i.e., $\nabla_{\mathbf{h}_f}\theta < 0$. Note that $\nabla_{\mathbf{h}_f}\theta = \langle \mathbf{h}_f, \nabla\theta\rangle = \langle -\nabla f, \nabla\theta\rangle = \nabla_{-\nabla\theta} f$. It suffices to show that $\nabla_{-\nabla\theta} f < 0$.

Towards this end, the following definitions are useful. Define $s = \text{sign}\left(\mathbf{u}^T\mathbf{u}_{\mathbf{A},1}\right)$. Then the vector $s\mathbf{u}_{\mathbf{A},1}$ is one of the two global minimizers that is the closest to $\mathbf{u}$. It can be also verified that $\theta = \cos^{-1}\langle \mathbf{u}, s\mathbf{u}_{\mathbf{A},1}\rangle$. Furthermore, suppose that $\theta \in (0, \pi/2)$. Define

$$\mathbf{h}_\theta = \frac{s\mathbf{u}_{\mathbf{A},1} - \mathbf{u}\cos\theta}{\sin\theta}, \text{ and } \mathbf{u}_\perp = \frac{\mathbf{u} - s\mathbf{u}_{\mathbf{A},1}\cos\theta}{\sin\theta}.$$

Clearly, vectors $\mathbf{h}_\theta$ and $\mathbf{u}_\perp$ are well-defined when $\theta \in (0, \pi/2)$. The relationship among $\mathbf{u}$, $\mathbf{u}_{\mathbf{A},1}$, $\mathbf{h}_\theta$ and $\mathbf{u}_\perp$ is illustrated in Figure A.1. Intuitively, the vector $\mathbf{h}_\theta$ is the tangent vector that pushes $\mathbf{u}$ towards the global minimizer $s\mathbf{u}_{\mathbf{A},1}$.

In the following, we show that $\nabla_{-\nabla\theta} f = \nabla_{\mathbf{h}_\theta} f$ if we restrict $\mathbf{u} \in \mathcal{U}_{m,1}$. By the definition of the directional derivative, one has[1]

$$\nabla_{-\nabla\theta}\mathbf{u} = \lim_{\epsilon\to 0}\frac{\mathbf{u} - \epsilon\nabla\theta}{\|\mathbf{u} - \epsilon\nabla\theta\|}.$$

[1]The denominator comes from the restriction that $\mathbf{u} \in \mathcal{U}_{m,1}$.

Note that

$$\begin{aligned}\nabla\theta &= \nabla\left(\cos^{-1}(\cos\theta)\right)\\ &= -\frac{1}{\sqrt{1-\cos^2\theta}}\nabla\left\langle \mathbf{u}, s\mathbf{u}_{\mathbf{A},1}\right\rangle = -\frac{1}{\sin\theta}\left(s\mathbf{u}_{\mathbf{A},1}\right).\end{aligned}$$

Since $s\mathbf{u}_{\mathbf{A},1} = \mathbf{u}\cos\theta + \mathbf{h}_\theta \sin\theta$, one has

$$\begin{aligned}\mathbf{u} - \epsilon\nabla\theta &= \mathbf{u} + \frac{\epsilon}{\sin\theta}\left(s\mathbf{u}_{\mathbf{A},1}\right)\\ &= \mathbf{u}\left(1 + \epsilon\cos\theta/\sin\theta\right) + \epsilon\mathbf{h}_\theta.\end{aligned}$$

Substitute it back to $\nabla_{-\nabla\theta}\mathbf{u}$. One has $\nabla_{-\nabla\theta}\mathbf{u} = \mathbf{h}_\theta$. In other words, if $\mathbf{u} \in \mathcal{U}_{m,1}$, then $\nabla_{-\nabla\theta} f = \nabla_{\mathbf{h}_\theta} f$.

To compute $\nabla_{\mathbf{h}_\theta} f$, note that $f(\mathbf{u}) = \left\|\mathbf{A} - \mathbf{u}\mathbf{w}_{\mathbf{u}}^T\right\|_F^2 = \|\mathbf{A}\|_F^2 - \left\|\mathbf{u}^T\mathbf{A}\right\|_2^2$. Now define

$$g(\mathbf{u}) = \left\|\mathbf{u}^T\mathbf{A}\right\|_2^2.$$

Then clearly $\nabla_{\mathbf{h}_\theta} f = -\nabla_{\mathbf{h}_\theta} g$. To proceed, we also decompose $\mathbf{A}$ as follows. Recall the SVD of $\mathbf{A}$ given by $\mathbf{A} = \sum_{i=1}^m \lambda_i \mathbf{u}_{\mathbf{A},i}\mathbf{v}_{\mathbf{A},i}^T$. Let $\mathbf{U}_{\mathbf{A},\perp} \in \mathcal{U}_{m,m-1}$ contain the left singular vectors corresponding to $\lambda_2, \cdots, \lambda_m$, i.e., $\mathbf{U}_{\mathbf{A},\perp} = [\mathbf{u}_{\mathbf{A},2}, \cdots, \mathbf{u}_{\mathbf{A},m}]$. Similarly define $\mathbf{V}_{\mathbf{A},\perp}$. Then,

