Dictionary Optimisation for Representing and Sensing Sparse Signals

Thesis submitted at the Imperial College London
in partial fulfillment of the requirements for
the doctorate degree of Electrical Engineering

by

Xiaochen Zhao
Department of Electrical and Electronic Engineering
Imperial College London

October 2015

Declaration of Originality

I hereby declare that this thesis was entirely my own work and that any additional sources of information have been duly cited. I hereby declare that any internet sources, published or unpublished works from which I have quoted or drawn reference have been reference fully in the text and in the contents list.

– Xiaochen Zhao

Copyright Declaration

Abstract

Compressed sensing takes advantage that most of the natural signals can be sparsely represented via linear transformations, therefore it is possible to have accurate reconstruction from sub-Nyquist samplings. The properties of the measurement matrix directly affect the relation between the sampling rate and the distortion of the reconstructions. People have been trying to either design measurement matrices from the signal statistics, or train the matrices from large amount of similar signals. Hence the relevant techniques they keep developing become very hot research topics.

This thesis focuses on discussing the impact of the measurement matrices on representing and sensing sparse signals. The full text is divided into four parts (presented in Chapter 2 to 5, respectively). In Chapter 2 we focus on the dictionary update stage in dictionary learning. Given observations of the sparse signals via an over-complete measurement matrix, , dictionary learning is to find this measurement matrix, i.e., dictionary, to accurately reconstruct the sparse signals. Usually a dictionary learning problem includes two stages that are implemented iteratively, sparse coding and dictionary update. Sparse coding is to fix the dictionary and update the sparse pattern of the estimated sparse signals. Dictionary update is to fix the sparse pattern and update the dictionary. We show that the failure of the update procedure to find a global optimum is not because of their converging to local minima or saddle points but to singular points. Afterwards, against this singularity issue, we revise the original objective function and propose a continuous counterpart. This modification is applied in the SimCO dictionary update framework and can be proved that in the limit case, the new objective function is the best possible lower semi-continuous approximation of the original. In Chapter 3 we present a joint source separation and dictionary learning algorithm to separate the noise corrupted mixed sources. The idea behind

is that for our different targeted sources, such as images and audios, have different sparse representations. We choose the deterministic scenarios, where the number of mixtures is not less than that of sources. The technique presented in Chapter 2 to alleviate singularity is used in the algorithm and we use examples to show its benefit. In Chapter 4, we notice that rely on the prior known statistics of the sparse signals, it is possible to allocate the sensing power accordingly to achieve the best possible performance. Given the non-uniform signal sparsity and the total power budget, we study how to optimally allocate the power across the columns of a Gaussian random measurement matrix so as to meet the reconstruction requirements. We revise the so called approximate message passing algorithm and quantify the MSE performance in the asymptotic regime. The obtained closed form of the optimal power allocation shows that in the presence of measurement noise, uniform power allocation is not optimal for non-uniformly sparse signals. In Chapter 5 we study distributed compressed sensing problem. We consider the scenarios where unequal number of measurements can be assigned for each signal block, and look for the optimal measuring rate allocation for recovering the sparse signals with common support. For simplification we assume the signals have Bernoulli-Gaussian distribution and again use AMP for analysis and obtain the exact phase transition curve in an asymptotic region. Interestingly, via the state evolution technique it can be shown that the rate region is concave, suggesting the corner points at the curve are optimal operating points and equal rate allocations is strictly sub-optimal. Besides the rate allocation, we also numerically quantify how the expected reconstruction error is affected by lack of enough measurements, the presence of Gaussian noise and the inter correlation across the signal blocks.

Acknowledgment

With my thesis almost complete, I would like to express my deep gratitude to a few people who have helped me and guided me through this PhD studying.

I want to give my first sincere thanks to my supervisor, Dr. Wei Dai. What an opportunity that I could meet him in Imperial College and became his student! His rigorous spirit of academic, creative research ideas and his guidance of perseverance led to my enormous progress during my nearly four-year doctoral study. I was really enjoying the way he started fascinating topics and encouraged me to find the solution of each small problem by myself. When I got lost, he enlightened me with just the right tips, until I found the way out and connected all the clues to complete the story. All by his constantly encouragement and turning tough research to hands-on puzzles can I keep a high studying enthusiasm. Although most of the discussions with him are of academic topics, he actually inspired me a more general view to analyse complicated problems, which I deem it really valuable in the long run.

I am also glad to have my signal processing lab mates. Jason Filos, Maxime Ferreira Da Costa, Zhen Gao, Tianyao Huang, Evripidis Karseras, Pan Li, Yang Lu, Zeqiang Ma, Peng Zhang and Guangyu Zhou all provided me hours of insightful discussions. Oftentimes from the conversations with them I got inspirations of solving the problems that once bothered me for a long time. I treasured the time that, via mutual help and cooperation, we finished a number of challenging tasks and meanwhile had growth together. I feel very grateful for their years accompanying, bringing me cheer and laughter.

Lastly I thank my parents and my grandparents. Even from thousands of miles away, they are always thinking of me. They played different roles to give me encouragement and comfort, at the same time reminded me the responsibility and sense of crisis. I won't be

here without their love and support. I want to especially thank my departed grandfather, Longzhang. He taught me the first sentence in English. He was also my personality maker. His upbringing and supervision to me is the main reason I demand high standards to myself. Although in the first place we had difference in opinion on my future study plan, finally his understanding and being supportive made me believe his selfless love to me. I want to dedicate this thesis to my grandfather, as a gift to repay his raise graciousness, and convey my truly love to him.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction of Compressed Sensing

Compressed sensing, also called as compressive sensing or compressive sampling, is an extensively developed technique not only limited in signal processing and information theory area in the recent ten years. It is well known that Nyquist sampling rate provides a lower bound of the sampling rate in order to completely recover a signal. This is true in the conventional sense, however in practice a lot of applications including imaging and video processing, sensing networks, bioengineering systems and analog to digital conversions usually process signals with the sparse property. By "sparse" we mean a multidimensional signal can be presented under a basis where most of the coordinate coefficients are zero. For example, consider a smooth image which contains millions of pixels and they are all non-zero. By transforming it to the wavelet domain, we may obtain that only thousands of the wavelet coefficients are significantly far from zero while the rest are very close to zero. This small fraction of "strictly" non-zero coefficients will be enough for catching most of the information from the original image. Therefore compressing, i.e., throwing out the close-to-zero coefficients, became a necessary procedure in order to remove these information redundancy. Compressed sensing algorithms merge sampling and compressing into one step. It vastly shortens the sensing time and will significantly cut down the resource budget in various infrastructure constructions.

Fundamental linear algebra tells that it is not possible to find a unique solution if there are more unknowns than equations. Thanks to the sparse property in large amount of practical signals, compressed sensing is able to provide exact recovery of high-dimensional signals by using a much smaller number of samplings, where each sample is a linear com-

bination of the signals. The underlying rules is that the sparsity of the signals can indeed decrease the necessary number of equations to find a unique sparsest solution. However besides the brute-force searching, which has been proved not to be able to find the solution in a polynomial time, finding efficient algorithms for the sparse signal reconstruction became hot topics. Cornerstone algorithms cross the past ten years include Basis Pursuit (BP) algorithm (which is commonly known as the ℓ_1-minimisation), Orthogonal Matching Pursuit (OMP) algorithm, Sparse Bayesian Learning (SBL) algorithm and Approximate Message Passing (AMP) algorithm, etc.

Different type of compressed sensing reconstruction algorithm show different sensitivity to the way of sampling. For most of the algorithms, a better sparse representation can decrease the number of sampling in a large extent. Therefore either based on the statistics of the signals that is know in advance to design good sampling strategy, i.e., sensing matrix design, or adaptively train a good sensing matrix according to the obtained measurements, i.e., dictionary learning, are the accompanying emerging research topics to finding "good" sparse representations.

1.1 Sparse Recovery Algorithms

We say a signal $\boldsymbol{x} \in \mathbb{R}^n$ is S sparse if there are S many or fewer non-zero components in $\boldsymbol{x}$. In mathematical form,

$$\|\boldsymbol{x}\|_0 \leq S,$$

where $S \leq n$ and $\|\cdot\|_0$ is the ℓ_0-pseudo norm which counts the number of non-zero elements in $\boldsymbol{x}$. Measure the sparse signal $\boldsymbol{x}$ from m many linear combination of its elements, where $m \leq n$. Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be the measurement matrix and $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ be the measurements. In compressed sensing we assume the sparsity S of the signal $\boldsymbol{x}$ is known a prior. We aim to solve the ℓ_0-minimisation problem

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0 \text{ subject to } \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}. \tag{1.1}$$

One necessary sufficient condition of (1.1) existing unique solution is that the measurement matrix $\boldsymbol{A}$ has rank larger than $2S$. A simple proof could be as follows. Assume $\boldsymbol{x}_1$

and $\boldsymbol{x}_2$ are both solutions of (1.1). $\boldsymbol{x}_1 - \boldsymbol{x}_2$ is at most $2S$ sparse and in contradiction $\boldsymbol{A}(\boldsymbol{x}_1 - \boldsymbol{x}_2) = \boldsymbol{0}$. However since ℓ_0-minimisation problem is well defined, there is no efficient way to solve it and is computationally known as an NP-complete problem.

One popular tool in compressed sensing is the Restricted Isometry Property (RIP). In linear algebra it characterises matrices which are nearly orthonormal, at least when operating on sparse vectors. It is used to describe whether a measurement matrix is suitable for compressed sensing problems.

Definition 1.1. (Restricted Isometry Property [18]) A matrix $\boldsymbol{A}$ satisfies the restricted isometry property with constant δ_S if for arbitrary given signal $\boldsymbol{x}$ with sparsity S,

$$(1 - \delta_S) \|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{A}\boldsymbol{x}\|_2^2 \leq (1 + \delta_S) \|\boldsymbol{x}\|_2^2 ,$$

where $0 \leq \delta_S \leq 1$.

In the RIP definition, the measurement matrix $\boldsymbol{A}$ performs a way to transform the signal $\boldsymbol{x}$ from the signal space to a smaller measurement space. By writing the RIP condition into form

$$(1 - \delta_S) \leq \frac{\|\boldsymbol{A}\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2} \leq (1 + \delta_S) ,$$

we see that when δ_S is close to zero there is no dramatic ℓ_2-norm change of the signal $\boldsymbol{x}$ due to the transformation and any possible transformation, i.e., consider any square sub-matrix of $\boldsymbol{A}$, is nearly isometric. RIP condition provides sufficient conditions for successfully reconstruction of compressed sensing problems. Yet the disadvantage is that generally to check weather a matrix satisfies RIP condition is an NP-compelete problem and RIP is not a necessary condition and the usually the constant δ_S's are loose.

1.1.1 ℓ_1-minimisation

One standard way to handle the compressed sensing problem is via ℓ_0-minimisation. Unfortunately ℓ_0-minimisation is an NP-complete problem and in general can not be solved in polynomial time. A convex relaxation reconstruction, first proposed by Chen. et. al. [22], namely Basis Pursuit (BP), is then considered as an efficient replacement. In BP the non-convex ℓ_0-pseudo norm is replaced by the convex ℓ_1-norm and hence the problem

formulation

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1 \text{ subject to } \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}. \tag{1.2}$$

Analysed by Donoho [32] and Candes. et. al [18], this relaxation often reconstruct the signal $\boldsymbol{x}$ successfully and the solution is same as the one given by (1.1). Basis Pursuit problem can be written in a linear form which can be solved efficiently via a few linear solvers.

Further introduce additive noise to the measurement. The system model changes to $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}$, where $\boldsymbol{w} \in \mathbb{R}^m$ represents for the additive measurement noise. In order to find the sparse solution of the signal via the relaxed reconstruction meanwhile try to remove the noise, one consider

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1 \text{ subject to } \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 \leq \epsilon. \tag{1.3}$$

This problem is known as as the basis pursuit denosing (BPDN). It was shown that if the RIP condition of the measurement matrix $\boldsymbol{A}$ performs well (e.g., $\delta_S + \delta_{2S} + \delta_{3S} < 1$ [18]) then the problem is solved with the robust performance. When there is no known information about the noise, we may simply use the Lagrangian unconstrained form to get an alternative problem formulation to for finding the sparse solution of $\boldsymbol{x}$,

$$\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \lambda \|\boldsymbol{x}\|_1 . \tag{1.4}$$

This problem is known as the least absolute shrinkage and selection operator (LASSO) problem. Parameter λ is used to tune the weight between the ℓ_2 least square term and the ℓ_1 term, controlling sparsity level of the solution. It is noteworthy that the solution of (1.4) is sensitive to the parameter λ, hence finding optimal λ is also a non-trivial task. Simultaneously optimising the parameter λ and finding the solution of (1.4) can be completed using the least angle regression stage wise (LARS) algorithm [38]. Approximate Message Passing can be considered an equivalent solver of the Lasso problem and will provide the optimised parameter λ when the algorithm finally converges.

1.1.2 Orthogonal matching pursuit (OMP)

The ℓ_1-minimisation algorithm can provide uniform guarantees over all sparse signals. It also works robustly for approximately sparse signals and with the presence of Gaussian noise. However the complexity of the optimisation procedure increases with n^3, where n is dimension of the signal, therefore may still cost relatively high. Another set of compressed sensing algorithms are called greedy algorithms. These iterative signal support update algorithm provides a lower computational complexity than BP, although they do not have provable uniform guarantees or stability. Usually given the same sampling rate, they are able to give recoverable result for sparser signals than BP.

Orthogonal matching pursuit (OMP) algorithm is one of the benchmark greedy algorithms. It iteratively improves the estimate of the signal by choosing the column of a matrix that has the most correlation with the residual. Consider model $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$, where each column of $\boldsymbol{A}$ is normalised and the sparsity S of signal $\boldsymbol{x}$ is known. OMP starts the estimate of the signal from $\boldsymbol{x}^0 = \boldsymbol{0}$ and set the residual as $\boldsymbol{r}^0 = \boldsymbol{y}$. The support set of the initial estimate is therefore $\mathcal{I}^0 = \oslash$. At each iteration t, one chooses a column of $\boldsymbol{A}$ according to

$$i = \arg\max_i \left| \langle \boldsymbol{r}^t, \boldsymbol{A}_{:,i} \rangle \right|,$$

and add it to the support set $\mathcal{I}^t = \mathcal{I}^{t-1} \cup \{i\}$. Then one finds the estimate and the residual at iteration t via

$$\boldsymbol{x}^t = \boldsymbol{A}^\dagger_{:,\mathcal{I}^t} \boldsymbol{y},$$

$$\boldsymbol{r}^t = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^t.$$

The iterations keep going until S many columns of $\boldsymbol{A}$ are selected into the support set.

Proposition 1.2. *(Theorem 3.1 in [28]) Suppose measurement matrix $\boldsymbol{A}$ satisfies the RIP condition of order $S+1$ with the isometry constant $\delta_{S+1} < \frac{1}{3\sqrt{S}}$. Then $\forall \boldsymbol{x} \in \mathbb{R}^n$ with sparsity $s \leq S$, OMP will recover $\boldsymbol{x}$ exactly from system $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ in S many iterations.*

Proposition 1.2 provides a loose sufficient bound for the measurement matrix for exact recovery. It is weaker than the sufficient condition for ℓ_1-minimisation.

1.1.3 Subspace pursuit (SP)

Subspace pursuit [24], as another famous greedy algorithm, has the similar idea as the OMP and have the same order of the computational complexity. The main difference is that instead of moving one column to the support set at each iteration, SP update the support set by simultaneously adding and removing columns. More precisely, in the initialisation, SP uses the S many columns of $\boldsymbol{A}$ that provides the largest residual, and name the support set as $\mathcal{I}^0$. Then at each iteration t, SP evaluates the residual, $\boldsymbol{A}^T\boldsymbol{r}^{t-1}$, then finds the indices corresponding to the S largest amplitudes and then stores them into a set $\mathcal{J}$. Then the support set of the signal estimate is updated by taking the union $\mathcal{I}^{t-1}\cup\mathcal{J}$ and leave S many indices corresponding to the S largest amplitudes. The iterations keep going until converge or a stop criteria is met. Another algorithm came up later than SP, called CoSaMP [76] have a similar idea of adding and removing indices from the support set. The major difference are the scaling up and down on the support set volume during each iteration update.

Proposition 1.3. *(Theorem 9 in [24]) Let $\boldsymbol{x}\in\mathbb{R}^n$ be such that $\|\boldsymbol{x}\|_0\leq T$, and let its corresponding measurement be $\boldsymbol{y}=\boldsymbol{A}\boldsymbol{x}+\boldsymbol{w}$, where $\boldsymbol{w}$ denotes the noise vector. Suppose the sampling matrix $\boldsymbol{A}$ satisfies RIP of order $3S$ with the isometry constant $\delta_{3S}<0.083$. Then the reconstruction error of the SP algorithm satisfies:*

$$\|\boldsymbol{x}-\hat{\boldsymbol{x}}\|_2\leq C\,\|\boldsymbol{w}\|_2\,,$$

where $C=\frac{1+\delta_{3S}+\delta_{3S}^2}{\delta_{3S}(1-\delta 3_S)}\leq 14.31$.

1.1.4 Approximate Message Passing (AMP)

Approximate message passing (AMP) proposed in [6], constructing and solving artificial denoising problems at each iteration, is a compressive sensing algorithm which has nice properties which statistically improve the convergence rate. Generally two key steps at

each iteration, signal denosing and resudial update, are included in AMP,

$$\boldsymbol{x}^{t+1} = \eta\left(\boldsymbol{x}^t + \boldsymbol{A}^T\boldsymbol{r}^t;\, \boldsymbol{\theta}^t\right),$$
$$\boldsymbol{r}^t = \boldsymbol{y} - \boldsymbol{A}x^t + \frac{1}{m}\eta'\boldsymbol{r}^{t-1},$$

where η represents the denoiser and $\eta' = \partial\eta/\partial\left(\boldsymbol{x}^t + \boldsymbol{A}^T\boldsymbol{r}^t\right)$. In iteration a term called Onsager term, i.e., $\frac{1}{m}\eta'\boldsymbol{r}^{t-1}$ is added on the residual. It enables us to predict accurate theoretical performance in the asymptotic regime, which some traditional criteria such as RIP cannot provide. In the latter part of our thesis, the techniques related to AMP will be thoroughly discussed and extended to compressed sensing structured sparse signals.

1.2 Applications

The theory of compressed sensing is booming and meanwhile penetrating into more and more applications. Advanced developed applications include compressive imaging, image impainting, biology, MRI imaging, radar signal processing, communications, error correcting and seismology, to name a few. In the following we briefly introduce several well researched applications.

1.2.1 Compressed Sensing in Image Processing

The most prevailing application of compressed sensing is for image processing. Images from various sources are usually sparse over some basis therefore can be compressed using compressed sensing techniques. Take digital cameras as an example, when a megapixel picture is taken, each pixel of the picture is captured by one sensor. Therefore each digital camera needs millions of sensors. And most of the time due to the storage limitation the pictures are compressed before being stored. During this procedure there is an obvious waste to throw a large percentage of captured pixels.Compressed sensing techniques developed novel strategies by sensing linear combination of the megapixels in order to complete the pixel capture and compression at the same time. In this way much fewer sensors are needed and there would be a much more efficient pixel collection. Based on this idea a variety of measurement methods and the corresponding decoding methods have been de-

veloped to meet different purposes. Not just staying in theory, actual hardware devices have been developed. Rice university successfully built the first prototype of "single-pixel" camera in 2006 [108]. This camera consists of a digital micro-mirror device, two lenses, a single photon detector and an analog-to-digital converter. The principle of the device is to use the lens and the digital micro-mirrors to generate inner products with random vectors. Then the photon detector looks after computing the measurement of the light that collected by the lens. Finally optical computer computes the linear measurements of the image and another digital computer will be in charge of reconstructing the image from the computation results. This "single-pixel" camera showed very impressive results. And if considering its potential extension to milti-pixels cases, the image quality will most likely outperform the traditional digital cameras.

Another compressive imaging area of people's particular interest is in magnetic resonance imaging (MRI). MRI images can be sparsely represented in Fourier domain thus compressed sensing can be used for reducing the scanning time. This is an attractive reason to apply compressed sensing to MRI imaging since as taking an MRI image of a tissue, organ or joint is taken, a person should stay still in the machine for a long time. This is a difficult task or even not possible for children and people who are in pain. By using the compressed sensing technique, the number of the samples can usually maintain in a low fraction compared to traditional MRI while keep the image quality therefore significantly decreased the scanning time. In addition, new result [2] shows that by carefully designing the measurement matrix, the number of samples taken to successfully recovering the MRI images can be further reduced. For more information about MRI compressed imaging please referred to [2] and the reference therein.

1.2.2 Sparse Signal Representation

Sparse representation received a lot discussion in signal processing. Image impainting is an typical example. Some of the early artworks, for example, the printing restored at the Renaissance, were damaged with scratches due to being improperly preserved. Cracks in photographs and dust spots in films are common phenomena. Also we may add or remove elements from pictures (e.g., removal of stamps on postcards or red-eyes on photos). These issues can be fixed by image impainting algorithms. Again because of the sparsity of image

in certain basis, the corrupted area on images can be manually selected out and filled-in using the compressed sensing technique. Elad, et. al. proposed algorithms in [69] and showed an example to successfully remove the red texts of a given photo. For more relative detailed please refer in [39, 69].

Blind source separation (BSS) is the second example. It has been investigated during the past two decades in a wide range of application fields such as speech and image separation. One motivation for BSS in early studies is to filter the voices when there are a few people talking at the same time. A few microphones are fixed in different positions, each can only record a different linear combination of the voices. The first few studies focus on the instantaneous and (over-)determined BSS problem. The problem is addressed under the framework of independent component analysis (ICA) [58], assuming that the sources are statistically independent. This has led to some well-known approaches, such as Infomax [9], maximum likelihood estimation [43], the maximum a posterior (MAP) [10], and FastICA [58]. Convoluted and/or underdetermined BSS problems have also been extensively studied especially in the speech processing applications, where the sensor measurements are usually modeled as convoluted (often underdetermined) mixtures of the original sources due to the presence of room reverberations (and often more sources than sensors).

While blind source separation decompose signals in high dimensions, for the underdetermined mixtures, prior knowledge is required to succeed in the separation. One of the most common prior knowledge would be the sparsity of the signals. The first step in that direction is taken by Thomas Blumensath and Mike Davies [11]. They pointed out several similarities between compressed sensing and source separation. By assuming the mixing system is known, they extended some of the results in compressed sensing to more general overcomplete sparse representations and studied the sensitivity of errors in the mixing system. Later in work [16, 14, 15, 1] the mixing system is assumed to be unknown and algorithms are design to both separate the source and reconstruct each of the corresponding signals.

1.2.3 Radar Signal Processing

Traditional radar systems use matched filter and high rate analog to digital converter for pulse signal processing. In order to successfully demodulate the signals, very high pulse

transmitting frequency are often restricted to avoid overlaps. This results in the accuracy limitation. In addition the conventional approaches of radar processing is complicated and costly. Compressive radar imaging lattices the time-frequency plane where each grid becomes an element in a signal vector. When there are a small number of targets in the scanning area, the receiving time-frequency signal vector can be treated sparse. Therefore compressed sensing algorithms can be used for sensing and reconstructing the target locations [55]. Besides time-frequency, space-frequency and time-space compressive sensing algorithms are also the hot research topics in radar imaging.

1.2.4 Bio-informatics

Compressed sensing can also be applied to biology to develop effective and low cost genetic sensing algorithms. For example, accurate identification on large number of genetic sequences usually require DNA microarrays for detection and classification [27]. In traditional DNA microarray designs, each genetic sensor can only match up one DNA base sequence with one particular organism in the target set. However there are always more than one organism that share very similar DNA base sequences. When this hybridization occurs, detection errors will happen due to the wrong classifications. At the same time the accuracy of classification and the unique identification design constraint the number of organisms being detected as well, because in this design the length of the DNA sequence and the number of targets are in direct proportion to the implementation time. For the bio-sensing experiments that large number of organisms needs to be identified, the time consumption could be very high. For another example, metagenomic research compares thousands of environmental and health-related samples by extracting and sequencing the rRNA amplifications and measuring their similarity under certain metrics. One of the important steps is to classify the operational taxonomic units within the sample. Methods to achieve this task requires hundreds of thousands of reads of the taxonomic assignments, which is a computationally costly work [64].

In the laboratory experiments, consider using DNA microarray to find biological agents in air or liquid samples that occupied by hostile adversaries. Among a large number of agents only very small amount will be used and present a significant concentration of the hostile adversaries at a given time and location. And furthermore, there may be only a

little fraction of the harmful agents are of our particular interest. This makes our targets very sparse compared to the whole samples. Mathematically, one can represent the DNA concentration of each organism as an element in a detected vector. As is mentioned, this vector is approximately sparse, which contains only a few significant entries. Compressive DNA microarrays are therefore naturally employed to detect the locations of the targets and can even determine their concentrations as well.

1.3 Measurement Design for Compressed Sensing

Standard compressed sensing theories have been thoroughly studied in the past few years. In these theories, sparse signals, incoherence and uniform random subsampling are usually three fundamental concepts. In many applications, e.g. Magnetic Resonance Imaging, the sampling scheme is fixed therefore the measurement matrix is coherent; distributed sensor networks, the signals is structured sparse or approximate sparse. In these scenarios one or more than one of the three conditions are not satisfied, therefore leaves a performance gap to the theoretical results. For those cases with physical constraint so that standard compressed sensing technique does not reach ideal performance. Therefore we need bespoke sampling strategies.

When many signals of the same type are available, a feasible idea is to adaptively learn the measurement matrix to best fit them. This research is called dictionary learning. Learned dictionary contains the characteristic information of the training signals, hence it shows strong robustness to sample and reconstruct other signals of the same type. Dictionary learning process is usually time consuming and requires large amount of computational cost. The success of dictionary learning algorithm highly depends on the number of the signals, and very low successful rate is usually unavoidable when the signal quantity is gravely insufficient. Increasing the successful rate in low demand of signals is one of the major targets of the dictionary learning research.

Taking another track, the problems can be unfolded from Bayesian formulation. That is, the measurement matrices can be designed in advance with the prior information of the signals. At the same time where Bayesian framework counts more information into solving the compressed sensing problem, it provides possibility to quantify the perspective sampling

and points out ways to adjust measurement matrices. For structured sparse signals, e.g., non-uniform sparse signals, one can allocate the sampling resource according to sparse level of change along the signals. In scenarios where distributed compressed sensing technique is applied, joint inter-signal distribution can help suggesting the adjustment of sampling rates on different signals.

Papers have given designed sampling algorithms and derivation of the performance for structured sparse signals. Most popular structured signals that were studied are non-uniformly sparse signals, including block sparse signals and hierarchical block sparse signals. Instead of considering a high dimensional signal as a whole, a block sparse signal is separated into a few sub-signals, where each sub-signal is treated as sparse with a given uniform sparsity. Similarly, hierarchical block sparse signal treats each of its sub-signal as a block sparse signal. Another key concerned signal is called group sparse signal. A high dimensional signal is cut into many small pieces with even lengths, where each piece is treated as one singleton, and the sparsity acts on the singletons rather than the signal elements. According to the given sparsity information, multilevel sampling strategies can be used for the sampling. If a large number of signals are available, one may use dictionary learning algorithms to adaptively design the measurement scheme so as to achieve better reconstruction performance than random measurements under the same sampling rates [41, 3, 25]. In [2], Adcock, et. al. presented three new concepts: asymptotic sparsity, asymptotic incoherence and multilevel random subsampling to replace the three traditional compressed sensing concepts. Based on the new theory, a more universalised framework is given, hence analysis of the measurement design problems right outside the standard compressed sensing theory margins become available.

1.4 Outline and Contributions

In the following we summarise the content and contributions for each chapter.

Chapter 2: Singularity issue in dictionary update and smoothed SimCO algorithm

Typical dictionary learning algorithms iteratively perform two stages: sparse coding and dictionary update. In this chapter we focus on the latter stage. We formulate dictionary update as an optimisation problem on a manifold, and particularly study one possible reason for the optimization procedures not always converging to a global optimum. An interesting result shown in our analysis is that the failure is not because of their converging to local minima or saddle points but to singular points, where the objective function is discontinuous. We further give an instructive example, studying on three mainstream dictionary update procedures, to support the above statement. Afterwords, against the singularity issue, we revise the original objective function and propose a continuous counterpart. This modification is applied in the SimCO dictionary update framework and hence we name it Smoothed SimCO. It can be proved that in the limit case, the new objective function is the best possible lower semi-continuous approximation of the original. In terms of line search implementations, we derived a correspondent Newton CG method. Simulations demonstrate the proposed method significantly outperforms the benchmark algorithms.

The chapter is based on work from the following publications:

- X. Zhao, G.Zhou and W. Dai, "Smoothed SimCO for Dictionary Learning: Handling the Singularity Issue", in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, 2013.

- X. Zhao, G. Zhou, W. Wang and W. Dai, "Weighted SimCO: A Novel Algorithm for Dictionary Update", in Proc. Sensor Signal Processing for Defense, 2012.

Chapter 3: Blind source separation based on dictionary learning

This chapter surveys recent works in applying sparse signal processing techniques, in particular dictionary learning algorithms to solve the blind source separation problem. For the proof of concepts, the focus is the scenario where the number of mixtures is not less than that of sources. Based on the assumption that the sources are sparsely represented by some dictionaries, we present a joint source separation and dictionary learning algorithm (SparseBSS) to separate the noise corrupted mixed sources with very little extra informa-

tion. We also discuss the singularity issue in the dictionary learning process which is one major reason for algorithm failure. Finally two approaches are presented to address the singularity issue.

The chapter is based on work from the following publications:

- 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, 2013.
- X. Zhao, G. Zhou, W. Wang and W. Dai, "Blind Source Separation Based on Dictionary Learning: A Singularity Aware Approach" in Advances in Modern Blind Source Separation Techniques: Theory and Applications, Springer, 2014.

Chapter 4: Power allocation in compressed sensing of non-uniformly sparse signals

In this chapter we study the problem of power allocation in compressed sensing when different components in the unknown sparse signal have different probability to be non-zero. Given the prior information of the non-uniform sparsity and the total power budget, we are interested in how to optimally allocate the power across the columns of a Gaussian random measurement matrix so as to obtain various reconstruction goals, e.g, minimise the total mean squared reconstruction error. Based on the state evolution technique originated from the work by Donoho, Maleki, and Montanari, we revise the so called approximate message passing (AMP) algorithm for the reconstruction and quantify the MSE performance in the asymptotic regime. Then the closed form of the optimal power allocation is obtained. The results show that in the presence of measurement noise, uniform power allocation, which results in the commonly used Gaussian random matrix with i.i.d. entries, is not optimal for non-uniformly sparse signals. Empirical results are presented to demonstrate the performance gain.

The chapter is based on work from the following publications:

- X. Zhao and W. Dai, "Power Allocation in Compressed Sensing of Non-uniformly Sparse Signals", in Proc. IEEE International Symposium of Information Theory , 2014.

- X. Zhao and W. Dai, "Compressed sensing non-uniformly sparse signals: An asymptotically optimal power allocation", in UCL-Duke Workshop on Sensing and Analysis of High-Dimensional Data, 2014.

Chapter 5: Joint approximate message passing for distributed compressed sensing

In this chapter, we study a novel joint sparse signal reconstruction approach for block sparse signals under the scenarios where unequal numbers of measurements are obtained for different signal blocks. This research can also be viewed as an extension of distributed compressed sensing (DCS) and carries more possibility in practically integrating techniques of saving measuring cost and improving reconstruction results. We consider Bernoulli-Gaussian signals and develop a group sparse signal reconstruction algorithm. The algorithm is based on approximate message passing (AMP), thus termed as joint-AMP. We use the state evolution technique to give analysis under asymptotic situation and show that, by fixing a total sensing resource, measurements equally assigned to each signal block is not an optimal strategy. Based on the phase transition analysis, we also give the estimated reconstruction error when lack of enough measurements as well as in the presence of Gaussian noise. In addition we introduce covariance among non-zeros part of the signal blocks to study the measurement amount change that affected by the dependence between signal blocks.

The chapter is based on work from the following publications:

- X. Zhao and W. Dai, "On Joint Recovery of Sparse Signals with Common Supports", in Proc. IEEE International Symposium of Information Theory , 2015.

- X. Zhao and W. Dai, "Joint Reconstruction for Distributed Compressed Sensing with Common Support Signals", in iCore Inaugural Workshop, 2015.

Chapter 6: Conclusions and Future work

Finally, we conclude the thesis by summarizing the main idea and elaborate on possible future work.

Contributions Outside the Scope of this Thesis

The author of this thesis has also contributed to some other compressed sensing related works which are not included in this thesis. These contributions can be found in the following publications.

- G. Zhou, X. Zhao and W. Dai, "Low Rank Matrix Completion: A Smoothed L0-search", Allerton Conference, 2012.
- X. Zhao, T. Lu and W. Dai, "Compressive Sensing Reconstruction Techniques with Magnitude Prior Information", in Proc. Sensor Signal Processing for Defense, 2011.

1.5 Notation and Abbreviations

We list notations and abbreviations that are frequently used throughout the thesis.

1.5.1 Notation

$\boldsymbol{x}$	column vector
$\boldsymbol{A}$	matrix
$\boldsymbol{A}^T$	matrix transpose
$\boldsymbol{A}^\dagger$	Moore-Oenrose pseudioinverse of matrix $\boldsymbol{A}$
$\boldsymbol{I}$	identity matrix
$\boldsymbol{X}_{:,i}$	i^{th} column of matrix $\boldsymbol{X}$
$\boldsymbol{D}_i$	in Chapter 2: $\boldsymbol{D}_{\text{supp}(\boldsymbol{X}_{:,i})}$
$\boldsymbol{A}_i$	in Chapter 5: a sub-matrix in diagonal matrix $\boldsymbol{A}$, where $\boldsymbol{A} = \text{diag}(\boldsymbol{A}_1, \boldsymbol{A}_2, ..., \boldsymbol{A}_K)$
$\boldsymbol{r}^t$	t^{th} iteration of update on vector $\boldsymbol{r}$ in an algorithm
$I, \mathcal{J}$	the signal support set
$f(\boldsymbol{D})$	scalar function of matrix $\boldsymbol{D}$
$p_X(x)$	probability of x
$P(\mathcal{A})$	probability of event $\mathcal{A}$
$\mathcal{N}(\mu, \sigma^2)$	Gaussian distribution with mean μ and variance σ^2
$\mathcal{U}(a, b)$	uniform distribution in intervial $[a, b]$
$\lambda_{\min}(\boldsymbol{D})$	minimum singular value of matrix $\boldsymbol{D}$

$\boldsymbol{x}_{\mathcal{I}}$ subvector of $\boldsymbol{x}$ with the element indices chosen from set $\mathcal{I}$

$\boldsymbol{A}_{\mathcal{I}}$ submatrix of $\boldsymbol{A}$ with the element columns chosen from set $\mathcal{I}$

$\|\boldsymbol{x}\|_2$ the ℓ_2-norm

$\|\boldsymbol{x}\|_1$ the ℓ_1-norm

$\|\boldsymbol{x}\|_0$ the ℓ_0-pseudo norm

$\|\boldsymbol{A}\|_F$ the Frobenius norm of matrix $\boldsymbol{A}$

$\mathrm{tr}(\boldsymbol{A}, \boldsymbol{B})$ trace of matrix $\boldsymbol{A}$ and $\boldsymbol{B}$

$\mathrm{supp}(\boldsymbol{x})$ the support set of $\boldsymbol{x}$

■ the end of proof

$\mathbf{0}$ zero vector

$\nabla f(\cdot)$ derivative of function $f(\cdot)$

$\nabla_\eta f(\cdot)$ derivative of function $f(\cdot)$ along direction η

$\bullet$ dot product of two matrices

$\otimes$ Kronecker product of two matrices

$[a, b]$ closed real value interval between a and b

(a, b) open real value interval between a and b

$[m]$ integer set $\{1, 2, 3, ..., m\}$

$\sup f(\cdot)$ supremum of function $f(\cdot)$

$\inf f(\cdot)$ infimum of function $f(\cdot)$

1.5.2 Abbreviations

AMP approximate message passing

AWGN additive white Gaussian noise

BMMCA blind multichannel morphological component analysis

BP basis pursuit

BSS blind source separation

CG conjugate gradient

CS compressed sensing

DCS distributed compressed sensing

ICA independent component analysis

IST iterative soft thresholding

JSM joint sparsity model

LASSO least absolute shrinkage and selection operator

MCA morphological component analysis

MMCA multichannel morphological component analysis

MMSE minimum mean squared error

MMV multiple measurement vectors

MOD method of optimal directions

MSE mean squared error

OMP orthogonal matching pursuit

PT phase transition

RIP restricted isometry property

SE state evolution

SCA sparse component analysis

SimCO simultaneous codeword optimization

SP subspace pursuit

SVD singular value decomposition

Chapter 2

Singularity Issue in Dictionary Update and the Smoothed SimCO Algorithm

2.1 Introduction

Sparse signal representation is a technique to approximate signals by only a small amount of chosen columns from an over-complete dictionary. It has been received wide interest in many research fields including signal processing, information theory, machine learning, etc. To increase the reconstruction performance, large number of great works touching upon source separation, signal denoising, coding, classification and inpainting have been done in the past two decades.

All these achievements mainly focus on solving two problems. The first problem is called sparse coding. For a given dictionary, where each of its column represents a codeword, one wants to find a set of good sparse coefficients which linearly combine dictionary codewords to approach the given training signals. ℓ_1 minimization or greedy algorithms such as basis pursuit (BP) [22], matching pursuit (MP) [71], orthogonal matching pursuit (OMP) [101, 81], regression shrinkage and selection (LASSO) [100], focal under determined system solver (FOCUSS) [46], subspace pursuit (SP) [24] and approximate message passing (AMP) [30] are to solve this sparse coding problem with different constraints. All these algorithms

shall work based on good constructed dictionaries. However problems like blind source separation and device calibration usually do not provide knowledge about the dictionary, thus for this type of problems the algorithms mentioned above become invalid. This obliged us to study the second problem: How do we determine the dictionary when it is partly or entirely not given?

In early approaches the dictionaries are generated from typical mathematical transforms, e.g., discrete Fourier transform (DCT), wavelets [89], curvelets [19], etc. Such predefined transforms are not targeted to particular training samples, thus do not always provide enough accurate reconstructions. The objective of dictionary learning is to find an over-complete dictionary to accurately reconstruct the training signals. Mainstream dictionary learning algorithms include two stages: *sparse coding stage* and *dictionary update stage*. The two stages are respectively responsible for updating the sparse coefficient and the dictionary but for some algorithms the division of work between the two stages is not that clear. In sparse coding stage the dictionary is fixed and the sparse coefficients are updated by ℓ_1 minimization and greedy approaches such as BP, OMP, SP, etc. In dictionary update stage, the dictionary is trained referring to the sparse coefficient obtained form the previous stage. The two-stage procedure are iterated until the convergence condition is met.

One of the earliest dictionary update scheme appeared in the method of optimal directions (MOD) [41] proposed by Engan et al. The main idea is as follows: iteratively implement sparse coding and dictionary update stage. In sparse coding stage, one fixes the dictionary and uses OMP or FOCUSS to update the sparse coefficients. Then in dictionary update stage, one fixes the obtained sparse coefficients and updates the dictionary. MOD was further modified to iterative least squares algorithm (ILS-DLA) [42] and recursive least squares algorithm (RLS-DLA) [91]. K-SVD algorithm [3], developed by Aharon et al., is another dictionary learning approach which can be viewed as a generalization of the K-means algorithm. In each iteration, the sparse coding stage does the same work as in MOD algorithm. Then in dictionary update stage, one fixes the sparsity pattern, and updates the dictionary and the nonzero coefficients simultaneously. In particular, the codewords in the dictionary are sequentially selected: the selected codeword and the corresponding row of the sparse coefficients are updated simultaneously by using singular value

decomposition (SVD). More recently, Dai et al. [25] considered the dictionary learning problem from a new perspective. They formulated dictionary learning as an optimization problem on manifolds and developed simultaneous codeword optimization (SimCO) algorithm. In each iteration SimCO allows multiple codewords of the dictionary to be updated with corresponding rows of the sparse coefficients jointly. This new algorithm can be viewed as a generalization of both MOD and K-SVD. Some other dictionary learning algorithms are also developed in the past decade targeting on various circumstances [78, 67, 66, 68]. For example, based on stochastic approximations, Mairal, et al. [68] proposed an online algorithm to address the problem with large data sets. This algorithm assumes that all the signals has the same statistical distribution. Each time one of the signals is introduced to refine the dictionary which obtained in the previous iteration. The whole update procedure is repeated until a stopping criterion is met (usually the dictionary converges).

Theoretical or in-depth analysis about the dictionary learning problem was meantime in progress as well. Gribonval et al. [48], Geng et al. [44] and Jenatton et al. [60] studied the stability and robustness of the objective function under different probabilistic modeling assumptions, respectively. In addition, one observation in [25] is that the dictionary update procedure may fail to converge to a minimizer. This is a common phenomenon happens in MOD, K-SVD and SimCO. Dai, et al. further observed that ill-conditioned dictionary is probably the reason leading to the failure. To address this issue, Regularized SimCO is proposed in [25]. The main idea is to add an l_2 norm of the sparse coefficients as a penalty term to the original objective function in dictionary update stage. This regularized technique is also applicable to other dictionary update procedures such as MOD. It is verified that the technique effectively mitigates the occurrence of ill-conditioned dictionary. The same approach was also considered in [114], however the discussion on the singularity issue was not detailed.

In this chapter, we will make further discussion on the dictionary update stage. We focus on the scenario that ill-conditioned dictionary occurs, and point out that singularity, rather than stationary points, is the major reason leading to the failure of dictionary learning algorithms. To avoid the singularity issues, we propose smoothing techniques on SimCO, termed *Smoothed SimCO*. Some theoretical derivations and proofs will be shown in this chapter to support the rationality and validity of this new algorithm. The major

contributions of this chapter are:

- An explicit example is provided to show that the benchmark algorithms fail to find a global minimum. Instead, theoretical derivations are made to show they all converge to ill-conditioned dictionaries.
- A continuous objective function is proposed to replace the original one. The new objective function results in significant improvement according to the numerical tests.
- We prove that the proposed objective function, in the limit, is the best possible lower semi-continuous approximation of the original one. The lower semi-continuity guarantees that the solution set is closed, which is required for a convergence of any optimization procedure. By contrast, the regularized objective function proposed in [25] does not have this property.
- A Newton CG method is designed to minimize the proposed objective function. It turns out that the corresponding computations are highly non-trivial. In this chapter, a second order implementing (the Newton CG method) for the Smoothed SimCO are derived. Numerical tests verify that our implementation achieves a good balance between convergence rate and computational complexity.

The remainder of this chapter is organized into six sections. In Section 2.2 we formulate the dictionary learning problem in the SimCO framework. In Section 2.3 we elaborate the singularity issue in dictionary learning. Moreover an explicit example is designed to show that mainstream algorithms including MOD, K-SVD and (Regularized) SimCO may fail. We propose the Smoothed SimCO algorithm in Section 2.4. In Section 2.5 a Newton CG implementation of the proposed algorithm is derived. Numerical examples and comparisons are given in section 2.6.

2.2 Dictionary Learning and the Framework of SimCO

In the dictionary learning problem, the goal is to find a dictionary $\boldsymbol{D} \in \mathbb{R}^{m\times d}$ and a sparse coefficient matrix $\boldsymbol{X} \in \mathbb{R}^{d\times n}$ to best represent the training samples $\boldsymbol{Y} \in \mathbb{R}^{m\times n}$. Each column of $\boldsymbol{D}$ represents for a *codeword* of the dictionary and each column of $\boldsymbol{Y}$ represents

a training sample. In practical applications the dictionary $\boldsymbol{D}$ is generally over-complete ($m < d$). This results in non-unique solutions of $\boldsymbol{X}$ unless certain constraints are posed. One widely used constraint is that $\boldsymbol{X}$ is sparse, i.e., most entries in $\boldsymbol{X}$ are zero. Following this constraint, the dictionary learning problem can be casted as

$$\inf_{\boldsymbol{D},\boldsymbol{X}} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F^2 + \|\boldsymbol{X}\|_0 , \tag{2.1}$$

where $\|\cdot\|_F^2$ denotes the Frobenius norm and $\|\cdot\|_0$ denotes the ℓ_0 pseudo-norm, which counts the number of non-zero elements.

Dictionary learning algorithms normally include two stages: sparse coding and dictionary update. In the sparse coding stage, one fixes the dictionary $\boldsymbol{D}$ and finds the sparse coefficients $\boldsymbol{X}$. That is,

$$\inf_{\boldsymbol{X}} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F^2 + \|\boldsymbol{X}\|_0 . \tag{2.2}$$

As the ℓ_0 pseudo-norm is non-convex, this optimization problem is difficult to solve [18]. In practice, one can either replace the term $\|\boldsymbol{X}\|_0$ with $\|\boldsymbol{X}\|_1$ ($\ell_1$norm) and turn (2.2) into a convex optimization problem, or employ greedy algorithms including IHT [12], OMP [81], SP [24], etc.