$$\begin{aligned}\mathbf{A} &= [\mathbf{u}_{\mathbf{A},1}, \mathbf{U}_{\mathbf{A},\perp}]\,\mathrm{diag}\left([\lambda_1, \cdots, \lambda_m]\right)\begin{bmatrix}\mathbf{v}_{\mathbf{A},1}^T\\ \mathbf{V}_{\mathbf{A},\perp}^T\end{bmatrix}\\ &= [\mathbf{u}_{\mathbf{A},1}, \mathbf{U}_{\mathbf{A},\perp}]\,[\mathbf{w}_{\mathbf{A},1}, \mathbf{W}_{\mathbf{A},\perp}]^T,\end{aligned}$$

where $\mathbf{w}_{\mathbf{A},i} = \lambda_i \mathbf{v}_{\mathbf{A},i}$ for $i = 1, \cdots, m$, and $\mathbf{W}_{\mathbf{A},\perp} = [\mathbf{w}_{\mathbf{A},2}, \cdots, \mathbf{w}_{\mathbf{A},m}]$. It is straightforward to verify that $\mathbf{w}_{\mathbf{A}}^T\mathbf{W}_{\mathbf{A},\perp} = \mathbf{0}$.

The function $g(\mathbf{u})$ can be decomposed into two parts. Note that

$$\begin{aligned} g(\mathbf{u}) &= \left\| \mathbf{u}^T \left[\mathbf{u}_{\mathbf{A},1}, \mathbf{U}_{\mathbf{A},\perp}\right] \left[\mathbf{w}_{\mathbf{A},1}, \mathbf{W}_{\mathbf{A},\perp}\right]^T \right\|_2^2 \\ &= \left\| \mathbf{u}^T \mathbf{u}_{\mathbf{A},1} \mathbf{w}_{\mathbf{A},1}^T \right\|_2^2 + \left\| \mathbf{u}^T \mathbf{U}_{\mathbf{A},\perp} \mathbf{W}_{\mathbf{A},\perp}^T \right\|_2^2 \\ &\quad + 2 \left\langle \mathbf{u}^T \mathbf{u}_{\mathbf{A},1} \mathbf{w}_{\mathbf{A},1}^T, \mathbf{u}^T \mathbf{U}_{\mathbf{A},\perp} \mathbf{W}_{\mathbf{A},\perp}^T \right\rangle \\ &= \left\| \mathbf{u}^T \mathbf{u}_{\mathbf{A},1} \mathbf{w}_{\mathbf{A},1}^T \right\|_2^2 + \left\| \mathbf{u}^T \mathbf{U}_{\mathbf{A},\perp} \mathbf{W}_{\mathbf{A},\perp}^T \right\|_2^2, \end{aligned}$$

where the last equality follows from that $\mathbf{W}_{\mathbf{A},\perp}^T \mathbf{w}_{\mathbf{A}} = \mathbf{0}$ and hence

$$\left\langle \mathbf{u}^T \mathbf{u}_{\mathbf{A},1} \mathbf{w}_{\mathbf{A},1}^T, \mathbf{u}^T \mathbf{U}_{\mathbf{A},\perp} \mathbf{W}_{\mathbf{A},\perp}^T \right\rangle = 0.$$

To further simplify $g(\mathbf{u})$, note that $\cos\theta = \left|\mathbf{u}^T \mathbf{u}_{\mathbf{A}}\right|$. Furthermore, it is straightforward to verify that the projection of $\mathbf{u}$ on $\operatorname{span}(\mathbf{U}_{\mathbf{A},\perp})$ is given by $\mathbf{U}_{\mathbf{A},\perp}\mathbf{U}_{\mathbf{A},\perp}^T \mathbf{u} = \mathbf{u}_\perp \sin\theta$. Define $\mathbf{u}_R = \mathbf{U}_{\mathbf{A},\perp}^T \mathbf{u}_\perp \in \mathbb{R}^{m-1}$. Then, $\|\mathbf{u}_R\| = 1$ and

$$\begin{aligned} &\left\| \mathbf{u}^T \mathbf{U}_{\mathbf{A},\perp} \mathbf{W}_{\mathbf{A},\perp}^T \right\|_2^2 \\ &= \sin^2\theta \left\| \mathbf{u}_\perp^T \mathbf{U}_{\mathbf{A},\perp} \mathbf{W}_{\mathbf{A},\perp}^T \right\|_2^2 \\ &= \sin^2\theta \mathbf{u}_R^T \operatorname{diag}\left(\left[\lambda_2^2, \cdots, \lambda_m^2\right]\right) \mathbf{u}_R^T. \end{aligned}$$

Hence,

$$g(\mathbf{u}) = \cos^2\theta \cdot \lambda_1 + \sin^2\theta \mathbf{u}_R^T \operatorname{diag}\left(\left[\lambda_2^2, \cdots, \lambda_m^2\right]\right) \mathbf{u}_R.$$

We are now ready to decide the sign of $\nabla_{\mathbf{h}_\theta} g$. It is straightforward to verify that

$$\nabla_{\mathbf{h}_\theta} \cos\theta = \lim_{\epsilon \to 0} \left\langle \frac{\mathbf{u} + \epsilon \mathbf{h}_\theta}{\sqrt{1+\epsilon^2}}, s\mathbf{u}_{\mathbf{A},1} \right\rangle = \sin\theta,$$

and similarly $\nabla_{\mathbf{h}_\theta} \sin\theta = -\cos\theta$. Therefore,

$$\begin{aligned} \nabla_{\mathbf{h}_\theta} \mathbf{u}_\perp &= \nabla_{\mathbf{h}_\theta} \left(\frac{\mathbf{u} - -\cos\theta s\mathbf{u}_{\mathbf{A},1}}{\sin\theta} \right) \\ &= \frac{\mathbf{h}_\theta \sin\theta + \mathbf{u}\cos\theta - s\mathbf{u}_{\mathbf{A},1}}{\sin^2\theta} \\ &= \frac{s\mathbf{u}_{\mathbf{A},1} - s\mathbf{u}_{\mathbf{A},1}}{\sin^2\theta} = \mathbf{0}, \end{aligned}$$

and $\nabla_{\mathbf{h}_\theta}\mathbf{u}_R = \nabla_{\mathbf{h}_\theta}\left(\mathbf{U}_{\mathbf{A},\perp}^T\mathbf{u}_\perp\right) = \mathbf{0}$. Hence, one has

$$\nabla_{\mathbf{h}_\theta} g = \sin 2\theta \left(\lambda_1 - \mathbf{u}_R^T \mathrm{diag}\left(\left[\lambda_2^2, \cdots, \lambda_m^2\right]\right)\mathbf{u}_R\right).$$

Note that

$$\begin{aligned}&\mathbf{u}_R^T \mathrm{diag}\left(\left[\lambda_2^2, \cdots, \lambda_m^2\right]\right)\mathbf{u}_R \\ &\leq \mathbf{u}_R^T \mathrm{diag}\left(\left[\lambda_2^2, \cdots, \lambda_2^2\right]\right)\mathbf{u}_R = \lambda_2 < \lambda_1.\end{aligned}$$

It can be concluded that when $\theta \in (0, \pi/2)$, $\nabla_{\mathbf{h}_\theta} g > 0$ and $\nabla_{\mathbf{h}_\theta} f = -\nabla_{\mathbf{h}_\theta} g < 0$. Lemma 6 is therefore proved.

Appendix B

Tables B.1, B.2 and B.3 below show the confidence intervals that correspond to the p-values in Tables 4.1, 4.2 and 4.3 respectively. The p-values and the confidence intervals were obtained from the t-test on the results of the compared methods.

	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 B.1: The confidence intervals corresponding to the p-values in Table 4.1 obtained from the t-test between the methods, where B = BP, M = MP, and L = L1LS.

	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 B.2: The confidence intervals corresponding to the p-values in Table 4.2 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/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)

Table B.3: The confidence intervals corresponding to the p-values in Table 4.3 obtained from the t-tests between the methods of SimCO, K-SVD and GAD, where S = SimCO, K = K-SVD, and G = GAD.

References

[1] V. Abolghasemi, S. Ferdowsi, and S. Sanei. Blind separation of image sources via adaptive dictionary learning. *IEEE Transactions on Image Processing*, 21(6):2921–2930, 2012.

[2] P. A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

[3] A. Adler, V. Emiya, M. G. Jafari, M. Elad, R. Gribonval, and M. D. Plumbley. Audio inpainting. *IEEE Transactions on Audio, Speech and Language Processing*, 20(3):922–932, 2011.

[4] M. Aharon and M. Elad. Sparse and redundant modeling of image content using an image-signature-dictionary. *SIAM Journal on Imaging Sciences*, 1(3):228–247, 2008.

[5] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions Signal Processing*, 54(11):4311–4322, 2006.

[6] A. Alinaghi, W. Wang, and P. Jackson. Integrating binaural cues and blind source separation method for separating reverberant speech mixtures. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 209–212, 2011.

[7] S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. *In Advances in Neural Information Processing Systems*, 8:757–763, 1996.

[8] S. Araki, S. Makino, H. Sawada, and R. Mukai. Underdetermined blind separation of convolutive mixtures of speech with directivity pattern based mask and ica. In *In Proceedings of 5th International Conference Independent Component Anal. and Blind Signal Separation*, pages 898–905, 2004.

[9] S. Araki, R. Mukai, S. Makino, and H. Saruwatari. The fundamental limitation of frequency domain blind source separation for convolutive mixture of speech. *IEEE Transactions on Speech Audio Processing*, 11:109–116, 2003.