The goal of the dictionary update stage is to update the dictionary. Mathematically, this can be formulated using the SimCO framework [25] as follows. Let $\operatorname{supp}(\boldsymbol{X})$ denote the sparsity pattern which is the index set of nonzero elements in $\boldsymbol{X}$, i.e., $\operatorname{supp}(\boldsymbol{X}) = \{(i,j) : \boldsymbol{X}_{i,j} \neq 0\}$. Define $\mathcal{X}$ as the set of coefficient matrices having sparse support $\operatorname{supp}(\boldsymbol{X})$:

$$\mathcal{X} = \left\{ \boldsymbol{X} \in \mathbb{R}^{d\times n} : \ \boldsymbol{X}_{i,j} = 0, \forall (i,j) \notin \operatorname{supp}(\boldsymbol{X}) \right\}. \tag{2.3}$$

Assume that columns of the dictionary are of unit ℓ_2-norm and hence the space of the dictionaries is given by

$$\mathcal{D} = \left\{ \boldsymbol{D} \in \mathbb{R}^{m\times d} : \ \|\boldsymbol{D}_{:,i}\|_2 = 1, \forall i \in [d] \right\}, \tag{2.4}$$

where $[d] = \{1, 2, \ldots, d\}$. Then the dictionary update stage is to solve the optimization

problem

$$\inf_{\boldsymbol{D}\in\mathcal{D}} f(\boldsymbol{D}) = \inf_{\boldsymbol{D}\in\mathcal{D}} \underbrace{\inf_{\boldsymbol{X}\in\mathcal{X}} \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2}_{f(\boldsymbol{D})}. \tag{2.5}$$

Different approaches to solve (2.5) include the MOD [41], K-SVD [3], and SimCO [25] algorithms. It is also noteworthy that the constraint (2.4) is important (see [25] for detailed discussions).

For the later analysis, it is convenient to write the objective function $f(\boldsymbol{D})$ as a sum of atomic functions. Let $\boldsymbol{X}_{:,i}$ and $\boldsymbol{Y}_{:,i}$ be the i^{th} column of $\boldsymbol{X}$ and $\boldsymbol{Y}$ respectively. Let $\text{supp}(\boldsymbol{X}_{:,i})$ be the index set of non-zero elements in $\boldsymbol{X}_{:,i}$, i.e., $\text{supp}(\boldsymbol{X}_{:,i}) = \{k : \boldsymbol{X}_{k,i} \neq 0\}$. Define $\boldsymbol{D}_i = \boldsymbol{D}_{\text{supp}(\boldsymbol{X}_{:,i})}$ as the sub-matrix of $\boldsymbol{D}$ containing the columns indexed by $\text{supp}(\boldsymbol{X}_{:,i})$. Then it holds that

$$\begin{aligned} f(\boldsymbol{D}) &= \sum_i \inf_{\boldsymbol{X}_{:,i}:\, \boldsymbol{X}\in\mathcal{X}} \|\boldsymbol{Y}_{:,i}-\boldsymbol{D}\boldsymbol{X}_{:,i}\|_2^2 \\ &= \sum_i \underbrace{\inf_{\boldsymbol{X}_{\text{supp}(\boldsymbol{X}_{:,i}),i}} \left\|\boldsymbol{Y}_{:,i}-\boldsymbol{D}_i\boldsymbol{X}_{\text{supp}(\boldsymbol{X}_{:,i}),i}\right\|_2^2}_{f_i(\boldsymbol{D}) \text{ or } f_i(\boldsymbol{D}_i)}. \end{aligned} \tag{2.6}$$

Since each atomic function involves a simple least squares problem, the optimal $\boldsymbol{X}_{\text{supp}(\boldsymbol{X}_{:,i}),i}$ has a closed-form formula given by

$$\boldsymbol{X}^*_{\text{supp}(\boldsymbol{X}_{:,i}),i} = \boldsymbol{D}_i^\dagger \boldsymbol{Y}_{:,i},$$

where the superscript † denotes the pseudo-inverse.

2.2.1 Regularized SimCO

The main idea of Regularized SimCO lies in the use of an additive penalty term to avoid singularity. An l_2 norm of the sparse coefficients to the original objective function in

dictionary update stage. Consider the objective function in (2.5),

$$\begin{aligned} f_\mu\left(\tilde{\boldsymbol{D}}\right) = & \min_{\boldsymbol{X}\in\mathcal{X}} \|\boldsymbol{D}\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \mu\|\boldsymbol{X}\|_F^2, \\ = & \min_{\boldsymbol{X}\in\mathcal{X}} \left\| \begin{bmatrix} \boldsymbol{Y} \\ \boldsymbol{0} \end{bmatrix} - \begin{bmatrix} \boldsymbol{D} \\ \sqrt{\mu}\boldsymbol{I} \end{bmatrix} \boldsymbol{X} \right\|_F^2. \end{aligned} \tag{2.7}$$

As long as $\mu \neq 0$ ($\mu > 0$ in our case), the block $\mu\boldsymbol{I}$ guarantees the full column rank of $\tilde{\boldsymbol{D}} = \begin{bmatrix} \boldsymbol{D}^T & \mu\boldsymbol{I} \end{bmatrix}^T$. Therefore, with the modified objective function $f_\mu\left(\tilde{\boldsymbol{D}}\right)$, there is no singular point so that gradient descent methods will only converge to stationary points.

This regularization technique is also applicable to MOD [25]. It is verified that this technique effectively mitigates the occurrence of ill-conditioned dictionary although at the same time some stationary points might be generated. To alleviate this problem, one can decrease gradually the regularization parameter μ during the optimization process [25]. In the end μ will decrease to zero. Nevertheless, it is still not guaranteed to converge to a global minimum. The explicit example constructed in the next subsection shows a failure of the Regularized SimCO. As a result, another method to address the singularity issue is introduced below.

2.3 The Singularity Issue in Benchmark Algorithms

According to our simulations (see Section 2.6 for more details), it has been observed that the failures of benchmark dictionary update algorithms are mainly due to singular points in the objective function rather than stationary points. In this section, an explicit example will be constructed to rigorously show how singularity affects the convergence of the dictionary update.

To start, we first formally define the singular dictionaries.

Assumption. Fix a sparsity support of $\boldsymbol{X}$. Assume that $\|\mathrm{supp}\,(\boldsymbol{X}_{:,i})\|_0 < m$, $1 \leq i \leq n$, i.e., all the sub-dictionary $\boldsymbol{D}_i$'s are tall matrices.

Definition 2.1. Given the above Assumption, a dictionary $\boldsymbol{D} \in \mathbb{R}^{m\times d}$ contains singular sub-dictionaries under $\mathrm{supp}\,(\boldsymbol{X})$ if there exists an $i \in [n]$ such that the corresponding sub-dictionary $\boldsymbol{D}_{\mathrm{supp}(\boldsymbol{X}_{:,i})}$ is column rank deficient, or equivalently, the minimum singular

value of $\boldsymbol{D}_{\text{supp}(\boldsymbol{X}_{:,i})}$, denoted as $\lambda_{\min}\left(\boldsymbol{D}_{\text{supp}(\boldsymbol{X}_{:,i})}\right)$, is zero.

This definition is motivated by the following facts.

Proposition 2.2. Given the above Assumption,

1. $f(\boldsymbol{D})$ is continuous for all $\boldsymbol{D}$'s that are not singular.
2. Suppose that $\boldsymbol{D}$ is singular, i.e., $\exists i$ such that $\text{rank}(\boldsymbol{D}_i) < \|\text{supp}(\boldsymbol{X}_{:,i})\|_0$. If among all these is, there exists one i such that $\boldsymbol{Y}_{:,i} \notin \text{span}(\boldsymbol{D}_i)$, then $f(\boldsymbol{D})$ is discontinuous.

The proof is given in Appendix A.1.

Now comes the explicit example. This example is designed in such a way that MOD, K-SVD, SimCO and regularized SimCO will converge to a singular point rather than the global minimum. Though this "hand-made" example may look artificial, the corresponding analysis is applicable to the general case.

In this example, the training sample matrix $\boldsymbol{Y}$ is obtained from $\boldsymbol{Y} = \boldsymbol{D}^*\boldsymbol{X}^*$ (there is no noise) where $\boldsymbol{D}^* \in \mathcal{D}$ and the sparsity patter Ω of $\boldsymbol{X}^*$ is given. To simplify the notations, denote Ω by a binary matrix $\boldsymbol{\Omega}$ where 0 means that the corresponding entry in $\boldsymbol{X}^*$ is zero and one implies otherwise. In particular,

$$\boldsymbol{Y} = \begin{bmatrix} 1 & 0 & 0.7 & 0 \\ 0 & 1 & 0.7 & 0 \\ 0 & 0 & -0.1 & 1 \\ 0 & 0 & -0.1 & 1 \end{bmatrix}, \ \boldsymbol{\Omega} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

Due to the particular structure of this problem, the optimal $\boldsymbol{D}^*$ and $\boldsymbol{X}^*$ can be obtained by using the first three columns of $\boldsymbol{Y}$ and $\boldsymbol{\Omega}$:

$$\boldsymbol{D}^* = \begin{bmatrix} 1 & 0 & 0.7 \\ 0 & 1 & 0.7 \\ 0 & 0 & -0.1 \\ 0 & 0 & -0.1 \end{bmatrix}, \ \boldsymbol{X}^* = \begin{bmatrix} 1 & 0 & 0 & 7 \\ 0 & 1 & 0 & 7 \\ 0 & 0 & 1 & -10 \end{bmatrix}. \tag{2.8}$$

To analyze the behavior of benchmark algorithms [41, 3, 25], we formulate the dictionary update problem in terms of the optimization problem stated in (2.5). To simplify the

analysis, we enforce further structures in the dictionary $\boldsymbol{D}$. Denote the i^{th} codeword of dictionary $\boldsymbol{D}$ by $\boldsymbol{d}_i$. Motivated from the $\boldsymbol{D}^*$ given in (2.8), we assume that

$$\boldsymbol{D}(\epsilon) = \left[\begin{array}{ccc} \boldsymbol{d}_1, & \boldsymbol{d}_2, & \boldsymbol{d}_3(\epsilon) \end{array} \right], \tag{2.9}$$

where $\boldsymbol{d}_1 = [1,\, 0,\, 0,\, 0]^T$, $\boldsymbol{d}_2 = [0,\, 1,\, 0,\, 0]^T$, and $\boldsymbol{d}_3(\epsilon) = \frac{1}{\sqrt{2}}\left[\sqrt{1-\epsilon^2},\, \sqrt{1-\epsilon^2},\, \epsilon,\, \epsilon\right]^T$ where $\epsilon \in [-1,\, 1]$. In this way, the dictionary is parametrized by only one parameter ϵ and the optimization problem is therefore

$$\inf_{\boldsymbol{D}\in\mathcal{D}} f(\boldsymbol{D}) = \inf_{\epsilon} f(\epsilon) = \inf_{\epsilon} \inf_{\boldsymbol{X}\in\mathcal{X}(\Omega)} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F^2. \tag{2.10}$$

It is clear that the global optimum $\epsilon^* = -0.1\sqrt{2}$ and that the dictionary $\boldsymbol{D}(\epsilon)$ is column rank deficient when $\epsilon = 0$.

With the above simplification, the analysis of benchmark algorithms is reduced to analyzing the convergence of a single parameter ϵ. To that end, we denote the initial value of ϵ by ϵ_0. For iterative algorithms, let ϵ_k denote the value of ϵ at the end of the k^{th} iteration. In the rest, we shall show that ϵ_k converges to zero, the singular dictionary, rather than the global optimum $\epsilon^* = -0.1\sqrt{2}$. The proof sketch is provided below while the details are postponed to Appendix A.2.

2.3.1 Maximum Optimal Directions (MOD)

MOD algorithm employs an optimization procedure that the dictionary and the sparse coefficients are alternately updated: fix $\boldsymbol{D}$ and update $\boldsymbol{X}$; then fix $\boldsymbol{X}$ and update $\boldsymbol{D}$. Applying MOD to our example, we derive that when the initial ϵ_0 is not appropriately chosen (in this case, $\epsilon_0 \in (0,\, 1]$), the algorithm will fail. The analysis given in Appendix *A.2* includes the update of ϵ_k via manually handling several least square problems. The result shows that after the iteration k, ϵ_k is given by

$$\epsilon_k = \epsilon_{k-1}\left(1 - 0.07\epsilon_{k-1} - 0.48\epsilon_{k-1}^2 + o(\epsilon_{k-1}^2)\right). \tag{2.11}$$

where $\epsilon_k \in (0,\, \epsilon_0)$. Note that$\epsilon_k$ is strictly smaller than ϵ_{k-1}. This implies with $k \to \infty$, ϵ_k converges to 0 rather than $-0.1\sqrt{2}$, i.e., the dictionary $\boldsymbol{D}$ does not converge to $\boldsymbol{D}^*$.

2.3.2 K-SVD

We initialize the dictionary with an $\epsilon_0 \in (0, 1]$. Let $\boldsymbol{x}_i^T$ denote the i^{th} row of $\boldsymbol{X}$. K-SVD sequentially update $\left(\boldsymbol{d}_i,\ \boldsymbol{x}_i^T\right)$s, $i \in [3]$, by using SVD. In our setting, $\left(\boldsymbol{d}_1,\ \boldsymbol{x}_1^T\right)$ and $\left(\boldsymbol{d}_2,\ \boldsymbol{x}_2^T\right)$ are optimized and fixed. It suffices to optimize $\left(\boldsymbol{d}_3,\ \boldsymbol{x}_3^T\right)$ only. To do that, one first cancels the effects from other codewords by computing $\boldsymbol{Y}^r = \boldsymbol{Y} - \sum_{i=1}^{2} \boldsymbol{d}_i \boldsymbol{x}_i^T$ and then updates $\left(\boldsymbol{d}_3,\ \boldsymbol{x}_3^T\right)$ by using the strongest singular vectors of $\boldsymbol{Y}^r$ (obtained from SVD). As has been detailed in Appendix *A.2*, the ϵ before and after the k^{th} iteration can be related via (2.11) again.

2.3.3 Primitive and Regularized SimCO

SimCO framework uses the line search methods for updates. For primitive SimCO, the dictionary $\boldsymbol{D}(0)$ is not of full column rank which implies a singular point at $f(0)$. Regularized SimCO, proposed in [25], adds a penalty term $f(\boldsymbol{D})$ with the motivation to make the alternated objective function to be continuous. Such modification alleviates the singularity issue, yet introduces some stationary points.

In the following we consider $\mu\|\boldsymbol{X}_{:,4}\|_F^2$ as the regularized term added to the original objective function in order to facilitate the analysis. Such regularized formulation removes the discontinuity of the objective function and we are able to calculate its derivatives. For a given $\boldsymbol{D}(\epsilon)$, let $\boldsymbol{X}(\epsilon)$ have the same setup as in the MOD case. The regularized objective function is written as

$$\inf_{\epsilon|\boldsymbol{D}\in\mathcal{D}} f_\mu(\epsilon) = \inf_{\epsilon|\boldsymbol{D}\in\mathcal{D}} \inf_{\boldsymbol{X}(\epsilon)} \|\boldsymbol{Y} - \boldsymbol{D}(\epsilon)\boldsymbol{X}(\epsilon)\|_F^2 + \mu\|\boldsymbol{X}_{:,4}(\epsilon)\|_F^2. \tag{2.12}$$

This is a least squares problem and it can be derived with

$$f_\mu(\epsilon) = \begin{cases} 2.02 & \text{when } \mu{=}0,\ \epsilon{=}0 \\ 3 - \frac{2\epsilon^2}{\mu+1-\frac{1-\epsilon^2}{\mu+1}} - 2\left(0.7\sqrt{1-\epsilon^2} - 0.1\epsilon\right)^2 & \text{otherwise.} \end{cases}$$

When $\mu = 0$, the problem degenerates to primitive SimCO. From Appendix *A.2* we know that whenever $\epsilon \in (0,\ 1\sqrt{2}]$, $f_0(\epsilon) < 1 < f_0(0) = 2.02$, and its derivative $f_0'(\epsilon) \geq$

0.28. Therefore we infer that the primitive SimCO update process will finally stagnate at $\epsilon = 0$ if the initial $\epsilon_0 \in (0,\ 1\sqrt{2}]$. We also show in Appendix *A.2* that whenever $\epsilon \in (0,\ 1\sqrt{2}]$ and $\mu \in \left(0,\ \min\left(\sqrt{1+100\epsilon^2},\ \sqrt{2}-1\right)\right)$, there always exist a $\bar{\epsilon} \in (0,\ \epsilon)$ with $f_\mu(\bar{\epsilon}) > f_\mu(\epsilon)$. Hence, line search methods do not make $\boldsymbol{D}(\epsilon_k)$ pass through $\boldsymbol{D}(0)$ as the iteration number $k \to \infty$ to reach the global minimum. Furthermore, as $\mu \to 0$ the dictionary $\boldsymbol{D}$ converges to the singular point as well.

2.4 Smoothing Technique

A smoothing technique is developed to address the singularity issue in this section. This is fundamentally different from the regularization technique proposed in [25] in two aspects. Firstly, as is shown in the explicit example from the last section, the additive penalty term in the regularization technique alleviates the singularity issue but introduces new stationary point, while in the smoothing technique productive terms are added which may not create new stationary point. Secondly, another nature being superior to the regularization technique is, we prove that in the limit case the proposed objective function, as the best lower semi-continuous approximation of the original one, guarantees that the solution set is closed.

2.4.1 Smoothed SimCO

Aiming at the singularity issue, we propose a new idea trying to remove the discontinuity in the objective function $f(\boldsymbol{D})$. Write $f(\boldsymbol{D})$ into a summation of atomic functions.

$$\begin{aligned} f(\boldsymbol{D}) &= \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F^2 \\ &= \sum_i \|\boldsymbol{Y}_{:,i} - \boldsymbol{D}_i \boldsymbol{X}_{\mathrm{supp}(\boldsymbol{X}_{:,i}),i}\|_2^2 \\ &= \sum_i f_i(\boldsymbol{D}_i), \end{aligned} \tag{2.13}$$

where each $f_i(\boldsymbol{D}_i)$ is termed as an atomic function. Let $\mathcal{I}$ be the index set corresponding to the $\boldsymbol{D}_i$'s of full column rank. Define an indicator function $\mathcal{X}_{\mathcal{I}}$ s.t. $\mathcal{X}_{\mathcal{I}}(i) = 1$ if $i \in \mathcal{I}$ and $\mathcal{X}_{\mathcal{I}}(i) = 0$ if $i \in \mathcal{I}^c$. Use $\mathcal{X}_{\mathcal{I}}(i)$ as a multiplicative modulation function and apply it to each $f_i(\boldsymbol{D}_i)$. Then one obtain

$$g_i\left(\lambda_{\min}\left(\boldsymbol{D}_i\right)\right)=\begin{cases}0 & \lambda_{\min}\leq\delta_1^{(i)}\\ 6\left(\frac{\lambda_{\min}-\delta_1^{(i)}}{\delta_2^{(i)}-\delta_1^{(i)}}\right)^5-15\left(\frac{\lambda_{\min}-\delta_1^{(i)}}{\delta_2^{(i)}-\delta_1^{(i)}}\right)^4+10\left(\frac{\lambda_{\min}-\delta_1^{(i)}}{\delta_2^{(i)}-\delta_1^{(i)}}\right)^3 & \delta_1^{(i)}<\lambda_{\min}\leq\delta_2^{(i)}\\ 1 & \lambda_{\min}>\delta_2^{(i)}\end{cases}. \tag{2.14}$$

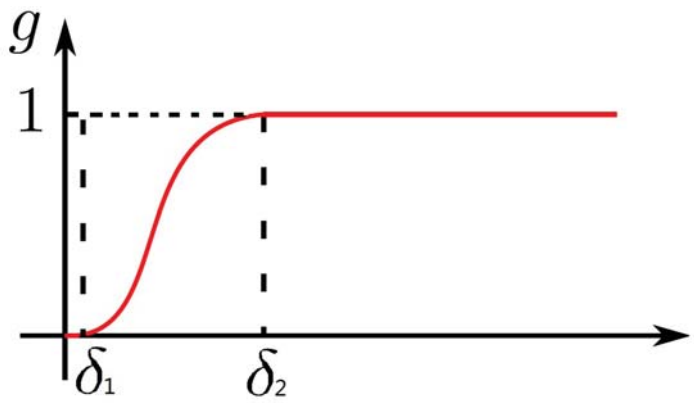

Figure 2.1: A illustrative shape of smoothed function $g\left(\cdot\right)$.

$$\bar{f}(\boldsymbol{D})=\sum_i f_i(\boldsymbol{D}_i)\mathcal{X}_{\mathcal{I}}\left(i\right)=\sum_{i\in\mathcal{I}}f_i(\boldsymbol{D}_i). \tag{2.15}$$

This new function $\bar{f}$ is actually the best possible lower semi-continuous approximation of f and there is no new stationary point created.

Note that $\mathcal{X}_{\mathcal{I}}\left(i\right)$ is a step function of $\boldsymbol{D}_i$. Therefore finding the derivative of $\bar{f}(\boldsymbol{D})$ is an ill-posed problem. The SimCO framework could not apply to $\bar{f}(\boldsymbol{D})$. In addition, when $\boldsymbol{D}$ is not singular but ill-conditioned, $\bar{f}(\boldsymbol{D})=f(\boldsymbol{D})$ will still be caught into slow convergence. To address the above two issues, we propose a continuous function $g_i\left(\boldsymbol{D}_i\right)$ and hence we reformulate the dictionary update stage to

$$\begin{aligned}\inf_{\boldsymbol{D}\in\mathcal{D}}\tilde{f}(\boldsymbol{D})&=\inf_{\boldsymbol{D}\in\mathcal{D}}\inf_{\boldsymbol{X}\in\mathcal{X}(\Omega)}\sum_i\|\boldsymbol{Y}_{:,i}-\boldsymbol{D}_i\boldsymbol{X}_{\boldsymbol{\Omega}(:,i),i}\|_F^2\cdot g_i\left(\boldsymbol{D}_i\right)\\&=\inf_{\boldsymbol{D}\in\mathcal{D}}\sum_i\underbrace{f_i(\boldsymbol{D}_i)\cdot g_i\left(\boldsymbol{D}_i\right)}_{\tilde{f}_i(\boldsymbol{D}_i)},\end{aligned} \tag{2.16}$$

where the shape of g_i is given in Figure *2.1*.

Let $\boldsymbol{\delta}_i=\left(\delta_1^{(i)},\delta_2^{(i)}\right)$, where $0\leq\delta_1^{(i)}\leq\delta_2^{(i)}$ are two thresholds. The continuous function $g_i\left(\boldsymbol{D}_i\right)$ is constructed as: 1) $g_i\left(\boldsymbol{D}_i\right)=0$ when $\lambda_{\min}\left(\boldsymbol{D}_i\right)<\delta_1^{(i)}$; 2) $g_i\left(\boldsymbol{D}_i\right)=1$ when

$\lambda_{\min}(\boldsymbol{D}_i) > \delta_2^{(i)}$; 3) $g_i(\boldsymbol{D}_i)$ is monotonically increasing; 4) $g_i(\boldsymbol{D}_i)$ is second order differentiable. Function $\lambda_{\min}(\boldsymbol{D}_i)$ indicates whether $\boldsymbol{D}_i$ is ill-conditioned. Here we provide a continuously second order differentiable polynomial $g_i(\boldsymbol{D}_i) = g_i(\lambda_{\min}(\boldsymbol{D}_i))$ in (2.14) as the smooth curve between $\left(\delta_1^{(i)}, \delta_2^{(i)}\right)$.

Note that the proposed polynomial is one of the proper choices since $g_i(\boldsymbol{D}_i)$ is continuous and second order differentiable in $\lambda_{\min} \in \mathbb{R}$. Usually $\boldsymbol{\delta}_i$'s are different for different i and the specific choices of them will be explained in Section 2.4.2. A formal description of $\tilde{f}(\boldsymbol{D})$ is given in Theorem *2.3.*

Theorem 2.3. Consider the smoothed objective function $\tilde{f}$ and the original objective function f defined in (2.16) and (2.13) respectively.

1. When $\delta_2^{(i)} > \delta_1^{(i)} > 0$, $\forall i$, $\tilde{f}(\boldsymbol{D})$ is continuous.
2. Consider the limit case where $\delta_1^{(i)}, \delta_2^{(i)} \to 0$ with $\delta_2^{(i)} > \delta_1^{(i)} > 0$, $\forall i$. The followings hold.
   (a) $\tilde{f}(\boldsymbol{D})$ and $f(\boldsymbol{D})$ differ only at the singular points.
   (b) $\tilde{f}(\boldsymbol{D})$ is the best possible lower semi-continuous approximation of $f(\boldsymbol{D})$.
3. *F*or any $a \in \mathbb{R}^+$, define the lower level set of a function $h(\boldsymbol{D})$: $\mathcal{D}_h(a) = \{\boldsymbol{D} : h(\boldsymbol{D}) \leq a\}$. It is provable that when $\delta_1^{(i)} = \delta_2^{(i)} \to 0$, $\mathcal{D}_{\tilde{f}}(a)$ is the closure of $\mathcal{D}_f(a)$.

The proof of Theorem A.3.1 is given in Appendix A.3. It is clear from the fact that $g_i(\boldsymbol{D}_i) = 1$ for all non-singular $\boldsymbol{D}_i$. Theorem A.3.3 specifies the property of the smoothed technique (Theorem A.3.2(b)) from the perspective of set theory. The theorem actually shows that the global optimum of $\tilde{f}(\boldsymbol{D})$ is also the global optimum of $f(\boldsymbol{D})$.

Functions $g_i(\boldsymbol{D}_i)$s smooth the discontinuity in $f(\boldsymbol{D})$. We call $g_i(\boldsymbol{D}_i)$ as a smoothing function. Hence this technique is termed as *Smoothed SimCO*.

The effect of adding $g_i(\boldsymbol{D}_i)$'s, intuitively speaking, is to open "tunnels" on $f(\boldsymbol{D})$ for the optimization process to pass through. The smaller $\delta_2^{(i)}$'s are, the better the function $\tilde{f}(\boldsymbol{D})$ approximates the function $f(\boldsymbol{D})$. Consequently the narrower the tunnels, and the slower the convergence rate. The function of threshold $\delta_1^{(i)}$'s are to eradicate the large computational cost when $\boldsymbol{D}_i$ is very ill-conditioned.

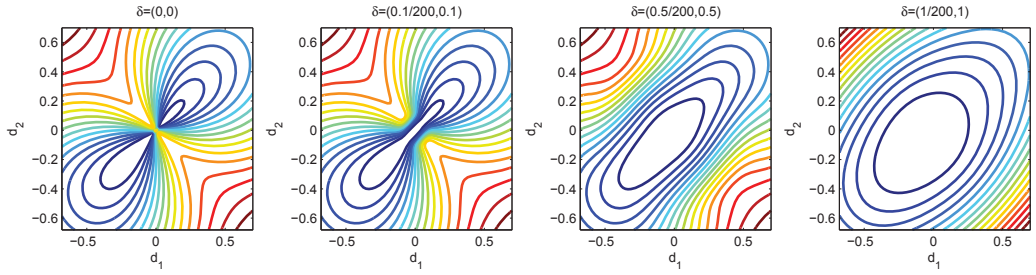

Figure 2.2: Four pairs of $\boldsymbol{\delta}$s selected from left to right are $\boldsymbol{\delta} = (0, 0)$, $\boldsymbol{\delta} = (0.1/200, 0.1)$, $\boldsymbol{\delta} = (0.5/200, 0.5)$ and $\boldsymbol{\delta} = (1/200, 1)$. The global minimum of the original objective function locates at $d_1 = d_2 = -0.1$. A singular point is found at $d_1 = d_2 = 0$ and line search methods are not able to make the dictionary updating pass through. With the increasing of $\boldsymbol{\delta}$, the area around the singular point is gradually replaced by a channel. Larger $\boldsymbol{\delta}$ opens wider 'tunnel' around the singular point but may affecting larger surrounding area. The area not closed to the singularity does not change.

In practice, we always choose a $\delta_2^{(i)} > \delta_1^{(i)} > 0$. The effect of positive $\delta_1^{(i)}$ and $\delta_2^{(i)}$, roughly speaking, is to remove the barriers created by singular points, and replace them with "tunnels", whose widths are controlled by $\delta_1^{(i)}$ and $\delta_2^{(i)}$, to allow the optimization process to pass through. The smaller the $\delta_2^{(i)}$'s are, the better $\tilde{f}$ approximates f, but the narrower the tunnels are, and the slower the convergence rate will be. As a result, the thresholds $\delta_2^{(i)}$'s should be properly chosen. Usually $\delta_1^{(i)}$'s are set to be not too large. We propose the relation $\delta_1^{(i)} = \delta_2^{(i)}/200$ for all $i \in [n]$ and use it in our empirical tests. In the following subsection we will discusses a particular way to choose $\delta_2^{(i)}$'s.

2.4.2 A Brief Discussion on the Choice of the Upper Thresholds

We first give an example to intuitively show the effect of parameter $\boldsymbol{\delta}_i$s to the convergence rate. Consider the explicit example proposed in Section *2.3*. With slight difference, here the 3^{rd} column of the dictionary is initialized

$$\boldsymbol{d}_3 = \begin{bmatrix} \sqrt{\frac{1-d_1^2-d_2^2}{2}} & \sqrt{\frac{1-d_1^2-d_2^2}{2}} & d_1 & d_2 \end{bmatrix}^T,$$

where $d_1^2 + d_2^2 \leq 1$, $d_1, d_2 \in \mathbb{R}^+$. Refer to book [77], we remark that a narrower "tunnel" in the objective function may cost more numerical steps to pass through and each step has large fluctuation in both size and direction. In other words, narrower "tunnels" lead to slower convergence rate. We sample four pairs of $\boldsymbol{\delta}$ and plot the contours of $\tilde{f}(\boldsymbol{D})$ against (d_1, d_2) in Figure *2.2*.

The above example implies the importance of choosing parameters $\boldsymbol{\delta}_i$'s. Too small $\boldsymbol{\delta}_i$'s may cause slow convergence rate and too large $\boldsymbol{\delta}_i$'s may cause finding wrong global optimum ($\tilde{f}(\boldsymbol{D}) \neq f(\boldsymbol{D})$ at the global minimum. The contour is changed in a larger extent and the global minimum is moved to somewhere else). Next we will use random matrix theory to give a formal discussion about the selection of $\boldsymbol{\delta}_i$'s.

We first argue that for different i, $\boldsymbol{\delta}_i$ should be different. Consider the case where $m = 100$, $\|\mathrm{supp}\,(\boldsymbol{X}_{:,1})\|_0 = 2 \ll m$ and $\|\mathrm{supp}\,(\boldsymbol{X}_{:,2})\|_0 = m$. Suppose that the dictionary $\boldsymbol{D}$ is randomly generated from the uniform distribution on $\mathcal{D}$.[1] It is clear that with high probability $\lambda_{\min}(\boldsymbol{D}_1)$ centers around 1 but $\lambda_{\min}(\boldsymbol{D}_2)$ is close to zero. Intuitively, the thresholds $\boldsymbol{\delta}_i$s should be chosen such that the modulation functions take effect (i.e., $g_i < 1$) with small but positive probability.

Generally speaking, it is difficult to quantify the probability of $\lambda_{\min}(\boldsymbol{D}_i)$s. Nevertheless, when m and $\|\mathrm{supp}\,(\boldsymbol{X}_{:,i})\|_0$ approaches infinity with a constant ratio, the distribution of $\lambda_{\min}$ will converge a distribution only dependent on the ratio $\|\mathrm{supp}\,(\boldsymbol{X}_{:,i})\|_0 / m$. In particular,

Proposition 2.4. *For any given m and $s_i = \|\mathrm{supp}\,(\boldsymbol{X}_{:,i})\|_0$ such that $s_i \leq m$, define $\mathcal{D}_{m,s_i}$ as the set containing all the matrices with unit norm columns. Randomly generate $\boldsymbol{D}_i$ from the uniform distribution on $\mathcal{D}_{m,s_i}$. Then as m, $s_i \to \infty$ simultaneously with $s_i/m \to c_i < 1$, the minimum singular values $\lambda_{\min}(\boldsymbol{D}_i)$ converges to $\tau_i \triangleq 1 - \sqrt{c_i}$ in probability.*

Proposition 2.5. *For any randomly generate $\boldsymbol{D}_i$ from the uniform distribution on $\mathcal{D}_{m,s_i}$. For each fixed $t > 0$, the following deviation bound holds*

$$\boldsymbol{P}\left(\lambda_{\min}(\boldsymbol{D}_i) < \tau_i + o(1) - t\right) \leq e^{-mt^2/2}.$$

The term $o(1)$ is a small and tending to zero as $m \to \infty$. The detailed proofs of the equivalent propositions (*Proposition 4 & 5*) can be found in [90] and [65] respectively. Though the results are asymptotic, they provide a good approximation for finite m and s_i. On the other hand, from *Proposition 5* we have the following remark.

Remark 2.6. When m and s_i are finite. For a random generalized true dictionary $\boldsymbol{D}_i$,

[1] The uniform distribution is well defined as $\mathcal{D}$ is a compact manifold.

$\boldsymbol{P}\left(\lambda_{\min}\left(\boldsymbol{D}_i\right) < \tau_i\right)$ increases with the decrease of m and s_i. Thus the true dictionary $\boldsymbol{D}_{true}$ may more likely have its minimum singular value smaller than $1 - \sqrt{c_i}$.

In practice, m and s_i are finite. To make the smoothed objective function $\tilde{f}(\boldsymbol{D})$ be as far as equivalent to $f(\boldsymbol{D})$ around the global optimum, in our implementation we make a relaxation on the threshold settings. We define constant $\alpha \in (0, 1)$ independent of i and set $\delta_2^{(i)} = \alpha\tau_i$.

2.5 Implementation of Smoothed SimCO

In this section, we present a Newton CG implementation to minimize the objective function $\tilde{f}\left(\boldsymbol{D}\right)$. As a second order line search method, Newton CG presents efficient performance in finding the optimum compared with other first order methods, such as gradient descent or conjugate gradient method. On the other hand, in actual applications it is generally possible to find a good warm starting point to initialize dictionary learning process, therefore significantly alleviate the drawback where the Newton CG being overwhelmed when the initial point is far from the optimum.

In order to facilitate the discussion, some useful notations are given as follows. We give function $h\left(\boldsymbol{D}\right) \in \mathbb{R}$, $\boldsymbol{D} \in \mathbb{R}^{m\times d}$ and $\boldsymbol{\eta} \in \mathbb{R}^{m\times d}$. Denote the gradient of $h\left(\boldsymbol{D}\right)$ by $\nabla h\left(\boldsymbol{D}\right)$. Denote the directional derivative of function $h\left(\boldsymbol{D}\right)$ along direction $\boldsymbol{\eta}$ [56] by $\nabla_{\boldsymbol{\eta}} h\left(\boldsymbol{D}\right)$. Furthermore we have $\nabla_{\boldsymbol{\eta}} h\left(\boldsymbol{D}\right) = \operatorname{trace}\left(\nabla h\left(\boldsymbol{D}\right), \boldsymbol{\eta}\right)$ [82].

Most optimization methods are based on line search strategies. The dictionaries at the beginning and the end of an iteration, denoted by $\boldsymbol{D}^{(k)}$ and $\boldsymbol{D}^{(k+1)}$ respectively, can be related via $\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} + \alpha^{(k)}\boldsymbol{\eta}^{(k)}$, where $\alpha^{(k)}$ is an appropriately chosen step size and $\boldsymbol{\eta}^{(k)}$ is the line search direction. Usual determinations of the step size $\alpha^{(k)}$ include *Armijo condition* and *Golden selection* [77]. The search direction $\boldsymbol{\eta}^{(k)}$ has many choices [77, 37] as well. In fact, the choice of $\boldsymbol{\eta}^{(k)}$ plays the key role in the convergence rate of the whole algorithm. Generally speaking, a Newton direction is preferred (compared with the gradient descent direction) [77]. In a standard Newton method, the computation of the Newton direction requires the *Hessian* of the objective function. Note that in the problem at hand, the dictionary $\boldsymbol{D}$ has size $m \times d$ and hence the corresponding Hessian is of size $md \times md$. To compute the Hessian explicitly, it requires large computational resources as

well as extra-ordinary storage resources. Such cost is enough to offset the advantage in convergence rate.

By contrast, Newton CG provides a means to compute the Newton direction without explicitly computing the Hessian. More specifically, the Newton CG method starts with the gradient descent direction $\boldsymbol{\eta}^{(0)}$ and iteratively refines it towards the Newton direction. In each line search step, instead of computing the Hessian $\nabla^2 \tilde{f}(\boldsymbol{D}) \in \mathbb{R}^{md \times md}$ explicitly, one only needs to compute $\nabla_{\boldsymbol{\eta}} \left(\nabla \tilde{f}(\boldsymbol{D}) \right) \in \mathbb{R}^{m \times d}$. The required computational and storage resources are therefore much less than working with the Hessian.

We consider efficient computations related to the Hessian matrix of an atomic function $f_i(\boldsymbol{D}_i) g_i(\boldsymbol{D}_i)$ of $\tilde{f}(\boldsymbol{D})$ and provide a detailed derivation of $\nabla_{\boldsymbol{\eta}} (\nabla (f_i(\boldsymbol{D}_i) g_i(\boldsymbol{D}_i)))$ under the assumption that $\boldsymbol{D}_i$ is a full column rank tall matrix.

We denote $f_i(\boldsymbol{D}_i)$ and $g_i(\boldsymbol{D}_i)$ by f_i and g_i respectively for simplification. Write $\nabla_{\boldsymbol{\eta}} (\nabla (f_i g_i))$ as the summation

$$\begin{aligned}
&\nabla_{\boldsymbol{\eta}} (\nabla (f_i \cdot g_i)) \\
=&\nabla_{\boldsymbol{\eta}} (\nabla f_i) g_i + f_i \nabla_{\boldsymbol{\eta}} (\nabla g_i) + \nabla f_i \nabla_{\boldsymbol{\eta}} g_i + \nabla_{\boldsymbol{\eta}} f_i \nabla g_i \\
=&\nabla_{\boldsymbol{\eta}} (\nabla f_i) g_i + f_i \nabla_{\boldsymbol{\eta}} (\nabla g_i) + \nabla f_i \mathrm{trace} (\nabla g_i, \boldsymbol{\eta}) + \mathrm{trace} (\nabla f_i, \boldsymbol{\eta}) \nabla g_i
\end{aligned} \tag{2.17}$$

Essentially in equation (2.17) there are four terms to be computationally derived individually: ∇f_i, $\nabla_{\boldsymbol{\eta}} (\nabla f_i)$, ∇g_i and $\nabla_{\boldsymbol{\eta}} (\nabla g_i)$. We will sequentially derive the four terms in the following and respectively formulate them in equation (2.18), (2.19), (2.21) and (2.22).

First we consider term ∇f_i and $\nabla_{\boldsymbol{\eta}} (\nabla f_i)$. They are relatively easy to be derived. The optimal

$$\boldsymbol{x}_i^* = \arg\min_{\boldsymbol{x}_i} \|\boldsymbol{y}_i - \boldsymbol{D}\boldsymbol{x}_i\|_2^2 = \boldsymbol{D}_i^\dagger \boldsymbol{y}_i.$$

Having that $\frac{\partial f}{\partial \boldsymbol{x}_i} |_{\boldsymbol{x}_i^*} = \boldsymbol{0}$, ∇f_i can be written as

$$\nabla f_i = \frac{\partial f}{\partial \boldsymbol{D}_i} + \frac{\partial f}{\partial \boldsymbol{x}_i^*} \frac{\partial \boldsymbol{x}_i^*}{\partial \boldsymbol{D}_i} = -2(\boldsymbol{y}_i - \boldsymbol{D}_i \boldsymbol{x}_i^*) \boldsymbol{x}_i^{*T} + \boldsymbol{0}. \tag{2.18}$$

To compute $\nabla_{\boldsymbol{\eta}} (\nabla f_i)$, we have

$$\begin{aligned}
\nabla_{\boldsymbol{\eta}} (\nabla f_i) =& 2 \nabla_{\boldsymbol{\eta}} (\boldsymbol{D}_i \boldsymbol{x}_i^* - \boldsymbol{y}_i) \boldsymbol{x}_i^{*T} + 2 (\boldsymbol{D}_i \boldsymbol{x}_i^* - \boldsymbol{y}_i) \nabla_{\boldsymbol{\eta}} \boldsymbol{x}_i^{*T} \\
=& 2 \boldsymbol{\eta} \boldsymbol{x}_i^* \boldsymbol{x}_i^{*T} + 2 \boldsymbol{D}_i \nabla_{\boldsymbol{\eta}} \boldsymbol{x}_i^* \boldsymbol{x}_i^{*T} + 2 (\boldsymbol{D}_i \boldsymbol{x}_i^* - \boldsymbol{y}_i) \nabla_{\boldsymbol{\eta}} \boldsymbol{x}_i^{*T},
\end{aligned} \tag{2.19}$$

where $\nabla_{\boldsymbol{\eta}} \boldsymbol{x}_i^*$ is obtained by,

$$\nabla_{\boldsymbol{\eta}} \boldsymbol{x}_i^* = -(\boldsymbol{D}_i^T \boldsymbol{D}_i)^{-1} \left(\left(\boldsymbol{D}_i^T \boldsymbol{\eta} + \boldsymbol{\eta}^T \boldsymbol{D}_i \right) \boldsymbol{D}_i^\dagger - \boldsymbol{\eta}^T \right) \boldsymbol{y}. \tag{2.20}$$

Now we derive the formulae of ∇g_i and $\nabla_{\boldsymbol{\eta}} (\nabla g_i)$. Since g_i is a function of $\lambda_r = \lambda_{\min}(\boldsymbol{D}_i)$, where $\boldsymbol{D}_i \in \mathbb{R}^{m \times r}$, we suppose that all other non-zero singular values are strongly larger than λ_r, i.e. $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_{r-1} > \lambda_r$ so as to guarantee that the 1st and 2nd derivative of $\lambda_{\min}$ is well defined (Otherwise see Remark (2.7), (2.8)). In formula (2.17), when $\lambda_r \leq \delta_1$ or $\lambda_r > \delta_2$, both ∇g_i and $\nabla_{\boldsymbol{\eta}} (\nabla g_i)$ are zero. When $\delta_1 < \lambda_r \leq \delta_2$,

$$\nabla g_i = \frac{\mathrm{d} g_i}{\mathrm{d} \lambda_r} \cdot \nabla \lambda_r, \tag{2.21}$$

$$\nabla_{\boldsymbol{\eta}} (\nabla g_i) = \frac{\mathrm{d}^2 g_i}{\mathrm{d} \lambda_r^2} \cdot \mathrm{trace}(\nabla \lambda_r, \boldsymbol{\eta}) \cdot \nabla \lambda_r + \frac{\mathrm{d} g_i}{\mathrm{d} \lambda_r} \cdot \nabla_{\boldsymbol{\eta}} (\nabla \lambda_r). \tag{2.22}$$

Note that g_i is a polynomial of λ_r in this interval, which means the computations of $\frac{\mathrm{d} g_i}{\mathrm{d} \lambda_r}$ and $\frac{\mathrm{d}^2 g_i}{\mathrm{d} \lambda_r^2}$ are straight forward. Therefore the key step to compute (2.21) and (2.22) is to determine $\nabla \lambda_r$ and $\nabla_{\boldsymbol{\eta}} (\nabla \lambda_r)$. In [79] one provides a smart way to obtain the calculation formula of $\nabla \lambda_r$ and $\nabla_{\boldsymbol{\eta}} (\nabla \lambda_r)$, nevertheless it only applies to the cases when $\boldsymbol{D}_i$ is a square matrix. In the following we base on the idea and extend the derivation to more general non-squared matrices. Note that the extension is the crux of determining the formulae of ∇g_i and $\nabla_{\boldsymbol{\eta}} (\nabla g_i)$. Yet in order to maintain the continuity of the contents, we arrange to list the main result below and leave the derivations to Appendix A.4.

$$\nabla \lambda_r(\boldsymbol{D}_i) = \frac{\partial \lambda_r}{\partial \boldsymbol{D}_i} = \boldsymbol{U}_{:,r} \boldsymbol{V}_{:,r}^T. \tag{2.23}$$

In terms of determining term $\nabla_{\boldsymbol{\eta}} (\nabla \lambda_r)$, computations under traditional Cartesian coordinates are tedious and time consuming. To mitigate the defects, we compute $\nabla_{\boldsymbol{\eta}} (\nabla \lambda_r)$, under our generated basis $\mathcal{B} = \left\{ \boldsymbol{B}^{ij} = \boldsymbol{U}_{:,i} \boldsymbol{V}_{:,j}^T,\ i \in [1, m],\ j \in [1, n] \right\}$. Consider the singular value decomposition of $\boldsymbol{\eta} = \boldsymbol{U} \boldsymbol{S} \boldsymbol{V}^T$. In Appendix A.4 we show that $\nabla_{\boldsymbol{\eta}} (\nabla \lambda_r)$ can be represented as a summation of only $(m + n - 2)$ terms on basis $\mathcal{B}$ (instead of mn terms

on traditional Cartesian coordinates):

$$\nabla_{\boldsymbol{\eta}} \left(\nabla \lambda_r \right) = \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{U}_{:,r} \right) \boldsymbol{V}_{:,r}^T + \boldsymbol{U}_{:,r} \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{V}_{:,r}^T \right). \tag{2.24}$$

where

$$\left(\nabla_{\boldsymbol{\eta}} \boldsymbol{U}_{:,r} \right) \boldsymbol{V}_{:,r}^T = \sum_{k=1}^{r-1} \frac{\lambda_r \boldsymbol{S}_{kr} + \lambda_k \boldsymbol{S}_{rk}}{\lambda_r^2 - \lambda_k^2} \boldsymbol{B}^{kr} + \sum_{k=r+1}^{m} \frac{\boldsymbol{S}_{kr}}{\lambda_r} \boldsymbol{B}^{kr}.$$

and

$$\boldsymbol{U}_{:,r} \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{V}_{:,r}^T \right) = \sum_{k=1}^{r-1} \frac{\lambda_r \boldsymbol{S}_{rk} + \lambda_k \boldsymbol{S}_{kr}}{\lambda_r^2 - \lambda_k^2} \cdot \boldsymbol{B}^{rk}.$$

Remark 2.7. When $\nabla \lambda_r = 0$, the objective function is not differentiable at $\boldsymbol{D}_i$. However, we know that $g_i = 0$ and $\frac{\mathrm{d}g}{\mathrm{d}\lambda_r} = 0$, one may set $\nabla f_i = 0$.