[10] S. Araki, H. Sawada, and S. Makino. K-means based underdetermined blind speech separation. In *Blind Speech Separation*, pages 243–270. Springer Netherlands, 2007.

[11] S. Arberet, R. Gribonval, and F. Bimbot. A robust method to count and locate audio sources in a multichannel underdetermined mixture. *IEEE Transactions on Signal Processing*, 58(1):121–133, 2010.

[12] J. A.Tropp. Greed is good: algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, 2004.

[13] A. D. Back and A. S. Weigend. A first application of independent component analysis to extracting structure from stock returns. *International Journal of Neural Systems*, 8(4):473–484, 1997.

[14] S. Baillet, J. C. Mosher, and R. M. Leahy. Electromagnetic brain mapping. *IEEE Signal Processing Magazine*, 18(6):14–30, 2001.

[15] R. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24(4):118–121, 2007.

[16] R. G. Baraniuk, E. J. Candès, M. Elad, and Y. Ma. Applications of sparse representation and compressive sensing. *Proceedings of the IEEE*, 98(6):906–909, 2010.

[17] R.G. Baraniuk, E. J. Candès, R. Nowak, and M. Vetterli. Compressed sensing. *IEEE Signal Processing Magazine*, 25(2):12–13, 2008.

[18] D. Barchiesi and M. D. Plumbley. Learning incoherent dictionaries for sparse approximation using iterative projections and rotations. In *Proceedings of ICML Workshop on Sparsity, Dictionaries and Projections in Machine Learning and Signal Processing*, 2012.

[19] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal Imaging Sciences*, 2(1):183–202, 2009.

[20] J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computatation*, 7:1129–1159, 1995.

[21] A. Belouchrani and J. F. Cardoso. Maximum likelihood source separation for discrete sources. In *Proceedings of European Signal Processing Conference*, pages 768–771, 1994.

[22] E. van den Berg and M. P. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912, 2008.

[23] E. van den Berg and M. P. Friedlander. Spgl1: Spectral projected gradient for l1 minimization. http://www.cs.ubc.ca/labs/scl/spgl1/, 2008.

[24] T. Blumensath and M. E. Davies. Sparse and shift-invariant representations of music. *IEEE Transactions Speech Audio Processing*, 14(1):50–57, 2006.

[25] T. Blumensath and M. E. Davies. Gradient pursuits. *IEEE Transactions on Signal Processing*, 56(6):2370–2382, 2008.

[26] J. Bobin, Y. Moudden, J. Starck, and M. Elad. Morphological diversity and source separation. *IEEE Signal Processing Letters*, 13(7):409–412, 2006.

[27] J. Bobin, J. Starck, J. Fadili, and Y. Moudden. Sparsity and morphological diversity in blind source separation. *IEEE transactions on Image Processing*, 16(11):2662–2674, 2007.

[28] P. Bofill and M. Zibulevsky. Underdetermined blind source separation using sparse representations. *Signal Processing*, 81(11):2353–2362, Nov. 2001.

[29] A. S. Bregman. *Auditory scene analysis.* MIT Press, September 1994.

[30] R. Cabeza and L. Nyberg. Imaging cognition ii: An empirical review of 275 pet and fmri studies. *Journal of Cognitive Neuroscience*, 22(1):1–47, 2000.

[31] C. Cadieu and B. A. Olshausen. Learning transformational invariants from time-varying natural images. In *Proceedings of Conference Neural Information Processing Systems*, 2008.

[32] E. J. Candès. Compressive sampling. *International Congress of Mathematics*, 3:1433–1452, 2006.

[33] E. J. Candès and D. L. Donoho. Curvelets - a surprisingly effective nonadaptive representation for objects with edges. *Curves and Surfaces*, 1999.

[34] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions Information Theory*, 52(2):489–509, 2006.

[35] E. J. Candès and T. Tao. Decoding by linear programming. *IEEE Transactions Information Theory*, 51(12):4203–4215, 2005.

[36] E. J. Candès and T. Tao. The dantzig selector: Statistical estimation when p is much larger than n. *Annals of Statistics*, 35:2313–2351, 2007.

[37] E. J. Candès and M. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21–30, 2008.

[38] A. Llagostera Casanovas, G. Monaci, P. Vandergheynst, and R. Gribonval. Blind audiovisual source separation based on sparse redundant representations. *IEEE Transactions Multimedia*, 12(5):358–371, 2010.

[39] V. Cevher and A. Krause. Greedy dictionary selection for sparse representation. In *Proceedings of NIPS Workshop on Discrete Optimization in Machine Learning*, Vancouver, Canada, December 2009.

[40] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

[41] C. E. Cherry. Some experiments on the recognition of speech, with one and with two ears. *The Journal of the Acoustical Society of America*, 25(5):975–979, 1953.

[42] A. Cichocki and S. Amari. *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications.* John Wiley, April 2003.

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

[44] P. Comon. Independent component analysis: A new concept. *Signal Processing*, 36:287–314, 1994.

[45] P. Comon. Blind channel identification and extraction of more sources than sensors. In *Advanced Signal Processing Algorithms, Architectures, and Implementations VIII*, volume 3461, pages 2–13, 1998.

[46] P. Comon. Blind identification and source separation in 2×3 under-determined mixtures. *IEEE Transactions on Signal Processing*, 52(1):11–22, 2004.

[47] M. Congedo, C. Gouypailler, and C. Jutten. On the blind source separation of human electroencephalogram by approximate joint diagonalization of second order statistics. *Clinical Neurophysiology*, 119(12):2677–2686, 2008.

[48] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker. *Digital Watermarking and Steganography.* Morgan Kaufmann Publishers, 2007.

[49] W. Dai, E. Kerman, and O. Milenkovic. A geometric approach to low-rank matrix completion. *IEEE Transactions on Information Theory*, 58:237–247, 2011.

[50] W. Dai, Y. Liu, and B. Rider. Quantization bounds on Grassmann manifolds and applications to MIMO systems. *IEEE Transactions on Information Theory*, 54(3):1108–1123, 2008.

[51] W. Dai and O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Transactions on Information Theory*, 55:2230–2249, 2009.

[52] W. Dai, T. Xu, and W. Wang. Simultaneous codeword optimization (simco) for dictionary update and learning. *IEEE Transactions on Signal Processing*, 60(12):6340–6353, 2012.

[53] I. Daubechies, R. DeVore, M. Fornasier, and S. Gunturk. Iteratively re-weighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 2009.

[54] J. W. Demmel and M. T. Heath. Applied numerical linear algebra. In *Society for Industrial and Applied Mathematics*. SIAM, 1997.

[55] M. N. Do and M. Vetterli. The contourlet transform: An efficient directional multiresolution image representation. *IEEE Transactions Image Processing*, 14(12):2091–2106, 2005.

[56] D. L. Donoho. Compressed sensing. *IEEE Transactions Information Theory*, 52(4):1289–1306, 2006.

[57] D. L. Donoho, Iddo Drori, V. Stodden, and Y. Tsaig. Sparselab. http://sparselab.stanford.edu/, 2005.

[58] D. L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions Information Theory*, 47:2845–2862, 2001.

[59] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1999.

[60] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[61] K. Engan, S. Aase, and J. H. Husèy. Method of optimal directions for frame design. In *IEEE International Conference Acoustics, Speech, and Signal Processing*, volume 5, pages 2443–2446, 1999.

[62] K. Engan, K. Skretting, and J. H. Husèy. Family of iterative LS-based dictionary learning algorithms, ILS-DLA, for sparse signal representation. *Digital Signal Processing*, 17(1):32–49, 2007.

[63] D. Fitzgerald and M. Cranitch. Sound source separation using shifted non-negative tensor factorisation. In *IEEE International Conference on Audio and Speech Signal Processing*, 2006.

[64] P. Foldiak. Forming sparse representations by local anti-Hebbian learning. *Biological Cybernetics*, 64:165–170, 1990.

[65] M. Gaeta and J. L. Lacoume. Source separaion without prior knowledge: the maximum likelihood solution. In *Proceedings of European Signal Processing Conference*, pages 621–624, 1990.

[66] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[67] Q. Geng, H. Wang, and J. Wright. On the local correctness of L-1 minimization for dictionary learning. *IEEE Transactions on Information Theory*, 2011.

[68] I. Gorodnitsky and B. Rao. Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm. *IEEE Transactions on Signal Processing*, 45(3):600–616, 1997.

[69] B. V. Gowreesunker and A. H. Tewfik. Blind source separation using monochannel overcomplete dictionaries. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 33 – 36, 2008.

[70] B. V. Gowreesunker and A. H. Tewfik. A novel subspace clustering method for dictionary design. In *Proceedings of International Conference on Independent Component Analysis and Signal Separation*, pages 34 – 41, 2009.

[71] R. Gribonval. Sparse decomposition of stereo signals with matching pursuit and application to blind separation of more than two sources from a stereo mixture. In *IEEE International Conference Acoustic, Speech Signal Processing*, volume 3, pages 3057–3060, 2002.

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

[73] R. Gribonval and M. Nielsen. Sparse representations in unions of bases. *IEEE Transactions on Information Theory*, 49(12):3320–3325, 2003.

[74] R. Gribonval and K. Schnass. Dictionary identification - sparse matrix factorisation via l1 minimisation. *IEEE Transactions on Information Theory*, 56(7):3523–3539, 2010.

[75] R. Gribonval and P. Vandergheynst. On the exponential convergence of matching pursuit in quasi-incoherent dictionaries. *IEEE Transactions on Information Theory*, 52(1):255–261, 2006.

[76] J. Herault and C. Jutten. Space or time adaptive signal processing by neural models. In *Proceedings AIP Conference: Neural Networks for Computing*, pages 206–211, 1986.