Remark 2.8. If the minimum singular value of $\boldsymbol{D}_i$ is a repetitive singular value, i.e., there exists more than one equally minimum singular value, then $\nabla \lambda_r$ is not well defined. However this happens with probability zero when the dictionary is randomly generated from the uniform distribution on $\mathcal{D}_{m,r} = \left\{ \boldsymbol{D}_i \in \mathbb{R}^{m \times r} : \ \boldsymbol{D}_i^T \boldsymbol{D}_i = \boldsymbol{I}_r \right\}$. Furthermore, even this happens during the optimization, directly applying $\nabla \lambda_r = \boldsymbol{U}_{:,r} \boldsymbol{V}_{:,r}^T$ does not introduce any practical issue in our simulations.

We present an outline of the Newton CG method (Algorithm 2.1) for the Smoothed SimCO. The line search method in conjunction with the Newton CG method is presented in Algorithm 2.2.

2.6 Empirical Tests

Definition 2.9. [25] Define the *condition number* of a dictionary $\boldsymbol{D}$ as:

$$\kappa \left(\boldsymbol{D} \right) = \max_{i \in [n]} \frac{\lambda_{\max} \left(\boldsymbol{D}_i \right)}{\lambda_{\min} \left(\boldsymbol{D}_i \right)},$$

where $\lambda_{\max} \left(\boldsymbol{D}_i \right)$ and $\lambda_{\min} \left(\boldsymbol{D}_i \right)$ represent the maximum and the minimum singular value of the sub-dictionary $\boldsymbol{D}_i$ respectively.

Algorithm 2.1 The Newton CG algorithm for Smoothed SimCO dictionary update: find the search direction.

Input: $\boldsymbol{D}$; **Output**: $\boldsymbol{\eta}$.
Define: $\mathcal{P}\left(\boldsymbol{\eta}_{:,i}\right) = (\boldsymbol{I} - \boldsymbol{D}_{:,i}\boldsymbol{D}_{:,i}^T)\boldsymbol{\eta}_{:,i}$, δ_2 defined in (2.6) and $\delta_2 = 200\delta_1$.
For $k = 0, 1, 2, ...$

Define tolerance $\epsilon_k = \min\left(0.5, \sqrt{\left\|\nabla\tilde{f}\right\|}\right)\left\|\nabla\tilde{f}\right\|$, where $\tilde{f}$ is defined in (2.16).
Set $\boldsymbol{z}_0 = \boldsymbol{0}$, $\boldsymbol{r}_0 = \nabla\tilde{f}$, $\boldsymbol{d}_0 = -\boldsymbol{r}_0 = -\nabla\tilde{f}$.
For $j = 0, 1, 2, ...$

Set $\boldsymbol{H}_j = \nabla_{\boldsymbol{d}_j}\left(\nabla\tilde{f}\right)$.
$\forall i$, let $\left(\boldsymbol{H}_j\right)_{:,i} = \mathcal{P}\left(\left(\boldsymbol{H}_j\right)_{:,i}\right)$.
If $\operatorname{tr}\left(\boldsymbol{d}_j^T\boldsymbol{H}_j\right) \leqslant 0$
If $j = 0$
return $\boldsymbol{\eta} = -\nabla\tilde{f}$.
else
return $\boldsymbol{\eta} = \boldsymbol{z}_j$.
Set $\alpha_j = \operatorname{tr}\left(\boldsymbol{r}_j^T\boldsymbol{r}_j\right)/\operatorname{tr}\left(\boldsymbol{d}_j^T\boldsymbol{H}_j\right)$.
Set $\boldsymbol{r}_{j+1} = \boldsymbol{r}_j + \alpha_j\boldsymbol{H}_j$.
If $\|\boldsymbol{r}_{j+1}\| < \epsilon_k$
return $\boldsymbol{\eta} = \boldsymbol{z}_{j+1}$.
Set $\beta_{j+1} = \operatorname{tr}\left(\boldsymbol{r}_{j+1}^T\boldsymbol{r}_{j+1}\right)/\operatorname{tr}\left(\boldsymbol{r}_j^T\boldsymbol{r}_j\right)$.
Set $\boldsymbol{d}_{j+1} = -\boldsymbol{r}_{j+1} + \beta_{j+1}\boldsymbol{d}_j$.
end
$\forall i$, let $\boldsymbol{\eta}_{:,i} = \mathcal{P}\left(\boldsymbol{\eta}_{:,i}\right)$.

In this section, we numerically test the performance of the Smoothed SimCO. We firstly illustrate a particular example which is considered as a difficult case for dictionary update and use it to show some index along updating process. Then in the next subsection we test the performance of Smoothed SimCO by giving various number of training samples. We also show the successful rate of the proposed algorithm under a large number of trails. In addition, we do all the same tests for MOD, K-SVD and Regularized SimCO for the propose of performance comparisons.

2.6.1 A Difficult Case: Show the Superiority of Smoothed SimCO

In this subsection, we choose a particular example to show the power of the proposed Smoothing technique. The main argument is that in this example mainstream algorithms including MOD, K-SVD and Primitive/Regularized SimCO converge to a singular dictionary. Smoothed SimCO, meanwhile, breaks the obstacle and successfully converge to a global optimum. In this example, the training samples $\boldsymbol{Y} \in \mathbb{R}^{16\times 66}$ are generated via

Algorithm 2.2 Line search method on deciding the step size for Smoothed SimCO dictionary update

Input: $f_0 = f(\boldsymbol{D})$, $\nabla f_0 = \nabla f(\boldsymbol{D})$,$\bar{\boldsymbol{\eta}}$, $\boldsymbol{D}^0 = \boldsymbol{D}$.
Output: $\boldsymbol{D} = \boldsymbol{D}^k$.
Initialize $t = 1$, $c = 1e-6$.
For $k = 1, 2, 3, ...$
Do compact SVD $\forall i$, $\bar{\boldsymbol{\eta}}_{:,i} = \boldsymbol{U}_i \boldsymbol{\Sigma}_i \boldsymbol{V}_i^T$.
Compute $\forall i$, $\boldsymbol{D}_{:,i}^k = (\boldsymbol{D}_{:,i}^{k-1} \boldsymbol{V}_i, \boldsymbol{U}_i) \begin{pmatrix} cos\,(\boldsymbol{\Sigma}_i) \\ sin\,(\boldsymbol{\Sigma}_i) \end{pmatrix} \boldsymbol{V}_i^T$. (Referring to Theorem 2.3 in [37])
If $f_k \leqslant f_{k-1} + c \cdot t \cdot \nabla f_{k-1}^T \bar{\boldsymbol{\eta}}$ (Armijo condition in [77])
return $\boldsymbol{D}^k$.
Set $t^k = 0.8 t^{k-1}$.
end

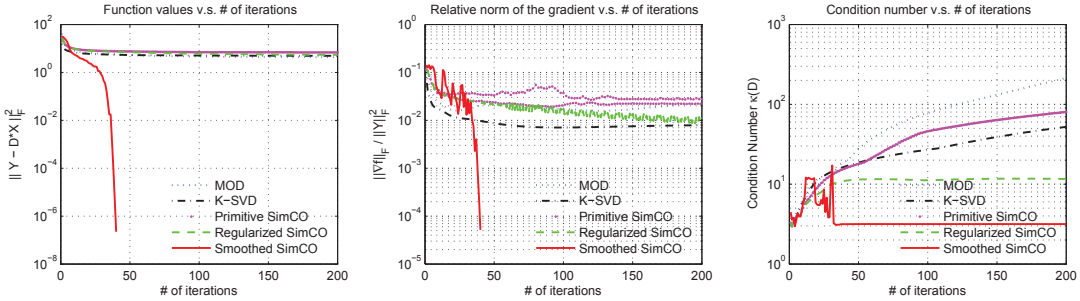

Figure 2.3: Starting with the same point, the convergence behavior of MOD, K-SVD, Primitive SimCO, Regularized SimCO and Smoothed SimCO are different. In this particular example, only Smoothed SimCO avoids converging to a singular point.

$\boldsymbol{Y} = \boldsymbol{D}_{\text{true}} \boldsymbol{X}_{\text{true}}$, where $\boldsymbol{D} \in \mathbb{R}^{16 \times 32}$ and $\boldsymbol{X} \in \mathbb{R}^{32 \times 66}$. Each column of $\boldsymbol{X}$ contains 4 non-zero entries. We assume the true sparsity pattern Ω of $\boldsymbol{X}$ is known a priori, and implement dictionary update algorithms. This assumption eliminate the probability that singularity issue is caused in the sparse coding stage. We start all the algorithms with the same particular choice of the dictionary $\boldsymbol{D}_0 \in \mathcal{D}$. For Regularized SimCO, we set the regularized parameter $\mu = 0.01$ [25].

Three indexes are considered in this test: $f(\boldsymbol{D})$, $\|\nabla f(\boldsymbol{D})\| / \|\boldsymbol{Y}\|_F^2$ and $\kappa(\boldsymbol{D})$. The results are demonstrated in Figure 2.3. The condition number for the true dictionary $\kappa(\boldsymbol{D}_{\text{true}}) \approx 3.17$. It is worth noting that for fair comparison in this test we consider terms $f(\boldsymbol{D})$ rather than $\tilde{f}(\boldsymbol{D})$ for Smoothed SimCO. This is reasonable since after all the smoothing function g is a technique handling the singularity issue and is equivalent to the objective function $f(\boldsymbol{D})$ when it is closed to the global optimum.

From the results shown in Figure 2.3, we comment as follows:

- From the objective function $f(\boldsymbol{D})$ we see that except Smoothed SimCO, the rest of the algorithms do not converge to zero (i.e., the global optimum), but stagnate at some other points instead.

- From the gradient term $\|\nabla f(\boldsymbol{D})\| / \|\boldsymbol{Y}\|_F^2$, we further analyze that Smoothed SimCO quickly decrease to below 10^{-4}. Plus the phenomenon shown in the $f(\boldsymbol{D})$ figure, we conclude that the Smoothed SimCO does find the global optimum for this example. Regularized SimCO, however stagnates at some stationary point on $f_\mu(\boldsymbol{D})$. If we consider the gradient term $\|\nabla f_\mu(\boldsymbol{D})\| / \|\boldsymbol{Y}\|_F^2$, we may see it is approaching to zero. And on objective function $f(\boldsymbol{D})$, it exhibits a slow decreasing in the gradient against the number of iterations.

- The condition number $\kappa(\boldsymbol{D})$ best describes the singularity issue. For MOD, K-SVD and Primitive SimCO, with the increase of iteration number, their $\kappa(\boldsymbol{D})$s rapidly increase to another magnitude order, which clearly reflects that they have been caught into the singularity issue. Regularized SimCO, however, almost stands still at a stationary point on $f_\mu(\boldsymbol{D})$, thus its $\kappa(\boldsymbol{D})$ remains around some value. Smoothed SimCO since changed the objective function $f(\boldsymbol{D})$ by reweighing all its summands. Numerically the change of its $\kappa(\boldsymbol{D})$ against the number of iteration is volatile as expected. After 40 iterations Smoothed SimCO finds the global optimum, thus afterwords its $\kappa(\boldsymbol{D})$ remains at a value which is equal to $\kappa(\boldsymbol{D}_{\text{true}})$.

2.6.2 Synthetic Data Analysis

The settings in this subsection are as follows. The training samples are generated according to $\boldsymbol{Y} = \boldsymbol{D}_{\text{true}}\boldsymbol{X}_{\text{true}} + \boldsymbol{W}$ where $\boldsymbol{W} \in \mathbb{R}^{m\times n}$ are Gaussian noise ($\boldsymbol{W} = \boldsymbol{0}$ for the noiseless case), where $m = 16$, $d = 32$. The true dictionary $\boldsymbol{D}_{\text{true}}$ is randomly generated from the uniform distribution on $\mathcal{D}$. Regarding the sparse coefficients, we assume that each column of $\boldsymbol{X}_{\text{true}}$ contains exactly $s = 4$ many non-zero elements of which the locations are randomly generated from the corresponding uniform distribution. The nonzero elements of $\boldsymbol{X}_{\text{true}}$ are randomly generated from the standard Gaussian distribution. To separate the effect of sparse coding, we also assume that the sparse coding stage is perfect, i.e., the true sparsity pattern Ω_{true} is given a priori.

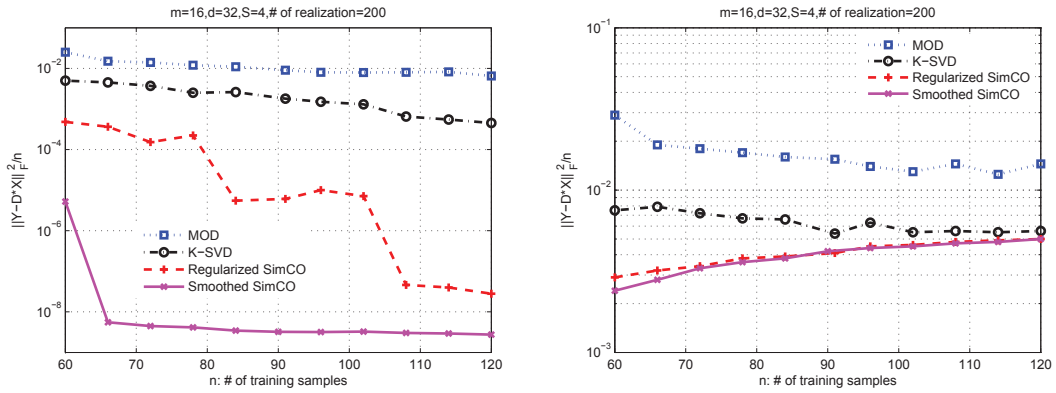

Figure 2.4: Performance comparison of dictionary update stage. Noiseless case (left) and noisy case with SNR=20dB (right). Note that there exist a lower bound of in the reconstruction error (right) which is proportional to the noise level. See also in [25]. The advantage of Smoothed SimCO is significant when the number of training samples is not large. The advantage gradually disappear when the train samples increase since the benchmark algorithms have lower probability converge to ill-conditioned dictionaries.

Both noiseless and noisy case are considered in the tests. Let $\hat{\boldsymbol{D}}$ and $\hat{\boldsymbol{X}}$ be the learned dictionary and the corresponding sparse coefficients, respectively. The normalized learning error is defined as $\left\|\boldsymbol{Y}-\hat{\boldsymbol{D}}\hat{\boldsymbol{X}}\right\|_F^2/n$. The criteria for success learning are designed for both cases using the normalized learning error: in the noiseless case, a success is claimed when $\left\|\boldsymbol{Y}-\hat{\boldsymbol{D}}\hat{\boldsymbol{X}}\right\|_F^2/n \leq \epsilon_e \|\boldsymbol{Y}\|_F^2$ where the constant ϵ_e is ideally zero but set to 10^{-6} in practice; for the noisy case, the criterion for a successful learning is given by $\left\|\boldsymbol{Y}-\hat{\boldsymbol{D}}\hat{\boldsymbol{X}}\right\|_F^2/n \leq \epsilon_n \|\boldsymbol{Y}\|_F^2$ where $\epsilon_n := \|\boldsymbol{W}\|_F^2/n/\|\boldsymbol{D}_{\text{true}}\boldsymbol{X}_{\text{true}}\|_F^2$.

Four algorithms: MOD, K-SVD, Regularized SimCO, and Smoothed SimCO, are compared in the tests. For each of the algorithms, 200 realizations are implemented. The maximum iteration number of each realization is set to 1000. For Regularized SimCO, the regularization constant is initially set as $\mu = 0.1$ and then reduced to $\mu/10$ after every 100 iterations. In Smoothed SimCO, the thresholds $\boldsymbol{\delta}_i$s are set to $(0.001, 0.2)$ for the first 500 iterations and then to $(0, 0)$ for the rest 500 iterations. (Note that $\boldsymbol{\delta}_i = \boldsymbol{\delta}_j$ for $\forall i, j \in [n]$ due to the simulation settings.)

The simulation results are presented in Figure 2.4, where the two sub-figures compare the normalized distortion in noiseless and 20dB noise case. The advantage of the proposed smoothed SimCO is clear for both cases and is particularly evident in the noiseless case.

In terms of successful rate, Smoothed SimCO reaches 100% successful rate when the number of training samples $n \geq 66$ while MOD and K-SVD could not achieve 100%

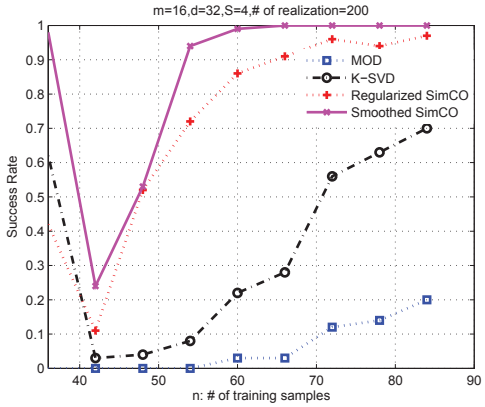

Figure 2.5: The successful rate of MOD, K-SVD, Regularized SimCO and Smoothed SimCO against the number of training samples. Each point shows the average value over 200 realizations. Among the four algorithms, Smoothed SimCO has the highest successful rate and the robustness covers more cases.

successful rate even when $n \geq 84$. It is also interesting to observe from Figure *2.5* that the dip in the successful rate when n is in the middle-range ($n = 40 \sim 60$). This is expected. On one hand, the successful rate should increase when the number of training samples becomes larger. On the other hand, when the number of training samples is extremely low, for example, $n \leq d$, the learning problem becomes trivial. Hence, the most difficult case is when n is in the middle-range. Having a deep study on the failure cases, they all converge to the dictionaries with large condition number, i.e. singular dictionaries. Take the realizations in Figure *2.5* with $n = 60$ training samples as examples, all the failures have $\kappa(\boldsymbol{D}) > 65$ for MOD , $\kappa(\boldsymbol{D}) > 37$ for K-SVD and $\kappa(\boldsymbol{D}) > 30$ for Regularized SimCO. On the contrary $\kappa(\boldsymbol{D}) < 8$ for all the successful cases (since all the true dictionaries used in the tests are of $\kappa(\boldsymbol{D}_{\text{true}}) < 8$). These results support the main reason of the dictionary update failures that put forward in this chapter.

Chapter 3

Blind Source Separation Based on Dictionary Learning

3.1 Introduction

Blind source separation (BSS) has been investigated during the last two decades, many algorithms have been developed and applied in a wide range of applications including biomedical engineering, medical imaging, speech processing, astronomical imaging and communication systems. Typically a linear mixture model is assumed where the mixtures $\boldsymbol{Z} \in \mathbb{R}^{r \times N}$ are described as $\boldsymbol{Z} = \boldsymbol{A}\boldsymbol{S} + \boldsymbol{V}$. Each row of $\boldsymbol{S} \in \mathbb{R}^{s \times N}$ is a source and $\boldsymbol{A} \in \mathbb{R}^{r \times s}$ models the linear combinations of the sources. The matrix $\boldsymbol{V} \in \mathbb{R}^{r \times N}$ represents additive noise or interference introduced during mixture acquisition and transmission.

Usually in the BSS problem the only known information is the mixtures $\boldsymbol{Z}$ and the number of sources. One needs to determine both the mixing matrix $\boldsymbol{A}$ and the sources $\boldsymbol{S}$, i.e., mathematically, one needs to solve

$$\min_{\boldsymbol{A},\boldsymbol{S}} \|\boldsymbol{Z} - \boldsymbol{A}\boldsymbol{S}\|_F^2 .$$

It is clear that such a problem has an infinite number of solutions, i.e., the problem is ill-posed. In order to find the true sources and the mixing matrix (subject to scale and permutation ambiguities), it is often required to add extra constraints to the problem formulation. For example, a well-known method called independent component analysis

(ICA) [58] assumes that the original sources are statistically independent. This has led to some widely used approaches, such as Infomax [9], maximum likelihood estimation [43], the maximum a posterior (MAP) [10], and FastICA [58].

Sparsity prior is another property that can be used for BSS. Most natural signals are sparse under some dictionaries. The mixtures, viewed as a superposition of sources, are in general less sparse compared to the original sources. Based on this fact, the sparse prior has been used in solving the BSS problem from various perspectives since 2001, e.g., sparse ICA (SPICA) [16] and sparse component analysis (SCA) [47]. In this approach, there is typically no requirement that the original sources have to be independent. As a result, these algorithms are capable of dealing with highly correlated sources, for example, in separating two superposed identical speeches, with one being a few samples delayed version of the other. Jourjine, et al., proposed a SCA based algorithm in [62] aiming at solving the anechoic problem. SCA algorithms look for a sparse representation under predefined bases such as discrete cosine transform (DCT), wavelet, curvelet, etc. Morphological component analysis (MCA) [93] and its extended algorithms for multichannel cases, Multichannel MCA (MMCA) [14] and Generalized MCA (GMCA) [15], are also based on the assumption that the original sources are sparse in different bases instead of explicitly constructed dictionaries. However these algorithms do not exhibit an outstanding performance since in most cases the predefined dictionaries are too general to offer sufficient details of sources when used in sparse representation.

A method to address this problem is to learn data-specific dictionaries. In [40], the authors advised to train a dictionary from the mixtures/corrupted-images and then decompose it into a few dictionaries according to the prior knowledge about the main components in different sources. This algorithm is used for separating images with different main frequency components (e.g., Cartoon and Texture images) and obtained satisfactory results in image denoising. Starck, et al. proposed in [83] to learn dictionary from a set of exemplar images for each source. Xu, et al., [112] proposed an algorithm which allows the dictionaries to be learned from the sources or the mixtures. In most BSS problems, however, dictionaries learned from the mixtures or from similar exemplar images rarely well represent the original sources.

To get more accurate separation results, the dictionaries should be adapted to the

unknown sources. The motivation is clear from the assumption that the sources are sparsely represented by some dictionaries. The initial idea of learning dictionaries while separating the sources was suggested by Abolghasemi, et al. [1]. They proposed a two-stage iterative process. In this process each source is equipped with a dictionary, which is learned in each iteration, right after the previous mixture learning stage. Considering the size of dictionaries being much larger than the mixing matrix, the main computational cost is on the dictionary learning stage. This two-stage procedure was further developed in Zhao, et al. [118]. The method was termed as SparseBSS, which employs a joint optimization framework based on the idea of SimCO dictionary update algorithm [25]. By studying the optimization problem encountered in dictionary learning, the phenomenon of singularity in dictionary update was for the first time discovered. Furthermore, from the viewpoint of the dictionary redundancy, SparseBSS uses only *one dictionary* to represent all the sources, and is therefore computationally much more efficient than using multiple dictionaries as in [1]. This joint dictionary learning and source separation framework is the focus of this chapter. This framework can be extended potentially to a convolutive or underdetermined model, e.g., apply clustering method to solve the the ill-posed inverse problem in underdetermined model [13] ; however, discussion on such an extension is beyond the scope of this chapter. In this chapter, we focus on over-determined/even-determined model.

The remainder of this chapter is organized as follows. Section 3.2 describes the framework of the BSS problem based on dictionary learning. The recently proposed algorithm SparseBSS is introduced and compared in detail with the related benchmark algorithm BMMCA.

3.2 Framework of Dictionary Learning based Blind Source Separation Problem

We consider the following linear and instantaneous mixing model. Suppose that there are s source signals of the same length, denoted by $\boldsymbol{s}_1,\ \boldsymbol{s}_2,\ \cdots,\ \boldsymbol{s}_s$ respectively, where $\boldsymbol{s}_i \in \mathbb{R}^{1\times N}$ is a row vector to denote the i^{th} source. Assume that these sources are linearly mixed into r observation signals, denoted by $\boldsymbol{z}_1,\ \boldsymbol{z}_2,\ \cdots,\ \boldsymbol{z}_r$ respectively where $\boldsymbol{z}_j \in \mathbb{R}^{1\times N}$. In the matrix format, denote $\boldsymbol{S} = \left[\boldsymbol{s}_1^T, \boldsymbol{s}_2^T, \cdots, \boldsymbol{s}_s^T\right]^T \in \mathbb{R}^{s\times N}$ and $\boldsymbol{Z} = \left[\boldsymbol{z}_1^T, \boldsymbol{z}_2^T, \cdots, \boldsymbol{z}_r^T\right]^T \in$

$\mathbb{R}^{r\times N}$. Then the mixing model is given by

$$\boldsymbol{Z} = \boldsymbol{A}\boldsymbol{S} + \boldsymbol{V}, \tag{3.1}$$

where $\boldsymbol{A} \in \mathbb{R}^{r\times s}$ is the mixing matrix and $\boldsymbol{V} \in \mathbb{R}^{r\times N}$ is denoted as zero mean additive Gaussian noise. We also assume that $r \geq s$, i.e., the under-determined case will not be discussed here.

3.2.1 Separation with Dictionaries Known in Advance

For some BSS algorithms, such as MMCA [14], orthogonal dictionaries $\boldsymbol{D}_i$'s are required to be known a priori. Each source $\boldsymbol{s}_i$ is assumed to be sparsely represented by a different $\boldsymbol{D}_i$. Hence we have $\boldsymbol{s}_i = \boldsymbol{D}_i\boldsymbol{x}_i$ with $\boldsymbol{x}_i$'s being sparse. Given the observation $\boldsymbol{Z}$ and the dictionaries $\boldsymbol{D}_i$'s, MMCA [14] aims to estimate the mixing matrix and sources, based on the following form:

$$\min_{\boldsymbol{A},\boldsymbol{S}} \|\boldsymbol{Z} - \boldsymbol{A}\boldsymbol{S}\|_F^2 + \sum_{i=1}^{n} \lambda_i \left\|\boldsymbol{s}_i\boldsymbol{D}_i^{\dagger}\right\|_1. \tag{3.2}$$

Here $\lambda_i > 0$ is the weighting parameter determined by the noise deviation σ, $\|\cdot\|_F$ represents the Frobenius norm, $\|\cdot\|_1$ is the ℓ_1 norm and $\boldsymbol{D}_i^{\dagger}$ denotes the pseudo-inverse of $\boldsymbol{D}_i$. Predefined dictionaries generated from typical mathematical transforms, e.g., DCT, wavelets and curvelets, do not target to particular sources, and thus do not always provide sufficiently accurate reconstruction and separation results. Elad, et al. [40] designed a method to first train a redundant dictionary by K-SVD algorithm in advance, and then decompose it into a few dictionaries, one for each source. This method works well when the original sources have components that are largely different from each other under some unknown mathematical transformations (e.g. Cartoon and Texture images under the DCT transformation). Otherwise the dictionaries found may not be appropriate in the sense that they may fit better to the mixtures rather than the sources.

3.2.2 Separation with Unknown Dictionaries

SparseBSS Algorithm Framework

According to the authors knowledge, BMMCA and SparseBSS are the two most recently BSS algorithms which implement the idea of performing source separation and dictionary learning simultaneously. We focus on Sparse BSS in this chapter. In SparseBSS, one assumes that all the sources can be sparsely represented under the same dictionary. In order to obtain enough training samples for dictionary learning, multiple overlapped segments (patches) of the sources are taken. To extract small overlapped patches from the source image $\boldsymbol{s}_i$, a binary matrix $\boldsymbol{P}_k \in \mathbb{R}^{n\times N}$ is defined as a patching operator[1] [118]. The product $\boldsymbol{P}_k \cdot \boldsymbol{s}_i^T \in \mathbb{R}^{n\times 1}$ is needed to obtain and vectorize the kth patch of size $\sqrt{n}\times\sqrt{n}$ taken from image $\mathcal{S}_i$. Denote $\boldsymbol{P} = [\boldsymbol{P}_1, ..., \boldsymbol{P}_K] \in \mathbb{R}^{n\times KN}$, where K is the number of patches taken from each image. Then the extraction of multiple sources $\boldsymbol{S}$ is defined as $\mathcal{P}(\boldsymbol{S}) = ([\boldsymbol{P}_1, ..., \boldsymbol{P}_K]) \cdot ([\boldsymbol{s}_1^T, \boldsymbol{s}_2^T, ..., \boldsymbol{s}_s^T] \otimes \boldsymbol{I}_K) = \boldsymbol{P} \cdot (\boldsymbol{S}^T \otimes \boldsymbol{I}_K) \in \mathbb{R}^{n\times Ks}$, where symbol $\otimes$ denotes the Kronecker product and $\boldsymbol{I}_K$ indicates the identity matrix. The computational cost associated with converting from images to patches is low. Each column of $\mathcal{P}(\boldsymbol{S})$ represents one vectorized patch. We sparsely represent $\mathcal{P}(\boldsymbol{S})$ by using only one dictionary $\boldsymbol{D} \in \mathbb{R}^{n\times d}$ and a sparse coefficient matrix $\boldsymbol{X} \in \mathbb{R}^{d\times Ks}$, which suggests $\mathcal{P}(\boldsymbol{S}) \approx \boldsymbol{D}\boldsymbol{X}$. This is different from BMMCA, where multiple dictionaries are used for multiple sources.

With these notations, the BSS problem is formulated as the following joint optimization problem

$$\min_{\boldsymbol{A},\boldsymbol{S},\boldsymbol{D},\boldsymbol{X}} \lambda \left\|\boldsymbol{Z} - \boldsymbol{A}\boldsymbol{S}\right\|_F^2 + \left\|\mathcal{P}^{\dagger}(\boldsymbol{D}\boldsymbol{X}) - \boldsymbol{S}\right\|_F^2. \tag{3.3}$$

The parameter λ is introduced to balance the measurement error and the sparse approximation error, and $\boldsymbol{X}$ is assumed to be sparse.

To find the solution of the above problem, we propose a joint optimization algorithm to iteratively update the following two pairs of variables $\{\boldsymbol{D}, \boldsymbol{X}\}$ and $\{\boldsymbol{A}, \boldsymbol{S}\}$ over two stages until a (local) minimizer is found. Note that in each stage there is only one pair of variables to be updated simultaneously by keeping the other pair fixed.

[1] Note that in this chapter $\boldsymbol{P}_k$ is defined as a patching operator for image sources. The patching operator for audio sources can be similarly defined as well.

- Dictionary learning stage

$$\min_{\boldsymbol{D},\boldsymbol{X}} \|\boldsymbol{DX} - \mathcal{P}(\boldsymbol{S})\|_F^2, \tag{3.4}$$

- Mixture learning stage

$$\min_{\boldsymbol{A},\boldsymbol{S}} \lambda \|\boldsymbol{Z} - \boldsymbol{AS}\|_F^2 + \|D\boldsymbol{X} - \mathcal{P}(\boldsymbol{S})\|_F^2. \tag{3.5}$$

Without being explicit in (3.3), a sparse coding process is involved where greedy algorithms such as orthogonal matching pursuit (OMP) [81] and subspace pursuit (SP) [24] are used to solve

$$\min_{\boldsymbol{X}} \|\boldsymbol{X}\|_0, \text{ s.t. } \|\boldsymbol{DX} - \mathcal{P}(\boldsymbol{S})\|_F^2 \leq \epsilon,$$

where $\|\boldsymbol{X}\|_0$ counts the number of nonzero elements in $\boldsymbol{X}$, the dictionary $\boldsymbol{D}$ is assumed fixed, and $\epsilon > 0$ is an upper bound on the sparse approximation error.

During the optimization, further constraints are made on the matrices $\boldsymbol{A}$ and $\boldsymbol{D}$. Consider the dictionary learning stage. Since the performance is invariant to scaling and permutations of the dictionary codewords (columns of $\boldsymbol{D}$), we follow the convention in the literature, e.g., [25], and enforce the dictionary to be updated on the set

$$\mathcal{D} = \left\{ \boldsymbol{D} \in \mathbb{R}^{n \times d} : \ \|\boldsymbol{D}_{:,i}\|_2 = 1, \ 1 \leq i \leq d \right\}, \tag{3.6}$$

where $\boldsymbol{D}_{:,i}$ stands for the i^{th} column of $\boldsymbol{D}$. A detailed description of the advantage by adding this constraint can be found in [25]. Sparse coding, once performed, provides the information about which elements of $\boldsymbol{X}$ are zeros and which are non-zeros. Define the sparsity pattern by $\Omega = \{(i,j) : \ \boldsymbol{X}_{i,j} \neq 0\}$, which is the index set of the nonzero elements of $\boldsymbol{X}$. Define $\mathcal{X}_\Omega$ as the set of all matrices conforming to the sparsity pattern Ω. This is the feasible set of the matrix $\boldsymbol{X}$. The optimization problem for the dictionary learning stage can be written as

$$\begin{aligned} \min_{\boldsymbol{D}\in\mathcal{D}} f_\mu(\boldsymbol{D}) &= \min_{\boldsymbol{D}\in\mathcal{D}} \ \min_{\boldsymbol{X}\in\mathcal{X}_\Omega} \|\boldsymbol{DX} - \mathcal{P}(\boldsymbol{S})\|_F^2 + \mu \|\boldsymbol{X}\|_F^2, \\ &= \min_{\boldsymbol{D}\in\mathcal{D}} \ \min_{\boldsymbol{X}\in\mathcal{X}_\Omega} \left\| \begin{bmatrix} \mathcal{P}(\boldsymbol{S}) \\ \boldsymbol{0} \end{bmatrix} - \begin{bmatrix} \boldsymbol{D} \\ \sqrt{\mu}\boldsymbol{I} \end{bmatrix} \boldsymbol{X} \right\|_F^2. \end{aligned} \tag{3.7}$$

The term $\mu \|\boldsymbol{X}\|_F^2$ introduces a penalty to alleviate the singularity issue.

In the mixture learning stage, similar to the dictionary learning stage, we constrain the mixing matrix $\boldsymbol{A}$ in the set

$$\mathcal{A} = \left\{ \boldsymbol{A} \in \mathbb{R}^{r \times s} : \ \|\boldsymbol{A}_{:,i}\|_2 = 1, \ 1 \leq i \leq s \right\}. \tag{3.8}$$

This constraint is necessary. Otherwise if the mixing matrix $\boldsymbol{A}$ is scaled by a constant c and the source $\boldsymbol{S}$ is inversely scaled by c^{-1}, then for any $\{\boldsymbol{A}, \boldsymbol{S}\}$ we can always find a solution $\{c\boldsymbol{A}, c^{-1}\boldsymbol{S} | c > 1\}$ which further decreases the objective function (3.3) from $\lambda \|\boldsymbol{Z} - \boldsymbol{A}\boldsymbol{S}\|_F^2 + \|D\boldsymbol{X} - \mathcal{P}(\boldsymbol{S})\|_F^2$ to $\lambda \|\boldsymbol{Z} - \boldsymbol{A}\boldsymbol{S}\|_F^2 + c^{-2} \|\boldsymbol{D}\boldsymbol{X} - \mathcal{P}(\boldsymbol{S})\|_F^2$. Now if we view the sources $\boldsymbol{S} \in \mathbb{R}^{s \times n}$ as a "sparse" matrix with the sparsity pattern $\Omega' = \{(i,j) : \ 1 \leq i \leq s, \ 1 \leq j \leq N\}$. Then the optimization problem for the mixture learning stage is exactly the same as that for the dictionary learning stage:

$$\begin{aligned} \min_{\boldsymbol{A} \in \mathcal{A}} f_\lambda(\boldsymbol{A}) =& \min_{\boldsymbol{A} \in \mathcal{A}} \min_{\boldsymbol{S} \in \mathbb{R}^{s \times n}} \lambda \|\boldsymbol{Z} - \boldsymbol{A}\boldsymbol{S}\|_F^2 + \left\| \mathcal{P}^\dagger(\boldsymbol{D}\boldsymbol{X}) - \boldsymbol{S} \right\|_F^2 \\ =& \min_{\boldsymbol{A} \in \mathcal{A}} \min_{\boldsymbol{S} \in \mathcal{X}_{\Omega'}} \left\| \begin{bmatrix} \sqrt{\lambda}\boldsymbol{Z} \\ \mathcal{P}^\dagger(\boldsymbol{D}\boldsymbol{X}) \end{bmatrix} - \begin{bmatrix} \sqrt{\lambda}\boldsymbol{A} \\ \boldsymbol{I} \end{bmatrix} \boldsymbol{S} \right\|_F^2, \end{aligned} \tag{3.9}$$

where the fact that $\mathbb{R}^{s \times n} = \mathcal{X}_{\Omega'}$ has been used. As a result, the SimCO mechanism can be directly applied. Here, we do not require the prior knowledge about the scaling matrix in front of the true mixing matrix [15], as otherwise required in MMCA and GMCA algorithms.

To conclude this subsection, we emphasize the following treatment of the optimization problems (3.7) and (3.9). Both of them involve a joint optimization over two variables, i.e., $\boldsymbol{D}$ and $\boldsymbol{X}$ for (3.7) and $\boldsymbol{A}$ and $\boldsymbol{S}$ for (3.9). Note that if $\boldsymbol{D}$ and $\boldsymbol{A}$ are fixed, then the optimal $\boldsymbol{X}$ and $\boldsymbol{S}$ can be easily computed by solving the corresponding least squares problems. Motivated by this fact, we write (3.7) and (3.9) as $\min_{\boldsymbol{D} \in \mathcal{D}} f_\mu(\boldsymbol{D})$ and $\min_{\boldsymbol{A} \in \mathcal{A}} f_\lambda(\boldsymbol{A})$ respectively, when $f_\mu(\boldsymbol{D})$ and $f_\lambda(\boldsymbol{A})$ are properly defined in (3.7) and (3.9). In this way, the optimization problems, at least from the surface, only involve one variable. This helps the discovery of the singularity issue and the developments of handling singularity. See Chapter 2 for details.

Implementation Details in SpaseBSS

Most optimization methods are based on line search strategies. The dictionaries at the beginning and the end of the k^{th} iteration, denoted by $\boldsymbol{D}^{(k)}$ and $\boldsymbol{D}^{(k+1)}$ respectively, can be related by $\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} + \alpha^{(k)}\boldsymbol{\eta}^{(k)}$ where $\alpha^{(k)}$ is an appropriately chosen step size and $\boldsymbol{\eta}^{(k)}$ is the search direction. The step size $\alpha^{(k)}$ can be determined by *Armijo condition* or *Golden selection* presented in [77]. The search direction $\boldsymbol{\eta}^{(k)}$ can be determined by a variety of gradient methods [77, 37]. The decision of $\boldsymbol{\eta}^{(k)}$ plays the key role which directly affects the convergence rate of the whole algorithm. Generally speaking, a Newton direction is a preferred choice (compared with the gradient descent direction) [77]. In many cases, direct computation of the Newton direction is computationally prohibitive. Iterative methods can be used to search the Newton direction. Take the Newton Conjugate Gradient (Newton CG) method as an example. It starts with the gradient descent direction $\boldsymbol{\eta}_0$ and iteratively refines it towards the Newton direction. Denote the gradient of $f_\mu(\boldsymbol{D})$ as $\nabla f_\mu(\boldsymbol{D})$. Denote $\nabla_{\boldsymbol{\eta}}(\nabla f_\mu(\boldsymbol{D}))$ as the directional derivative of $\nabla f_\mu(\boldsymbol{D})$ along $\boldsymbol{\eta}$ [56]. In each line search step of the Newton CG method, instead of computing the Hessian $\nabla^2 f_\mu(\boldsymbol{D}) \in \mathbb{R}^{md\times md}$ explicitly, one only needs to compute $\nabla_{\boldsymbol{\eta}}(\nabla f_\mu(\boldsymbol{D})) \in \mathbb{R}^{m\times d}$. The required computational and storage resources are therefore much reduced.

When applying the Newton CG to minimize $f_\mu(\boldsymbol{D})$ in (3.7), the key computations are summarized below. Denote $\tilde{\boldsymbol{D}} = \begin{bmatrix} \boldsymbol{D}^T & \mu\boldsymbol{I} \end{bmatrix}^T$ and let $\Omega(:,j)$ be the index set of nonzero elements in $\boldsymbol{X}_{:,j}$. We consider $\tilde{\boldsymbol{D}}_i = \tilde{\boldsymbol{D}}_{:,\Omega(:,i)} \in \mathbb{R}^{(m+r)\times r}$ with $m > r$. Matrix $\tilde{\boldsymbol{D}}_i$ is a full column rank tall matrix. We denote

$$f_i(\tilde{\boldsymbol{D}}_i) = \min_{\boldsymbol{x}_i} \|\boldsymbol{y}_i - \tilde{\boldsymbol{D}}\boldsymbol{x}_i\|_2^2$$

and the optimal

$$\boldsymbol{x}_i^* = \arg\min_{\boldsymbol{x}_i} \|\boldsymbol{y}_i - \tilde{\boldsymbol{D}}\boldsymbol{x}_i\|_2^2.$$

Denote $\tilde{\boldsymbol{D}}_i^\dagger$ as the pseudo-inverse of $\tilde{\boldsymbol{D}}_i$. As discussed in the last chapter, we can compute $\nabla f_i(\tilde{\boldsymbol{D}}_i)$, $\nabla_{\boldsymbol{\eta}}\left(\nabla f_i(\tilde{\boldsymbol{D}}_i)\right)$ and $\nabla_{\boldsymbol{\eta}}\boldsymbol{x}^*$ via (2.18), (2.19) and (2.20), respectively. From the definition of $\tilde{\boldsymbol{D}}_i$, $\boldsymbol{D}_i$ is a sub-matrix of $\tilde{\boldsymbol{D}}_i$, therefore $\nabla f_i(\boldsymbol{D}_i)$ and $\nabla_{\boldsymbol{\eta}}(\nabla f_i(\boldsymbol{D}_i))$ are also respectively sub-matrices of $\nabla f_i(\tilde{\boldsymbol{D}}_i)$ and $\nabla_{\boldsymbol{\eta}}\left(\nabla f_i(\tilde{\boldsymbol{D}}_i)\right)$, i.e., $\nabla f_i(\boldsymbol{D}_i) = \left(\nabla f_i(\tilde{\boldsymbol{D}}_i)\right)_{1:m,:}$

and $\nabla_{\boldsymbol{\eta}}\left(\nabla f_i(\boldsymbol{D}_i)\right) = \left(\nabla_{\boldsymbol{\eta}}\left(\nabla f_i(\tilde{\boldsymbol{D}}_i)\right)\right)_{1:m,:}$.

In addition, it is also worth noting that the SpaseBSS model, using one dictionary to sparsely represent all the sources will get almost the same performance as using multiple but same-sized dictionaries when the dictionary redundancy $\frac{d}{n}$ is large enough. As a result it is reasonable to train only one dictionary for all the sources. An obvious advantage for using one dictionary is that the computational cost does not increase when the number of sources increases.

3.2.3 Blind MMCA and its Comparison to SparseBSS

BMMCA [1] is another recently proposed BSS algorithm based on adaptive dictionary learning. Without knowing dictionaries in advance, BMMCA algorithm also trains dictionaries from the observed mixture $\boldsymbol{Z}$. Inspired by the hierarchical scheme used in MMCA and the update method in K-SVD, the separation model in BMMCA is made up of a few rank-1 approximation problems, where each problem targets on the estimation of one particular source

$$\min_{\boldsymbol{A}_{:,i},\boldsymbol{s}_i,\boldsymbol{D}_i,\boldsymbol{X}_i} \lambda \left\|\boldsymbol{E}_i - \boldsymbol{A}_{:,i}\boldsymbol{s}_i\right\|_F^2 + \left\|\boldsymbol{D}_i\boldsymbol{X}_i - \mathcal{R}\left(\boldsymbol{s}_i\right)\right\|_2^2 + \mu \left\|\boldsymbol{X}_i\right\|_0. \tag{3.10}$$

Different from the operator $\mathcal{P}$ defined earlier in SparseBSS algorithm, the operator $\mathcal{R}$ in BMMCA is used to take patches from only one estimated image $\boldsymbol{s}_i$. $\boldsymbol{D}_i$ is the trained dictionaries for representing source $\boldsymbol{s}_i$. $\boldsymbol{E}_i$ is the residual which can be written as

$$\boldsymbol{E}_i = \boldsymbol{Z} - \sum_{j \neq i} \boldsymbol{A}_{:,j}\boldsymbol{s}_j. \tag{3.11}$$

Despite being similar in problem formulation, BMMCA and SpaseBSS differ in terms of whether the sources share a single dictionary in dictionary learning. In the SparseBSS algorithm, only one dictionary is used to provide sparse representations for all sources. BMMCA requires multiple dictionaries, one for each source. In the mixing matrix update, BMMCA imitates the K-SVD algorithm by splitting the steps of update and normalization. Such two-step based approach does not bring the expected optimality of $\boldsymbol{A} \in \mathcal{A}$, thereby giving inaccurate estimation, while SparseBSS keeps $\boldsymbol{A} \in \mathcal{A}$ during the optimiza-

tion process. In BMMCA, the authors claim that the ratio between the parameter λ and the noise standard deviation σ is fixed to 30, which will not guarantee good estimation results at various noise levels.

3.3 Algorithm Testing on Practical Applications

In this section we present numerical results of the SparseBSS method compared with some other mainstream algorithms. We first focus on speech separation where an equal determined case will be considered. Then we show an example for blind image separation, where we will consider an overdetermined case.

In the speech separation case, two mixtures are used which are the mixtures of two audio sources. Two male utterances in different languages are selected as the sources. The sources are mixed by a 2×2 random matrix $\boldsymbol{A}$ (with normalized columns). For the noisy case, a 20 dB Gaussian noise was added to the mixtures. See Figure 3.1 for the sources and mixtures.

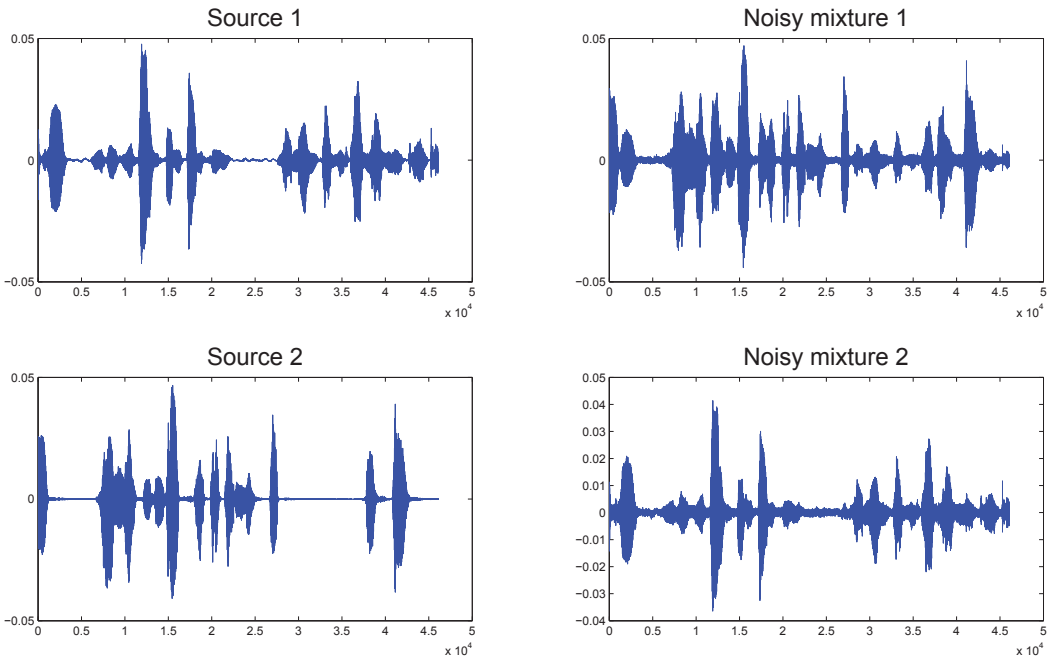

Figure 3.1: Two speech sources and the corresponding noisy mixtures (20 dB Gaussian noise).

We compare SparseBSS with two benchmark algorithms including FastICA and QJADE [20]. The BSSEVAL toolbox [107] is used for the performance measurement. In particular, an estimated source $\hat{\boldsymbol{s}}$ is decomposed as $\hat{\boldsymbol{s}} = \boldsymbol{s}_{\text{target}} + \boldsymbol{e}_{\text{interf}} + \boldsymbol{e}_{\text{noise}} + \boldsymbol{e}_{\text{artif}}$, where $\boldsymbol{s}_{\text{target}}$ is the true source signal, $\boldsymbol{e}_{\text{interf}}$ denotes the interferences from other sources, $\boldsymbol{e}_{\text{noise}}$ represents the deformation caused by the noise, and $\boldsymbol{e}_{\text{artif}}$ includes all other artifacts introduced by the separation algorithm. Based on the decomposition, three performance criteria can be defined: the source-to-distortion ratio $\text{SDR} = 10\log_{10}\frac{\|\boldsymbol{s}_{\text{target}}\|^2}{\|\boldsymbol{e}_{\text{interf}}+\boldsymbol{e}_{\text{noise}}+\boldsymbol{e}_{\text{artif}}\|^2}$, the source-to-interference ratio $\text{SIR} = 10\log_{10}\frac{\|\boldsymbol{s}_{\text{target}}\|^2}{\|\boldsymbol{e}_{\text{interf}}\|^2}$, and the source-to-artifact ratio $\text{SAR} = 10\log_{10}\frac{\|\boldsymbol{s}_{\text{target}}+\boldsymbol{e}_{\text{interf}}+\boldsymbol{e}_{\text{noise}}\|^2}{\|\boldsymbol{e}_{\text{artif}}\|^2}$. Among them, the SDR measures the overall performance (quality) of the algorithm, and the SIR focuses on the interference rejection. We investigate the gains of SDRs, SARs and SIRs from the mixtures to the estimated sources. For example, $\triangle SDR = SDR_{out} - SDR_{in}$, where SDR_{out} is calculated from its definition and SDR_{in} is obtained by letting $\hat{\boldsymbol{s}} = \boldsymbol{Z}$ with the same equation. The results (in dB) are summarized in Table 3.1.

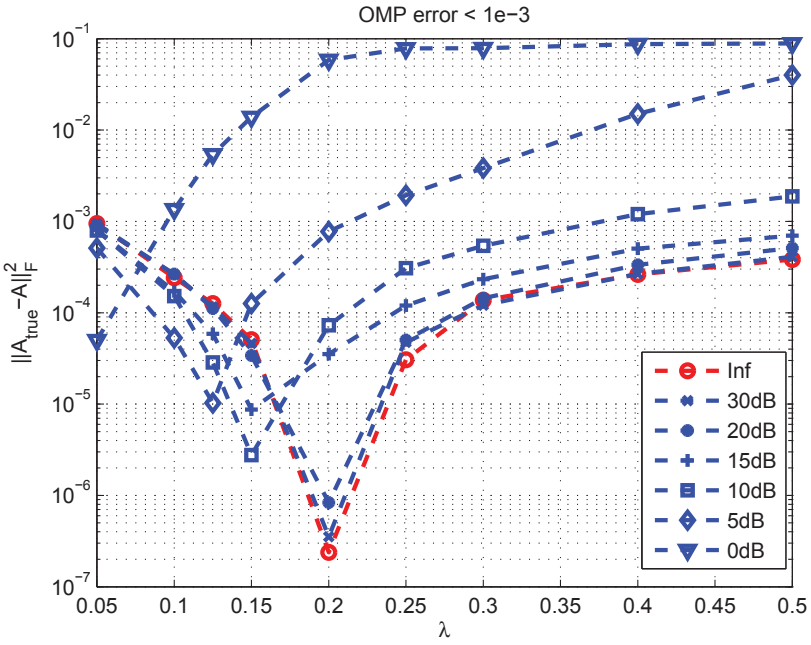

Figure 3.2: Relation of the parameter λ in SparseBSS problem to the estimation error of the mixing matrix under different noise levels. The signal-to-noise ratio (SNR) is defined as $\rho = 10\log_{10}\|\boldsymbol{AS}\|_F^2 / \|\boldsymbol{V}\|_F^2$ dB.

	△SDR	△SIR	△SAR
QJADE	60.661	60.661	-1.560
FastICA	57.318	57.318	-0.272
SparseBSS	**69.835**	**69.835**	**1.379**

(a) The noiseless case for three BSS algorithms.

	△SDR	△SIR	△SAR
QJADE	7.453	58.324	-1.245
FastICA	7.138	40.789	-1.552
SparseBSS	**9.039**	**62.450**	**0.341**

(b) The noisy case for three BSS algorithms.

Table 3.1: Separation performance of the SparseBSS algorithm as compared to FastICA and QJADE. The proposed SparseBSS algorithm performs better than the benchmark algorithms. Table 3.1a. For the same algorithm, the $\triangle SDR$ and $\triangle SIR$ are the same in noiseless case. The $\triangle SDRs$ and $\triangle SIRs$ for all the tested algorithms are large and similar, suggesting that all the compared algorithms perform very well. The artifact introduced by SparseBSS is small as its $\triangle SAR$ is positive. Table 3.1b. In the presence of noise with SNR = 20 dB, SparseBSS excels the other algorithms in $\triangle SDR$, $\triangle SIR$ and $\triangle SAR$. One interesting phenomenon is that the $\triangle SDRs$ are much smaller than those in the noiseless case, implying that the distortion introduced by the noise is trivial. However, SparseBSS still has better performance.

The selection of λ is an important practical issue since it is related to the noise level and largely affects the algorithm performance. From the optimization formulation (3.3), it is clear that with a fixed SNR, different choices of λ may give different separation performance. To show this, we use the estimation error $\left\|\boldsymbol{A}_{\text{true}} - \hat{\boldsymbol{A}}\right\|_F^2$ of the mixing matrix to measure the separation performance, where $\boldsymbol{A}_{\text{true}}$ and $\hat{\boldsymbol{A}}$ are the true and estimated mixing matrices, respectively. The simulation results are presented in Figure 3.2. Consistent with the intuition, simulations suggest that the smaller the noise level the larger the optimal value of λ. The results in Figure 3.2 help in setting λ when the noise level is known a priori.

Next, we show an example for blind image separation, where we consider an overdetermined case. The mixed images are generated from two source images using a 4×2 full rank column normalized mixing matrix $\boldsymbol{A}$ with its elements generated randomly according to a Gaussian process. The mean squared errors (MSEs) is used to compare the reconstruction performance of the candidate algorithms when no noise is added. MSE is defined as $MSE = (1/N)\left\|\chi - \tilde{\chi}\right\|_F^2$, where χ is the source image and $\tilde{\chi}$ is the reconstructed image.

The lower the MSE, the better the reconstruction performance. Table 3.2 illustrates the results of four tested algorithms. For the noisy case, a Gaussian white noise was added to the four mixtures with $\sigma = 10$. We use the Peak Signal-to-Noise Ratio (PSNR) to measure the reconstruction quality, which is defined as, $PSNR = 20\log_{10}(\frac{MAX}{\sqrt{MSE}})$, where MAX indicates the maximum possible pixel value of the image, (e.g., $MAX = 255$ for a uint-8 image). Higher PSNR indicates better quality. The noisy observations are illustrated in Figure 3.3. (b)[2].

[2]For the BMMCA test, a better performance was demonstrated in [1]. We point out that here a different true mixing matrix is used. And further more, in our tests the patches are taken with a 50% overlap (by shifting 4 pixels from the current patch to the next) while in [1] the patches are taken by shifting only one pixel from the current patch to the next.

Figure 3.3: Two classic images, *Lena* and *Boat* were selected as the source images, which are shown in (a). The mixtures are shown in (b). The separation results are shown in (c)-(f). We compared SparseBSS with other benchmark algorithms: FastICA [57], GMCA [15] and BMMCA [1]. We set the overlap percentage equal to 50% for both BMMCA and SparseBSS. The recovered source images by the SparseBSS tend to be less blurred as compared to the other three algorithms.

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

Table 3.2: Achieved MSEs of the four blind source separation algorithms in a noiseless case.

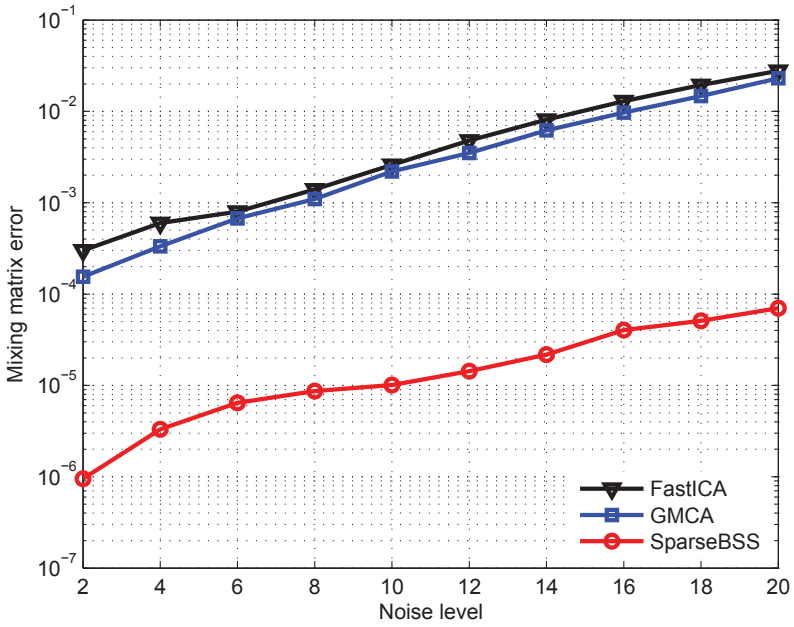

Figure 3.4: Compare the performance of estimating the mixing matrix for all the methods in different noise standard deviation σs. In this experiment, σ varies from 2 to 20. The performance of GMCA is better than that of FastICA. The curve for BMMCA is not available as the setting for the parameters is too sophisticated and inconsistent for different σ to obtain a good result. SparseBSS outperforms the compared algorithms.

At last, we show another example of blind image separation to demonstrate the importance of the singularity aware process. In this example, we use two classic images *Lena* and *Texture* as the source images (3.5(a)). Four noiseless mixtures were generated from the sources. The separation results are shown in 3.5(b) and (c). Noting that images like *Texture* contain a lot of frequency components corresponding to a particular frequency. Hence an initial dictionary with more codeword corresponding to the particular frequency may better estimate these images. From this motivation, in 3.5(b) the initial dictionary is generated from an over-complete DCT dictionary but contain more high frequency code-

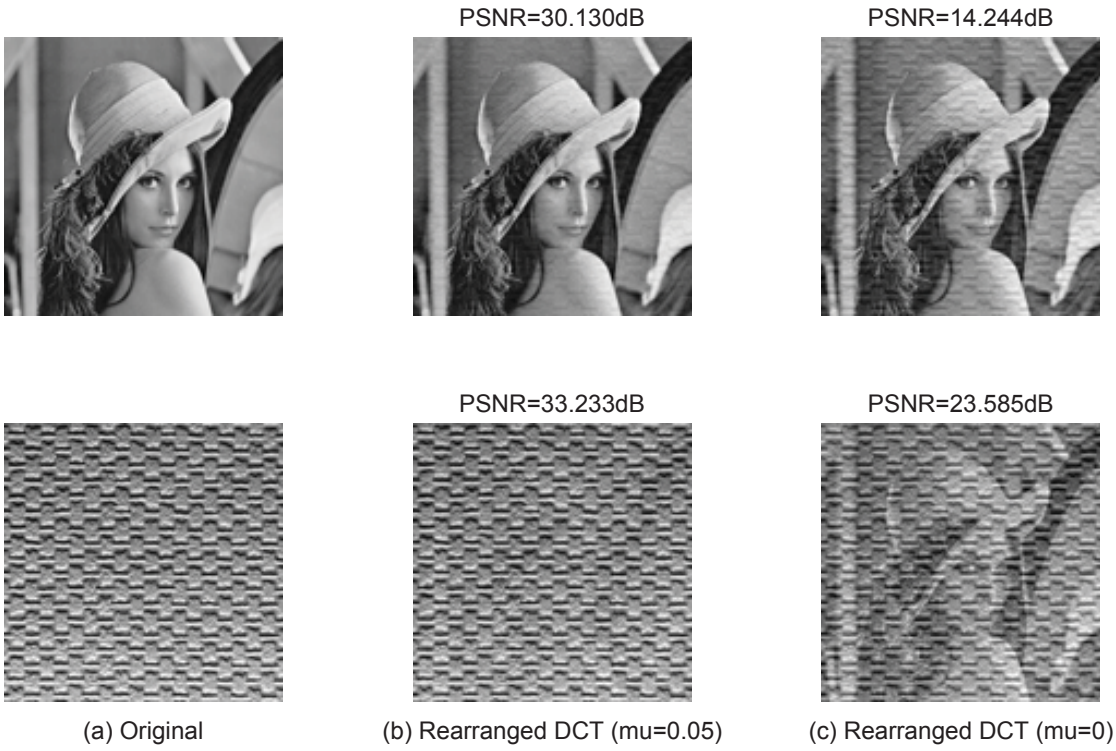

Figure 3.5: The two source images *Lena* and *Texture* are shown in (a). The separation results are shown in (b) and (c). The comparison results demonstrate the importance of the singularity aware process.

words. Such choice can bring better separation results. At the same time, the very similar dictionary codewords bring the risk of singularity issue.

The major difference between 3.5(b) and (c) is that: in 3.5(b) the Regularized SimCO process ($\mu = 0.05$) is introduced, while in 3.5(c) there is no regularized term in the dictionary learning stage. As one can see from the numerical results, 3.5(b) performs much better than 3.5(c). By checking the condition number when the regularized term is not introduced ($\mu = 0$), the value stays in a high level as expected (larger than 40 in this example). This shows the necessity of considering the singularity issue in BSS and the effectiveness of the proposed singularity aware approach.

Chapter 4

Power Allocation in Compressed Sensing of Non-uniformly Sparse Signals

4.1 Introduction

This chapter focuses on the compressed sensing model. The observation $\boldsymbol{y} \in \mathbb{R}^m$ is measured from the linear model

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}, \tag{4.1}$$

where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ $(m < n)$ is the measurement matrix, $\boldsymbol{x} \in \mathbb{R}^n$ is the unknown sparse signal, and $\boldsymbol{w} \in \mathbb{R}^m$ is the white Gaussian noise with covariance $\sigma^2 \boldsymbol{I}$. We are particularly interested in non-uniformly sparse signals where different signal components may have different nonzero probabilities. Such signals arise in many practical scenarios. For example, in the multiple-source localization problem, the sources (corresponding to nonzero signal components) are often clustered in certain areas. For natural images, the nonzero wavelet coefficients form a tree structure [4]. In video surveillance, the signals from adjacent frames share many nonzero components [106]. Using the non-uniformly sparsity appropriately can help improve the compressed sensing reconstruction performance, see [92, 35] for examples. Similar concept named variable density sampling techniques in MRI could reduce the frequency of signal sensing.

In this work we focus on the measurement matrix design problem when non-uniformly sparse signals are involved. More specifically, given a total power budget (sum of ℓ_2 norm squares across all measurement matrix columns), we are interested in the optimal power allocation across the columns of a Gaussian random measurement matrix to minimize the reconstruction error. Similar problems have been considered in the adaptive sensing setup where non-uniformly sparse statistics are generated in the initial sensing process and that information is used to design the measurement matrices in later stages. Examples include [52], [88], [99], and [109], to name a few. Different from adaptive sensing, we assume that the non-uniformly sparse statistics are given a priori, which can be viewed as a simplification of adaptive sensing. As we shall show later, this simplification allows a closed form formula to compute the asymptotically optimal power allocation policy under certain assumptions.

Our technique originates from the so-called approximate message passing (AMP) algorithm and the associated analysis developed by Donoho et al. [31]. AMP assumes no power allocation, that is, the entries of the measurement matrix are generated from i.i.d. Gaussian random variables. The key element of the theoretical analysis is the so called state evolution. It quantifies exactly the under-sampling rates when perfect reconstruction is possible (referred as the phase transition curve [32]), or the worst-case reconstruction mean squared error (MSE) for a given noise variance (referred to as minimax MSE) [34]. The same technique has been applied to non-uniformly sparse signals in [92] and block separable signals in [35], and also been extended to more general channel models [85, 86]. With power allocation, the measurement matrix in this chapter does not contain i.i.d. Gaussian entries. It can be viewed as special cases of the generalised channel model.

The main contribution of this chapter is the asymptotically optimal power allocation to minimize the reconstruction MSE. More specifically, we revise the standard AMP algorithm to accommodate non-uniformly sparse signals and Gaussian measurement matrices with power allocation. The reconstruction MSE of the revised AMP algorithm has been exactly quantified in an asymptotic regime. Based on it, the asymptotically optimal power allocation policy is derived. Note that the presented analysis is mainly for the worst case as it results in closed-form formulas. The analysis can be generalised for more practical scenarios with minor modifications and produce satisfactory results according to our

simulations.

4.2 Introduction of Approximate Message Passing

Convex optimization algorithms are widely used in compressed sensing and achieve fairly good sparsity-undersampling tradeoff. Yet these algorithms can be computationally expensive which is intolerable in important large-scale applications. Fast iterative thresholding algorithms are studied as alternatives to convex optimization for large-scale problems. Unfortunately they do not provide sparsity-undersampling tradeoffs as good as convex optimization. Approximate message passing algorithm (AMP) proposed in [31] is the first algorithm that offers both the low complexity of IST and the reconstruction power of basis pursuit. It has been proved to be effective in reconstructing sparse signals from a small number of incoherent linear measurements.

The AMP framework involves a soft thresholding function and the associated MSE analysis. Consider a scalar system $y = x + w$ where $x \sim p_\epsilon$ and $w \sim \mathcal{N}\left(0, \sigma^2\right)$. Given vector y, AMP employs the soft thresholding function

$$\hat{x} = \eta\left(y;\ \theta\right) \triangleq \begin{cases} y - \theta & \text{if } y > \theta, \\ y + \theta & \text{if } y < -\theta, \\ 0 & \text{otherwise,} \end{cases} \tag{4.2}$$

to estimate x, where $\theta \geq 0$ is a threshold. Consider the reconstruction MSE

$$M\left(p_\epsilon, \sigma^2\right) = \inf_{\theta \geq 0} \mathbb{E}\left\{(\hat{x} - x)^2\right\},$$

where the threshold θ is optimally chosen for the given prior distribution p_ϵ and noise variance σ^2. Introduce the three-point mixture dustribution

$$p_{\epsilon,\mu} = \frac{\epsilon}{2}\delta_{-\mu} + (1 - \epsilon)\,\delta_0 + \frac{\epsilon}{2}\delta_{+\mu}, \tag{4.3}$$

where δ_c is the Delta function centered at c. It can be shown that among all sparse distributions in the family of $\mathcal{F}_\epsilon$ (4.14), the (*worst*) one that results in the maximum recon-

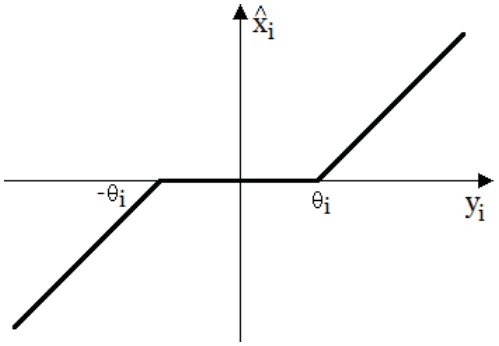

Figure 4.1: Soft thresholding function with threshold θ_i.

struction MSE is when $\mu = \infty$. Denote the worst case (*least favorable*) prior distribution by $p_\epsilon^\#$ ($p_\epsilon^\# = p_{\epsilon,\infty}$). The associated reconstruction MSE has the nice property

$$M\left(p_\epsilon^\#, \sigma^2\right) = \sigma^2 M\left(p_\epsilon^\#, 1\right) = \sigma^2 M^\#(\epsilon), \tag{4.4}$$

where $M^\#(\epsilon) \triangleq M\left(p_\epsilon^\#, 1\right)$ is introduced to simplify the notations and referred to as *minimax MSE*. A closed form to compute $M^\#(\epsilon)$ for an $\epsilon \in (0,1)$ has been given in [74]. The optimal threshold is of the form $\theta = \alpha\sigma$ where α is a constant only dependent on nonzero probability ϵ.

Remark 4.1. To analyse the more general case, the three-point mixture $p_{\epsilon,\mu}$ with finite μ becomes important. The associated scaling rule is given by $M\left(p_{\epsilon,\mu}, \sigma^2\right) = \sigma^2 M\left(p_{\epsilon,\mu/\sigma}, 1\right)$, and reconstruction MSE of $\sigma^2 = 1$ also has an explicit form. Despite the nice forms for the scalar case, the state evolution for overall performance analysis turns out more complicated.

Based on the results for the scalar case, the AMP algorithm to recover sparse $\boldsymbol{x}$ from compressed sensing measurements (4.1) has been derived [31]:

$$\boldsymbol{x}^{t+1} = \eta\left(\boldsymbol{x}^t + \boldsymbol{A}^T\boldsymbol{r}^t;\ \boldsymbol{\theta}^t\right), \tag{4.5}$$

$$\boldsymbol{r}^t = \boldsymbol{y} - \boldsymbol{A}x^t + \frac{1}{m}\left\|\boldsymbol{x}^t\right\|_0 \boldsymbol{r}^{t-1}, \tag{4.6}$$

where the superscript t denotes the t-th iteration. As n, $m \to \infty$ simultaneously with a constant ratio $m/n \to \delta$, a closed-form formula to compute the minimax MSE $\frac{1}{n}\mathbb{E}\left\{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2^2\right\}$ has been derived in [7]. It is noteworthy that the algorithm (4.5,4.6) and the analysis are based on the assumption that the matrix $\boldsymbol{A}$ contains i.i.d. Gaussian entries.

Consider the signal model (4.1). Compared to IST algorithm shown in, AMP only adds one additional term $\frac{1}{m}\left\|\boldsymbol{x}^t\right\|_0 \boldsymbol{r}^{t-1}$.

$$\boldsymbol{x}^{t+1} = \eta\left(\boldsymbol{x}^t + \boldsymbol{A}^T\boldsymbol{r}^t;\ \boldsymbol{\theta}^t\right), \tag{4.7}$$

$$\boldsymbol{r}^t = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^t, \tag{4.8}$$

Further denote the sparsity value by $\rho = \frac{k}{m}$ and the sampling rate by $\delta = \frac{m}{n}$. Extensive numerical work reported in [74] show that AMP achieves a $\rho - \delta$ (sparsity-sampling) tradeoff matching the theoretical tradeoff which has been proved for LP-based reconstruction (e.g. basis pursuit). Referring to Figure 1, [30], the $\rho - \delta$ parameter space are separated into two areas by the phase boundary curve: one where the massing passing approach is successful in accurately reconstruction and the other where it is unsuccessful. These curves is shown to separate the $\rho - \delta$ space identically to the ones from LP-based reconstruction. A strong theoretical support called *state evolution* formalism is also available to accurately predict the dynamical behavior of numerous observables of the AMP algorithm.

The choices of $\boldsymbol{\theta}^t$ and $b^t = \frac{1}{\delta}\eta'\left(x^{t-1} + \boldsymbol{A}^T r^{t-1};\ \boldsymbol{\theta}^{t-1}\right)$ is closely connected with the derivations of AMP. More generally, they also have tight connection between AMP and LASSO.

Proposition 4.2. *Let* $(\boldsymbol{x}^*,\ \boldsymbol{r}^*)$ *be a fixed point of the iteration* (4.7) *and* (4.8) *for* $\boldsymbol{\theta}^t = \boldsymbol{\theta}$, $b^t = b$ *fixed. Then* $\boldsymbol{x}^*$ *is a minimum of the LASSO cost function for* $\boldsymbol{\lambda} = \boldsymbol{\theta}\,(1-b)$.

Proof. From equ. (4.7), we get the fixed point condition

$$\boldsymbol{x}^* + \boldsymbol{\theta}\frac{\partial\left\|\boldsymbol{x}^*\right\|_1}{\partial \boldsymbol{x}^*} = \boldsymbol{x}^* + \boldsymbol{A}^T r^t.$$

From equ. (4.8), we get

$$\boldsymbol{r}^*\,(1-b) = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^*.$$

Combine the two equations, we get

$$(1-b)\,\boldsymbol{\theta}\frac{\partial\left\|\boldsymbol{x}^*\right\|_1}{\partial \boldsymbol{x}^*} = \boldsymbol{A}^T\left(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^*\right).$$

Therefore we can set $\boldsymbol{\lambda} = \boldsymbol{\theta}(1-b)$. □

It is mentioned in [74] that if a found sequence $\left(\boldsymbol{x}^t,\ b^t\right)$ that converge and the estimates $\boldsymbol{x}^t$ converges as well, then it is guaranteed that the limit of the AMP iteration is a LASSO optimum.

4.3 State Evolution and the Phase Transition Boundary

In this section we introduce theorem 4.3. It shows that the behavior of AMP algorithms can be monitored via state evolution iterations. Definition 4.4 shows the necessary conditions on signals, measurement matrices and noise where rigorous proof of the AMP performance is possible.

Theorem 4.3. *[6] Let* $\{\boldsymbol{x}(n),\ \boldsymbol{w}(n)\,, \boldsymbol{A}(n)\}_{n\in\mathbb{N}}$ *be a converging sequence of instance with the entries of* $\boldsymbol{A}(n)$ *i.i.e. normal with mean zero and variance* $1/m$*, while the signal* $\boldsymbol{x}(n)$ *and noise vectors* $\boldsymbol{w}(n)$ *satisfy the hypotheses of Definition 1. Let* $\psi_1 :\ \mathbb{R}\to\mathbb{R}$*,* $\psi_2 :\ \mathbb{R}\times\mathbb{R}\to\mathbb{R}$ *be pseudo-Lipchitz functions. Finally, let* $\left\{\boldsymbol{x}^t\right\}$*,* $\left\{\boldsymbol{r}^t\right\}$ be the sequence of estimate and residuals produced by AMP. Then almost surely

$$\lim_{n\to\infty}\frac{1}{m}\sum_{a=1}^{m}\psi_1\left(r_a^t\right) = \mathbb{E}\left\{\psi_1\left(\tau_t Z\right)\right\},$$

$$\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\psi_2\left(\boldsymbol{x}_i^{t+1},\ \boldsymbol{x}_i\right) = \mathbb{E}\left\{\psi_2\left(\eta\left(X_0+\tau_t Z\right);\theta_t\right),\ X_0\right\},$$

where $Z\sim(0,1)$ is independent of $X_0\sim p_0$.

Definition 4.4. The sequence of instances $\{\boldsymbol{x}(n),\ \boldsymbol{w}(n)\,, \boldsymbol{A}(n)\}_{n\in\mathbb{N}}$ indexed by n is said to be a converging sequence if $\boldsymbol{x}(n)\in\mathbb{R}^n$, $\boldsymbol{w}(n)\in\mathbb{R}^m$, $\boldsymbol{A}(n)\in\mathbb{R}^{m\times n}$ with $m=m(n)$ is such that $m/n\to\delta\in(0,\ \infty)$, and in addition the following conditions hold

1. The empirical distribution of the entries of $\boldsymbol{x}(n)$ converges weakly to a probability measure p_0 on $\mathbb{R}$ with bounded second moment. Further $n^{-1}\sum_{i=1}^{n}x_i(n)^2\to\mathbb{E}_{p_0}\left\{X_0^2\right\}$.

2. The empirical distribution of the entries of $\boldsymbol{w}(n)$ converges weakly to a probability measure p_W on $\mathbb{R}$ with bounded second moment. Further $m^{-1}\sum_{i=1}^{m}x_i(n)^2\to$

$\mathbb{E}_{p_W}\left\{W^2\right\} \equiv \sigma^2$.

3. If $\{\boldsymbol{e}_i\}_{1\leq i\leq n}$, $\boldsymbol{e}_i \in \mathbb{R}^n$ denotes the canonical basis, then $\lim_{n\leftarrow\infty} \min_{i\in[n]} \ \|\boldsymbol{A}(n)\boldsymbol{e}_i\|_2 = 1$.

The rigorous proof is shown in [8] which supports the rationality of the state evolution. We will clarify it in the following. It will be extended to our future study. Consider the IST algorithm and recognize the input to the denoiser.

$$\begin{aligned}\boldsymbol{x}^t + \boldsymbol{A}^T\boldsymbol{r}^t &= \boldsymbol{x}^t + \boldsymbol{A}^T\left(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^t\right)\\ &= \boldsymbol{x}^t + \boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{x}^{\text{true}} + \boldsymbol{w} - \boldsymbol{A}\boldsymbol{x}^t\right)\\ &= \boldsymbol{x}^t + \left(\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right)\left(\boldsymbol{x}^{\text{true}} - \boldsymbol{x}^t\right) + \boldsymbol{A}^T\boldsymbol{w}.\end{aligned}$$

Denote $\boldsymbol{e}^t = \left(\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right)\left(\boldsymbol{x}^{\text{true}} - \boldsymbol{x}^t\right) + \boldsymbol{A}^T\boldsymbol{w}$. Then

$$\boldsymbol{x}^{t+1} - \boldsymbol{x}^{\text{true}} = \eta\left(\boldsymbol{x}^t + \boldsymbol{A}^T\boldsymbol{r}^t;\ \boldsymbol{\theta}^t\right) - \boldsymbol{x}^{true}. \tag{4.9}$$

Each entry of $\left(\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right) \sim \mathcal{N}(0,\ 1/m)$. Then we have for $\boldsymbol{e}^t$: 1) $\mathbb{E}\left\{\boldsymbol{e}_i^t\right\} = 0$; 2) $\mathbb{E}\left\{\boldsymbol{e}_i^t\boldsymbol{e}_j^t\right\} = 0,\ i \neq j$; 3)

$$\mathbb{E}\left\{\left|\boldsymbol{e}_i^t\right|^2\right\} = \frac{1}{m}\left\|\boldsymbol{x}^{\text{true}} - \boldsymbol{x}^t\right\|_2^2 + \sigma^2.$$

The denoising step using function $\eta(\cdot)$ for very large n will be

$$\boldsymbol{x}^{t+1} - \boldsymbol{x}^{\text{true}} = \eta\left(\boldsymbol{x}^{\text{true}} + \tau_t\boldsymbol{z};\ \boldsymbol{\theta}^t\right) - \boldsymbol{x}^{\text{true}}. \tag{4.10}$$

where $\boldsymbol{z} \sim (0,\ \boldsymbol{I})$ and the variance parameters

$$\tau_t^2 \triangleq \frac{1}{m}\mathbb{E}\left\{\left\|\boldsymbol{x}^{\text{true}} - \boldsymbol{x}\right\|_2^2\right\} + \sigma^2. \tag{4.11}$$

$$\tau_{t+1}^2 = \frac{1}{m}\mathbb{E}\left\{\left\|\eta\left(\boldsymbol{x}^{\text{true}} + \tau^t\boldsymbol{z};\ \boldsymbol{\theta}_i^t\right) - \boldsymbol{x}^{\text{true}}\right\|_2^2\right\} + \sigma^2. \tag{4.12}$$

The above two equations give the state evolution of AMP algorithms.

To do further study based on the state evolution, one set $\sigma^2 = 0$ and define the soft thresholding MSE per coordinate as

$$\operatorname{mse}\left(\tau_t^2;\ p_0,\ \alpha\right) \triangleq \frac{1}{n}\mathbb{E}\left\{\left[\eta\left(\boldsymbol{x}^{\text{true}} + \tau_t \boldsymbol{z};\ \alpha\boldsymbol{\tau}_t\right) - \boldsymbol{x}^{\text{true}}\right]^2\right\}.$$

Note that $\mathcal{F}_{\epsilon_i}$ is scale invariant,

$$\begin{aligned}&\sup_{p_0 \in \mathcal{F}_\epsilon} \operatorname{mse}\left(\tau_t^2;\ p_0,\ \alpha\right)\\ &= \sup_{p_0 \in \mathcal{F}_\epsilon} \frac{1}{n}\tau_t^2 M\left(\epsilon, \alpha\right)\end{aligned} \tag{4.13}$$

where $p^{\#} = \frac{\epsilon}{2}\delta_{-\infty} + (1-\epsilon)\,\delta_0 + \frac{\epsilon}{2}\delta_{+\infty}$.

Minimizing $M\left(\epsilon, \alpha\right)$, the limit is

$$\lim_{\epsilon \to 0} \frac{\alpha^{\#}}{\sqrt{2\log\left(\epsilon^{-1}\right)}} = 1.$$

Define the minimax threshold MSE

$$M^{\#}\left(\epsilon\right) = \inf_{\alpha > 0} M\left(\epsilon, \alpha\right).$$

The infimum is obtained when $p = \frac{\epsilon}{2}\delta_{-\infty} + (1-\epsilon)\,\delta_0 + \frac{\epsilon}{2}\delta_{+\infty}$.

$$\begin{aligned}M^{\#}\left(\epsilon\right) &= \mathbb{E}\left\{\left[\eta\left(\boldsymbol{x} + z;\ \alpha\right) - \boldsymbol{x}\right]^2\right\}\\ &= \left\{\epsilon\left(1+\alpha^2\right) + (1-\epsilon)\left[2\left(1+\alpha^2\right)\Phi\left(-\alpha\right) - 2\alpha\phi\left(\alpha\right)\right]\right\}.\end{aligned}$$

Combine (4.11) and (4.13). Then for each $\delta \in [0,\ 1]$, let $\rho_{\text{MSE}}\left(\delta\right)$ be the value of ρ solving

$$M^{\#}\left(\rho\delta\right) = \delta.$$

The explicit expression for the phase boundary curve $(\delta,\ \rho_{\mathrm{MSE}}(\delta))$ is provided in the following.

$$\begin{cases} G_\epsilon(\alpha) = \left\{\epsilon\left(1+\alpha^2\right) + (1-\epsilon)\left[2\left(1+\alpha^2\right)\Phi(-\alpha) - 2\alpha\phi(\alpha)\right]\right\} = \delta \\ \frac{\partial G_\epsilon(\alpha)}{\partial \alpha} = \left\{2\alpha\epsilon + (1-\epsilon)\left[4\alpha\Phi(-\alpha) + 2\left(1+\alpha^2\right)\Phi'(-\alpha) - 2\phi(\alpha) - 2\alpha\phi'(\alpha)\right]\right\} = 0 \end{cases}$$

where $\phi(\alpha) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\alpha^2}{2}}$ and $\Phi(\alpha) = \int_{-\infty}^{\alpha} \phi(x)\, dx$. It is easy to derive that $\phi(\alpha) = \alpha\phi(\alpha)$ and $\Phi'(\alpha) = \phi(\alpha)$. Thus the solution of the above equations is

$$\begin{cases} \delta = \frac{2\phi(\alpha)}{\alpha + 2(\phi(\alpha) - \alpha\Phi(-\alpha))} \\ \rho = \frac{\phi(\alpha) - \alpha\Phi(-\alpha)}{\phi(\alpha)} \end{cases}.$$

4.4 A Simple Example

In standard compressed sensing (CS) settings, the entries of the measurement matrix $\boldsymbol{A}$ are generated from i.i.d. Gaussian random variables. However, this may not be optimal in terms of reconstruction distortion when the unknown signal $\boldsymbol{x}$ is non-uniformly sparse, i.e., the probabilities for different entries to be nonzero may be different. Consider the example where $\boldsymbol{x} = [\boldsymbol{x}_{\mathcal{I}_1},\ \boldsymbol{x}_{\mathcal{I}_2}]$ and the entries in $\boldsymbol{x}_{\mathcal{I}_1},\ \boldsymbol{x}_{\mathcal{I}_2} \in \mathbb{R}^{n/2}$ have different nonzero probabilities. In an extreme case, suppose that the entries in $\boldsymbol{x}_{\mathcal{I}_1}$ share the same prior distribution with strictly positive nonzero probability while all the entries in $\boldsymbol{x}_{\mathcal{I}_2}$ are zeros. Fix the total power budget, i.e., the squared ℓ_2-norm of each row of the measurement matrix is fixed to a constant. Different from the equal power allocation in standard compressed sensing, a more sensible way is to spend no sensing power on the zero components in $\boldsymbol{x}_{\mathcal{I}_2}$ but allocate all sensing power evenly to the columns corresponding to $\boldsymbol{x}_{\mathcal{I}_1}$. In Figure 4.2, we generate an extreme case example by assuming that the sparse signal has two equal length parts, and the second part contains all zeros. Then we use two different sensing matrices for the sampling. The middle figure shows the reconstruction from the observation where a uniformly random Gaussian sensing matrix is used. The bottom figure shows the reconstruction where only the left half of the sensing matrix is randomly Gaussian and the rest is zero. The results show an obvious advantage by adapting the non-uniformly sensing matrix for the compressed sensing.

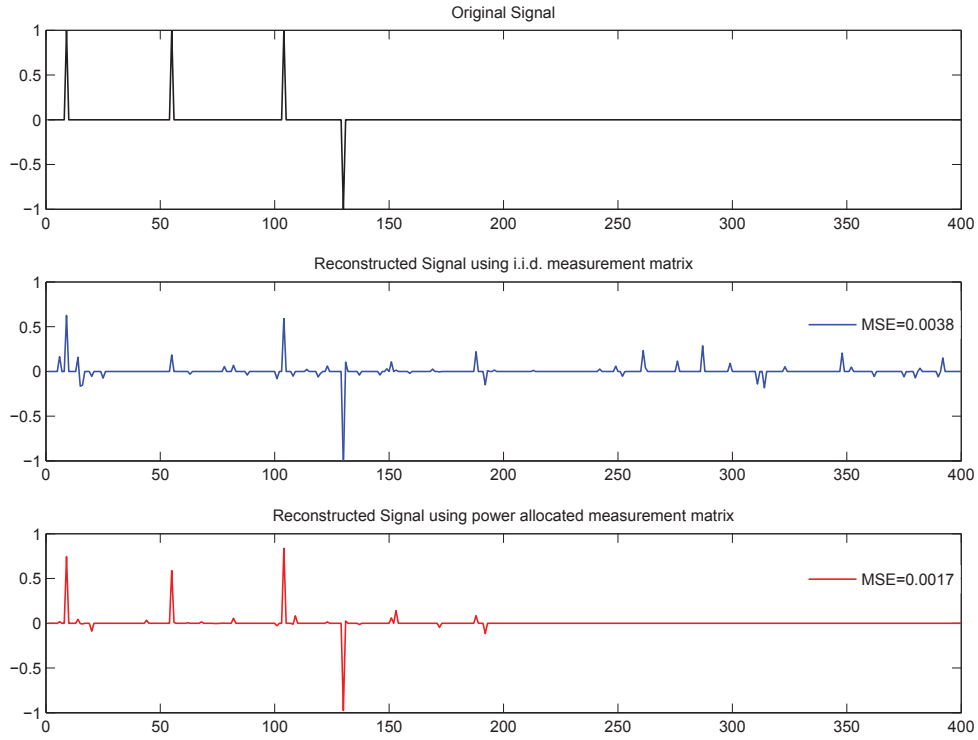

Figure 4.2: Intuitive example for reconstructing two-block non-uniformly sparse signals. Top figure: original signal; middle figure: reconstruction with equal allocation on $\boldsymbol{A}_{\mathcal{I}_1}$ and $\boldsymbol{A}_{\mathcal{I}_2}$; bottom figure: reconstruction with all power allocated on $\boldsymbol{A}_{\mathcal{I}_2}$.

4.5 Revised AMP with Given Power Allocation

Based on the above simple example which motivates our study, we give the formal setting of the problem as follows. Let

$$\mathcal{F}_\epsilon = \{p : \ p\{0\} = 1 - \epsilon\} \tag{4.14}$$

be the family of probability distribution with a mass $1-\epsilon$ at zero. Assume a block-sparsity signal $\boldsymbol{x} = [\boldsymbol{x}_{\mathcal{I}_1}; \boldsymbol{x}_{\mathcal{I}_2}; ...; \boldsymbol{x}_{\mathcal{I}_s}]$ where $p_{\epsilon_i} \in \mathcal{F}_{\epsilon_i}$ and $p_{\epsilon_i} = p_{\epsilon_j}$ if $i,\ j \in \mathcal{I}_k$, $k \in [s]$. For the purpose of power allocation, suppose that each column of $\boldsymbol{A}$, denoted by $\boldsymbol{A}_i$, $i \in [n]$, contains entries generated from i.i.d. Gaussian random variables with $\mathcal{N}\left(0,\ \sigma_i^2/m\right)$. Fix a total power budget $\Sigma_{i=1}^n \sigma_i^2 = n$. The goal is to minimize the reconstruction error subject

to the total power budget,

$$\min_{\sigma_1^2,\cdots,\sigma_n^2} \frac{1}{n}\mathbb{E}\left\{\|\hat{\boldsymbol{x}}-\boldsymbol{x}\|_2^2\right\}, \text{ s.t. } \sum_{i=1}^{n}\sigma_i^2 = n, \tag{4.15}$$

where $\hat{\boldsymbol{x}}$ is the compressed sensing reconstruction.

When coming to power allocation, the original AMP algorithm (4.5,4.6) needs to be tailored. It has been assumed that a column of $\boldsymbol{A}$, say $\boldsymbol{A}_i$, contains entries generated from i.i.d. $\mathcal{N}\left(0,\ \sigma_i^2/m\right)$. The original AMP is not optimal any more as different columns may have different ℓ_2-norm. The revised AMP, termed as AMP.P($\boldsymbol{\epsilon}$), is given by

$$\boldsymbol{x}^{t+1} = \eta\left(\boldsymbol{x}^t + \boldsymbol{\Theta}^{-2}\boldsymbol{A}^T\boldsymbol{r}^t;\ \boldsymbol{\Theta}^{-1}\boldsymbol{\theta}^t\right), \tag{4.16}$$

$$\boldsymbol{r}^t = \boldsymbol{y} - \boldsymbol{A}x^t + \frac{1}{m}\left\|\boldsymbol{x}^t\right\|_0\boldsymbol{r}^{t-1}, \tag{4.17}$$

where $\boldsymbol{\Theta}^2 \triangleq \text{diag}\left(\sigma_1^2,\sigma_2^2,...,\sigma_n^2\right)$. The major difference from the standard one is the terms $\boldsymbol{\Theta}^{-2}$ and $\boldsymbol{\Theta}^{-1}$ in (4.16). It is noteworthy that the revised AMP is not particularly designed for the worst case though the later analysis is.

4.5.1 Derivations

The derivation of the AMP.P($\boldsymbol{\epsilon}$) follows from the same idea behind the standard AMP [74]. Describe the statistical relationship between $\boldsymbol{x}$ and $\boldsymbol{y}$ by a bipartite graph, which includes variable nodes indexed by $i \in [n]$ for variables x_i and factor nodes indexed by $a \in [m]$ corresponding to observations y_a. Denote the message passed from the factor node a to the variable node i by $r_{a\to i}^t$ and that from the variable node i to the factor node a by $x_{i\to a}^t$, where the superscript t denotes the t^{th} iteration. It can be verified that [74]

$$r_{a\to i}^t = y_a - \sum_{j\in[n]\backslash i} A_{aj}x_{j\to a}^t, \tag{4.18}$$

$$x_{i\to a}^{t+1} = \frac{1}{\sigma_i^2}\eta_t\left(\sum_{b\in[m]\backslash a} A_{bi}r_{b\to i}^t\right), \tag{4.19}$$

where for notational convenience, $\eta\left(\cdot,\theta_t\right)$ is simplified to $\eta_t\left(\cdot\right)$ henceforth. The crux of the AMP is to approximate these messages so that the computational complexity can be significantly reduced.