[77] K. Huang and S. Aviyente. Sparse representation for signal classification. In *Conference Neural Information Processing Systems*, 2007.

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

[79] A. Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, May 1999.

[80] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley-Interscience, May 2001.

[81] A. Hyvarinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9:1483–1492, 1997.

[82] M. G. Jafari and M. D. Plumbley. Speech denoising based on a greedy adaptive dictionary algorithm. In *European Signal Processing Conference*, pages 1423–1426, 2009.

[83] M. G. Jafari and M. D. Plumbley. Fast dictionary learning for sparse representations of speech signals. *The Journal of Selected Topics in Signal Processing*, 5(5):1025–1031, 2011.

[84] T. Jan, W. Wang, and D.L. Wang. A multistage approach to blind separation of convolutive speech mixtures. *Speech Communication*, 53:524–539, 2011.

[85] S. Ji, Y. Xue, and L. Carin. Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 56(6):2346–2356, 2008.

[86] P. Jost, P. Vandergheynst, S. Lesage, and R. Gribonval. BMoTIF: An efficient algorithm for learning translation invariant dictionaries. In *IEEE International Conference Acoustic, Speech, Signal Processing*, volume 5, pages 857–860, 2006.

[87] A. Jourjine, S. Rickard, and O. Yilmaz. Blind separation of disjoint orthogonal signals: Demixing n sources from 2 mixtures. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2985–2988, 2000.

[88] C. Jutten and J. Herault. Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24:1–10, 1991.

[89] K. Kayabol, E. E. Kuruoglu, and B. Sankur. Bayesian separation of images modeled with MRFs using MCMC. *IEEE Transactions on Image Processing*, 18(5):982–994, 2009.

[90] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, 2007.

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

[92] E. Kokiopoulou and P. Frossard. Semantic coding by supervised dimensionality reduction. *IEEE Transactions Multimedia*, 10(5):806–818, 2008.

[93] M. Kowalski, E. Vincent, and R. Gribonval. Beyond the narrowband approximation: Wideband convex methods for under-determined reverberant audio

source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 18(7):1818–1829, 2010.

[94] K. Kreutz-Delgado, J. Murray, B. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Computation*, 15(2):349–396, 2003.

[95] K. Labusch, E. Barth, and T. Martinetz. Robust and fast learning of sparse codes with stochastic gradient descent. *IEEE Journal on Selected Topics in Signal Processing*, 5(5):1048–1060, 2011.

[96] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788, October 1999.

[97] E. LePennec and S. Mallat. Sparse geometric image representations with bandelets. *IEEE Transactions Image Processing*, 14(4):423–438, 2005.

[98] M. S. Lewicki, T. J. Sejnowski, and H. Hughes. Learning overcomplete representations. *Neural Computation*, 12:337–365, 2000.

[99] R. Linsker. An application of the principle of maximum information preservation to linear systems. In *Proceedings of AIP Conference: Neural Networks for Computing*, pages 206–211, 1986.

[100] Y. Luo, W. Wang, J. A. Chambers, S. Lambotharan, and I. K. Proudler. Exploitation of source nonstationarity in underdetermined blind source separation with advanced clustering techniques. *IEEE Transactions on Signal Processing*, 54(6-1):2198–2212, 2006.

[101] B. Mailhè, D. Barchiesi, and M. D. Plumbley. INK-SVD: Learning incoherent dictionaries for sparse representations. In *Proc IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3573–3576, 2012.

[102] B. Mailhè, S. Lesage, R. Gribonval, F. Bimbot, and P. Vandergheynst. Shift invariant dictionary learning for sparse representations: Extending K-SVD. In *European Signal Processing Conference*, volume 4, 2008.

[103] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:16–60, 2010.

[104] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *IEEE International Conference Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[105] S. Makino, T. W. Lee, and H. Sawada. *Blind Speech Separation.* Springer, 2007.

[106] S. Makino, H. Sawada, R. Mukai, and S. Araki. Blind source separation of convolutive mixtures of speech in frequency domain. *IEICE Transactions Fundamentals*, E88-A:1640–1655, Jul. 2005.

[107] S. G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

[108] M. I. Mandel, R. J. Weiss, and D. P. W. Ellis. Model based expectation-maximization source separation and localization dictionaries. *IEEE Transactions on Audio Speech and Language Processing*, 18(2):382–394, 2010.

[109] C. M. Michel, M. M. Murray, G. Lantz, S. Gonzalez, L. Spinelli, and R. Peralta. EEG source imaging. *Clinical Neurophysiology*, 115(10):2195–2222, 2004.

[110] H. Mohimani, M. Babaie-Zadeh, and C. Jutten. A fast approach for overcomplete sparse decomposition based on smoothed ℓ_0 norm. *IEEE Transactions on Signal Processing*, 57(1):289–301, 2009.