In the approximation, only $\mathcal{O}(1)$ and $\mathcal{O}\left(n^{-1/2}\right)$ terms are kept and all smaller terms are omitted. Here, it is assumed that both n and m are large and $\delta \triangleq m/n$ is a constant strictly positive. Since $A_{a,i} \sim \mathcal{N}\left(0, \sigma_i^2/m\right)$, it is clear $A_{a,i}$ is of $\mathcal{O}\left(n^{-1/2}\right)$. Note that $r_{a\to i}^t = y_a - \sum_{j\in[n]} A_{aj} x_{j\to a}^t + A_{ai} x_{i\to a}^t$ where only the last term (of $\mathcal{O}\left(n^{-1/2}\right)$) depends on i. One can write $r_{a\to i}^t = r_a^t + \delta r_{a\to i}^t$ where r_a^t is of $\mathcal{O}(1)$ and both $\delta r_{a\to i}^t$ is of $\mathcal{O}\left(n^{-1/2}\right)$. By similar arguments, it holds that $x_{i\to a}^t = x_i^t + \delta x_{i\to a}^t$, where again, x_i^t is of $\mathcal{O}(1)$ and $\delta x_{i\to a}^t$ is of $\mathcal{O}\left(n^{-1/2}\right)$. Keeping only $\mathcal{O}(1)$ and $\mathcal{O}\left(n^{-1/2}\right)$ terms, the equations (4.18) and (4.19) become

$$r_a^t + \delta r_{a\to i}^t = y_a - \sum_{j\in[n]} A_{aj}\left(x_j^t + \delta x_{j\to a}^t\right) + A_{ai} x_i^t, \tag{4.20}$$

$$x_i^{t+1} + \delta x_{i\to a}^{t+1} = \frac{1}{\sigma_i^2}\eta_t\left(\sum_{b\in[m]} A_{bi}\left(r_b^t + \delta r_{b\to i}^t\right) - A_{ai} r_a^t\right). \tag{4.21}$$

From (4.20), it is straightforward to recognize that

$$r_a^t = y_a - \sum_{j\in[n]} A_{aj}\left(x_j^t + \delta x_{j\to a}^t\right); \tag{4.22}$$

$$\delta r_{a\to i}^t = A_{ai} x_i^t. \tag{4.23}$$

By Taylor expansion of $\eta_t(\cdot)$, Equation (4.21) becomes

$$\begin{aligned} x_i^t + \delta x_{i\to a}^t = {} & \frac{1}{\sigma_i^2}\eta_t\left(\sum_{b\in[m]} A_{bi}\left(r_b^t + \delta r_{b\to i}^t\right)\right) + \\ & \frac{1}{\sigma_i^2} A_{ai} r_a^t \eta_t'\left(\sum_{b\in[m]} A_{bi}\left(r_b^t + \delta r_{b\to i}^t\right)\right), \end{aligned} \tag{4.24}$$

from which it is clear that

$$x_i^{t+1} = \frac{1}{\sigma_i^2}\eta_t\left(\sum_{b\in[m]} A_{bi}\left(r_b^t + \delta r_{b\to i}^t\right)\right); \tag{4.25}$$

$$\delta x_{i\to a}^{t+1} = \frac{1}{\sigma_i^2} A_{ai} r_a^t \eta_t'\left(\sum_{b\in[m]} A_{bi}\left(r_b^t + \delta r_{b\to i}^t\right)\right). \tag{4.26}$$

Substitute (4.23) into (4.25) and (4.26) into (4.22). Again omit the terms smaller than

$\mathcal{O}\left(n^{-1/2}\right)$. We have

$$x_i^{t+1} = \frac{1}{\sigma_i^2}\eta_t\left(\sigma_i^2 x_i^t + \left(\boldsymbol{A}^T\boldsymbol{r}^t\right)_i\right), \tag{4.27}$$

$$r_a^t = y_a - \sum_{j\in[n]} A_{aj}x_j^t + \sum_{j\in[n]} \frac{A_{aj}^2}{\sigma_j^2}\eta'_{t-1}\left(\sigma_j^2 x_j^{t-1} + \left(\boldsymbol{A}^T\boldsymbol{r}^{t-1}\right)_j\right) r_a^{t-1}. \tag{4.28}$$

Note that for large n, $\boldsymbol{A}_{aj}^2 \approx \sigma_j^2/m$. The last term on the right hand side of Equation (4.28) can be approximated as

$$\sum_{j\in[n]} \frac{1}{m}\eta'_{t-1}\left(\sigma_j^2 x_j^{t-1} + \left(\boldsymbol{A}^T\boldsymbol{r}^{t-1}\right)_j\right) r_a^{t-1} = \frac{1}{m}\left\|\boldsymbol{x}^t\right\|_0 r_a^{t-1}. \tag{4.29}$$

Combine Equation (4.27), (4.28), and (4.29). We obtain the AMP.P($\boldsymbol{\epsilon}$) iterations described by (4.16) and (4.17).

4.6 Reconstruction MSE and A Heuristic Derivation

We analyze the MSE performance of AMP.P($\boldsymbol{\epsilon}$). We focus on the minimax MSE as the analysis can be highly simplified thanks to the property (4.4). As the rigorous analysis [7] is still too arduous, we follow the heuristic proof in [74] which is much easier to describe and highlights the key ideas.

The main results can be summarized as follows. Consider the asymptotic region where $(m,n) \to \infty$ simultaneously with a constant ratio $m/n \to \delta$. Assume the block sparsity structure described before with $|\mathcal{I}_i|/n \to c_i$ for some constant c_i. Consider the least favorable prior $p^{\#}(\epsilon_i)$, $i \in [n]$, and suppose that $\lim_{(m,n)\to\infty} \frac{1}{m}\sum_{i=1}^{n} M^{\#}(\epsilon_i) < 1$. The minimax MSE of the revised AMP algorithm is given by

$$\frac{1}{n}\mathbb{E}\left\{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2^2\right\} \doteq \frac{\frac{1}{n}\sum_{i=1}^{n} M^{\#}(\epsilon_i)/\sigma_i^2}{1 - \frac{1}{m}\sum_{i=1}^{n} M^{\#}(\epsilon_i)}\sigma^2, \tag{4.30}$$

where the symbol $\doteq$ denotes the equality in the aforementioned asymptotic region.

Remark 4.5 (*Relation with the Previous Result*). Consider the uniformly sparse signal $\boldsymbol{x}$

with $\epsilon_i = \epsilon_j$ for all $i, j \in [n]$. The minimax MSE in (4.30) becomes

$$\frac{M^{\#}(\epsilon)}{1 - M^{\#}(\epsilon)/\delta}\sigma^2,$$

which consists with the result given in [34].

Remark 4.6 (*Phase-Transition for the Noiseless Case*). For noiseless case, $\sigma^2 = 0$. Consider the same asymptotic region as specified before with additionally $\Sigma_{i=1}^{n}\epsilon_i/m \to \rho$. The phase-transition curve that separates the sparsity-undersampling $(\rho - \delta)$ plane [74] is given by

$$\frac{1}{n}\sum_{i=1}^{n} M^{\#}(\epsilon_i) \doteq \delta.$$

That is, the reconstruction is exact if and only if $\frac{1}{n}\sum M^{\#}(\epsilon) < \delta$. This result is consistent with the one in [92]. Furthermore, note the phase transition curve is independent of σ_i^2. It can be concluded that power allocation will not affect the phase transition curve when there is no noise.

4.6.1 The heuristic derivation

The heuristic derivation of (4.30) starts with the iterative algorithm that the term $\frac{1}{m}\left\|\boldsymbol{x}^t\right\|_0 \boldsymbol{r}^{t-1}$ in (4.17) is omitted, i.e.,

$$\boldsymbol{x}^{t+1} = \eta_t\left(\boldsymbol{x}^t + \boldsymbol{\Theta}^{-2}\boldsymbol{A}^T\boldsymbol{r}^t\right), \tag{4.31}$$

$$\boldsymbol{r}^t = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^t. \tag{4.32}$$

Meantime, it also poses an artificial assumption that the matrix $\boldsymbol{A}$ at different iterations are independently generated. Note in reality the matrix $\boldsymbol{A}$ is fixed for all the iterations. The heuristic derivation gives the correct analysis as adding term (4.29) will make the residue noise from different iterations independent.

To proceed, the input of the thresholding function in (4.31) can be written as

$$\begin{aligned}\boldsymbol{x}^t + \boldsymbol{\Theta}^{-2}\boldsymbol{A}^T\boldsymbol{r}^t &= \boldsymbol{x}^t + \boldsymbol{\Theta}^{-2}\boldsymbol{A}^T\left(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^t\right)\\ &= \boldsymbol{x} + \boldsymbol{e}^t,\end{aligned} \tag{4.33}$$

where $\boldsymbol{e}^t \triangleq \left(\boldsymbol{\Theta}^{-2}\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right)\left(\boldsymbol{x} - \boldsymbol{x}^t\right) + \boldsymbol{\Theta}^{-2}\boldsymbol{A}^T\boldsymbol{w}$. The explicit form of the matrix $\left(\boldsymbol{\Theta}^{-2}\boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{I}\right)$ in $\boldsymbol{e}^t$ is

$$\begin{bmatrix} \sigma_1^{-2}\boldsymbol{A}_1^T\boldsymbol{A}_1 - 1 & \sigma_1^{-2}\boldsymbol{A}_1^T\boldsymbol{A}_2 & \cdots \\ \sigma_2^{-2}\boldsymbol{A}_2^T\boldsymbol{A}_1 & \sigma_2^{-2}\boldsymbol{A}_2^T\boldsymbol{A}_2 - 1 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}.$$

It can be verified that each diagonal entry $\sigma_i^{-2}\boldsymbol{A}_i^T\boldsymbol{A}_i - 1$ is approximately normal with zero mean and variance $2/m$; each off-diagonal entry $\sigma_i^{-2}\boldsymbol{A}_i^T\boldsymbol{A}_j$, $i \neq j$, has zero mean and variance $\sigma_i^{-2}\sigma_j^2/m$. By the fact that $\boldsymbol{w} \sim \mathcal{N}\left(0,\, \sigma^2\boldsymbol{I}\right)$, the following properties hold: 1) $\mathbb{E}\left\{e_i^t\right\} = 0$; 2) $\mathbb{E}\left\{e_i^t e_j^t\right\} = 0$, $i \neq j$; 3) for large n, define $\tilde{\tau}_{t,i}^2 \triangleq \mathbb{E}\left\{\left|e_i^t\right|^2\right\}$, where

$$\mathbb{E}\left\{\left|e_i^t\right|^2\right\} \doteq \frac{1}{\sigma_i^2}\left(\sum_{j=1}^{n}\frac{\sigma_j^2}{m}\mathbb{E}\left\{\left|x_j - x_j^t\right|_2^2\right\} + \sigma^2\right).$$

This helps in quantifying the MSE at the $(t+1)^{th}$ iteration:

$$\tilde{\tau}_{t+1,i}^2 \doteq \frac{1}{\sigma_i^2}\left(\sum_{j=1}^{n}\frac{\sigma_j^2}{m}\mathbb{E}\left\{\left|x_j - \eta_t\left(x_j + e_j^t\right)\right|_2^2\right\} + \sigma^2\right).$$

From the definition of $M^{\#}\left(\epsilon_j\right)$ in (4.4),

$$\mathbb{E}\left\{\left|x_j - \eta_t\left(x_j + e_j^t\right)\right|_2^2\right\} = M^{\#}\left(\epsilon_j\right)\tilde{\tau}_{t,j}^2. \tag{4.34}$$

As a result, when the steady state ($\tilde{\tau}_{t,j} = \tilde{\tau}_{t+1,j}$) is reached,

$$\tilde{\tau}_i^2 \doteq \frac{1}{\sigma_i^2}\left(\frac{1}{m}\sum_{j=1}^{n}\sigma_j^2 M^{\#}\left(\epsilon_j\right)\tilde{\tau}_j^2 + \sigma^2\right),\ i \in [n]. \tag{4.35}$$

The explicit form to compute $\tilde{\tau}_i^2$ can be computed by observing that for all $i \in [n]$,

$\tilde{\tau}_i^2\sigma_i^2 = \sum_{j=1}^{n} \frac{\sigma_j^2}{m} M^{\#}(\epsilon_j)\,\tilde{\tau}_j^2 + \sigma^2$ which is a constant independent of i. Hence,

$$\tilde{\tau}_i^2 \doteq \frac{\sigma^2}{\sigma_i^2} \cdot \frac{1}{1 - \frac{1}{m}\sum_{i=1}^{n} M^{\#}(\epsilon_i)},\ i \in [n]. \tag{4.36}$$

Combine (4.36) with the state evolution (4.34). We obtain

$$\frac{1}{n}\mathbb{E}\left\{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2^2\right\} \doteq \frac{1}{n}\sum_{i=1}^{n} M^{\#}(\epsilon_i)\,\tilde{\tau}_i^2,$$

which gives (4.30).

4.7 Optimal Power Allocation Strategy

Based on the derived minimax MSE, the optimal power allocation can be achieved. In particular, the power allocation can be formulated as a constrained optimization problem

$$\min_{\sigma_i,\, i\in[n]} \frac{\frac{1}{n}\sum_{i=1}^{n} M^{\#}(\epsilon_i)/\sigma_i^2}{1 - \frac{1}{m}\sum_{i=1}^{n} M^{\#}(\epsilon_i)}\sigma^2,\ \text{s.t.}\ \sum_{i=1}^{n}\sigma_i^2 = n.$$

As σ_i^2's are the only variables, focus on the numerator of the objective function. By the *Cauchy-Schwarz* inequality, one has

$$\begin{aligned}\sum_{i=1}^{n}\frac{M^{\#}(\epsilon_i)}{\sigma_i^2} &= \sum_{i=1}^{n}\frac{M^{\#}(\epsilon_i)}{\sigma_i^2}\cdot\frac{1}{n}\sum_{i=1}^{n}\sigma_i^2 \\ &\geq \frac{1}{n}\left(\sum_{i=1}^{n}\sqrt{M^{\#}(\epsilon_i)}\right)^2,\end{aligned} \tag{4.37}$$

where the equality holds if and only if $\sqrt{M^{\#}(\epsilon_i)} = c\sigma_i^2$ for some constant c. Recall the total power constraint $\sum\sigma_i^2 = n$. The constant c can be characterized and the optimal power allocation is given by

$$\sigma_i^2 = \frac{\sqrt{M^{\#}(\epsilon_i)}}{\frac{1}{n}\sum_{i=1}^{n}\sqrt{M^{\#}(\epsilon_i)}},\ i \in [n]. \tag{4.38}$$

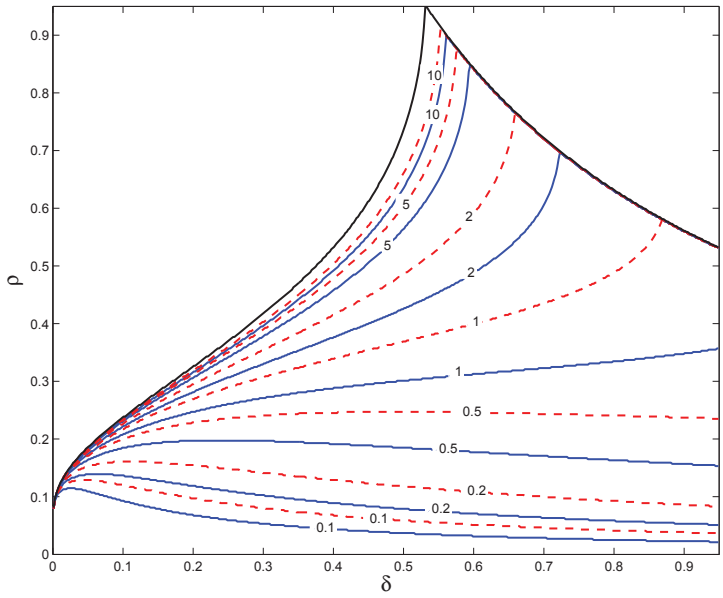

Figure 4.3: Reconstruction error contours for a sparse signal with two even-length blocks where the sparsity ratio $\epsilon^{(1)}/\epsilon^{(2)} = 100$. The blue solid lines and the red dashed lines respectively present the minimax MSEs $\{0.1, 0.2, 0.5, 1, 2, 5, 10\}$ before and after the power allocation. The phase-transition curve for noiseless case is given by the black line. The upper right curved area is the inadmissible area under the sparsity ratio 100. It is necessary to point out that for the curves with MSE$<$1 shown in the figure are not monotonic increasing with ρ. This means that by fixing ρ there exist two sampling rate to reach the same MSE. The curves without power allocation result are consistent with the ones shown in [74]

4.8 Discussion on Reconstruction Error

4.8.1 Theoretical Reconstruction Error

For theoretical demonstration of the effects of power allocation, we assume that the unknown sparse signal can be divided into two even-length blocks where the sparsity ratio is given by $\epsilon^{(1)}/\epsilon^{(2)} = 100$. Consider the least favorable prior $p^{\#}_{\epsilon^{(1)}}$ and $p^{\#}_{\epsilon^{(2)}}$. Normalize the noise variance by setting $\sigma^2 = 1$. Let $\delta = m/n$ and $\rho = \frac{1}{m}\sum \epsilon_i$. In Figure 4.3, the minimax MSE contours before and after the power allocation are respectively given by blue solid lines and red dashed lines. The phase-transition curve for noiseless case is given by the black line. We see that for the all pairs of $(\rho,\ \delta)$ under the phase-transition curve, the obtained reconstruction errors decreased after power allocation. Above the phase-transition bound the state evolution does not converge. The reconstruction error goes to infinity.

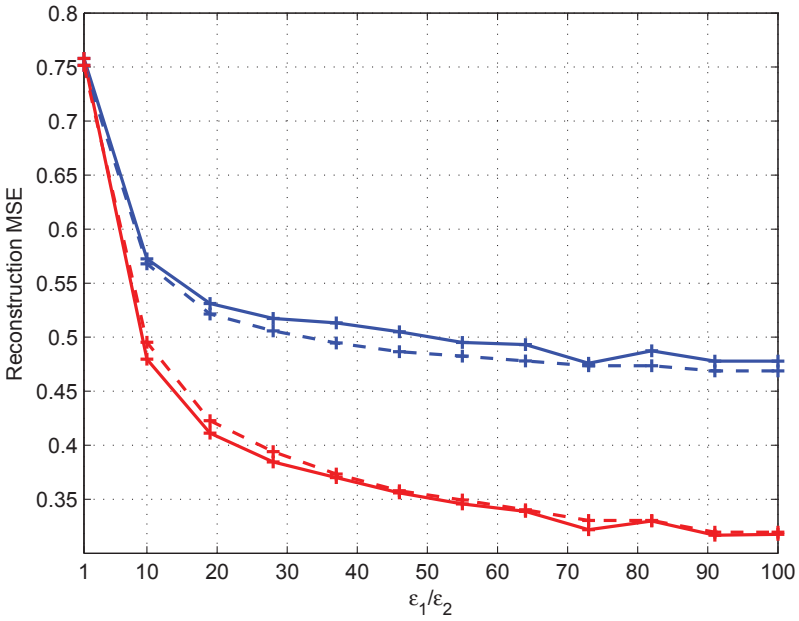

Figure 4.4: MSE against sparsity ratio for sparse signals with two even-length blocks. Blue and red solid lines are the MSE before and after power allocation. Dashed lines are the corresponding theoretical prediction.

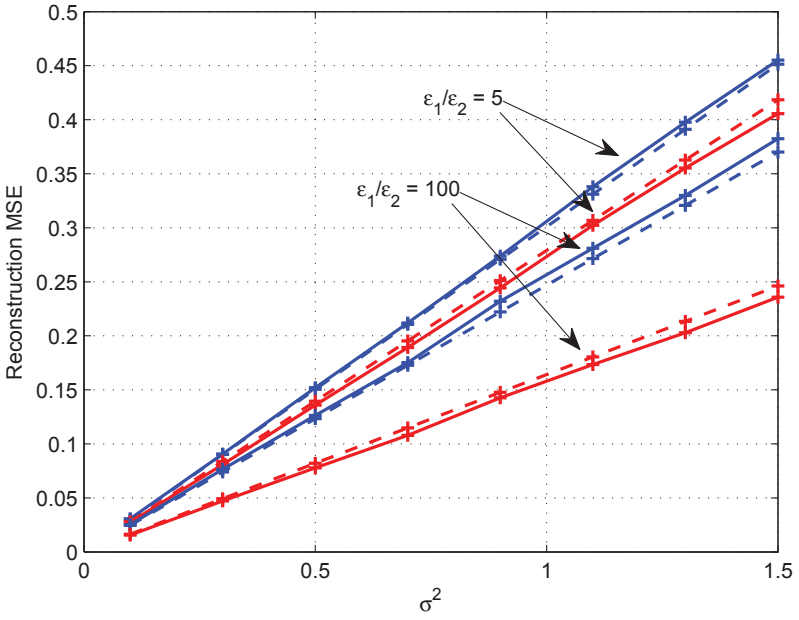

Figure 4.5: MSE against noise variance for sparse signals with two even-length blocks. Number of realizations is 100. Blue and red solid lines are MSE curves before and after power allocation. Dashed lines are the corresponding theoretical prediction.

4.8.2 Empirical Studies

The least favorable prior involves Diracs at $\pm\infty$. It is not practical to numerically generate a sparse signal from such a prior. To avoid this difficulty, the authors of [34] defined the so called *a-least favorable* prior as the distribution $p_{\epsilon,\mu} \in \mathcal{F}_\epsilon$ such that the corresponding MSE satisfies $M_a(\epsilon) = (1-a)\, M^{\#}(\epsilon)$, where $0 < a \ll 1$. Given an a, the value of μ can be computed via the explicit form of the MSE of the three-point mixture.

We set $a = 0.02$ which is the same as that in [34]. Let $m = 2000$ and $n = 4000$. Assume a sparse signal with two even-length blocks, i.e., $n_1 = n_2 = n/2$. The sparsity ratio is defined as $\epsilon^{(1)}/\epsilon^{(2)}$. The signal $\boldsymbol{x}$ is randomly generated (100 realizations) from the sparse prior. For each realization, the AMP.P($\boldsymbol{\epsilon}$) algorithm is applied for reconstruction to obtain $\hat{\boldsymbol{x}}$. In Figure 4.4, we fix $\rho = 0.18$ but vary the sparsity ratio $\epsilon^{(1)}/\epsilon^{(2)}$. We compare the reconstruction MSE $\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2^2/n$. From the presented results, the average MSE after power allocation is always smaller. The performance gain becomes larger when the sparsity ratio increases. Theoretical predictions drawn as dashed curves are very close to the curves obtained from simulations. In Figure 4.5, we aim to demonstrate the linear relationship between the reconstruction MSE and the noise variance, predicted by (4.30). The settings are the same to those for Figure 4.4 except that $\rho = 0.1$ and $\epsilon^{(1)}/\epsilon^{(2)} = 5$ and 100. From the simulations, the linear relationship is confirmed.

4.9 Power Allocation for Another Objective: Contour Enhancement

A similar analysis to that eariler we produced in this chapter can also be applied to optimization problems with other objective functions so as to meet various objectives. For example, due to some actual purposes one may need to reconstruct an image by far as possible to keep its main characteristics. Imagine a scenario where textures from an old book needs to be extracted. If we only focus on reconstructing the image by minimising the whole reconstruction error, the reconstruction will blur texture figures from which further extraction fails. To avoid this an alternative objective function is required to balance the reconstruction minimisation and the texture blurring. Notices that most images have

tree (non-uniform) structure sparsity in the wavelet domain [53] and the contours of the images usually lies in the most sparse layer. We explain that the amplitude of the non-zero elements in the wavelet domain can be statistically treated at the same level. Then the more sparse layer in the tree makes less contributions if a uniformly random measurement matrix is used in the linear observation system. As a result the sparse layer is less possible to be correctly recovered. One reasonable approach would be considering each of the layer in the wavelet domain as a signal block. Instead of minimising the overall reconstruction MSE, we normalise the MSE of each signal block and minimise their summation:

$$\mathrm{P_{CE}} = \min_{\sigma_1^2,\cdots,\sigma_n^2} \frac{1}{n}\sum_{s=1}^{S} \frac{\mathbb{E}\left\{\left\|\boldsymbol{\Psi}\left(\hat{\boldsymbol{x}}\right)_{\mathcal{I}_s} - \boldsymbol{x}_{\mathcal{I}_s}\right\|_2^2\right\}}{\mathbb{E}\left\{\left\|\boldsymbol{x}_{\mathcal{I}_s}\right\|_2^2\right\}}, \text{ s.t. } \sum_{i=1}^{n}\sigma_i^2 = n. \tag{4.39}$$

where signal model is assumed to be $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{\Psi}\left(\boldsymbol{x}\right) + \boldsymbol{w}$. $\boldsymbol{\Psi}\left(\boldsymbol{x}\right)$ is the wavelet transform of $\boldsymbol{x}$. This optimisation problem evens up the contribution of each layer, therefore provide a good balance between reconstruction accuracy of each layer and the overall MSE.

We can also use the minimax risk to find the optimality for problem (4.39):

$$\mathrm{P_{CE}} = \min_{\sigma_i^2\mathrm{s}} \frac{1}{n}\sum_{i=1}^{n} \frac{\mathbb{E}\left\{\left\|\hat{\boldsymbol{x}}_i - \boldsymbol{x}_i\right\|_2^2\right\}}{\epsilon_i} = \frac{\frac{1}{n}\sum_{i=1}^{n} M^{\#}\left(\epsilon_i\right)/\left(\epsilon_i\tau_i^2\right)}{1 - \frac{1}{m}\sum_{i=1}^{n} M^{\#}\left(\epsilon_i\right)}\sigma^2,$$

where the minimum achieves at

$$\sigma_i^2 = \frac{\sqrt{M^{\#}\left(\epsilon_i\right)/\epsilon_i}}{\frac{1}{n}\sum_{i=1}^{n}\sqrt{M^{\#}\left(\epsilon_i\right)/\epsilon_i}}, \; i \in [n]. \tag{4.40}$$

To demonstrate, we choose a text extraction case. See the original image (the left image in Figure 4.6), we transfer it via a standard wavelet matrix $\boldsymbol{\Psi}$ to obtain the non-uniform sparse signal and calculate the sparsity of each layer. Then we multiply an i.i.d. random Gaussian matrix $\boldsymbol{G}$ with $\delta = 0.5$ on the left of the wavelet matrix. Using the obtained signal sparsity, we apply the power allocation (4.40) to the multiplied measurement matrix $\boldsymbol{G}\boldsymbol{\Psi}$. The revised AMP is then called for measurements from $\boldsymbol{G}\boldsymbol{\Psi}$ with and without power allocation. The reconstructed results are shown in Figure 4.6, where the middle image uses

the measurement matrix without power allocation and the right result uses the one with power allocation. Our test results show a highlight effect on the texts observed from the right image, which makes it easier for the text extraction as an input.

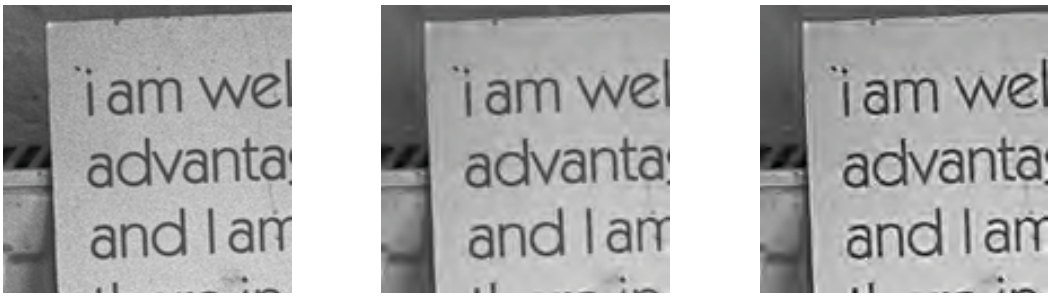

Figure 4.6: Compare reconstruction result with and without enhancement. Left: original; Middle: reconstruction without enhancement; Right: reconstruction with enhancement

Chapter 5

Quantifying the Asymptotic Performance of Distributed Compressed Sensing

5.1 Introduction

Distributed compressed sensing (DCS) [5] is a extended research of compressed sensing to reconstruct sparse signals with correlations. Consider a compressed sensing setup with several sensors measuring several sparse correlated signals, where each signal is measured via a linear transform with additive noise. DCS wishes to reconstruct the sparse signals through knowing the correlations to outperform traditional compressed sensing. Typical models of the signal correlation include sparse common component plus innovations [5], common sparse supports [5], non-sparse common component plus sparse innovations [5], sparse common component [75], sparse common supports plus innovations [96] and sparse common supports with correlations plus innovations [97].

In this chapter we consider common support signals, i.e., assume all the signals have a same sparse support. DCS with this type of signals can be applied in applications such as magnetoencephalography [23], DOA estimation [104], parallel magnetic resonance imaging (pMRI) [105], distributed sensor networks [115] and distributed video sensing [29]. Common support signals are also widely discussed in group lasso problem [116] and mul-

tiple measurement vectors (MMV) problem [23], where the signals are called group/block sparse signals and the linear systems for the signal observation are different from DCS. Most existing DCS algorithms for common support signals are the derivatives from greedy compressed sensing algorithms such as orthogonal matching pursuit (OMP) [81] and subspace pursuit (SP) [26]. The typical algorithms include simultaneous orthogonal matching pursuit (S-OMP) [103, 102], side-information based OMP (SiOMP) [117], distributed and collaborative OMP (DC-OMP) [110], distributed SP (DiSP) and distributed predictive SP (DPrSP) [96], etc. All these algorithms can use restricted isometry property (RIP) for the performance analysis. In terms of implementation they also carry the same thought: each signal updates the estimation of its support set and share to the other signals, then via a designed voting mechanism the common support of the signals is decided either sequentially or in parallel. This procedure is then iterated until a stop criteria is met.

Block sparse signals are usually a category to describe signals with common support. The earliest research problem focusing on compressive sensing of block sparse signals is called group Lasso which is proposed by Yuan and Lin [116]. Group Lasso extended the Lasso problem and was used to help with the regression model selections. The model was afterwards applied to generalized linear regression [87], logistic regression [73] and so on. Stojnic et. al. analysed the optimal number of measurements required for the block sparse signals given that the measurement matrix is i.i.d. Gaussian[94, 95]. A sharp lower bound was also derived in the asymptotically regime in their work. Baron et. al. considered slightly different scenarios that the measurement matrices contain block diagonal structure [5]. One of the cases they considered, which also belongs to our focus in this chapter, is that each signal block is independently measured. Based on this, algorithms such as simultaneous orthogonal matching pursuit (SOMP) [103, 102] and relaxed belief propagation(BP) algorithm [63] can be used to jointly reconstruct the signals. An adapted version of SOMP algorithm is proposed in [5]. The topic related to such signal model is termed as distributed compressed sensing. Ji et. al. started from the view of machine learning to analyse the same problem and proposed multitasking learning framework [61]. The essence is to use relevant vector machine as the driver to pursuit the training result to be sparse. In applications when all the sparse blocks are obtained from the same measurement matrix, the problem turns to multiple measurement vectors which is firstly

proposed by Cotter et. al. [23]. That topic also gained extensive attention, see [21, 119]. Other compressed sensing methods of interest related to block sparse signals can be referred to [96] and the references therein.

Other than greedy technique based algorithms, in this chapter we choose to use approximate message passing (AMP) [30] framework for signal recovery and performance analysis. This is motivated by the associated low computational complexity and the capability of quantifying the minimum sensing rate for exact recovery, i.e., the phase transition. AMP takes advantage of the properties in i.i.d. random measurement matrices and use reasonable approximations to vastly simplify the computations of message passing algorithms in graphical theories. In the mathematical expression, an 'Onsager' term is added to the iterative soft-thresholding (IST) iterations to eliminate the dependence brought in by the iterative updates. The AMP technique has been previously applied to analyse group Lasso [36, 98] and MMV problem [119]. However the analysis turns out to be significantly different in the heterogeneous DCS case where the numbers of measurements and the noise levels at different sensors are different. In the heterogeneous case, the symmetry in the decoder and performance evaluation breaks down and so does the techniques in [36, 98, 119]. Furthermore, as known by the author, AMP based MMV algorithm requires small innovative difference between each two blocks in the measurement matrix so as to reach high reconstruction successful rate[119]. In [49] the author discussed multi-terminal compressed sensing models and extended the associated state evolution disciplines. Sparse common component plus innovations model is considered and the results is compared with Renyi information dimension analysis.

This chapter firstly focuses on common sparse supports model with only knowing the uniform sparsity characteristic across the signal elements. We employ the least favorite distribution for block sparse signals and derive the corresponding AMP reconstruction algorithm from the bipartite graph analysis. Using the state evolution technique the phase transition bound is shown as a set of curves, each curve represents for signal with a specific number of blocks. Then we add a very common continuous-plus-discrete probability distribution prior on the sparse signal. The corresponding decoder has been derived with an explicit closed form. The exact phase transition can be evaluated via numerical integrals and the rate regions for exact recovery have been characterised. It turns out that

an equal allocation of the number of measurements across sensors is strictly suboptimal. Furthermore, our approach allows to quantify the effect of the correlation among nonzero components from different sparse signals. This is important for many practical scenarios [59, 97] but was absent before according to the authors' knowledge. The numerical algorithm provided in this chapter is also applicable for tracking the reconstruct mean squared error assuming Gaussian noise with known heterogeneous variance is added on each signal.

The remainder of the chapter is organized as follows. We introduce the DCS system model in section 5.2 and give the scalar case analysis in section 5.3.1. In section 5.3.2 we infer the approximate message passing algorithm for the common support support sparse model. In section 5.3.3 we introduce the phase transition for the derived algorithm and extend the result to limit as the number of blocks approaching to infinity. In section 5.4 and 5.5 we discuss the difficulty by employing the model to a heterogeneous case thus we propose an alternative but more specific signal prior distribution, Bernoulli-Gaussian distribution. We analyse the system model with its scalar case estimator derivation. Then in section 5.6 and 5.7 we give the corresponding AMP algorithm and use the state evolution to draw the phase transition. In the same sections we propose strategies to minimise the reconstruction error by giving a total sampling resource budget. Finally in section 5.8 we give the numerical study for comparison to the theoretical results. Image processing example is also demonstrated to show the effect of our optimisation strategies.

5.2 System Model

Consider the DCS scenario with $K \in \mathbb{Z}^+$ sensors. For each sensor k measurements are given by

$$\boldsymbol{y}_k = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{w}_k, \tag{5.1}$$

where $\boldsymbol{y}_k \in \mathbb{R}^{m_k}$ is the measurement vector. Here we an ambiguous definition, where $\boldsymbol{A}_k \in \mathbb{R}^{m_k \times n}$ denotes the measurement matrix (rather than a column vector), $\boldsymbol{x}_k \in \mathbb{R}^n$ stands for the unknown sparse signal, and $\boldsymbol{w}_k \in \mathbb{R}^{m_k}$ is the additive white Gaussian noise with mean zero and the covariance matrix $\sigma_k^2 \boldsymbol{I}$, $k \in [K]$.

Assume that the unknown signals $\boldsymbol{x}_k$'s are from related phenomena, which suggests that they share the same dimension and exhibit inter-signal correlations. In this chapter,

we are particularly interested in the joint sparsity model (JSM) type 2 proposed in [5], that is, $\boldsymbol{x}_k$'s share the common sparse support. Let $x_{k,i}$ be the i-th component of $\boldsymbol{x}_k$ and define $\text{supp}(\boldsymbol{x}_k) = \{i \in [n] : x_{k,i} \neq 0\}$ be the support of $\boldsymbol{x}_k$. The JSM type 2 model assumes that $\text{supp}(\boldsymbol{x}_1) = \text{supp}(\boldsymbol{x}_2) = \cdots = \text{supp}(\boldsymbol{x}_K)$. Define $\boldsymbol{x}_{\cdot,i} = [x_{1,i}, x_{2,i}, \cdots, x_{K,i}]^T \in \mathbb{R}^K$ which groups the i-th components from different k's together. Write the overall system model as

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}, \tag{5.2}$$

where $\boldsymbol{A} = \text{diag}(\boldsymbol{A}_1, \boldsymbol{A}_2, \cdots, \boldsymbol{A}_K)$ is a block diagonal matrix, $\boldsymbol{y} = \left[\boldsymbol{y}_1^T, \boldsymbol{y}_2^T, \cdots, \boldsymbol{y}_K^T\right]^T$, $\boldsymbol{x} = \left[\boldsymbol{x}_1^T, \boldsymbol{x}_2^T, \cdots, \boldsymbol{x}_K^T\right]^T$, and $\boldsymbol{w} = \left[\boldsymbol{w}_1^T, \boldsymbol{w}_2^T, \cdots, \boldsymbol{w}_K^T\right]^T$. Then the unknown signal $\boldsymbol{x}$ follows the well-known block sparse structure [116] where the signal $\boldsymbol{x}$ is divided into blocks given by $\boldsymbol{x}_{\cdot,i}$'s and the components in a block are either simultaneously zero or simultaneously nonzero.

5.3 AMP for DCS Reconstruction – Homogeneous System Model

5.3.1 Scalar Case Analysis

In order to derive the AMP algorithm for DCS reconstruction and analyse its performance, the "scalar" case is studied in this section where $m_1 = m_2 = \cdots = m_K = 1$, $n = 1$, and $\boldsymbol{A}_1 = \boldsymbol{A}_2 = \cdots = \boldsymbol{A}_K = 1$. With this assumption, the signal model is simplified to

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{w} \in \mathbb{R}^K, \tag{5.3}$$

where $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Assume there is no more information on the signals known in advance. We consider the least favorable distribution as article [33] shows it statistically gives the worst case minimum mean squared reconstruction error. We generalise this idea and start with a K scalar signal model including signal x_is, $i \in [K]$. Each x_i is independently generated and the ℓ_2-norm $\|\boldsymbol{x}\|_2$ has the least favorable distribution:

$$p(\|\boldsymbol{x}\|_2) = (1-\epsilon)\,\delta_0 + \epsilon\delta_\infty. \tag{5.4}$$

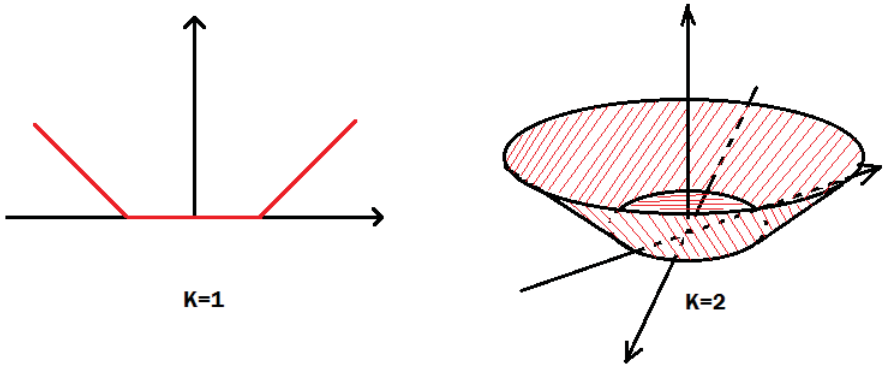

Figure 5.1: Soft-shresholding function shows the shrinkage of the signal amplitudes for cases $K = 1$ and 2.

The following soft-thresholding function (5.5) for denoising,

$$\eta_\alpha(\boldsymbol{y}) = \begin{cases} \frac{\|\boldsymbol{y}\|-\alpha}{\|\boldsymbol{y}\|}\boldsymbol{y} & \text{if } \|\boldsymbol{y}\| > \alpha \\ 0 & \text{otherwise} \end{cases}. \tag{5.5}$$

In [36] this function is simply mentioned and used for the group Lasso problem. This is a reasonable denoiser since in the case when $K = 1$, we use the thresholding function by shrinking the absolute value of the observation y and keep its sign. This is exactly the thresholding function used in [30]. Because of the isotropy of the signals in the model settings herein, we consider keeping the sign of each element and shrink the ℓ_2-norm $\|\boldsymbol{x}\|_2$. The analysis given in [70] shows the use of block thresholding function in complex AMP. It is equivalent to our $K = 2$ scalar case. The extension to $K > 2$ cases can be found in [36, 51, 17]. See Figure (5.1) for a pictorial generalisation from $K = 1$ to $K = 2$. In the extreme context, one assumes the isotropic signal $\boldsymbol{x}$ is generated from the least favorable distribution (5.4) and this gives an asymptotic upper bound reconstruction MSE using (5.5). In the Appendix B.1 we will spend some space for deriving the MSE closed form.

5.3.2 Inference via Message Passing for Common Support Signal Model

We start with the bipartite graph for the distributed compressed model and extend the theory of the previous section to the vector case. The key step is to introduce the min-sum algorithm to solve the optimization problem shown in (5.6). Min-sum algorithm is a popular optimization algorithm for graph-structured cost functions. More importantly

approximation techniques are added to simplify the complexity of the whole algorithm. The main procedure is followed by [74] with additional consideration of the message passing among multi-layers in Figure (B.3). Figure (B.3) gives the bipartite graph for DCS problem. It consists K many layers, each representing for a factor nodes $\boldsymbol{y}_i$ to variable nodes $\boldsymbol{x}_i$.

Consider model (5.2) and set $\boldsymbol{w} = \mathbf{0}$ for the moment. We aim to solve the DCS problem written as

$$\min_{\boldsymbol{x}} \frac{1}{2}\sum_{k=1}^{K} \|\boldsymbol{y}_k - \boldsymbol{A}_k\boldsymbol{x}_k\|_2^2 + \lambda \sum_{i=1}^{n} \|\boldsymbol{x}_{\cdot,i}\|_2$$
$$\text{s.t. } \forall k, l \in [K]\,,\ \text{Supp}\,(\boldsymbol{x}_k) = \text{Supp}\,(\boldsymbol{x}_l)\,, \tag{5.6}$$

where $\boldsymbol{x}_{\cdot,i}$ is a block of the signal components defined as $\boldsymbol{x}_{\cdot,i} \triangleq [x_{1,i}, x_{2,i}, \cdots, x_{K,i}]^T$. For block k, we have the decomposed cost function

$$\mathcal{C}_{\boldsymbol{A},y}\,(\boldsymbol{x}_k) \equiv \frac{1}{2}\sum_{a=1}^{m} (y_{k,a} - \boldsymbol{A}_{k,a}\boldsymbol{x}_k)^2 + \lambda \sum_{i=1}^{n} \|\boldsymbol{x}_{\cdot,i}\|_2\,,$$

where $\boldsymbol{A}_{k,a}$ represents the a^{th} column of matrix $\boldsymbol{A}_k$. The min-sum algorithm updates read

$$J_{i\to a}^{t+1}\,(x_{k,i}) \cong \lambda\,\|\boldsymbol{x}_{\cdot,i}\|_2 + \sum_{b\neq a} \hat{J}^t{}_{b\to i}\,(x_{k,i})\,,$$

$$\hat{J}^t{}_{a\to i}\,(x_{k,i}) \cong \min_{x_{j\neq i}} \left\{ \frac{1}{2}\,(y_{k,a} - \boldsymbol{A}_{k,a}\boldsymbol{x}_k)^2 + \sum_{j\neq i} J_{j\to a}^{t}\,(x_{k,j}) \right\}.$$

Since $J_{i\to a}^{t+1}\,(x_{k,i})$ and $\hat{J}^t{}_{a\to i}\,(x_{k,i})$ are both convex, we can simplifying it by quadratic approximation. Notice that amplitude of elements in $\boldsymbol{A}_k$ are assumed of order $O\left(\frac{1}{\sqrt{m}}\right)$, which are statistically much smaller than 1. As a result we use a second order Taylor expansion for $\hat{J}^t{}_{a\to i}\,(x_{k,i})$ (assume that $\hat{J}^t{}_{a\to i}\,(0) = 0$):

$$\hat{J}^t{}_{a\to i}\,(x_{k,i}) \cong -\alpha_{k,a\to i}^{t}\,(\boldsymbol{A}_{k,a,i}x_{k,i}) + \frac{1}{2}\beta_{k,a\to i}^{t}\,(\boldsymbol{A}_{k,a,i}x_{k,i})^2 + O\left((\boldsymbol{A}_{k,a,i}x_{k,i})^3\right).$$

where $\alpha_{k,a\to i}^{t}$ and $\beta_{k,a\to i}^{t}$ are the introduced coefficient of the first and second order term

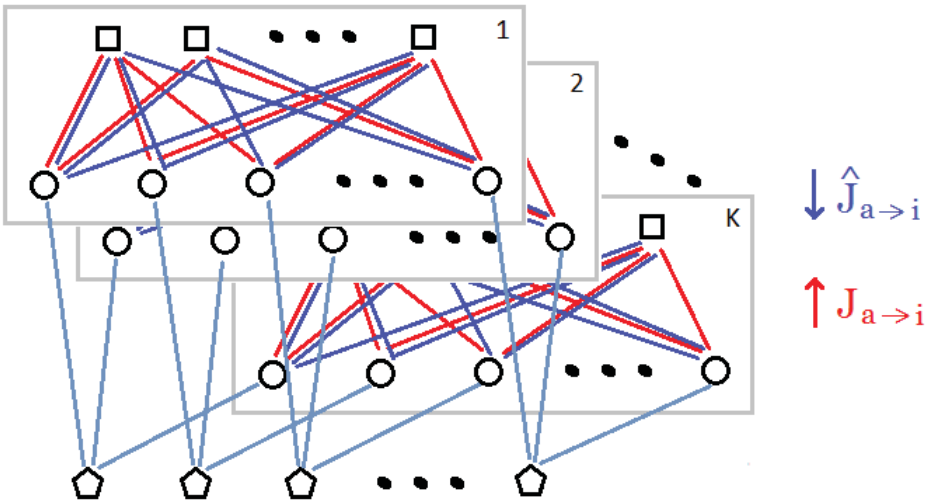

Figure 5.2: The factor sketch graph for the DCS problem. The circles are the variable nodes. The squares are the factor nodes. The pentagons are the nodes constraining all the K signals share the same support.

in the above expansion. Substitute $\hat{J}^t{}_{a\to i}\left(x_{k,i}\right)$ into $J^{t+1}_{i\to a}\left(x_{k,i}\right)$ we have

$$\begin{aligned} J^{t+1}_{i\to a}\left(x_{k,i}\right) &\cong \lambda\left\|\boldsymbol{x}_{\cdot,i}\right\|_2 - \left(\sum_{b\neq a}\boldsymbol{A}_{k,b,i}\alpha^t_{k,b\to i}\right)x_{k,i} + \frac{1}{2}\left(\sum_{b\neq a}\boldsymbol{A}_{k,b,i}\beta^t_{k,b\to i}\right)x^2_{k,i} + O\left(n\left(\boldsymbol{A}_{k,\cdot,i}x_{k,i}\right)^3\right) \\ &\cong \frac{1}{2\gamma^{t-1}_{k,i\to a}}\left(x_{k,i} - x^{t-1}_{k,i\to a}\right)^2 + O\left(\left(x_{k,i} - x^{t-1}_{k,i\to a}\right)^3\right). \end{aligned}$$

where $\boldsymbol{A}_{k,\cdot,i}$ represents the i^{th} row of matrix $\boldsymbol{A}_k$. The second $\cong$ holds due to taking the second order Taylor expansion for $\hat{J}^t{}_{a\to i}\left(x_{k,i}\right)$ and set $x^{t-1}_{k,i\to a} = \arg\min_{x_{k,i}}\ \hat{J}^t{}_{a\to i}\left(x_{k,i}\right)$. Now the problem is transferred to solve $x^t_{k,i\to a}$ and $\gamma^t_{k,i\to a}$. Take the derivative of $J^t_{i\to a}\left(x_{k,i}\right)$:

$$\frac{\partial J^t_{i\to a}\left(x_{k,i}\right)}{\partial x_{k,i}} = \frac{\lambda x_{k,i}}{\left\|\boldsymbol{x}_{\cdot i}\right\|_2} - \left(\sum_{b\neq a}\boldsymbol{A}_{k,b,i}\alpha^t_{k,b\to i}\right) + \left(\sum_{b\neq a}\boldsymbol{A}^2_{k,b,i}\beta^t_{k,b\to i}\right)x_{k,i} = 0.$$

Define $u_k = \frac{\sum_{b\neq a}\boldsymbol{A}_{k,b,i}\alpha^t_{k,b\to i}}{\sum_{b\neq a}\boldsymbol{A}^2_{k,b,i}\beta^t_{k,b\to i}}$, $v_k = \frac{\lambda}{\sum_{b\neq a}\boldsymbol{A}^2_{k,b,i}\beta^t_{k,b\to i}}$. The above equation turns to

$$x_{k,i} - u_k + v_k\cdot\frac{x_{k,i}}{\left\|\boldsymbol{x}_{\cdot i}\right\|_2} = 0.$$

Note the signal model assumes $\boldsymbol{x}_{\cdot,i}$ having the isotropic distribution consists all zeros (or non-zeros) at the same time. We may equivalently focus on $\left\|\boldsymbol{x}_{\cdot,i}\right\|_2$ instead and only

care about applying a thresholding function on it. In this way we expect $\boldsymbol{x}_{\cdot i} \| \boldsymbol{u}$, i.e., it implies that $v_k = v_l$ for $\forall k, l \in [K]$. Therefore we can remove the subscript on v. Let $\boldsymbol{u} = [u_1,\ u_2,\ ...,\ u_k]^T$. We have

$$\boldsymbol{x}_{\cdot i} \left(1 + \frac{v}{\|\boldsymbol{x}_{\cdot i}\|_2} \right) = \boldsymbol{u}.$$

Refer to (5.5) and solve the above equation and set $\boldsymbol{x}^t_{\cdot, i \to a} = \boldsymbol{x}_{\cdot i}$,

$$\boldsymbol{x}^t_{\cdot, i \to a} = \eta\left(\boldsymbol{u},\ v\right) = \begin{cases} \left(1 - \frac{v}{\|\boldsymbol{u}\|_2}\right) \boldsymbol{u} & \text{if } \|\boldsymbol{u}\|_2 > v \\ 0 & \text{otherwise} \end{cases}.$$

Also compute $\boldsymbol{\gamma}^t_{k, i \to a} = \eta^{'}\left(u_k,\ v\right)$, where

$$\eta'\left(u_k,\ v\right) \begin{cases} \frac{\left(u_k^2 - \|\boldsymbol{u}\|_2^2\right) v}{\|\boldsymbol{u}\|_2^3} + 1 & \text{if } \|\boldsymbol{u}\|_2 > v \\ 0 & \text{otherwise} \end{cases}.$$

We use the same way as shown in [74], use $\boldsymbol{x}^t_{\cdot, i \to a}$ and $\boldsymbol{\gamma}^t_{i \to a}$ to represent $\alpha^t_{k, b \to i}$s and $\beta^t_{k, b \to i}$s, yielding

$$\alpha^t_{k, b \to i} = \frac{1}{\sum_{i \neq j} \boldsymbol{A}^2_{k,a,j} \gamma^t_{k, i \to a}} \left\{ y_{k,a} - \sum_{i \neq j} \boldsymbol{A}_{k,a,j} x^t_{k, i \to a} \right\},$$

$$\beta^t_{k, b \to i} = \frac{1}{\sum_{i \neq j} \boldsymbol{A}^2_{k,a,j} \gamma^t_{k, i \to a}}.$$

Then noticing the weak dependence among both terms $\boldsymbol{A}^2_{k,a,j} \gamma^t_{k, i \to a}$ and terms$\boldsymbol{A}^2_{k,a,j} \beta^t_{k, i \to a}$. Let $r^t_{k, i \to a} = \alpha^t_{k, i \to a} / \beta^t_{k, i \to a}$, we can write the message passing iterations as

$$r^t_{k, i \to a} = y_a - \sum_{i \neq j} \boldsymbol{A}^2_{k,a,j} x^t_{k, i \to a},$$

$$x^t_{k, i \to a} = \eta \left(\sum_{b \neq a} \boldsymbol{A}^2_{k,a,j} r^t_{k, b \to i},\ \theta^t \right),$$

where θ^t is treated as independent of k.

The above equations lead to our AMP algorithm for DCS. The different from the classic

AMP in form is the element-wise thresholding function $\eta(\cdot)$.

$$\boldsymbol{x}_k^{t+1} = \eta\left(\boldsymbol{x}_k^t + \boldsymbol{A}_k^T \boldsymbol{r}_k^t; \theta^t\right), \tag{5.7}$$

$$\boldsymbol{r}_k^t = \boldsymbol{y}_k - \boldsymbol{A}_k \boldsymbol{x}_k^t + b_k^t \boldsymbol{r}_k^{t-1}, \tag{5.8}$$

where $b_k^t = \frac{1}{mK}\sum_j \eta'\left(\boldsymbol{x}_{k,j}^{t-1} + \left(\boldsymbol{A}_k^T \boldsymbol{r}_k^{t-1}\right)_j; \theta^{t-1}\right)$. Furthermore, in high dimensional statistics as $n \to \infty$, all the $\boldsymbol{x}_k$'s have the same distribution and so are all the $\boldsymbol{A}_k$'s which imply that $b_k^t = b_l^t$, $\forall k, l \in [K]$. We define $b^t \triangleq b_k^t$.

Although the above inference is built upon the DCS model (5.2), the choice of parameters θ^t and b^t also have close connections with the problem (5.6). We formalise the proposition below, where general sequences $\left\{\theta^t\right\}_{t\geq 0}$ and $\left\{b^t\right\}_{t\geq 0}$ can be used as long as there fix points $\left(\boldsymbol{x}^t, \boldsymbol{r}^t\right)$ can be found via the iterations (5.7) and (5.8).

Proposition 5.1. *Let* $(\boldsymbol{x}^*,\ \boldsymbol{r}^*)$ *be a fixed point of the above two iterations* (5.7) *and* (5.8) *for* $\theta^t = \theta$, $b^t = b$ *fixed,* $k \in [K]$. *Then* $\boldsymbol{x}^*$ *is a minimum of cost function in the problem* (5.6) *for*

$$\lambda = \theta(1-b).$$

Proof. From equation (5.7) we get the fixed point condition

$$\boldsymbol{x}_k^* + \theta_k \boldsymbol{v}_k = \boldsymbol{x}_k^* + \boldsymbol{A}_k^T \boldsymbol{r}_k^*,$$

for $\boldsymbol{v}_k \in \mathbb{R}^n$ such that$\boldsymbol{v}_k$ is a sub-gradient of the ℓ_1-norm at $\boldsymbol{x}_k^*$, i.e.,

$$v_{k,i} = \begin{cases} \operatorname{sign}\left(x_{k,i}^*\right) & \text{if } \boldsymbol{x}_i^* \neq 0 \\ c \in [-1, +1] & \text{otherwise} \end{cases}.$$

From equation (5.8) we get $(1-b)\,\boldsymbol{r}_k^* = \boldsymbol{y}_k - \boldsymbol{A}_k \boldsymbol{x}_k^*$. Substituting in the above equation, we have

$$\theta(1-b)\,\boldsymbol{v}_k = \boldsymbol{A}_k^T\left(\boldsymbol{y}_k - \boldsymbol{A}_k \boldsymbol{x}_k^*\right). \tag{5.9}$$

Note the stationary condition of problem (5.6) can be written as

$$\sum_{k=1}^{K} \boldsymbol{A}_k^T \left(\boldsymbol{y}_k - \boldsymbol{A}_k \boldsymbol{x}_k^*\right) = \frac{\partial \sum_{i=1}^{n} \lambda \left\|\boldsymbol{x}_{\cdot,i}\right\|_2}{\partial \boldsymbol{x}}$$
$$= \lambda \sum_{k=1}^{K} \boldsymbol{v}_k. \tag{5.10}$$

Consider the connection with the message passing min-sum algorithm. Combine and simplify (5.9) and (5.10) we get

$$\theta\left(1-b\right) = \lambda,$$

which is also the fixed condition of the AMP algorithm. □

5.3.3 Phase Transition Limit as $K \to \infty$

In [94] the sampling limit of group Lasso problem is shown as the number of blocks $K \to \infty$. Similar results for compressed sensing block sparse signals are also given in [72, 80]. We are interested in the phase transition limit as $K \to \infty$ of applying AMP algorithm to the homogeneous DCS problem. Below we will show the derivation and compare the obtained conclusion to the benchmark results in [72, 80]. The phase transitions we obtained in this section turn out to be the same as the group Lasso limit and leave large gap to the information theoretical limit. This result presents the motivation for us to study the heterogeneous case in the next section, i.e., unevenly allocating sampling rate for the signal blocks. We will show that it is a possible way to overcome the gap.

We formulate the goal in this section into the following proposition.

Proposition 5.2. *Consider system given in* (5.2). *Given the formula of* $M_K\left(\epsilon, \alpha\right)$ *in* (B.2), *and let* $\delta = \frac{m}{n}$, *where* $m = m_1 = m_2 = ... = m_K$. *Further assume there is no additive noise, i.e.,* $\boldsymbol{w} = \boldsymbol{0}$. *We show that as* $K \to \infty$, *the sampling rate* δ *is lower bounded by*

$$\delta = 1 - \frac{\alpha^2}{K}, \tag{5.11}$$

in order to achieve the accurate reconstruction.

The proof of this proposition is given in Appendix B.2. From the above results, we see that when $K \to \infty$, the phase transition curve depends on value $\frac{\alpha}{\sqrt{K}}$. Therefore we can

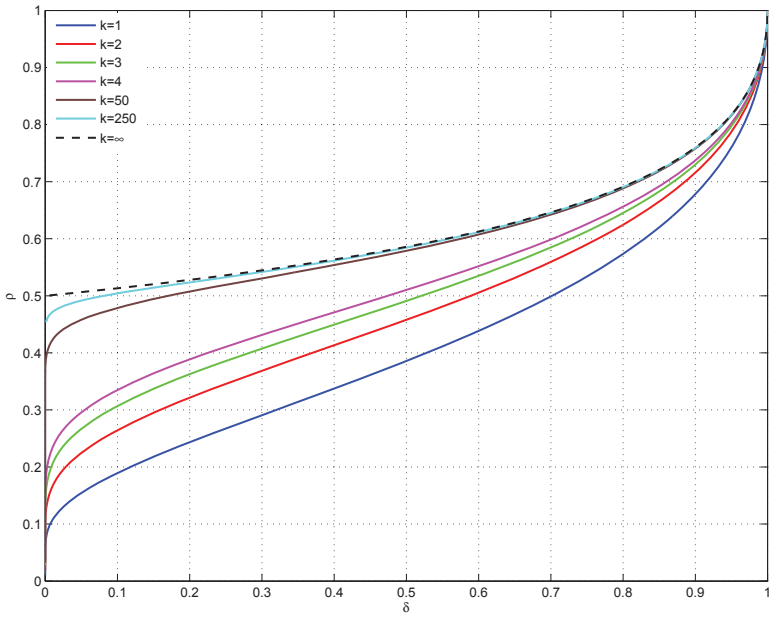

Figure 5.3: Phase transitions of DCS with least favorable group sparse distributed signals. The deconstruction algorithm is DCS-AMP.

draw the theoretical $\rho - \delta$ curve in Figure (5.3).

Techniques based on the coding theory of Reed-Solomon codes can be employed to determine any signal $\boldsymbol{x}$ with sparsity ϵ in (5.2) for any δ and any $\epsilon \leq \frac{\delta}{2}$ in polynomial time as the measurement matrix is chosen uniformly at random from a given distribution. It is easy to see that ρ can not be greater than $\frac{1}{2}$ if we want $\boldsymbol{x}$ to be uniquely recoverable. The complexity of algorithms from [80, 13] is roughly $O\left(n^3K^3\right)$. If the measurement matrix is designed based on the techniques related to the coding/decoding contents then the complexity of recovering $\boldsymbol{x}$ can be decreased to $O\left(nK\right)$ (see [113] and references therein). However, these algorithms usually do not allow the sampling rate δ to get close to $\frac{1}{2}$. In the DCS settings (5.2) the measurement matrix $\boldsymbol{A}$ is designed to be a block diagonal matrix and the theoretical bound shown in [80, 13] is achieved. The resulting phase transitions admit with the ones given in [80, 13] and [94] but complexity of recovering $\boldsymbol{x}$ is $O\left(n^2K^2\right)$ which outperforms the benchmarks.

5.4 AMP for DCS reconstruction – Heterogeneous System Model

This section is to further analyse the block soft thresholding function and show the technical difficulty on applying it to the least favorable distributed block sparse signals to the heterogeneous system model. We will overcome this is issue by introducing a signal prior and deriving the MMSE estimator to replace the thresholding function. We still consider the system model (5.1) and follow the same description in Section (5.2). More generally we allow heterogeneous cases where different sensors may collect different number of measurements and subject to different noise level, i.e. we assume $m_k \neq m_l$, $\forall k, l \in [K]$.

Remove the equal variance constraint in (5.3) and allow the variance being different among the noise elements. Note that here we do not have prior distribution on $\boldsymbol{x}$, we may define $\boldsymbol{\Lambda} = \text{diag}\,(\sigma_1, ..., \sigma_K)$ and normalise the noise variance on all elements. Therefore we are still fine to utilise (5.5) except the input is replaced by $\boldsymbol{\Lambda}^{-1}\boldsymbol{y}$. Finally scaling back the output by $\boldsymbol{\Lambda}$, we actually obtain an estimate $\hat{\boldsymbol{x}}$ by using the following function

$$\begin{aligned}\hat{\boldsymbol{x}} &= \eta\left(\boldsymbol{y};\, \boldsymbol{\Lambda}^{-1}, \alpha\right) \\ &= \begin{cases} \frac{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{y}\right\| - \alpha}{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{y}\right\|}\boldsymbol{y} & \text{if } \left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{y}\right\| > \alpha \\ 0 & \text{otherwise} \end{cases} \\ &\triangleq \left(1 - \frac{\alpha}{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{y}\right\|_2}\right)_+ . \end{aligned} \tag{5.12}$$

On determining the parameter α, we still consider the least favorable distribution (5.4) and assume an isotropic probabilistic characteristic on $\boldsymbol{x}$. The upper bound MSE

$$\begin{aligned} M_K^{\#} &= \mathbb{E}\left\{\left\|\eta\left(\boldsymbol{X} + \boldsymbol{W};\, \boldsymbol{\Lambda}^{-1}, \alpha\right) - \boldsymbol{X}\right\|_2^2\right\} \\ &= (1-\epsilon)\sum_{k=1}^{K}\frac{1}{K}\mathbb{E}\left\{\left(\sqrt{\sum_{l=1}^{K}\sigma_l^{-2}W_l^2} - \alpha\right)_+^2\right\}\sigma_k^2 \\ &\quad + \epsilon\alpha^2\mathbb{E}\left\{\frac{\sum_{k=1}^{K}X_k^2}{\sum_{k=1}^{K}\sigma_k^{-2}X_k^2}\right\} + \epsilon\sum_{k=1}^{K}\sigma_k^2, \end{aligned}$$

where a details can be found in Appendix B.3. A major difficulty we meet is that with

the change of σ_k's during AMP iterations. Since the scale invariant property does not hold for the heterogeneous case, the optimal threshold α needs to be updated according in each iteration. This α calculations are via numerical search which cost a lot, therefore it impedes the later procedures of determining the state evolution in AMP.

The aforementioned problem can be avoided if we switch from the Lasso type denoiser to a more convenient one. Specifically we assume the Bernoulli-Gaussian model:

$$p_{\boldsymbol{X}_{\cdot,i}}\left(\boldsymbol{x}_{\cdot,i};\epsilon,\mathbf{0},\boldsymbol{\Sigma}_X\right)=(1-\epsilon)\,\delta_{\boldsymbol{x}_{\cdot,i}=\mathbf{0}}+\epsilon f_G\left(\boldsymbol{x}_{\cdot,i};\mathbf{0},\boldsymbol{\Sigma}_X\right), \tag{5.13}$$

where

$$f_G\left(\cdot;\boldsymbol{\mu},\boldsymbol{\Sigma}\right)=\left|2\pi\boldsymbol{\Sigma}\right|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)^T\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)\right), \tag{5.14}$$

where $\delta_{\boldsymbol{x}=0}$ denotes the Dirac delta at $\boldsymbol{x}=0$, and $f_G\left(x;0,\boldsymbol{\Sigma}_X\right)$ is the Gaussian density function with zero mean and covariance matrix $\boldsymbol{\Sigma}_X$. Here, the parameters in $\boldsymbol{\Sigma}_X$ are introduced heterogeneously. However in this chapter we only analyse models with heterogeneous noises. Since we note that the performance analysis will highly depend on the signal-to-noise ratio, the signal energy can be normalised by letting $\boldsymbol{\Sigma}_X=\boldsymbol{I}$, and remain the signal-to-noise ratio unchanged by adjusting the noise variance σ_k^2's. It can be verified that the models before and after adjustment have one-to-one linear correspondence. We also note that the statistical modeling allows more sophisticated structure than in the JSM type 2 modeling. Besides the common sparse support, one may assume that the nonzero components from $\boldsymbol{x}_{\cdot,i}$ are correlated, i.e., $\boldsymbol{\Sigma}_X$ is not diagonal. In the rest of this chapter, we will firstly focus on (5.13) and present results related to the case $\boldsymbol{\Sigma}_X=\boldsymbol{I}$. Then we further spend some sparse to discuss the case when $\boldsymbol{\Sigma}_X\neq\boldsymbol{I}$.

5.5 Scalar Case Analysis for Heterogeneous Model

For compositional convenience, we define

$$R_k=\frac{1}{1+\sigma_k^2}, \tag{5.15}$$

which is an invertible function of the SNR at the k-th sensor.

The MMSE estimator and the associated MSE are derived as follows. The MMSE estimator of x_k, denoted by $\hat{x}_k = g_k\left(\boldsymbol{y};\boldsymbol{\sigma}^2\right)$, is given by the conditional expectation

$$\begin{aligned}\hat{x}_k = g_k\left(\boldsymbol{y};\boldsymbol{\sigma}^2\right) &= \mathbb{E}\left[X_k|\boldsymbol{Y}=\boldsymbol{y}\right] = \int x_k \cdot p_{\boldsymbol{X}|\boldsymbol{Y}}\left(\boldsymbol{x}|\boldsymbol{y}\right)\mathrm{d}\boldsymbol{x} \\ &= \frac{1}{1+\frac{1-\epsilon}{\epsilon}\prod_\ell \frac{1}{\sqrt{1-R_\ell}}\exp\left(-\frac{R_\ell}{2\sigma_\ell^2}y_\ell^2\right)}R_k y_k, \end{aligned} \tag{5.16}$$

where the notation $\int$ denotes multivariate integral with respect to x_1, x_2, ..., x_k on their supports $\mathbb{R}^K$. The associated MSE is then given by

$$\begin{aligned} M_{k,K} &= \mathbb{E}\left[\left(\hat{X}_k - X_k\right)^2 |\boldsymbol{Y}=\boldsymbol{y}\right] \\ &= \epsilon\left(1 - R_k \cdot I_{k,K}\right), \end{aligned} \tag{5.17}$$

where $\boldsymbol{R} = \left[R_1, R_2, \cdots, R_K\right]^T$,

$$I_{k,K} = \int \frac{\prod_l f_G\left(x_l;0,1\right)}{1+\frac{1-\epsilon}{\epsilon}\prod_l \frac{1}{\sqrt{1-R_l}}\exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right)} x_k^2 d\boldsymbol{x}, \tag{5.18}$$

and the subscript K of $M_{k,K}$ and $I_{k,K}$ is introduced to emphasise that the corresponding value depends on all the K components in the $\boldsymbol{R}$ vector.

It is hence clear that the MSE performance at the k-th sensor is affected by the SNRs at other sensors. We will see in Lemma 5.5 that it specifies the impacts: for a given sensor k, the higher the SNRs at other sensors are, the smaller MSE (the better performance) the sensor k gets. This is intuitively correct and introduced by the block sparse structure.

In addition, we include here some details that are required for the low complexity implementation proposed in the next section. Define

$$g_k'\left(\boldsymbol{y};\boldsymbol{\sigma}^2\right) := \frac{\partial}{\partial y_k} g_k\left(\boldsymbol{y};\boldsymbol{\sigma}^2\right) = \frac{\partial}{\partial y_k}\mathbb{E}\left[X_k|\boldsymbol{Y}=\boldsymbol{y}\right]. \tag{5.19}$$

It can be verified that

$$g_k'\left(\boldsymbol{y};\boldsymbol{\sigma}^2\right) = \frac{g_k\left(\boldsymbol{y};\boldsymbol{\sigma}^2\right)}{y_k} + \frac{g_k^2\left(\boldsymbol{y};\boldsymbol{\sigma}^2\right)}{\sigma_k^2} \cdot \frac{1-\epsilon}{\epsilon}\prod_l \frac{1}{\sqrt{1-R_l}}\exp\left(-\frac{R_l}{2\sigma_l^2}y_l^2\right). \tag{5.20}$$

Furthermore, we also consider the more general case given by (5.14) where $\boldsymbol{\Sigma}_X$ is not necessarily diagonal. In the Appendix we provide the computed result of the MMSE estimate $\mathbb{E}\left[X_k|\boldsymbol{Y}=\boldsymbol{y}\right]$, the associate MSE and the partial derivative of the MSE based on the general model (5.13).

Compared with the Group Lasso denoiser, it is possible to use MMSE denoiser to improve the performance to the reconstruction algorithm. For example, the message passing decoder in [84] which uses conditional mean of the signal as the prior information and achieves the optimal phase-transition threshold in the presence of additive Gaussian noise. Using MMSE denoiser, an accessible computation form of the mean squared error can be derived based on a given signal prior distribution. Given the signal distribution, the denoiser (5.16) can be designed which evidently brings better MSE than soft-thresholding functions. In the extreme case, we may assume the block sparse signal with each element having Bernoulli-Gaussian distribution $p_X\left(x;\epsilon,0,\sigma_g^2\to\infty\right)$ pursuing to a Bernoulli-Uniform distribution. The MMSE denoiser based upon that can be considered as an alternative to soft-thresholding functions in the scenario that the sparse signal distribution is unknown.

5.5.1 Special Cases for Computation Simplification

As $M_{k,K}$ in (5.17) involves K-variate integrals, the performance evaluation becomes challenging when K gets large. Here we discuss two special cases where the computations can be simplified.

One special case is that all the noise variances σ_k^2's are identical except one. Without loss of generality we assume $\sigma_1^2 \neq \sigma_2^2 = \sigma_3^2 = ... = \sigma_K^2$. For a fixed value z and consider any given $x_2, ..., x_K$ having $\sqrt{\sum_{k=2}^K x_k^2} = z \in \mathbb{R}^+$, we can use variable substitution to calculate (5.18) and consequently the $K-1$ multiple integral of $x_2, ..., x_K$ is replaced by a simple integral of a Chi distribution of variable z:

$$I_{1,K} = \int \frac{f_G(x_1;0,1)\, f_\chi(z;1,K-1)\, x_1^2}{1+\frac{1-\epsilon}{\epsilon\prod_{l\neq 1}\sqrt{R_l}} f_G(x_1;0,1)\, f_\chi\left(z;\frac{1-R_l}{R_l},K-1\right)}\mathrm{d}x_1\mathrm{d}z, \tag{5.21}$$

$$I_{2,K} = \int \frac{f_G(x_1;0,1)\, f_\chi(z;1,K-1)\, \frac{z^2}{K-1}}{1+\frac{1-\epsilon}{\epsilon\prod_{l\neq 1}\sqrt{R_l}} f_G(x_1;0,1)\, f_\chi\left(z;\frac{1-R_l}{R_l},K-1\right)}\mathrm{d}x_1\mathrm{d}z, \tag{5.22}$$

where $l \neq 1$, $\Gamma(\cdot)$ is the Gamma function and better

$$f_\chi\left(z;\sigma^2,k\right) = \frac{2^{1-\frac{k}{2}} z^{k-1} e^{-\frac{z^2}{2\sigma^2}}}{\sigma^k \Gamma(k/2)}.$$

This variable substitution will make it possible to analyse the theoretical bounds of block structured sparse signals with $K > 2$ in this particular form, by merely using double integrals.

The other special case is the homogeneous case, i.e., assume all the noise variance σ_k^2's are equal. This is a more general case and was used to analyse the state evolution for the Group Lasso problem [98]. By using the similar variable substitution, one assumes $\sqrt{\sum_{k=1}^K x_k^2} = z$. We are able to replace the K multiple integral (5.18) by one integral of Chi distribution:

$$I_{k,K} = \iint \frac{f_\chi(z;1,K)\, z^2}{1+\frac{1-\epsilon}{\epsilon}\prod_l \frac{1}{\sqrt{R_l}} f_\chi\left(z;\frac{1-R_l}{R_l},K\right)}\mathrm{d}z, \tag{5.23}$$

Remark 5.3. The expected MSE for each signal block converge with increasing K:

$$\lim_{K\to\infty} M_{k,K}(\boldsymbol{\tau}) = \epsilon \tag{5.24}$$

The detailed derivations of the above two special cases are given in Appendix B.4.

5.6 The AMP Based Reconstruction Algorithms

5.6.1 Joint Reconstruction

One of the key ideas of AMP is that at each iteration one computes a noisy observation of the true signal which is modeled as the true signal corrupted by a simple AWGN channel. In particular, at the t-th iteration, let $\tilde{\boldsymbol{x}}^t = \boldsymbol{x} + \boldsymbol{z}^t$ where $\boldsymbol{z}^t \in \mathbb{R}^n$ is the equivalent Gaussian

noise. Consider a block of the signal components $\boldsymbol{x}_{\cdot,i} = [x_{1,i}, x_{2,i}, \cdots, x_{K,i}]^T$. Then

$$\tilde{\boldsymbol{x}}^t_{\cdot,i} = \boldsymbol{x}_{\cdot,i} + \boldsymbol{z}^t_{\cdot,i}. \tag{5.25}$$

If the components of the noise $\boldsymbol{z}$ are independent and the statistics are known (which are true according to the state evolution analysis), then the model (5.25) coincides with the correspondent scalar case model. The MMSE estimator (5.16) can be applied and the MSE of the estimate can be computed via (5.17). The way to compute $\tilde{\boldsymbol{x}}^t$ was derived in [84] and is detailed in Algorithm 5.1.

To complete the algorithm design, one needs to characterise the statistics of the equivalent noise $\boldsymbol{z}^t$. This can be achieved by the state evolution analysis in the next section under certain conditions. Specifically, the equivalent noise vectors $\boldsymbol{z}^t_k$'s, $\forall k \in [K]$, are independent and approximated Gaussian distributed with distribution $\mathcal{N}\left(\boldsymbol{0}, \tau^t_k \boldsymbol{I}\right)$, $\forall k \in [K]$, where the variance τ^t_k can be computed in multiple ways. Let $\boldsymbol{\tau}^{t-1} = \left[\tau^{t-1}_1, \tau^{t-1}_2, \cdots, \tau^{t-1}_K\right]^T \in \mathbb{R}^K$ contain the equivalent noise variances from the $(t-1)$-th iteration. Compute $\boldsymbol{R}^{t-1} = \left[R^{t-1}_1, R^{t-1}_2, \cdots, R^{t-1}_K\right]^T \in [0,1]^K$ by substituting τ^{t-1}_k into (5.15). The MSE at the $(t-1)$-th reconstruction is given by $\boldsymbol{M}^{t-1} = \left[M^{t-1}_{1,K}, M^{t-1}_{2,K}, \cdots, M^{t-1}_{K,K}\right]$ where $M^{t-1}_{k,K}$ can be computed via (5.17). Then

$$\tau^t_k = M^{t-1}_{k,K} / \left(\frac{m_k}{n}\right) + \sigma^2_k, \ \forall k \in [K].$$

However, the computation of $M^{t-1}_{k,K}$ involves multiple integrations which may be computational challenging. Lower-complexity alternatives are needed.

One way to simplify the computation of τ^t_k is as follows. Recall the definitions of $g_k(\cdot;\cdot) \in \mathbb{R}$ and $g'_k(\cdot;\cdot) \in \mathbb{R}$ in (5.16) and (5.19) respectively. For a given $\tilde{\boldsymbol{x}} \in \mathbb{R}^{Kn}$ and $\boldsymbol{\tau} \in \mathbb{R}^K$, with slight abuse of notations, define

$$g_k(\tilde{\boldsymbol{x}}; \boldsymbol{\tau}) = \left[g_k(\tilde{\boldsymbol{x}}_{\cdot,1}; \boldsymbol{\tau}), g_k(\tilde{\boldsymbol{x}}_{\cdot,2}; \boldsymbol{\tau}), \cdots, g_k(\tilde{\boldsymbol{x}}_{\cdot,n}; \boldsymbol{\tau})\right]^T \in \mathbb{R}^n,$$

Algorithm 5.1 The Joint Reconstruction Based on AMP (Type I strategy)

Input: $\boldsymbol{y}$, $\boldsymbol{A}$, ϵ, σ_g^2, and $\boldsymbol{\sigma}^2 = \left[\sigma_1^2, \cdots, \sigma_K^2\right]$.
Initialisation:
$\boldsymbol{x}^0 = \boldsymbol{0}$, $\boldsymbol{r}^0 = \boldsymbol{y}$, $\tilde{\boldsymbol{x}}^0 = \boldsymbol{x}^0 + \boldsymbol{A}^T\boldsymbol{r}^0$, $\tau_k^0 = \epsilon\sigma_g^2/\left(\frac{m_k}{n}\right) + \sigma_k^2$, $\forall k \in [K]$.
Iteration: Let $t = 1, 2, \cdots$, until the stop criteria are met.

$$\boldsymbol{x}_k^t = g_k\left(\tilde{\boldsymbol{x}}^{t-1}; \boldsymbol{\tau}^{t-1}\right), \ \forall k \in [K]. \tag{5.26}$$

$$u_k^t = \frac{1}{m_k}\left\langle g_k'\left(\tilde{\boldsymbol{x}}^{t-1}; \boldsymbol{\tau}^{t-1}\right)\right\rangle, \ \forall k \in [K]. \tag{5.27}$$

$$\tau_k^t = u_k^t + \sigma_k^2, \ \forall k \in [K]. \tag{5.28}$$

$$\boldsymbol{r}_k^t = \boldsymbol{y}_k - \boldsymbol{A}_k\boldsymbol{x}_k^t + u_k\boldsymbol{r}_k^{t-1}, \ \forall k \in [K]. \tag{5.29}$$

$$\tilde{\boldsymbol{x}}_k^t = \boldsymbol{x}_k^t + \boldsymbol{A}_k^T\boldsymbol{r}_k^t, \ \forall k \in [K]. \tag{5.30}$$

and similarly define $g_k'\left(\tilde{\boldsymbol{x}}; \boldsymbol{\tau}\right) \in \mathbb{R}^n$. Then using ([84], Lemma 2), we have

$$\begin{aligned}\tau_k^t &= \frac{n}{m_k}\mathbb{E}\left\{\left\|\tilde{\boldsymbol{x}}_{\cdot,k}^{t-1} - \boldsymbol{x}_{\cdot,k}\right\|_2^2 | g_k\left(\tilde{\boldsymbol{x}}_{\cdot,k}; \boldsymbol{\tau}\right)\right\} + \sigma_k^2 \\ &\simeq \frac{\tau_k^{t-1}}{m_k}\left\langle g_k'\left(\tilde{\boldsymbol{x}}^{t-1}; \boldsymbol{\tau}^{t-1}\right)\right\rangle + \sigma_k^2,\end{aligned}$$

where $\left\langle g_k'\left(\tilde{\boldsymbol{x}}; \boldsymbol{\tau}\right)\right\rangle$ sums up all the elements in $g_k'\left(\tilde{\boldsymbol{x}}; \boldsymbol{\tau}\right) \in \mathbb{R}^n$. In this way, the computation of τ_k^t does not involve multiple integrations but elementary algebra.

With above notations, the joint reconstruction algorithm is given in Algorithm 5.1.

5.6.2 An Alternative Update Strategy

Joint AMP algorithm uses the benefit of common support information to improve the reconstruction performance. However in each iteration (5.26) to (5.30) will be executed for K many times and the computational cost linearly increases with the signal and observation dimensions. Consider a setting with no noise present and the observation dimension is large enough, it is possible to find a way to decrease the computational cost while still keep an accurate reconstruction. We propose the second type of Joint AMP algorithm (Type II strategy, presented in Algorithm 5.2) to balance the performance and the computations. The basic idea is that instead of simultaneously update all the signal blocks, in each loop we focus on updating one signal k till its equivalent noise τ_k stop decreasing, followed by updating the rest of the signals for one time. Then we repeat the loop until all the signals converge. Lemma 5.5, which will be given in section 5.7, guarantees that after

Algorithm 5.2 The Joint Reconstruction Based on AMP (Type II strategy)

Input: $\boldsymbol{y}$, $\boldsymbol{A}$, ϵ, σ_g^2, $\boldsymbol{\sigma}^2 = \left[\sigma_1^2, \cdots, \sigma_K^2\right]$ and Joint update tolerance ε.
Initialisation:
$\boldsymbol{x}^0 = \boldsymbol{0}$, $\boldsymbol{r}^0 = \boldsymbol{y}$, $\tilde{\boldsymbol{x}}^0 = \boldsymbol{x}^0 + \boldsymbol{A}^T\boldsymbol{r}^0$, $\tau_k^0 = \epsilon\sigma_g^2/\left(\frac{m_k}{n}\right) + \sigma_k^2$, $\forall k \in [K]$;
Find $m_j = \max(m_1, ..., m_K)$.
Iteration: Let $t = 1, 2, \cdots$.
Set $\mathcal{J} = \begin{cases} [K], & \text{if } \tau_j^{t+1} - \tau_j^t < \varepsilon; \\ \{j\}, & \text{otherwise.} \end{cases}$

$$\boldsymbol{x}_k^t = g_k\left(\tilde{\boldsymbol{x}}^{t-1}; \boldsymbol{\tau}^{t-1}\right), \ \forall k \in \mathcal{J}.$$

$$u_k^t = \frac{1}{m_k}\left\langle g_k'\left(\tilde{\boldsymbol{x}}^{t-1}; \boldsymbol{\tau}^{t-1}\right)\right\rangle, \ \forall k \in \mathcal{J}.$$

$$\tau_k^t = u_k^t + \sigma_k^2, \ \forall k \in \mathcal{J}.$$

$$\boldsymbol{r}_k^t = \boldsymbol{y}_k - \boldsymbol{A}_k\boldsymbol{x}_k^t + u_k\boldsymbol{r}_k^{t-1}, \ \forall k \in \mathcal{J}.$$

$$\tilde{\boldsymbol{x}}_k^t = \boldsymbol{x}_k^t + \boldsymbol{A}_k^T\boldsymbol{r}_k^t, \ \forall k \in \mathcal{J}.$$

updating the signals $l \neq k$, τ_k decreases as well. This strategy actually provides a different optimization path for $\boldsymbol{x}_k$'s which saves their number of updates on average. For noise free cases, this adjustment on the number of iterations of the signal $\boldsymbol{x}_k$'s via the sampling rates $\boldsymbol{\delta}_k$'s, it is possible to make it cost much less time than type I strategy to achieve accurate reconstruction. The Joint AMP algorithm with type II strategy is presented in Algorithm 5.2 and the dedication of the detailed analysis is provided in the next section.

5.7 Phase Transition via State Evolution

On the convergence study of AMP, each iteration can be simplified to an update of a scalar denosing problem. The input of the denoiser can be written as the true signal plus an equivalent additive noise. When the measurement matrix has i.i.d. Gaussian entries and the matrix size is large enough, the equivalent additive noise is also shown as i.i.d. Gaussian and is independent of the true signal [8]. Such appealing feature, termed as state evolution, allows us to track the phase transition (PT) as well as the MSE of the AMP algorithms. Generally speaking, PT is the curve which quantitatively divide the sparsity-sampling area into recovery achievable and unachievable region. Further since the MSE is a function of the sparsity rate ϵ and the sampling rate δ, one may check the convergence point to see whether a given δ is sufficient large for the noise free accurate reconstruction

of a given ϵ. In our DCS problem, we use the same concept, except that we take multiple signals sharing identical support as the additional information. More concretely, we convey this information into the denoiser for the updates. Following similar procedure we can still use the state evolution technique to track the equivalent noise on each signal block.

In this section we follow the approach in [30] by completing all the analysis in the asymptotic regime, i.e., given m_k and n defined in (5.1), we assume $m_k, n \to \infty$ with $m_k/n \to \delta_k$ where δ_k is a constant $\forall k \in [K]$, and the measurement matrices $\boldsymbol{A}_i$'s are generated from the standard Gaussian random matrix ensemble. We list the main results with illuminating explanations of the relation to the reconstruction algorithm, while leave their rigorous proof to the appendices.

State evolution is built upon the equivalent signal model specified in (5.25), which can be re-written in the following form

$$\tilde{\boldsymbol{x}}_k^t = \boldsymbol{x}_k + \boldsymbol{z}_k^t, \tag{5.31}$$

where $\boldsymbol{z}_k^t$ are the equivalent white Gaussian noise $\mathcal{N}\left(\boldsymbol{0}, \tau_k^t \boldsymbol{I}\right)$ and, with a slight abuse of notations, τ_k^t can be defined as

$$\tau_k^t = \frac{1}{n}\left\|\tilde{\boldsymbol{x}}_k^t - \boldsymbol{x}_k\right\|_2^2. \tag{5.32}$$

Furthermore, due to the independence among $\boldsymbol{A}_k$'s, it can be proved that $\boldsymbol{z}_k^t$ and $\boldsymbol{z}_l^t$ are independent for $1 \le k \ne l \le K$. Recall the model for the scalar case. It can be shown that the value τ_k^t can be obtained via scalar case analysis:

$$\begin{aligned}\tau_k^t &\to \frac{1}{\delta_k}\mathbb{E}\left\{\left(g_k\left(\boldsymbol{x} + \boldsymbol{z}^{t-1};\ \boldsymbol{\tau}^{t-1}\right) - x_k\right)^2\right\} + \sigma_k^2 \\ &= \frac{M_{k,K}^{t-1}}{\delta_k} + \sigma_k^2 \end{aligned} \tag{5.33}$$

almost surely, where the convergence holds as $g_k(\cdot)$ is Lipchitz continuous [8, Theorem 1], $\boldsymbol{z}^{t-1} \in \mathbb{R}^K$, and $\boldsymbol{z}^{t-1} \sim \mathcal{N}\left(\boldsymbol{0}, \operatorname{diag}\left(\cdots, \tau_k^{t-1}, \cdots\right)\right)$. Note that $M_{k,K}^{t-1}$ is a function of $\boldsymbol{\tau}^{t-1}$. Equation (5.33) gives the state evolution of the equivalent noise variance. The complete procedure for state evolution is presented in Algorithm 5.3. It is worth pointing out that

Algorithm 5.3 The State Evolution of Joint AMP.

Input: $\boldsymbol{\delta} = [\delta_1, ..., \delta_K]$, ϵ, σ_g^2, and $\boldsymbol{\sigma}^2 = \left[\sigma_1^2, \cdots, \sigma_K^2\right]$.
Initialisation:
$\tau_k^0 = \epsilon\sigma_g^2/\delta_k + \sigma_k^2$, $\forall k \in [K]$.
Iteration: Let $t = 1, 2, \cdots$, until the stop criteria are met.

- $\tau_k^{t+1} = M_{k,K}^t/\delta_k + \sigma_k^2$, $\forall k \in [K]$.
- Compute $M_{k,K}^{t+1}$ via (5.17), $\forall k \in [K]$.

the result is equivalent to joint AMP for our DCS model, provided that the signals on the same location of each group are generated via a same distribution and the number of measurement of each group is equal.

In the asymptotic analysis, the exact recovery of x_k is defined as the case where $\lim_{t\to\infty}\tau_k^t = 0$ or equivalently $\lim_{t\to\infty}M_{k,K}^t = 0$. Note that $M_{k,K}^{t-1}$ is a function of $\boldsymbol{\tau}^{t-1}$, whose values can be computed from $\boldsymbol{\tau}^{t-2}$ and depend on the values of δ_l's, $l \neq k$. Therefore we can define the following.

Definition 5.4. The phase transition is described by the vector $\boldsymbol{\delta}^* = [\cdots, \delta_k^*, \cdots]^T$ such that exact reconstruction is guaranteed if $\boldsymbol{\delta} \geq \boldsymbol{\delta}^*$ (holds element-wisely), and exact reconstruction is impossible if there exists k that $\delta_k < \delta_k^*$ and $\delta_l = \delta_l^*$, $\forall l \neq k$.

Lemma 5.5. *For given $1 \leq k \neq l \leq K$, assume that R_k, $R_\ell \in (0,1)$ ($R_k := \frac{1}{1+\tau_k}$) or equivalently $0 < \tau_k, \tau_l < \infty$. It holds that*

$$\frac{\partial M_{k,K}}{\partial R_l} < 0, \quad \textit{or equivalently } \frac{\partial M_{k,K}}{\partial \tau_l} > 0. \tag{5.34}$$

As a direct consequence,

$$M_{k,K}\left(\tau_l < \infty\right) < M_{k,K}\left(\tau_l = \infty\right),$$

which shows the benefit of joint reconstruction.

Using this lemma, of which the proof is given in Appendix A, we confirm the existence of $\boldsymbol{\delta}$ to guarantee exact recovery as follows. Consider a trivial sufficient condition for the exact recovery of $\boldsymbol{x}_k$:

$$\tau_k^t = M_{k,K}^{t-1}/\delta_k < \tau_k^{t-1}, \ \forall t \in \mathbb{N}^+, \tag{5.35}$$

where the superscript t represents for the t^{th} iteration. Lemma 5.5 and (5.35) tell that closer estimation to the true signal $\boldsymbol{x}_l$ (i.e., smaller $\left\|\boldsymbol{x}_l^t - \boldsymbol{x}_l\right\|_2^2$, $l \neq k$) requires smaller minimum sampling rate δ_k for accurate reconstruction on $\boldsymbol{x}_k$. The following proposition states that $M_{k,K}/\tau_k$ is uniformly upper bounded for $\tau_k \in \mathbb{R}^+$. It can be seen from (5.33) that the sampling rate δ_k's jointly decide the success and the convergence rate of the reconstruction. We will unfold the analysis around (5.35), then show that the phase transitions of joint AMP can be numerically determined on the base of the following discovered propositions.

Proposition 5.6. *For a given $k \in [K]$, fix all τ_l, $l \neq k$. Then the function $M_{k,K}/R_k$ is continuous and admits a maximum. The same is true for $M_{k,K}/\tau_k$.*

This proposition is easy to be verified by checking the limits $\lim_{R_k \to 0} M_{k,K}/R_k = 0$, $\lim_{R_k \to \infty} M_{k,K}/R_k = \epsilon > 0$ and $\lim_{R_k \to \infty} \partial M_{k,K}/\partial R_k = +\infty$. This proposition, combined with 5.35, enlightens the following one to determine lower bounds of δ_k's that guarantee the exact recovery.