[111] G. Monaci, P. Vandergheynst, and F. T. Sommer. Learning bimodal structure in audio-visual data. *IEEE Transactions on Neural Networks and Learning Systems*, 20(12):1898–1910, 2009.

[112] K. Nakadai, H. G. Okuno, and H. Kitano. Real-time sound source localization and separation for robot audition. In *Proceedings IEEE International Conference on Spoken Language Processing*, pages 193–196, 2002.

[113] S. Nam, M. E. Davies, M. Elad, and R. Gribonval. Cosparse analysis modeling - uniqueness and algorithms. In *IEEE International Conference Acoustic, Speech Signal Processing*, pages 5804–5807, 2011.

[114] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, May 2009.

[115] D. Nion and L. D. Lathauwer. An enhanced line search scheme for complex-valued tensor decompositions. application in ds-cdma. *Signal Processing*, 88(3):749–755, 2008.

[116] J. Nocedal and S. J. Wright. *Numerical Optimization.* New York: Springer, 1999.

[117] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, 2006.

[118] B. A. Olshausen, C. F. Cadieu, and D. K. Warland. Learning real and complex overcomplete representations from the statistics of natural images. *Proceedings of SPIE*, 7446, 2009.

[119] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[120] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1. *Vis. Res.*, 37(23):3311–3325, 1997.

[121] B. Ophir, M. Elad, N. Bertin, and M. D. Plumbley. Sequential minimal eigenvalues: An approach to analysis dictionary learning. In *Proceedings of 19th European Signal Processing Conference*, pages 1465–1469, 2011.

[122] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of the 27 th Annual Asilomar Conference on Signals, Systems, and Computers*, pages 40–44, 1993.

[123] M. S. Pedersen, D. Wang, J. Larsen, and U. Kjems. Two-microphone separation of speech mixtures. *IEEE Transactions on Neural Networks*, 19(3):475–492, 2008.

[124] T. Peleg, Y. C. Eldar, and M. Elad. Exploiting statistical dependencies in sparse representations for signal recovery. *IEEE Transactions on Signal Processing*, 60(5):2286–2303, 2012.

[125] G. Peyré and J. Fadili. Learning analysis sparsity priors. In *Proceedings of SAMPTA*, 2011.

[126] T. Pham, P. Garrat, and C. Jutten. Separation of a mixture of in dependent sources through a maximum likelihood approach. In *Proceedings of European Signal Processing Conference*, pages 771–774, 1992.

[127] M. D. Plumbley. Dictionary learning for L1-exact sparse coding. *Lecture Notes in Computer Science*, 4666:406–413, 2007.

[128] M. D. Plumbley, T. Blumensath, L. Daudet, R. Gribonval, and M. E. Davies. Sparse representations in audio and music: From coding to source separation. *Proceedings of IEEE*, 98(6):995–1005, 2010.

[129] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing*, chapter 10.2. Golden Section Search in One Dimension. Cambridge University Press, 3rd edition, 2007.

[130] R. Rubinstein, A. M. Bruckstein, and M. Elad. Dictionaries for sparse representation modelling. *Proceedings of IEEE*, 98(6):1045–1057, 2010.

[131] R. Rubinstein, T. Peleg, and M. Elad. Analysis k-svd: A dictionary-learning algorithm for the analysis sparse model. *IEEE Transactions on Signal Processing*, 61(3):661–677, 2013.

[132] P. Sallee and B. A. Olshausen. Learning sparse multiscale image representations. In *Conference Neural Information Processing Systems*, 2003.

[133] H. Sawada, S. Araki, and S. Makino. Underdetermined convolutive blind source separation via frequency bin-wise clustering and permutation alignment. *IEEE Transactions Audio Speech and Language Processing*, 19(3):516–527, 2011.

[134] P. Schmid-Saugeon and A. Zakhor. Dictionary design for matching pursuit and application to motion-compensated video coding. *Circuits and Systems for Video Technology*, 14(6):880–886, 2004.

[135] K. Schnass and P. Vandergheynst. A union of incoherent spaces model for classification. In *IEEE International Conference Acoustics, Speech, and Signal Processing*, pages 5490–5493, 2010.

[136] I. W. Selesnick, R. G. Baraniuk, and N. C. Kingsbury. The dual-tree complex wavelet transform. *IEEE Signal Process Magazine*, 22(6):123–151, 2005.

[137] K. Skretting and K. Engan. Recursive least squares dictionary learning algorithm. *IEEE Transactions on Signal Processing*, 58(4):2121–2130, 2010.

[138] P. Smaragdis. Blind separation of convolved mixtures in the frequency domain. In *Neurocomput.*, pages 21–34, 1998.

[139] P. Smaragdis. Non-negative matrix factor deconvolution, extraction of multiple sound sources from monophonic inputs. In *Proceedings of 5th International Conference on Independent Component Analysis and Blind Signal Separation*, pages 494–499, 2004.

[140] P. Smaragdis and J. C. Brown. Nonnegative matrix factorization for polyphonic music transcription. In *IEEE International Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180, 2003.