Proposition 5.7. *Let $\delta_1, \delta_2 ..., \delta_K \geq \epsilon$. Consider the expected mean squared error $M_{k,K}$ as a function of δ_k and τ_k. Exact recovery of $\boldsymbol{x}$ is guaranteed if for any given $\boldsymbol{\tau} \in (0, +\infty)^K$, there always exists one $M_{k,K}$, $k \in [K]$, such that*

$$\frac{M_{k,K}}{\tau_k} < \delta_k. \tag{5.36}$$

It's worth mentioning that for any given $\boldsymbol{\tau}$, there are K quantities $\frac{M_{k,K}}{\tau_k}$, $k = 1, 2 ..., K$. Proposition (5.7) only needs one of the K inequalities hold and it doesn't have to be with the same k for different $\boldsymbol{\tau}$'s. This brings benefit at AMP updates since in each iteration the equivalent noise vector $\boldsymbol{\tau}^t$ changes therefore for at each iteration we only need the union condition $\cup_{k=1}^{K} \left(\frac{M_{k,K}}{\delta_k} < \tau_k \right)$, which is more flexible than any of the condition $\frac{M_{k,K}}{\delta_k} < \tau_k$. An intuitive explanation behind Proposition 5.7 is that, after each iteration, we can always find one τ_k in $\boldsymbol{\tau}$ is reduced. Follow Lemma 5.5, which shows that smaller τ_k will decrease $M_{l,K}$, $k \neq l$, therefore the whole $\boldsymbol{\tau}$ reduces. This cooperation mode between the signals is the key to relax the requirement for accurate reconstruction, i.e., the lower bounds for the sampling rates will decrease compared to individual decoding strategy. In practice, given sampling rates δ_k's, we can theoretically determine that whether the joint reconstruction

is successful via checking the convergence of the following process, see Algorithm 5.3.

The type II update strategy given in Algorithm (5.2) considers not simultaneously update all the signals in model (5.2) but keep update one signal and only update the rest when necessary. We treat the estimations of $\boldsymbol{x}_{l\neq k}$'s as the supportive information, which according to Lemma 5.5 can move up/down the transition bound for reconstruction signal $\boldsymbol{x}_k$. Notice the sharp transition feature of the AMP successful reconstruction rate, we can check the change of τ_k between two iterations to decide whether the $\boldsymbol{x}_{l\neq k}$'s needs to be updated in the next iteration. If τ_k is large but the change compared to the last iteration is very small, it means that the phase transition supported by the current estimates $\boldsymbol{x}_{l\neq k}$ is not good enough, therefore an update of these estimates is needed. This strategy, listed in Algorithm (5.2), has the same phase transition performance as Algorithm (5.1) but achieves faster reconstruction speed. A simulating comparison is given in the next section to see the actual improvements.

In the extreme case we may have $\tau_l \to 0$ or equivalently $R_l \to 1$. Then $I_{k,K} \to 0$, i.e., $M_{k,K} \to \epsilon$. It shows that if an accurate reconstruction on signal $\boldsymbol{x}_l$ is obtained then joint algorithm will drive the reconstructions of $\boldsymbol{x}_k$, $k \neq l$, to be accurate with requiring sampling rates $\delta_k \geq \epsilon$. This is intuitively correct since all the $\boldsymbol{x}_k$'s share the same support, an accurate reconstruction on $\boldsymbol{x}_l$ brings the true support. Then the reconstruction on the rest $\boldsymbol{x}_k$'s are just ordinary least squares problems.

The following corollary is motivated by the Proposition 4 which guarantee the accurate reconstruction of $\boldsymbol{x}$. From Lemma 5.5, for $\tau_j > \tau_j^{'}$ and $\tau_k^* = \arg\max_{\tau_k} \left(\frac{M_{k,K}}{\tau_k}\right)|_{\tau_j^{'}}$, $j \neq k$, the following inequality holds

$$\frac{M_{k,K}}{\tau_k^*}|_{\tau_j^{'}} < \frac{M_{k,K}}{\tau_k^*}|_{\tau_j} < \max_{\tau_k}\left(\frac{M_{k,K}}{\tau_k}\right)|_{\tau_j}. \tag{5.37}$$

Combine (5.37), Lemma 5.5 and Proposition 5.6, we can choose one $k \in [K]$, maximise $\frac{M_{k,K}}{\tau_k}$ and minimize the rest $\frac{M_{l,K}}{\tau_l}$'s, $l \neq k$ to find the overall minimal sampling rate. The problem to optimise this total sampling rate $\delta = \sum_{k=1}^{K}$ is listed in Corollary 5.8.

Corollary 5.8. *Minimizing the overall sampling rate δ which guarantees accurate recovery*

for joint AMP can be settled via

$$\delta = \min_{\tau_2,\ldots,\tau_k} \left(\max_{\tau_1} \left(\frac{M_{1,K}}{\tau_1} \right) + \sum_{k=2}^{K} \frac{M_{k,K}}{\tau_k} \right) \quad \text{s.t. } \frac{M_{k,K}}{\tau_k} \geq \epsilon,\ \forall k \in [K]\,. \tag{5.38}$$

Note that by choosing any variable τ_j, both $\max_{\tau_k}\left(\frac{M_{k,K}}{\tau_k}\right)$ and $\frac{M_{k,K}}{\tau_k}$ are monotone increasing with $\tau_{j\neq k}$, and there exist a maximum on $\frac{M_{j,K}}{\tau_j}$. Therefore 5.38 is not a tricky optimization problem. Corollary (5.8) is not obvious and we usually need to use coordinate descent methods to find the solution. However for some special cases, e.g., given in (5.21) and (5.22), we have determined equation for this problem: e.g., simply consider function $f_1(\tau_j) = \max_{\tau_1}\left(\frac{M_{1,K}}{\tau_1}\right)$ and $f_2(\tau_j) = \sum_{k=2}^{K} \frac{M_{k,K}}{\tau_k}$, where f_1 represents the sampling rate for the first signal, and f_2 represents the rest of the signals, $i \neq 1$. Then $\delta_{\min}$ admits where the equation $(K-1)\cdot \partial f_1/\partial \tau_j + \partial f_2/\partial \tau_j = 0$ holds.

In Figure 5.4, we compare the PTs of several decoding strategies, including individual decoding, sequential decoding, the joint decoding proposed in Algorithm 5.1, and the optimal decoding. Here, the individual decoding is referred to the case that each sensor does not consider the side information from the other sensors and perform AMP decoding using only the measurements at this particular sensor. In sequential decoding, one chooses a sensor and apply the individual decoding to the measurements at the sensor, and then in decoding the data at the other sensor, one uses both the decoded signal from the first sensor and the raw data from the second sensor. The optimal decoding is essentially ℓ_0-minimisation, or exhaustive search. It results in impractical computational complexity but gives the best PT, i.e., $\delta_k^* = \epsilon,\ \forall k \in [K]$, in the information theoretical sense. This limit can be verified using information dimension concept [111] (See a short derivation in Appendix B.8). From the results presented in Figure 5.4, joint decoding is substantially better than either individual decoding or sequential decoding by taking the raw data from all sensors into account. It does not compete with the optimal decoder but it is practical. Another interesting observation is the shape of the PT curve of joint decoding. It is concave in the δ_1-δ_2 plane, which suggests that equally allocate measurements by given a total measurement budget is strictly suboptimal. Instead the optimal policy is to allocate

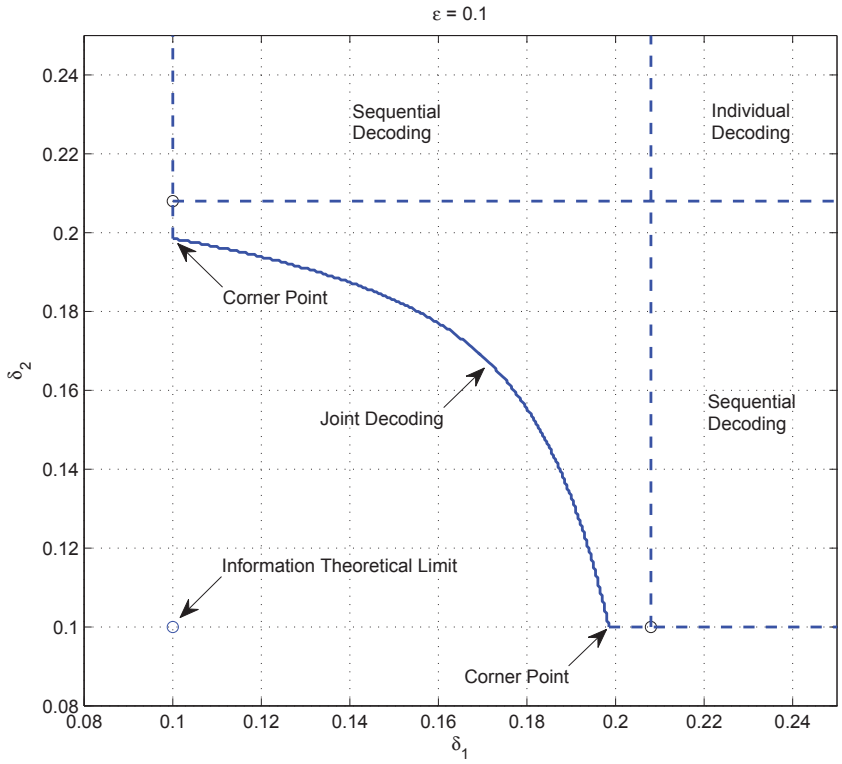

Figure 5.4: The phase transitions for case $K = 2$ using joint AMP, from where the area right above is the achievable area. The sparsity rates are $\epsilon = 0.1$. The signal is assumed as Bernoulli-Gaussian distributed.

$\delta_\ell = \epsilon$ (given by information theoretical limit) to all but one sensors $\ell \neq k$ and let δ_k be chosen to allow an exact recovery (denoted as "corner points" in the figure).

The concavity of PT curve can be observed for some other prior distributions, Here we consider two-block signals with Bernoulli-Uniform $p_{X_1,X_2}(x_1,x_2) = (1-\epsilon)\,\delta_{(0,0)} + \frac{\epsilon}{4}U_{X_1}(-1,1)\cdot U_{X_2}(-1,1)$. Then we do exactly the same derivations for the phase transitions and results in a similar curve (shown in Figure 5.5) as in 5.4. Based on this test, we conjecture that it is not because of Bernoulli-Gaussian distribution but, more generally, a mixed (discontinuous-continuous) distribution providing the benefit of unequal measurement allocation. A vitrifaction of this conjecture will potentially bring decent practical value as it covers a large range of block sparse signals.

In Figure 5.6 we show that for block sparse signals with Bernoulli-Uniform, allocation at "corner points" is also better than equal allocation for multiple group sparse signals ($K \geq 2$). We fix the the sparsity rates $\delta_k = \epsilon$, $k = 2\tilde{}K$, $K = 2, 3, 5, 10$. Phase transitions

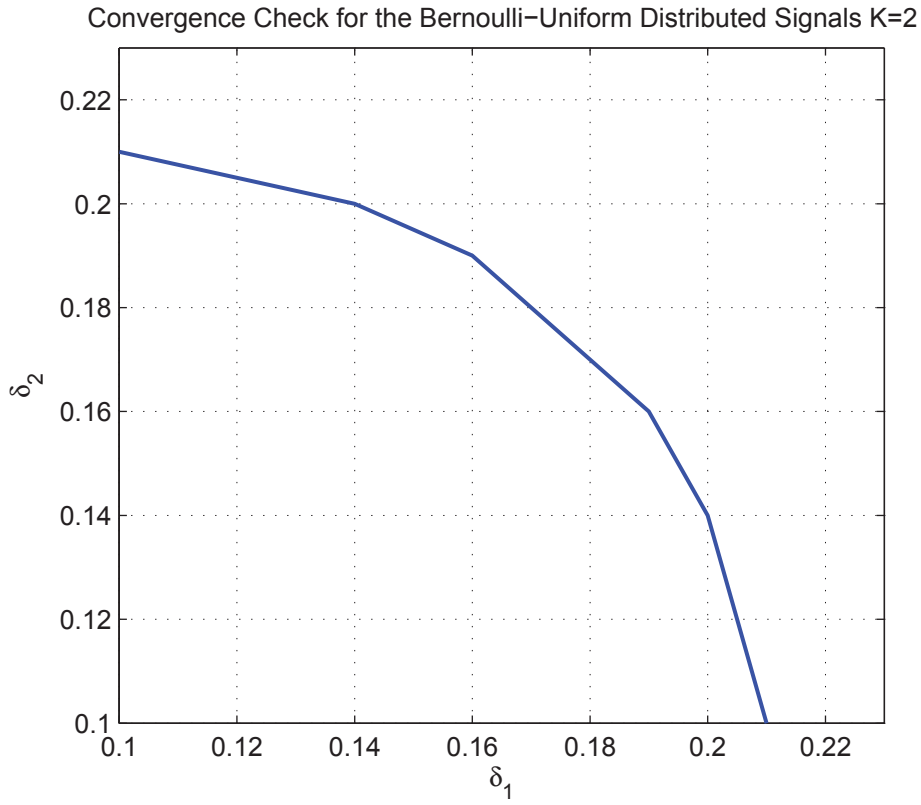

Figure 5.5: The phase transitions for case $K = 2$ using joint AMP. The sparsity rates are $\epsilon = 0.1$. The signal is assumed as Bernoulli-Uniform distributed.

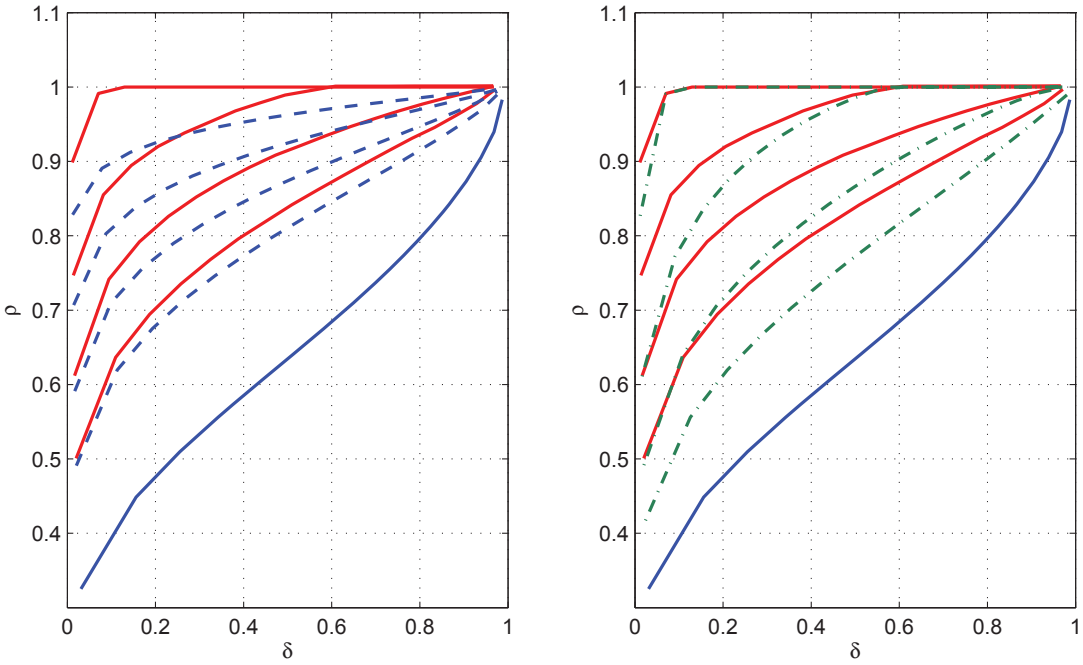

Figure 5.6: Theoretical Phase Transitions of joint AMP with common support signals. Allocations at corner points are presented in red lines, sequential decoding are in blue dashed lines and the equal allocations are in green lines. Each group of lines from bottom to top are for $K = 2, 3, 5, 10$. Phase Transition Curve for signal block case is also plotted in blue solid line, i.e., $K = 1$.

for joint decoding at "corner points" and equal allocation case are compared with sequential decoding. The performance gain of joint decoding is noticeable for all plotted cases. It can also been seen from Figure 5.6 that both joint and sequential decoding will reach the theoretical limit when $K \to \infty$.

In practice, it is interesting to study the case that the nonzero $\boldsymbol{x}_{\cdot,i} \in \mathbb{R}^K$ contains correlated components, i.e., the model in (5.13) with a non-diagonal covariance matrix $\boldsymbol{\Sigma}$. For simplicity, only consider the case $K = 2$. Define the correlation $\nu := \mathrm{cov}\left(X_{1,i}, X_{2,i}\right)/\sigma_g^2$. The corresponding joint decoding AMP algorithm can be obtained and the associated PT can be quantified. Figure 5.7 depicts the PT curves for different ν obtained from Algorithm 5.3 where the derivations of terms $M_{k,K}$'s are shown in Appendix B.6 and computed by using numerical integrals. Consistent with intuition, the numbers of measurements required for exact recovery present an opposite tendency to ν. In the extreme case when $\nu = 1$, $\boldsymbol{x}_1 = \boldsymbol{x}_2$ and the PT curve becomes a straight line. The concavity of the curves when $\nu \neq 1$ shows that best allocation for the signals are always at the corner points.

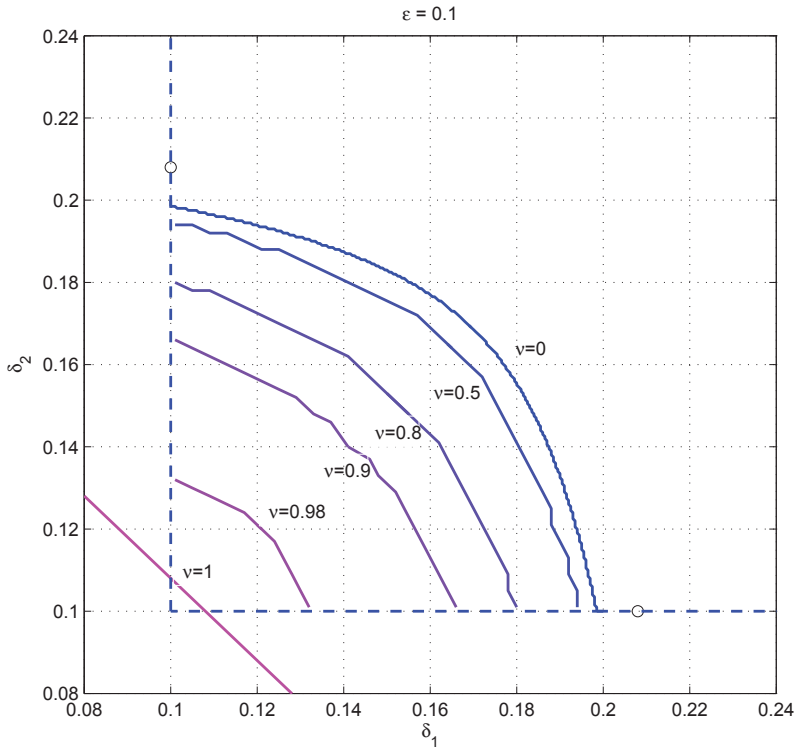

Figure 5.7: The minimum achieved sampling rates for common sparse support signals ($K = 2$, $\epsilon = 0.1$) with given inter-correlation $\nu \in [0,\ 1]$.

When additive white Gaussian noise is considered in model 5.2, terms τ_k^t in will still converge (see in Proposition 6) and can be traced via the state evolution 5.33. Therefore we can predict the reconstruction MSE for different sparse signal blocks.

Proposition 5.9. *Given signal model* (5.1) *with the sparsity rate* ϵ*, the sampling rate* δ_k*s and noise variance* σ_k^2*. Then via joint-AMP (Algorithm 1), the reconstruction MSE for signal block* $\boldsymbol{x}_k$ *converges to* $M_{k,K}^{t\to\infty}$*,* $k \in [K]$*, where*

$$\lim_{t\to\infty} \frac{M_{k,K}^t}{\tau_k^t - \sigma_k^2} = \delta_k, \text{ and } \frac{\partial M_{k,K}^t}{\partial \tau_k^t} < \delta_k. \tag{5.39}$$

The limit equation in (5.39) is essentially another way of writing the convergent state of (5.33). We also need to ensure that term $\frac{M_{k,K}^t}{\tau_k^t - \sigma_k^2}$ doesn't decrease with the update of τ_k^t, therefore $\partial\left(\frac{M_{k,K}^t}{\tau_k^t - \sigma_k^2}\right)/\partial\tau_k^t < 0$, which gives the inequality in (5.39). A practical way to find the convergent MSE can be obtained from Algorithm 5.3.

5.8 Numerical Study

Phase transitions are used to describe the underlying principles of a compressed sensing reconstruction algorithm in the asymptotic regime. Actual uses in handling large dimensional data usually reach very close performance to the theoretical results. In this section all the simulation tests use randomly generated group sparse signal $\boldsymbol{x}$ via Bernoulli-Gaussian distribution (5.13). From the above inference we know that finding the theoretical phase transition needs to compute integral 5.18. Due the limitation of the computer tools at hand which does not support double integral on interval $[-\infty, \infty]$, we instead look for an appropriate interval for the integral by using the 6σ rules of the Gaussian distribution. It is shown in Appendix B.7 that the calculation accuracy is enough of our demand (the tolerance will be less than $1 - \frac{\epsilon}{1-\epsilon}(1 - 7.3 \times 10^{-8}) \cdot \left(1 - 2 \times 10^{-9}\right)^{K-1} \prod_l (1 - R_l)$).

We firstly evaluate the accurate recovery of joint AMP. By doing this we choose single signal and two signals models. For the two signal models, we choose three decoding strategy including joint AMP with equal sampling rates, joint AMP with two signals with sampling rates at "corner points" and sequential decoding (using AMP) as the comparison objects, respectively. The measurement matrix $\boldsymbol{A}_k$'s are drawn from the i.i.d. Gaussian distribution

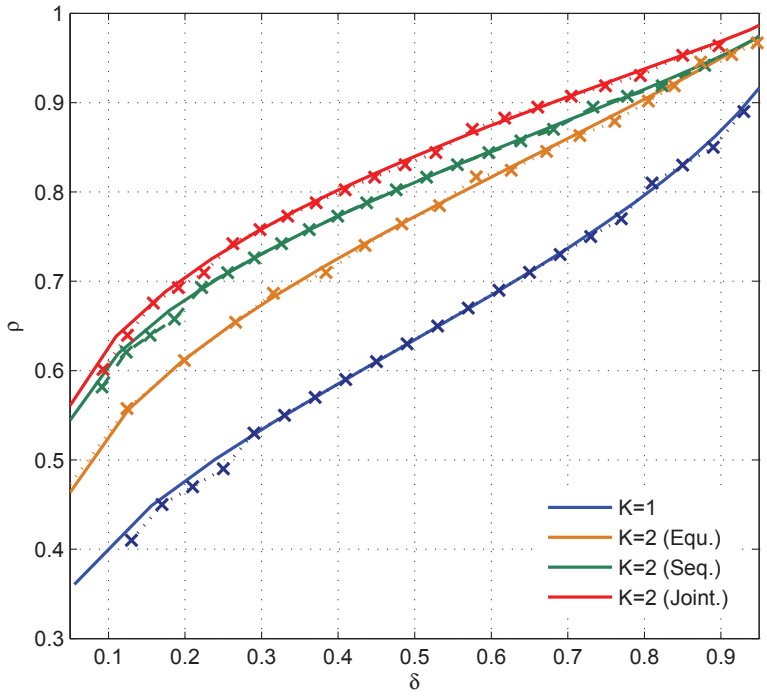

Figure 5.8: Numerical results (dashed lines) compare with the theoretical curves. Here all the curves are shown in ρ against δ. $n = 1000$. Each point is averaged from 100 realizations.

$\mathcal{N}\left(0, \frac{1}{m_k}\right)$. Figure 5.8 shows the predicted and the empirical phase transitions for the four comparison objects. Our experiment interval is chosen in $(0.05, 0.95)$. Each signal block contains $n = 1000$ elements. The simulation result for each point is averaged from 100 repetitions. It is shown that the predicted results is quite accurate compared to the empirical ones. The performance gain between joint decoding at "corner points" and sequential decoding (red and green curves) is mainly reflected when the total sampling $\delta < 0.7$.

In Figure 5.9 we compare the performance gain between the joint AMP and sequential decoding. We set the signal sparsity rate $\epsilon = 0.3$, fix $\delta_k = \epsilon$, $k = 2 \sim K$. The length of each signal block $n = 200$ and the number of blocks $K = 2, 4, 8, 16$. Each test is repeated by 400 times. The figures are plot under reconstruction rate against number of measurement per signal block. The rapid rising part of the curve located more on the left means the better the performance. Comparing the vertical interval of each pair of joint to sequential decoding, we can see that joint decoding outperforms sequential decoding and the gain

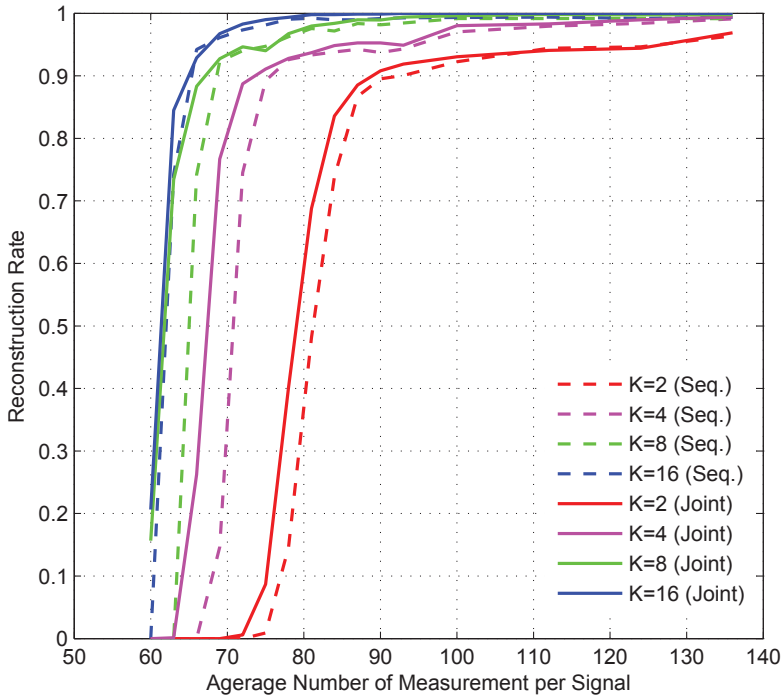

Figure 5.9: The performance gain from sequential decoding (dashed lines) to joint decoding (solid lines). We choose signal blocks $K = 2, 4, 8, 16$ in our trails. We compare the reconstruction successful rate with the average number of measurement per signal block. $\epsilon = 0.3$, $n = 200 \cdot K$. Each point is the average from 400 realizations.

becomes larger when there are more signals participate into the decoding process.

Further we compare the time saving by using type II update strategy. In Figure 5.10 we choose $\epsilon = 0.1$, $n = 1000$ and let $K = 2 \sim 30$. No noise is added and the the figures plot the time consumed for accurate recoveries. Note that in each iteration type II strategy contains $\frac{1}{K}$x computation cost of type I, however Type II strategy considers the update right above the phase transitions, it usually takes more iterations than the type I strategy. Overall we can see linear increasing trend for both strategies and the slope for type I is about 5x of type II.

In Figure 5.11 and 5.12 we test the noise sensitivity predictive performance. We consider Bernoulli Gaussian distributed signals. By state evolution the reconstruction MSE is always finite and we are able to predict it. We choose sparse signal with $K = 2$, each signal has length 1000 and choose sparsity rate $\epsilon = 0.1$ and 0.5, and then compare the empirical MSE (eMSE) and the theoretical MSE (fMSE) curves from state evolution by changing the sampling rates. 10dB white Gaussian noise is added on the observations $\boldsymbol{y}$.

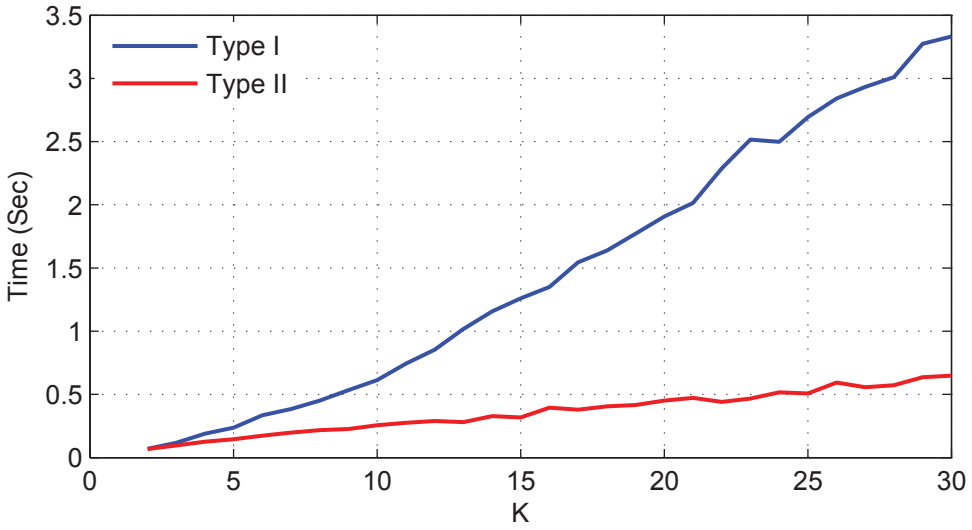

Figure 5.10: The consuming time between joint AMP type I & II update strategy with increasing the number K.

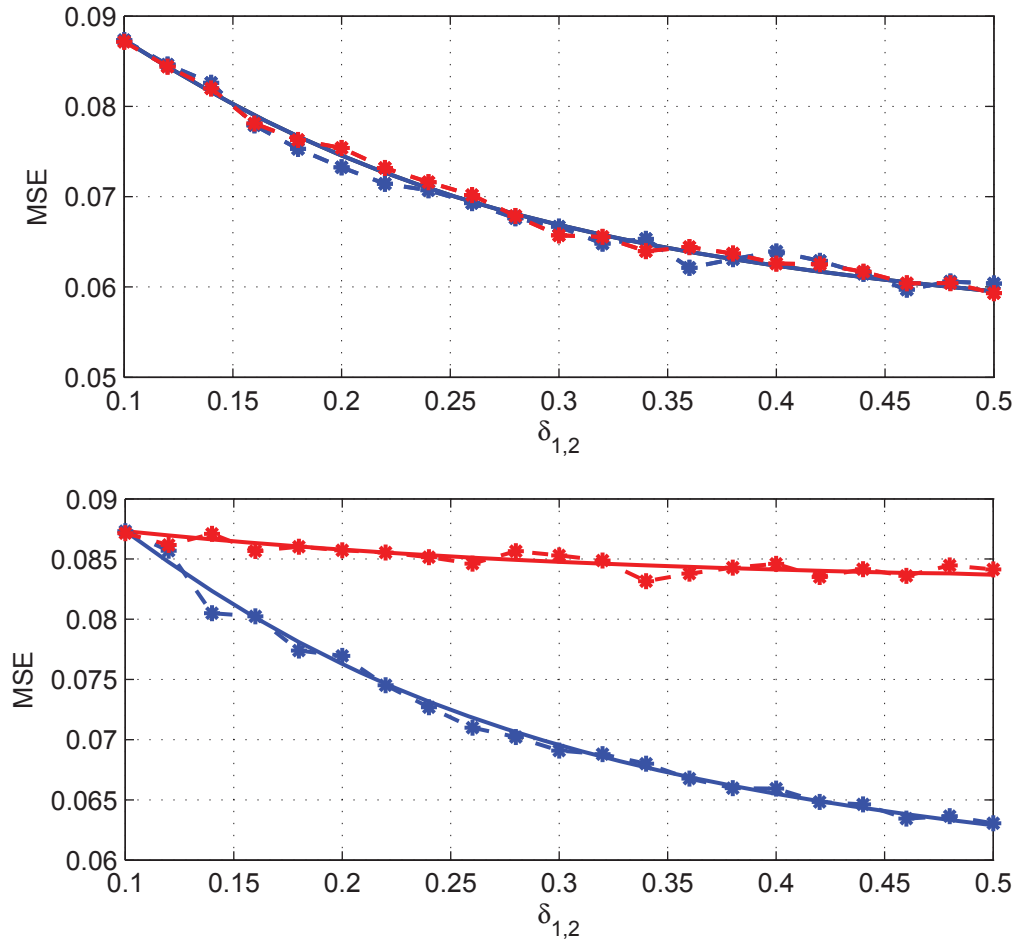

Figure 5.11: Noise sensitivity test, theoretical results in solid lines and empirical results in dashed lines. $\epsilon = 0.1$, $\nu^2 = 0$. Up:$\delta_1 = \delta_2$, down: $\delta_2 = \epsilon$.

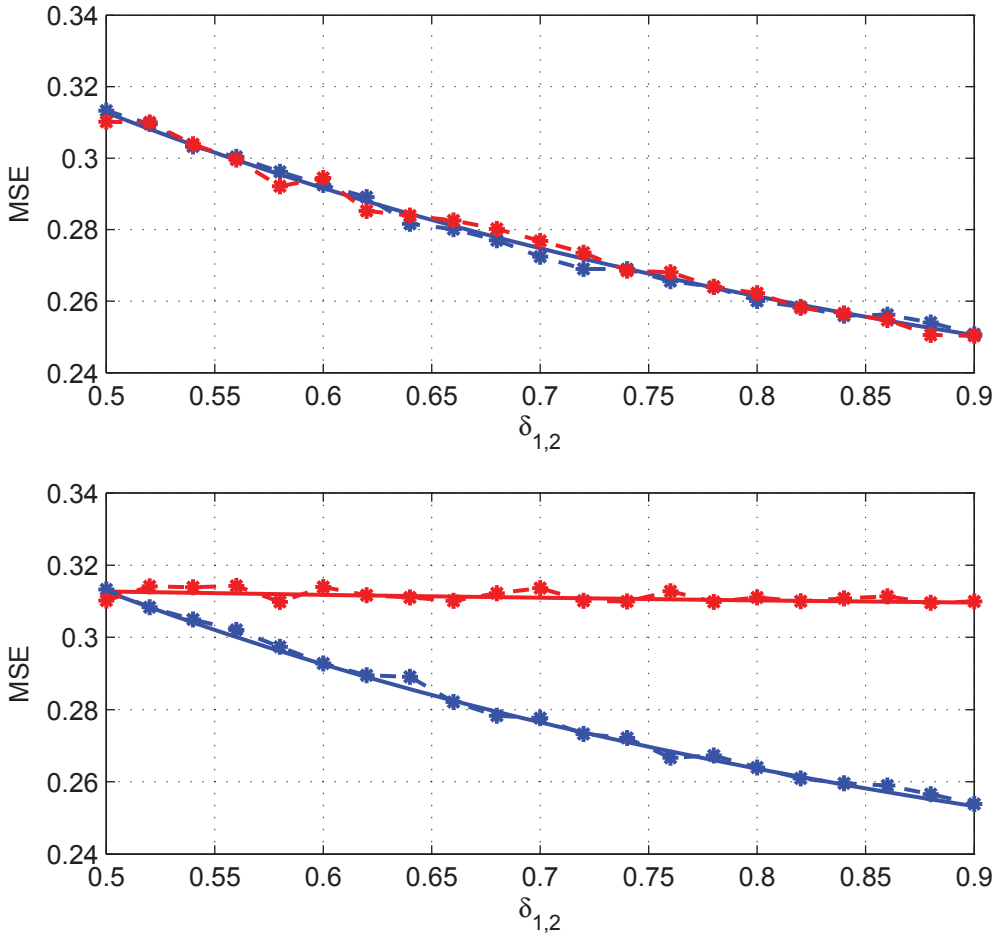

Figure 5.12: Noise sensitivity test, theoretical results in solid lines and empirical results in dashed lines. $\epsilon = 0.5$, $\nu^2 = 0$. Up:$\delta_1 = \delta_2$, down: $\delta_2 = \epsilon$.

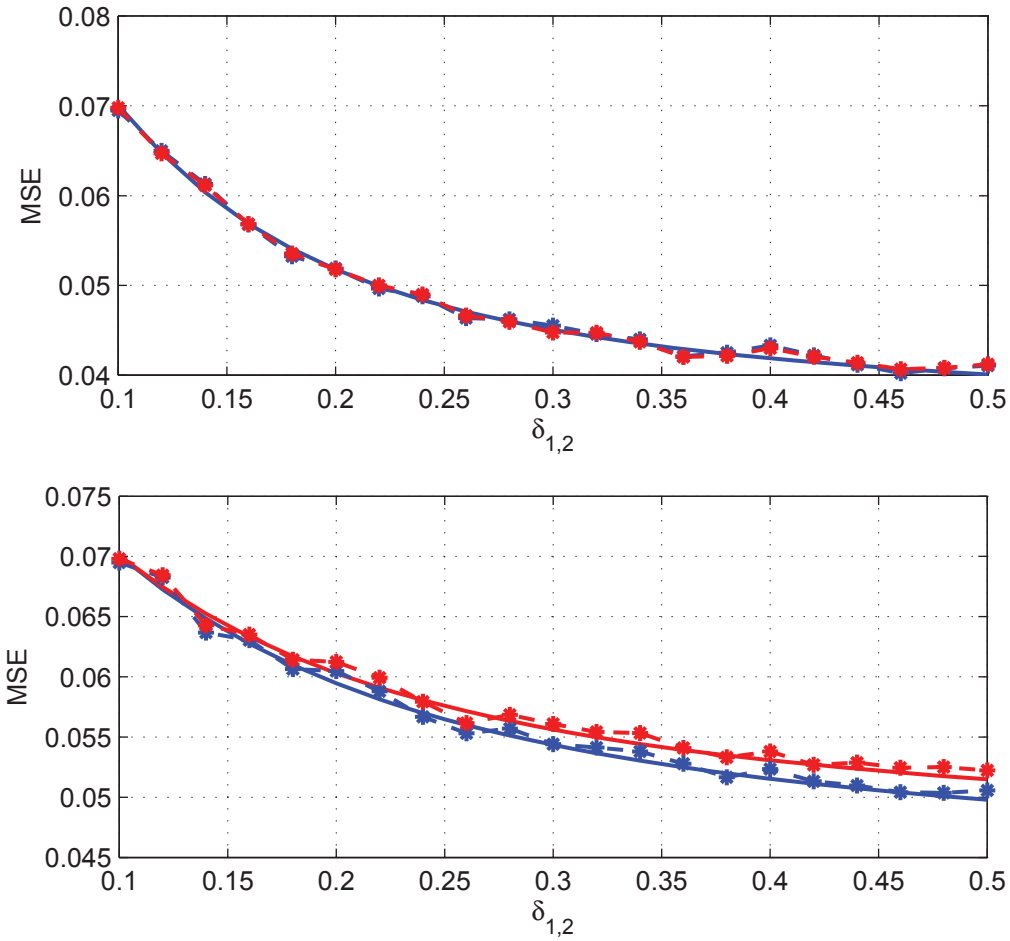

Figure 5.13: Noise sensitivity test, theoretical results in solid lines and empirical results in dashed lines. $\epsilon = 0.1$, $\nu^2 = 0.9$. Up:$\delta_1 = \delta_2$, down: $\delta_2 = \epsilon$.

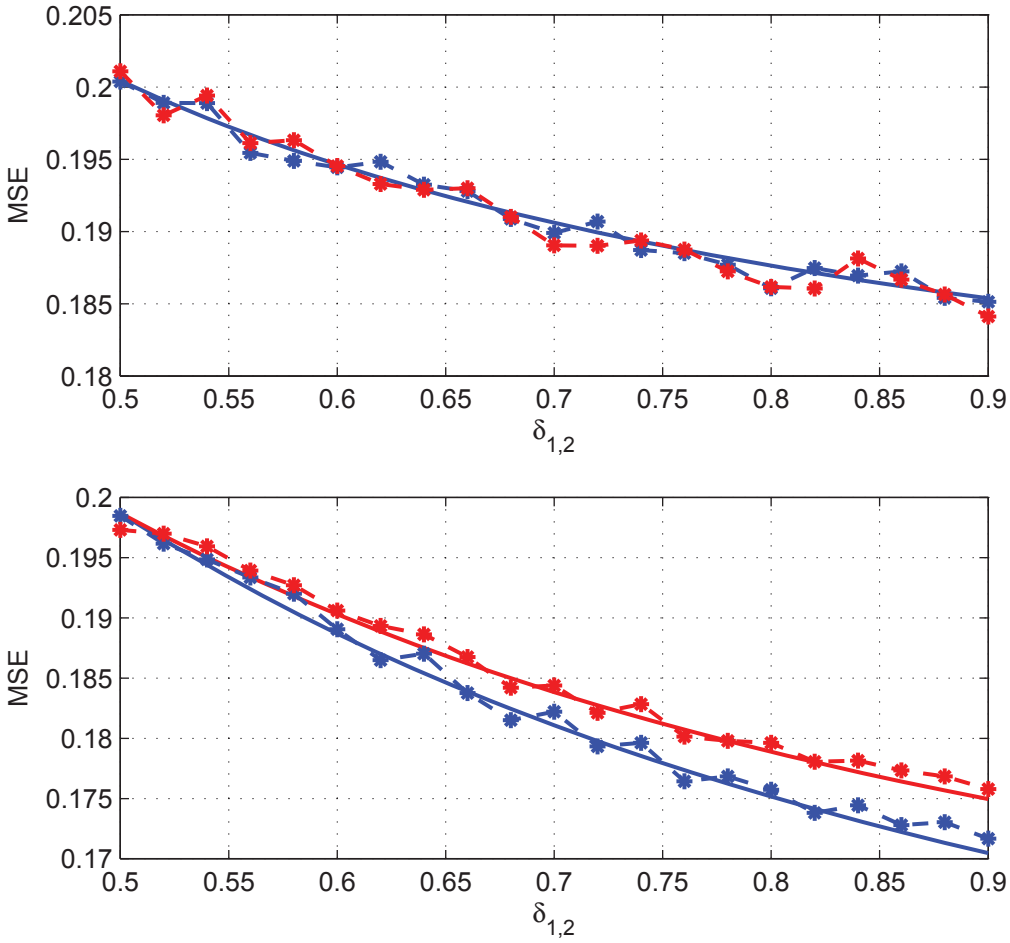

Figure 5.14: Noise sensitivity test, theoretical results in solid lines and empirical results in dashed lines. $\epsilon = 0.5$, $\nu^2 = 0.9$. Up:$\delta_1 = \delta_2$, down: $\delta_2 = \epsilon$.

δ_1	δ_2	fMSE_1	fMSE_2	eMSE_1	eMSE_2
0.051	0.051	0.093	0.093	0.092	0.093
0.078	0.055	0.087	0.091	0.087	0.091
0.105	0.055	0.078	0.091	0.079	0.091
0.149	0.053	0.061	0.088	0.062	0.089
0.192	0.051	0.035	0.084	0.037	0.084
0.074	0.074	0.087	0.087	0.087	0.087
0.102	0.070	0.079	0.087	0.079	0.087
0.148	0.066	0.061	0.085	0.062	0.085
0.192	0.062	**0.034**	0.081	**0.037**	0.081
0.097	0.097	0.079	0.079	0.079	0.080
0.146	0.093	0.060	0.077	0.061	0.077
0.194	0.086	0.029	0.071	0.031	0.070

Table 5.1: Comparison between the MSEs from formula (fMSE) and empirical results (eMSE) of two-group sparse signals. Each group has length 1000 and the empirical results are averaged from 1000 Monte Carlo realizations. The sparsity rate is chosen as $\epsilon = 0.1$. The undersampling rates are given in column δ_1 and δ_2.

In both Figure 5.11 and 5.12, we keep $\delta_1 = \delta_2$ in the upper figure only change δ_1 while keep $\delta_2 = \epsilon$ in the lower figure. A short conclusion on this experiment is, the eMSEs and the fMSEs match well. Larger measurement on one signal is more supportive for denosing the other. Further we take inter-correlation between signal non-zero components, i.e., set $\boldsymbol{\Sigma} \neq \boldsymbol{I}$ in model 5.13 and repeat the tests. The results also match well as we predicted.

In [30] one considered the worst distributed signal for the phase transition bound predictions, thus when there is not enough measurement the reconstruction MSE eventually blows up to infinity. However in this chapter we consider Bernoulli-Gaussian distributed signals. By state evolution the reconstruction MSE is always finite and predictable. We choose sparse signal with two blocks, each block has length 1000 and choose sparsity rate $\epsilon = 0.1,\ 0.5$, and then compare the empirical MSE (eMSE) and the theoretical MSE (fMSE) from state evolution by changing the undersampling rates (see Table 5.1 and Table 5.2). A short conclusion on this experiment would be, the eMSEs and the fMSEs match well. We also provide a three block signal comparison between the fMSEs and eMSEs (see Table 5.3 and Table 5.4). For each block we increase the length to 2000 and keep the sparsity rate $\epsilon = 0.1$ and 0.5 respectively. One of our observations is, as there are more signal blocks joining in the model, in order to get the reconstruction MSE close enough to the theoretical prediction it requires longer length for each signal block.

δ_1	δ_2	fMSE_1	fMSE_2	eMSE_1	eMSE_2
0.130	0.130	0.433	0.433	0.433	0.432
0.264	0.130	0.358	0.432	0.358	0.431
0.383	0.129	0.285	0.430	0.284	0.431
0.502	0.128	0.205	0.428	0.206	0.428
0.600	0.127	**0.134**	0.424	**0.140**	0.424
0.244	0.244	0.368	0.368	0.369	0.368
0.376	0.241	0.288	0.367	0.288	0.366
0.496	0.238	0.207	0.363	0.208	0.364
0.604	0.237	0.124	0.356	0.124	0.356
0.359	0.359	0.295	0.295	0.295	0.295
0.513	0.353	0.190	0.290	0.192	0.291
0.630	0.343	0.095	0.282	0.099	0.283

Table 5.2: Comparison between the MSEs from formula (fMSE) and empirical results (eMSE) of two-group sparse signals. Each group has length 1000 and the empirical results are averaged from 1000 Monte Carlo realizations. The sparsity rate is chosen as $\epsilon = 0.5$. The undersampling rates are given in column δ_1 and δ_2.