[141] P. Smaragdis, B. Raj, and M.V. Shashanka. Sparse and shift-invariant feature extraction from non-negative data. In *IEEE International Conference on Audio and Speech Signal Processing*, pages 2069–2072, 2008.

[142] J. Starck, M. Elad, and D.L. Donoho. Redundant multiscale transforms and their application for morphological component analysis. *Advances in Imaging and Electron Physics*, 132:287–348, 2004.

[143] P. Sudhakar. *Sparse Models and Convex Optimisation for Convolutive Blind Source Separation.* Ph.D. thesis, February 2011.

[144] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

[145] P. Tichavský and Z. Koldovský. Weight adjusted tensor method for blind separation of underdetermined mixtures of nonstationary sources. *IEEE Transactions on Signal Processing*, 59(3):1037–1047, 2011.

[146] I. Toèić and P. Frossard. Dictionary learning for stereo image representation. *IEEE Transactions Image Processing*, 20(4):921–934, 2011.

[147] I. Toèić and P. Frossard. Dictionary learning: what is the right representation for my signal. *IEEE Signal Processing Magazine*, 28(2):27–38, March 2011.

[148] A. Tonazzini, L. Bedini, E. E. Kuruoglu, and E. Salerno. Blind separation of autocorrelated images from noisy mixtures using MRF models. In *International Symposium on Independent Component Analysis and Blind Signal Separation*, pages 675–680, 2003.

[149] J. Tropp. *Topics in sparse approximations.* PhD thesis, University of Texas at Austin, 2004.

[150] J. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007.

[151] L. Vielva, D. Erdogmus, and J. C. Prancipe. Underdetermined blind source separation using a probabilistic source sparsity model. In *In 2nd International Workshop on Independent Component Analysis and Blind Signal Separation*, pages 675–679, 2001.

[152] E. Vincent, S. Araki, and P. Bofill. The 2008 signal separation evaluation campaign: A community-based approach to large-scale evaluation. In *International Conference on Independent Component Analysis and Signal Separation*, pages 734–741, 2009.

[153] E. Vincent, R. Gribonval, and C. Févotte. Performance measurement in blind

audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 14(4):1462–1469, 2006.

[154] D. Wang and G. J. Brown, editors. *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications.* Wiley-IEEE Press, September 2006.

[155] W. Wang, A. Cichocki, and J. A. Chambers. A multiplicative algorithm for convolutive non-negative matrix factorization based on squared euclidean distance. *IEEE Transactions on Signal Processing*, 57:2858–2864, 2009.

[156] W. Wang, Y. Luo, S. Sanei, and J. A. Chambers. Note onset detection via nonnegative factorization of magnitude spectrum. *EURASIP Journal on Advances in Signal Processing*, pages 447–452, 2008.

[157] W. Wang, S. Sanei, and J. A. Chambers. Penalty function based joint diagonalization approach for convolutive blind separation of nonstationary sources. *IEEE Transactions on Signal Processing*, 53(5):1654–1669, 2005.

[158] D. P. Wipf and B. D. Rao. Sparse bayesian learning for basis selection. *IEEE Transactions on Signal Processing*, 52(8):2153–2164, 2004.

[159] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[160] T. Xu and W. Wang. A compressed sensing approach for underdetermined blind audio source separation with sparse representation. In *Proceedings of IEEE International Workshop on Statistical Signal Processing*, pages 493 – 496, 2009.

[161] T. Xu and W. Wang. A block-based compressed sensing method for underdetermined blind speech separation incorporating binary mask. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2022 – 2025, 2010.

[162] T. Xu and W. Wang. Methods for learning adaptive dictionary for underdetermined speech separation. In *Proceedings of IEEE 21st International Workshop on Machine Learning for Signal Processing*, 2011.

[163] T. Xu, W. Wang, and W. Dai. Fast dictionary learning algorithm via codeword clustering and hierarchical sparse coding. In *9th IMA International Conference on Mathematics in Signal Processing*, 2012.

[164] M. Yaghoobi, T. Blumensath, and M. E. Davies. Dictionary learning for sparse approximations with the majorization method. *IEEE Transactions on Signal Processing*, 57(6):2178–2191, 2009.

[165] M. Yaghoobi, L. Daudet, and M. E. Davies. Parametric dictionary design for sparse coding. *IEEE Transactions Signal Processing*, 57(12):4800–4810, 2009.

[166] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies. Noise aware analysis operator learning for approximately cosparse signals. In *IEEE International Conference Acoustic, Speech Signal Processing*, pages 5409–5412, 2012.

[167] O. Yilmaz and S. Rickard. Blind separation of speech mixtures via time-frequency masking. *IEEE Transactions on Signal Processing*, 52:1830–1847, 2004.

[168] M. Zibulevsky and B. A. Pearlmutter. Blind source separation by sparse decomposition of a signal dictionary. *Neural Computation*, 13(4):863–882, 2001.