δ_1	δ_2	δ_3	fMSE_1	fMSE_2	fMSE_3	eMSE_1	eMSE_2	eMSE_3
0.041	0.041	0.041	0.094	0.094	0.094	0.094	0.095	0.095
0.068	0.045	0.045	0.087	0.092	0.092	0.089	0.093	0.093
0.095	0.045	0.045	0.079	0.091	0.091	0.081	0.092	0.092
0.139	0.043	0.043	0.061	0.090	0.090	0.064	0.091	0.091
0.064	0.064	0.064	0.087	0.087	0.087	0.089	0.089	0.089
0.092	0.060	0.060	0.078	0.087	0.087	0.081	0.089	0.089
0.138	0.056	0.056	0.059	0.086	0.086	0.064	0.088	0.088
0.087	0.087	0.087	0.076	0.076	0.076	0.081	0.081	0.081
0.136	0.083	0.083	**0.054**	0.074	0.074	**0.062**	0.078	0.078

Table 5.3: Comparison between the MSEs from formula (fMSE) and empirical results (eMSE) of three-group sparse signals. Each group has length 2000 and the empirical results are averaged from 1000 Monte Carlo realizations. The sparsity rate is chosen as $\epsilon = 0.1$. The undersampling rates are given in column δ_1, δ_2 and δ_3, where $\delta_2 = \delta_3$.

δ_1	δ_2	δ_3	fMSE_1	fMSE_2	fMSE_3	eMSE_1	eMSE_2	eMSE_3
0.100	0.100	0.100	0.457	0.457	0.457	0.449	0.448	0.449
0.234	0.100	0.100	0.370	0.446	0.446	0.375	0.449	0.447
0.353	0.099	0.099	0.296	0.445	0.445	0.304	0.447	0.447
0.472	0.098	0.098	0.216	0.443	0.443	0.225	0.446	0.445
0.570	0.097	0.097	0.143	0.441	0.441	0.154	0.443	0.443
0.214	0.214	0.214	0.374	0.374	0.374	0.383	0.385	0.384
0.346	0.211	0.211	0.289	0.373	0.373	0.305	0.383	0.383
0.466	0.208	0.208	0.205	0.371	0.371	0.226	0.381	0.381
0.574	0.207	0.207	**0.121**	0.365	0.365	**0.146**	0.376	0.375

Table 5.4: Comparison between the MSEs from formula (fMSE) and empirical results (eMSE) of three-group sparse signals. Each group has length 2000 and the empirical results are averaged from 1000 Monte Carlo realizations. The sparsity rate is chosen as $\epsilon = 0.5$. The undersampling rates are given in column δ_1, δ_2 and δ_3, where $\delta_2 = \delta_3$.

Before finishing this chapter we provide two example of applying the DCS algorithm to image processing. The aim is to confirm the advantage of uneven sampling allocation and its usage in practice. In the first example, we present two 128×128 pixel images with geometric patterns in them. The patterns located at the same coordinate positions but have different gray values. Since both images are in gray scale, the background shown in black has value zero and therefore the two images together make up a two-block sparse signal with common support. We do not append pre-transformation for the moment and compress the two images via model (5.2). We choose two pairs of sampling rate (δ_1, δ_2): 1) $\delta_1 = \delta_2 = 0.55$ and 2) $\delta_1 = 0.65$, $\delta_2 = 0.45$. Note that the sparsity of both images is $\epsilon = 0.43$, thus the second pair of sampling rate is selected very closed to the "corner point", which we deem to be the optimal sampling strategy. Although the pixels contain values which do not satisfy the Bernoulli Gaussian distribution, we stick to using the Bernoulli Gaussian MMSE estimator for our AMP algorithm. It may worsen the final results but do not affect what we tend to explain. Implement the joint AMP for the two pictures, we obtain the results listed in the second and the fourth column in Figure 5.15. We can easily observe a huge difference on the reconstruction quality. The sampling allocation at "corner point" almost perfectly reconstructs the images while the equal allocation does not. We also did experiment by individually reconstruct each of the two images. The results show that even the one with $\delta_1 = 0.65$ is not able to obtain a good reconstruction. This speaks for the effect of joint reconstruction algorithm and the sampling allocation strategy.

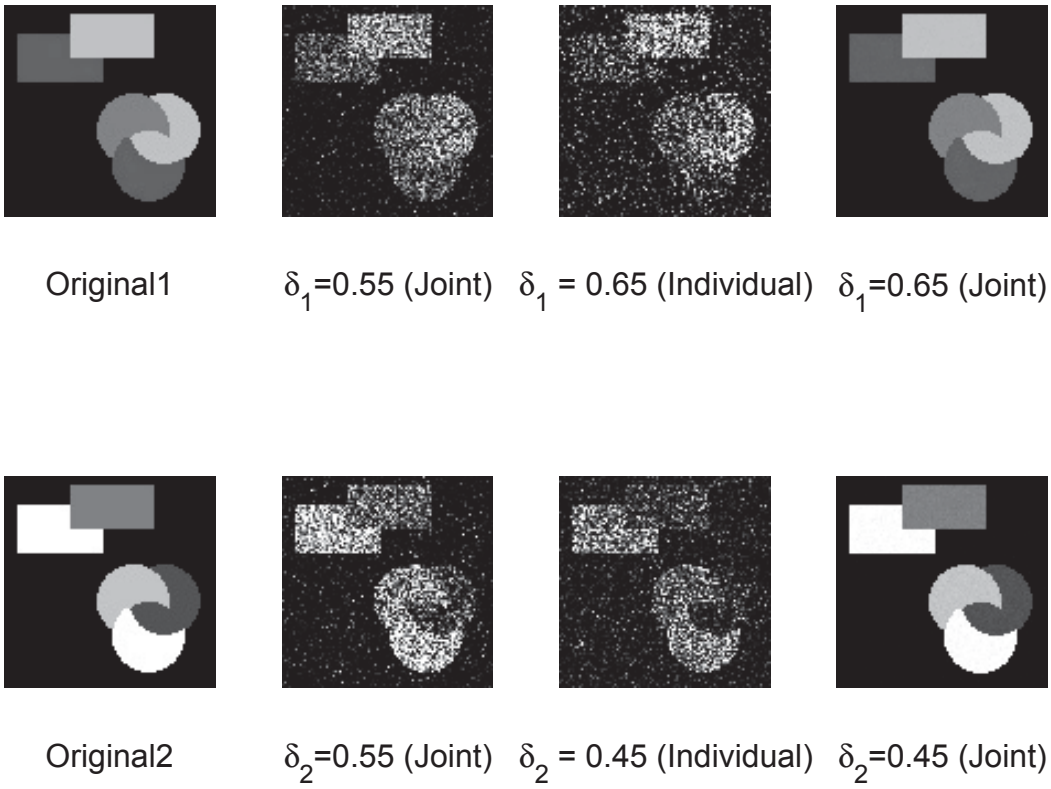

Figure 5.15: Joint reconstruction and individual reconstruction of two images with the same sparse support under a given basis ($\epsilon = 0.43$). Column 1: original images; column 2-4: reconstructed images from samples with given sampling rate δ_1 and δ_2.

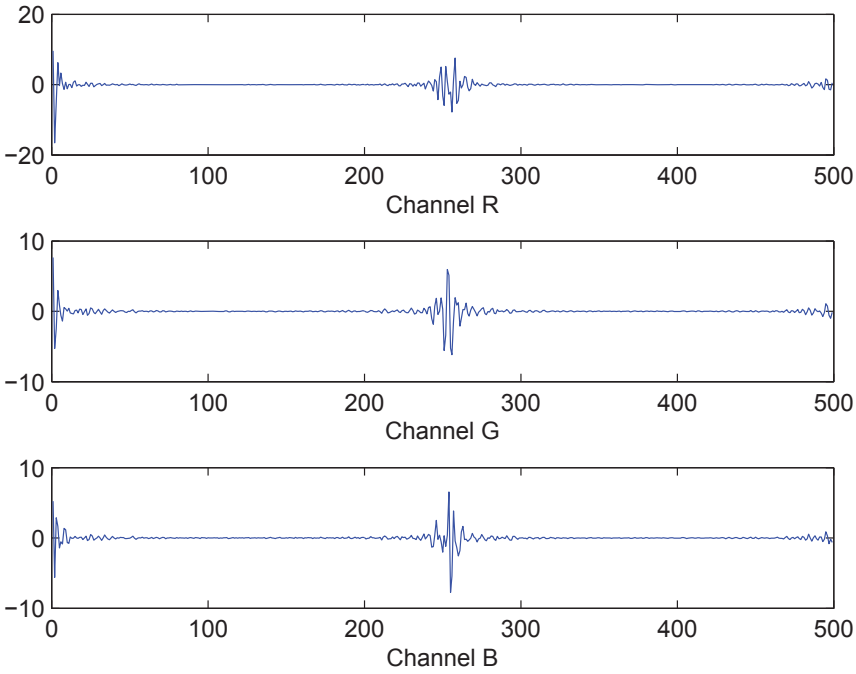

Figure 5.16: The first frequency component of the original bird picture in Red ,Green and Blue channel, respectively

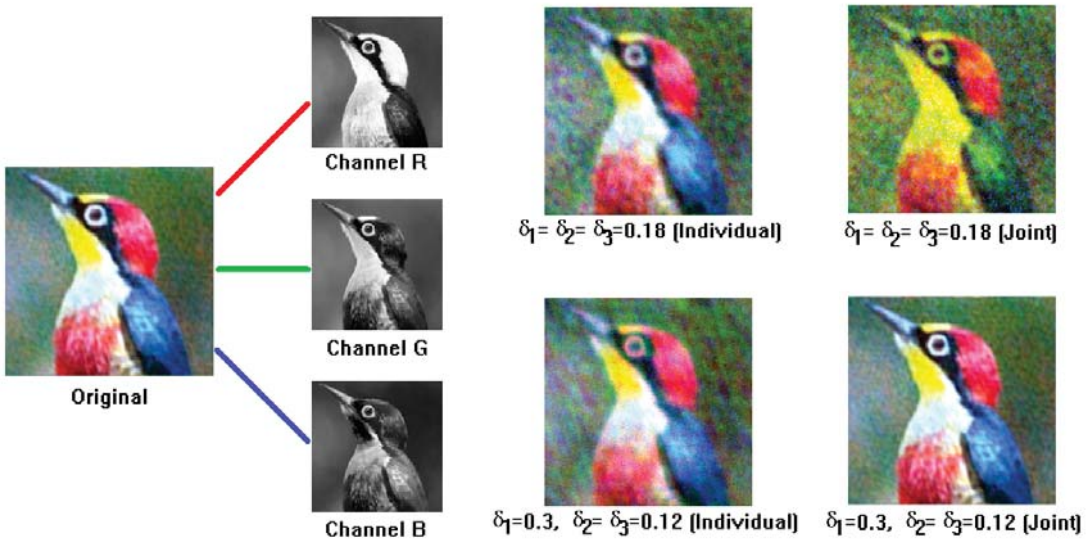

Figure 5.17: RGB image reconstruction (assuming $\epsilon = 0.1$ for each channel). This figure compares the equal resource allocation for each channel and the near optimal allocation (near "corner point").

In the second example we test RGB images. It is known that an RGB image can be decomposed into three gray scale image, representing for the red, green and blue channel, respectively. We pass the three channels via DCT linear transformation and obtain three sparse signals in frequency domain (see fig. 5.16). The three signals actually have very similar amplitude allocations which can be considered as they share the same sparse pattern. Therefore we compress them and use the DCS signal model then apply the joint AMP algorithm. The results show that by fixing the total sampling budget, equal allocation strategy observes a blurred image with a strong color aberration. The "corner point" strategy obtains a much better result. By comparing the joint and individual reconstruction algorithm, we found that individual reconstruction results in more blurry reconstruction as expected. This result is quite meaningful since it points out a possible future CMOS design for digital cameras, as it suggests arrays with uneven proportion of color sensors.

Chapter 6

Conclusions

In this final section we summarise the whole thesis and point some potential problem for future study.

In chapter 2 we have discussed the singularity issue occurring in dictionary update problem as an inherent property. We argued that it is the main issue leading to the failure of dictionary update algorithms. We proposed a smoothing technique under the SimCO framework to address the singularity problem and applied Newton CG method for the new algorithm implementation. Numerical test on both the singularity of mainstream algorithms and the improved performance demonstration of Smoothed SimCO algorithm are presented at the end of the chapter.

In chapter 3, we briefly introduced a development of the blind source separation algorithms based on dictionary learning. In particular, we focus on the SparseBSS algorithm and the optimization procedures. The singularity issue which might lead to the failure of these algorithms. At the same time there are still some open questions to be addressed.

In dictionary learning, it remains open how to find an optimum choice of the redundancy factor $\tau = d/n$ of the over-complete dictionary. A higher redundancy factor leads to either more sparse representation or more precise reconstruction. Moreover, one has to consider the computational capabilities when implementing the algorithms. From this point of view, it is better to keep the redundancy factor low. In the simulation, we have used 64 by 256 dictionary, which gives the redundancy factor $\tau = 256/64 = 4$. This choice is empirical: the sparse representation results are good and the computational cost is limited. A rigorous analysis on the selection of τ is still missing.

The relation between the parameters λ, ϵ and noise standard deviation σ is also worth investigating. As presented in the first experiment on blind audio separation, the relation between λ and σ is discussed when the error bound ϵ is fixed in the sparse coding stage. One can roughly estimate the value of the parameter λ assuming the noise level is known a priori. Similar investigation is undertaken in [1], where the authors claim that when $\lambda \approx \sigma/30$, the algorithm achieved similar reconstruction performance under various σ's. From another perspective, the error bound ϵ is proportional to the noise standard deviation. It turns out that once a well approximated relation between ϵ and σ is obtained, one may get more precise estimation of parameter λ, rather than keeping ϵ fixed. This analysis, therefore, is counted as another open question.

In chapter 4 we consider the power allocation problem for non-uniformly sparse signals. We first show in the presence of noise, i.i.d. Gaussian random measurement matrix may not be optimal in minimizing the reconstruction MSE. Then we considered how to allocate a given total power across the columns of the measurement matrix. Given a power allocation, we derived the AMP.P($\boldsymbol{\epsilon}$) algorithm, and quantitatively analyzed the corresponding minimax MSE. Based on it, the optimal power allocation policy has been identified. Both theoretical and empirical results are presented with the clear consistency and verified the performance gain. Further we change the objective function and show that our power allocation scheme also finds an optimal strategy. This change shows the generality and the potential wide usage of our contribution.

In chapter 5 we studied the rate allocation problem for DCS scenario. Given the sparse signal distribution, we worked on finding block sparse measurement matrices where joint approximate message passing can give asymptotically optimal reconstruction. It is suitable for both equal and unequal numbers of measurements for different signal blocks. We considered Bernoulli-Gaussian distribution as the signal prior and analysed the theoretical phase transition curves of the reconstruction algorithm in the asymptotic regime by using the state evolution technique. The results show that, by fixing a total sensing resource, we can enjoy a benefit by assigning unequal measurements to each signal block. We proposed propositions to discuss how to find exact reconstructions by using as few measurements as possible. We also gave reconstruction error estimations when independent additive Gaussian noise is present in the signal model. Finally we introduced inter-correlations among

non-zeros part of the signal blocks. This is to study how the numbers of measurement required for each signal block are affected by the inter-correlation.

The work in chapter 5 has a good potential value and brings interesting prospective problems that have not been reached. We currently assume that measurement matrices are independent to each other. A more complicated case would be assuming all the signal blocks sharing the same measurement matrix, i.e., the multiple measurement vectors (MMV) problem. It is worth mentioning that the extension from the DCS case to the MMV case is not tricky since strong correlation is introduced between measurement matrices. One may also generalise the signal prior to arbitrary distribution to study the phase transition behavior of the problem. Furthermore in our work we assume all the parameters are known in advance. However in real applications these parameters will probably be tuned from more advanced techniques. Therefore a combination of the joint AMP and the EM algorithm could be a useful research direction.

Appendix A

Proofs of Propositions in Chapter 2

A.1 Proof of Proposition 2.2

1. Represent $f(\boldsymbol{D}) = \sum_i f_i(\boldsymbol{D}_i)$. To prove $f(\boldsymbol{D})$ is continuous when $\boldsymbol{D}$ is non-singular, it suffices to show that for all i, $f_i(\boldsymbol{D}_i)$ is continuous.

When $\boldsymbol{D}$ is not singular, $\boldsymbol{D}_i$ is of full column rank (i.e., $\text{rank}(\boldsymbol{D}_i) = |\Omega(:,i)|$) and therefore $\left(\boldsymbol{D}_i^T \boldsymbol{D}_i\right)^{-1}$ exists. Referring to (2.6),

$$f_i(\boldsymbol{D}_i) = \left\| \boldsymbol{Y}_{:,i} - \boldsymbol{D}_i \left(\boldsymbol{D}_i^T \boldsymbol{D}_i\right)^{-1} \boldsymbol{D}_i^T \boldsymbol{Y}_{:,i} \right\|_2^2.$$

Note that $\boldsymbol{D}_i$ and $\boldsymbol{D}_i^T$ are continuous in $\boldsymbol{D}_i$, and additions, multiplications and combinations of continuous functions are continuous. To show $f_i(\boldsymbol{D}_i)$ is continuous, it is sufficient to prove that the inverse operator $\boldsymbol{A}^{-1}$ is continuous in $\boldsymbol{A}$ when $\boldsymbol{A}$ is of full rank.

Now we shall show the continuity of the inverse operator. Consider the case that we only change the $(i,j)^{th}$ element of $\boldsymbol{A}$ by δ. The resulting matrix can be written as $\boldsymbol{A} + \delta \boldsymbol{e}_i \boldsymbol{e}_j^T$. $\boldsymbol{e}_i \in \mathbb{R}^d$ is a column vector where its i^{th} element is one and all the other elements are zero.

The difference between $\boldsymbol{A}^{-1}$ and $\left(\boldsymbol{A} + \delta \boldsymbol{e}_i \boldsymbol{e}_j^T\right)^{-1}$ can be quantified by using Woodbury

identity [45]:

$$\begin{aligned}
&\boldsymbol{A}^{-1} - \left(\boldsymbol{A} + \delta \boldsymbol{e}_i \boldsymbol{e}_j^T\right)^{-1} \\
=&\boldsymbol{A}^{-1} - \boldsymbol{A}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{e}_i \left(\delta^{-1} + \boldsymbol{e}_j^T \boldsymbol{A}^{-1} \boldsymbol{e}_i\right)^{-1} \boldsymbol{e}_j^T \boldsymbol{A}^{-1} \\
=&\delta \boldsymbol{A}^{-1}\boldsymbol{e}_i \boldsymbol{e}_j^T \boldsymbol{A}^{-1} / \left(1 + \delta \boldsymbol{e}_j^T \boldsymbol{A}^{-1} \boldsymbol{e}_i\right),
\end{aligned}$$

which can be made arbitrary small when $\delta \to 0$. This proves the continuity of the inverse operator $\boldsymbol{A}^{-1}$, hence the first part of the proposition.

2. Suppose that $\exists\, i$ such that $\operatorname{rank}(\boldsymbol{D}_i) < |\Omega(:, i)|$ and $\boldsymbol{Y}_{:,i} \notin \operatorname{span}(\boldsymbol{D}_i)$. Fix such i, we first show that for this particular i, $f_i(\boldsymbol{D}_i)$ is not continuous.

That the vector $\boldsymbol{Y}_{:,i} \notin \operatorname{span}(\boldsymbol{D}_i)$ implies $f_i(\boldsymbol{D}_i) > 0$. We shall construct $\boldsymbol{D}_i(\epsilon)$ such that $\boldsymbol{D}_i(\epsilon) \to \boldsymbol{D}_i$ as $\epsilon \to 0$, but $f_i(\boldsymbol{D}_i(\epsilon)) = 0$. Since $\boldsymbol{D}_i$ is not of column full rank, the singular value decomposition of $\boldsymbol{D}_i$ can be written as $\boldsymbol{D}_i = \sum_{j=1}^{r} \lambda_j \boldsymbol{u}_j \boldsymbol{v}_j^T$, where $r < |\Omega(:, i)|$. Denote $\boldsymbol{y}_r = \left(\boldsymbol{I} - \boldsymbol{D}_i \boldsymbol{D}_i^{\dagger}\right) \boldsymbol{Y}_{:,i}$. Since $\boldsymbol{Y}_{:,i} \notin \operatorname{span}(\boldsymbol{D}_i(\epsilon))$, $\boldsymbol{y}_r \neq \boldsymbol{0}$. Let $\bar{\boldsymbol{y}}_r = \boldsymbol{y}_r / \|\boldsymbol{y}_r\|_2$. We construct $\boldsymbol{D}_i(\epsilon) = \boldsymbol{D}_i + \epsilon \bar{\boldsymbol{y}}_r \boldsymbol{v}_{\perp}^T$, where the vector $\boldsymbol{v}_{\perp}$ is of unit ℓ_2-norm and orthogonal to all $\boldsymbol{v}_j$s, $j \in [r]$. It is clear that $\boldsymbol{Y}_{:,i} \in \operatorname{span}(\boldsymbol{D}_i(\epsilon))$ when $\epsilon \neq 0$. Then we have

$$\begin{cases} f_i(\boldsymbol{D}_i(\epsilon)) = 0, & \text{when } \epsilon \neq 0 \\ f_i(\boldsymbol{D}_i(\epsilon)) > 0, & \text{when } \epsilon = 0 \end{cases},$$

and $\boldsymbol{D}_i(\epsilon) \to \boldsymbol{D}_i$. This implies the discontinuity of the atomic function $f_i(\boldsymbol{D}_i)$.

Now come back to the overall function

$$f(\boldsymbol{D}) = f_i(\boldsymbol{D}_i) + \sum_{j \neq i} f_j(\boldsymbol{D}_j).$$

It is clear that

$$\operatorname{span}(\boldsymbol{D}_j(0)) \subseteq \lim_{\epsilon \to 0} \operatorname{span}(\boldsymbol{D}_j(\epsilon)).$$

Then we have $f_j(\boldsymbol{D}_j(\epsilon)) \leq f_j(\boldsymbol{D}_j(0))$ by the definition of projection.

$$\begin{aligned}\lim_{\epsilon\to 0} f(\boldsymbol{D}(\epsilon)) &= \lim_{\epsilon\to 0} f_i(\boldsymbol{D}_i(\epsilon)) + \sum_{j\neq i} \lim_{\epsilon\to 0} f_j(\boldsymbol{D}_j(\epsilon)) \\ &\leq \lim_{\epsilon\to 0} f_i(\boldsymbol{D}_i(\epsilon)) + \sum_{j\neq i} f_j(\boldsymbol{D}_j(0)) \\ &< f_i(\boldsymbol{D}_i(0)) + \sum_{j\neq i} f_j(\boldsymbol{D}_j(0)) \\ &= f(\boldsymbol{D}(0)).\end{aligned}$$

The discontinuity of $f(\boldsymbol{D})$ is hence proved.

A.2 Analysis of the Example in Section 2.3

Analysis of the MOD algorithm in Section 2.3.1

This appendix details the update formula (2.11) of variable ϵ_k from ϵ_{k-1} under the MOD optimization framework.

First we fix the dictionary $\boldsymbol{D}(\epsilon_{k-1})$ (2.9) and find the optimal coefficient matrix $\boldsymbol{X}(\epsilon_{k-1})$. Recall the form of $f(\boldsymbol{D})$ in (2.5). The first two columns of $\boldsymbol{X}(\epsilon_{k-1})$ depend on $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$, and therefore remain unchanged during the update. Hence $\boldsymbol{X}(\epsilon_{k-1})$ can be written as

$$\boldsymbol{X}(\epsilon_{k-1}) = \begin{bmatrix} 1 & 0 & 0 & x_1 \\ 0 & 1 & 0 & x_1 \\ 0 & 0 & x_2 & x_3 \end{bmatrix}, \tag{A.1}$$

where x_1, x_2, $x_3$need to be calculated by solving the least squares problem involved in $f(\boldsymbol{D})$. Via long but straightforward algebra, the above least squares problem canbe solved:

$$x_1 = -\frac{\sqrt{1-\epsilon_{k-1}^2}}{\epsilon_{k-1}},\ x_2 = 1.4\sqrt{\frac{1-\epsilon_{k-1}^2}{2}} - 0.2\frac{\epsilon_{k-1}}{\sqrt{2}},\ x_3 = \frac{\sqrt{2}}{\epsilon_{k-1}}. \tag{A.2}$$

Now we fix the updated $\boldsymbol{X}(\epsilon_{k-1})$ and update dictionary $\boldsymbol{D}(\epsilon)$. The MOD optimization

framework updates the dictionary via

$$\epsilon_k = \arg \min_{\epsilon \in [-1\ 1]} \|\boldsymbol{Y} - \boldsymbol{D}(\epsilon)\, \boldsymbol{X}(\epsilon_{k-1})\|_F^2.$$

As it is again a least squares problem, a long but straightforward computations show that

$$\epsilon_k = \frac{0.1x_2 - x_3}{x_1 x_3 - 0.7x_2} \tag{A.3}$$

Substitute $(A.2)$ into $(A.3)$. Finally we reach the formula in (2.11).

Analysis of the K-SVD algorithm in Section 2.3.2

This appendix details the update of the dictionary $\boldsymbol{D}$ under the K-SVD optimization framework.

We firstly compute $\boldsymbol{Y}^r = \boldsymbol{Y} - \sum_{i=1}^{2} \boldsymbol{d}_i \boldsymbol{x}_i^T$. Note that in this example $\boldsymbol{Y}_{:,1}$ and $\boldsymbol{Y}_{:,2}$ depend only on the fixed and optimized $\left(\boldsymbol{d}_1,\ \boldsymbol{x}_1^T\right),\ \left(\boldsymbol{d}_2,\ \boldsymbol{x}_2^T\right)$. To update the dictionary (more precisely, to update $\boldsymbol{d}_3$), we can focus on the last two columns of $\boldsymbol{Y}^r$, denoted by

$$\boldsymbol{M} = \begin{bmatrix} 0.7 & \frac{\sqrt{1-\epsilon^2}}{\epsilon} \\ 0.7 & \frac{\sqrt{1-\epsilon^2}}{\epsilon} \\ -0.1 & 1 \\ -0.1 & 1 \end{bmatrix}. \tag{A.4}$$

Then SVD is used to update $\boldsymbol{d}_3$, that is, we solve

$$\min_{\boldsymbol{d}_3 \in \mathcal{D},\, \boldsymbol{x}_3} \left\|\boldsymbol{M} - \boldsymbol{d}_3 \boldsymbol{x}_3^T\right\|_F^2. \tag{A.5}$$

The vector $\boldsymbol{d}_3$ will be the eigenvector corresponding to the largest eigenvalue of $\boldsymbol{M}\boldsymbol{M}^T$ (denoted by $\lambda_{\max}^2$):

$$\left(\boldsymbol{M}\boldsymbol{M}^T\right) \boldsymbol{d}_3 = \lambda_{\max}^2 \boldsymbol{d}_3. \tag{A.6}$$

In general, it is difficult to get the closed form results for $(A.6)$. However, the $\boldsymbol{M}$ matrix in $(A.6)$ has a particular structure, i.e., the 2nd and 4th rows are repetition of the 1st and 3rd rows respectively. The eigen-decomposition can be obtained in a closed form.

In particular, let $\tilde{\boldsymbol{M}} = [\boldsymbol{M}_{1,:}; \boldsymbol{M}_{3,:}]$. Consider the eigen-decomposition

$$\left(\tilde{\boldsymbol{M}}\tilde{\boldsymbol{M}}^T\right)\tilde{\boldsymbol{d}} = \tilde{\lambda}^2_{\max}\tilde{\boldsymbol{d}}. \tag{A.7}$$

Then it can be verified that $\lambda^2_{\max} = 2\tilde{\lambda}^2_{\max}$. Vector

$$\boldsymbol{d}_3 = [d_1,\ d_1,\ d_2,\ d_2]/\sqrt{2}, \tag{A.8}$$

where d_1 and d_2 are the two entries of vector $\tilde{\boldsymbol{d}}$.

In particular, $\tilde{\boldsymbol{M}}\tilde{\boldsymbol{M}}^T = [a,\ c;\ c,\ b]$ where $a = 0.49 + \frac{1-\epsilon^2}{\epsilon^2}$, $b = 1.01$ and $c = -0.07 + \frac{\sqrt{1-\epsilon^2}}{\epsilon}$. It can be verified that for any matrix of the form $[a,\ c;\ c,\ b]$, its maximum eigenvalue is given by

$$\tilde{\lambda}^2_{\max} = \frac{(a+b) + (a-b)\sqrt{1 + 4\frac{c^2}{(a-b)^2}}}{2}$$

and the corresponding eigenvactor $\tilde{\boldsymbol{d}} = [d_1,\ d_2]^T$ satisfies

$$\frac{d_2}{d_1} = \frac{\tilde{\lambda}_{max} - a}{c} \text{ and } d_1^2 + d_2^2 = 1. \tag{A.9}$$

Substitute the values of $a,\ b,\ c$ into the formula $(A.8)$ and $(A.9)$. Using the Taylor approximation, it can be verified that

$$\frac{\sqrt{1-\epsilon_k^2}}{\epsilon_k} = \frac{a-b}{c} + \frac{c}{a-b} - \frac{c^3}{(a-b)^3} + o\left(\frac{c^3}{(a-b)^3}\right). \tag{A.10}$$

Substitute $a\left(\epsilon_{k-1}\right)$, $b = 1.01$ and $c\left(\epsilon_{k-1}\right)$ denoted above into $(A.10)$. After straightforward computations we obtain

$$\epsilon_k = \epsilon_{k-1}\left(1 - 0.07\epsilon_{k-1} - 0.48\epsilon_{k-1}^2 + o\left(\epsilon_{k-1}^2\right)\right).$$

Analysis of Regularized SimCO in Section 2.3.3

This appendix shows that Regularized SimCO may fail in this explicit example and converge to a singular point. More specifically, we will first prove this proposition for a fixed parameter μ by showing a sufficient property in the objective function on which there ex-

ists a local maximizer. Then we will explain the reason of failure even under the cooling schedule of parameter μ.

In Regularized SimCO, parameter $\mu > 0$. The solution of the least square problem (2.12) is

$$f_\mu(\epsilon) = 3 \underbrace{-2(0.7\sqrt{1-\epsilon^2} - 0.1\epsilon)^2}_{f_1(\epsilon)} \underbrace{- \frac{2\epsilon^2}{\mu + 1 - \frac{1-\epsilon^2}{\mu+1}}}_{f_2(\epsilon)}.$$

We look for the derivatives of $f_\mu(\epsilon)$. Compute the derivative of $f_1(\epsilon)$ and $f_2(\epsilon)$,

$$f_1'(\epsilon) = 4\left(0.7\sqrt{1-\epsilon^2} - 0.1\epsilon\right)\left(0.1 + 0.7\frac{\epsilon}{\sqrt{1-\epsilon^2}}\right),$$

$$f_2'(\epsilon) = -4(1+\mu)\left((1+\mu)^2 - 1\right)\epsilon / \left((1+\mu)^2 - 1 + \epsilon^2\right)^2.$$

The following statements about f_1' and f_2' can be verified.

- $f_1'(0) = 0.28$, $f_2'(0) = 0$ and $f_2'(\epsilon) < 0$ for $\epsilon \in (0, 1)$.
- By studying the second order derivative $f_1''(\epsilon)$, it can be shown that $f_1'(\epsilon)$ monotonically increases as ϵ increases in $\left[0, 0.1\sqrt{2}\right]$. A uniform upper bound on $f_1'(\epsilon)$ for $\epsilon \in \left[0, 0.1\sqrt{2}\right]$ can be obtained by $f_1'(\epsilon) \le f_1'\left(0.1\sqrt{2}\right) = 0.384\sqrt{2}$.
- By studying the second order derivative $f_2''(\epsilon)$, it can be shown that $f_2'(\epsilon)$ monotonically decreases as ϵ increases in $\left[0, \sqrt{\left((1+\mu)^2 - 1\right)/3}\right]$.
- For any given $\mu < \sqrt{2} - 1$, let $\epsilon_\mu = \sqrt{(1+\mu)^2 - 1}/10$. Then

$$f_2'(\epsilon_\mu) = -0.4\frac{100^2}{101^2}\frac{1+\mu}{\sqrt{(1+\mu)^2 - 1}} < -0.384\sqrt{2}.$$

Combine the above results. Note that $f_\mu'(\epsilon) = f_1'(\epsilon) + f_2'(\epsilon)$. One has

$$f_\mu'(0) > 0, \text{ and } f_\mu'(\epsilon_\mu) < 0.$$

As a result, there must exists a *local maximizer* $\epsilon_{\max} \in (0, \epsilon_\mu)$ such that $f_\mu'(\epsilon_{\max}) = 0$.

For any given initial $\epsilon_0 \in \left(0, 0.1\sqrt{2}\right)$ and any given $0 < \mu < \min\left(\sqrt{1+100\epsilon_0^2} - 1, \sqrt{2} - 1\right)$, it is clear that

$$\epsilon_\mu = \sqrt{(1+\mu)^2 - 1}/10 < \epsilon_0.$$

From previous discussion, we show there exists a local maximizer $\epsilon_{\max}(\mu) \in (0, \epsilon_\mu)$ on $f_\mu(\epsilon)$.

Now we take the cooling schedule into consideration. Via further studying on $f'_\mu(\epsilon)$, it can be shown that $\epsilon_{\max}(\mu)$ decreases as μ decreases. As a result, right after each time μ decreases, the new local maximizer is always in front through the path to the global minimum. With $\mu \to 0$, the problem gradually degenerate to Primitive SimCO. Hence the update will fail and finally converge to the singular point $\boldsymbol{D}(0)$.

A.3 Proof of Theorem 2.3

Theorem 2.3.1: First we show the continuity of $\tilde{f}_i(\boldsymbol{D}_i)$. Consider the limit

$$\begin{aligned} &\lim_{\boldsymbol{D}_\epsilon \to \boldsymbol{D}_i} \tilde{f}_i(\boldsymbol{D}_i) - \tilde{f}_i(\boldsymbol{D}_\epsilon) \\ =&\lim_{\boldsymbol{D}_\epsilon \to \boldsymbol{D}_i} \left(f_i(\boldsymbol{D}_i) - f_i(\boldsymbol{D}_\epsilon)\right) g_i(\boldsymbol{D}_i) - f_i(\boldsymbol{D}_\epsilon)\left(g_i(\boldsymbol{D}_\epsilon) - g_i(\boldsymbol{D}_i)\right). \end{aligned} \tag{A.11}$$

- When $\boldsymbol{D}_i$ is a singular, $g_i(\boldsymbol{D}_i) = 0$ and $\lim_{\boldsymbol{D}_\epsilon \to \boldsymbol{D}_i} g_i(\boldsymbol{D}_i) - g_i(\boldsymbol{D}_\epsilon) = 0$.
- Otherwise, $\lim_{\boldsymbol{D}_\epsilon \to \boldsymbol{D}_i} f_i(\boldsymbol{D}_i) - f_i(\boldsymbol{D}_\epsilon) = 0$ and $\lim_{\boldsymbol{D}_\epsilon \to \boldsymbol{D}_i} g_i(\boldsymbol{D}_i) - g_i(\boldsymbol{D}_\epsilon) = 0$.

Therefore, the limit $(A.11)$ is always equal to zero, i.e., $\tilde{f}_i(\boldsymbol{D}_i)$ is continuous everywhere. The continuity of $\tilde{f}_i(\boldsymbol{D}_i)$ implies that $\mathcal{D}_{\tilde{f}_i}$ is a closed set, i.e., $\overline{\mathcal{D}_{\tilde{f}_i}} = \mathcal{D}_{\tilde{f}_i}$.

Theorem 2.3.3: $\overline{\mathcal{D}_f} = \mathcal{D}_{\tilde{f}} \Leftrightarrow \left(\overline{\mathcal{D}_f} \subseteq \mathcal{D}_{\tilde{f}}\right) \cap \left(\mathcal{D}_{\tilde{f}} \supseteq \overline{\mathcal{D}_f}\right)$. Note that $\tilde{f}(\boldsymbol{D})$ can be written as a summation of finite many atomic functions (2.16). The statements in the theorem hold if the statement hold for each atomic function $\tilde{f}_i(\boldsymbol{D}_i)$. We separate the proof into two parts.

Part I: We prove $\mathcal{D}_{\tilde{f}_i} \supseteq \overline{\mathcal{D}_{f_i}}$. Note that $\mathcal{D}_{\tilde{f}_i} = \{f_i \cdot g_i \le a\} \supset \{f_i \le a\} \cap \{g_i \le 1\} = \{f_i \le a\} = \mathcal{D}_{f_i}$. And $\mathcal{D}_{\tilde{f}_i}$ is a closed set. Therefore we have

$$\mathcal{D}_{\tilde{f}_i} \supset \mathcal{D}_{f_i} \Rightarrow \mathcal{D}_{\tilde{f}_i} = \overline{\mathcal{D}_{\tilde{f}_i}} \supseteq \overline{\mathcal{D}_{f_i}}.$$

Part II: We prove $\mathcal{D}_{\tilde{f}_i} \subseteq \overline{\mathcal{D}_{f_i}}$. From part I, we know that $\mathcal{D}_{\tilde{f}_i} \supset \mathcal{D}_{f_i}$. Here to prove $\mathcal{D}_{\tilde{f}_i} \subseteq \overline{\mathcal{D}_{f_i}}$ is equivalent to prove $\mathcal{D}_{\tilde{f}_i} \backslash \mathcal{D}_{f_i} \subseteq \overline{\mathcal{D}_{f_i}} \backslash \mathcal{D}_{f_i}$. More specifically, we prove for $\forall \boldsymbol{D}_i \in \mathcal{D}_{\tilde{f}_i} \backslash \mathcal{D}_{f_i}$, it also holds $\boldsymbol{D}_i \in \overline{\mathcal{D}_{f_i}} \backslash \mathcal{D}_{f_i}$.

Refer to *Theorem A.3.2.* When $\delta_1 = \delta_2 \to 0$, for $\forall \boldsymbol{D}_i \in \mathcal{D}_{\tilde{f}_i} \backslash \mathcal{D}_{f_i}$, $\boldsymbol{D}_i$ is column rank deficient. W.L.O.G, we may assume that $\boldsymbol{D}_i = [\boldsymbol{d}_1, ..., \boldsymbol{d}_l, \boldsymbol{d}_{l+1}, ..., \boldsymbol{d}_n]$ where $\boldsymbol{d}_{l+1}, ..., \boldsymbol{d}_n$ is a maximal linear independent group of $\boldsymbol{D}_i$, and $\forall \boldsymbol{d}_j$, $j = 1, ..., l$, $\boldsymbol{d}_j = \sum_{p=l+1}^{n} c_{j,p} \boldsymbol{d}_p$. Then the proof is equivalent to prove that there exist a sequence $\boldsymbol{D}_i(\epsilon_k) \in \mathcal{D}_{f_i}$, where

$$\left\{ \boldsymbol{D}_i(\epsilon_k) \triangleq \boldsymbol{D}_i + \boldsymbol{\Delta D}_i \cdot \epsilon_k | \; k = 1, 2, ... \text{ and } \epsilon_1 > \epsilon_2 > ... > 0 \right\}$$

such that $\lim_{k \to \infty} \boldsymbol{D}_i(\epsilon_k) \to \boldsymbol{D}_i$[1].

Proof. Generate $\boldsymbol{\Delta D}_i = \left[\underbrace{\boldsymbol{y}, ..., \boldsymbol{y}}_{l}, \boldsymbol{d}_{l+1}, ..., \boldsymbol{d}_n \right]$.

$$\boldsymbol{D}_i(\epsilon_k) = [\boldsymbol{d}_1 + \epsilon_k \boldsymbol{y}, ..., \boldsymbol{d}_l + \epsilon_k \boldsymbol{y}, (1+\epsilon_k)\boldsymbol{d}_{l+1}, ..., (1+\epsilon_k)\boldsymbol{d}_n].$$

Because $\boldsymbol{D}_i(\epsilon_k)$ is column rank deficient, it always hold that

$$f_i(\boldsymbol{D}_i(\epsilon_k)) = \min_{\boldsymbol{x}(\epsilon_k)} \|\boldsymbol{y} - \boldsymbol{D}_i(\epsilon_k)\boldsymbol{x}(\epsilon_k)\|_2^2 = 0 \leq a,$$

where

$$\boldsymbol{x}(\epsilon_k) = \frac{1}{l \cdot \epsilon_k} \left[1, ..., 1, -\frac{1}{(1+\epsilon_k)} \sum_{j=1}^{l} c_{j,l+1}, ..., -\frac{1}{(1+\epsilon_k)} \sum_{j=1}^{l} c_{j,n} \right]^{\mathrm{T}}.$$

Therefore we proved $\boldsymbol{D}_i(\epsilon_k) \in \mathcal{D}_{f_i}$. Plus the fact that $\lim_{k \to \infty} \boldsymbol{D}_i(\epsilon_k) \to \boldsymbol{D}_i$, we proved $\boldsymbol{D}_i \in \overline{\mathcal{D}_{f_i}} \backslash \mathcal{D}_{f_i}$, i.e., the proof of Theorem A.3.3 is completed. □

[1] To simplify the proof we do not normalize each column of $\boldsymbol{D}_i(\epsilon_k)$. The proof is also established if $\boldsymbol{D}_i(\epsilon_k)$ is column normalized.

A.4 Derivation of $\nabla_{\boldsymbol{\eta}}(\nabla\lambda_r)$ in Section 2.5

Proposition A.1. *(Orthonormal Basis) Let $\boldsymbol{E}^{ij} \in \mathbb{R}^{m\times n}$ be the matrix of which the entry at row-k and column-l is given by*

$$\left(\boldsymbol{E}^{ij}\right)_{kl} = \begin{cases} 1 & \text{if } k = i, \text{ and } l = j, \\ 0 & \text{otherwise.} \end{cases}$$

Then $\left\{\boldsymbol{E}^{ij} : 1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n\right\}$ is an orthonormal basis for $\mathbb{R}^{m\times n}$.

Let $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ be the left and right singular vectors of $\boldsymbol{A}$ respectively, where both $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal. Then

$$\left\{\boldsymbol{B}^{ij} = \boldsymbol{U}_{:,i}\boldsymbol{V}_{:,j}^T : 1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n\right\}$$

forms another orthonormal basis for $\mathbb{R}^{m\times n}$.

Proposition A.2. *Let f be a mapping defined as $f : \mathbb{R}^{m\times n} \to \mathbb{R}$, $\boldsymbol{X} \to f(\boldsymbol{X})$. Suppose that f is smooth in a neighborhood of $\boldsymbol{X}$. Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$. Then the directional derivative*

$$\nabla_{\boldsymbol{A}} f(\boldsymbol{X}) = \lim_{t\to 0} \frac{f(\boldsymbol{X} + \boldsymbol{A}t) - f(\boldsymbol{X})}{t}.$$

Write $\boldsymbol{A}$ as a linear combination: $\boldsymbol{A} = \sum_i c_i \boldsymbol{B}^i$. For $\forall i$, $\boldsymbol{B}^i \in \mathbb{R}^{m\times n}$. Then

$$\nabla_{\boldsymbol{A}} f(\boldsymbol{X}) = \sum_i c_i \nabla_{\boldsymbol{B}^i} f(\boldsymbol{X}).$$

Proof. By the definition of the gradient, $f(\boldsymbol{X} + \boldsymbol{A}t) = f(\boldsymbol{X}) + \langle \nabla f(\boldsymbol{X}), \boldsymbol{A}\rangle \cdot t + o(t^2)$ when $|t|$ is sufficiently small. Consider the directional derivative, we have

$$\begin{aligned} \nabla_{\boldsymbol{A}} f(\boldsymbol{X}) &= \lim_{t\to 0} \frac{f(\boldsymbol{X} + \boldsymbol{A}t) - f(\boldsymbol{X})}{t} \\ &= \lim_{t\to 0} \langle \nabla f(\boldsymbol{X}), \boldsymbol{A}\rangle + o(t) \\ &= \lim_{t\to 0} \sum_i c_i \frac{f(\boldsymbol{X} + \boldsymbol{B}^i t) - f(\boldsymbol{X})}{t} \\ &= \sum_i c_i \nabla_{\boldsymbol{B}^i} f(\boldsymbol{X}). \end{aligned}$$

□

Consider the singular value decomposition (SVD) of the matrix $\boldsymbol{D}_i$. It can be written as $\boldsymbol{D}_i = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T$, where $\boldsymbol{U} \in \mathbb{R}^{m\times m}$, $\boldsymbol{V} \in \mathbb{R}^{r\times r}$, contain the left and right singular vectors, respectively, and $\boldsymbol{\Lambda} \in \mathbb{R}^{m\times r}$ where the diagonal elements are sorted singular values. Denote $\boldsymbol{\eta} \in \mathbb{R}^{m\times r}$ The derivative

$$\boldsymbol{U}^T \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{D}_i\right) \boldsymbol{V} = \boldsymbol{U}^T \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{U}\right) \boldsymbol{\Lambda} + \nabla_{\boldsymbol{\eta}} \boldsymbol{\Lambda} + \boldsymbol{\Lambda} \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{V}^T\right) \boldsymbol{V}. \tag{A.12}$$

This equation, as well as the following properties, are the key to compute the first and second order derivatives of λ_r.

- The matrix $\boldsymbol{U}^T \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{U}\right)$ is skew-symmetric, i.e. $\boldsymbol{U}^T \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{U}\right) = -\boldsymbol{U}^T \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{U}\right)^T$. This can be verified by differentiating both sides of $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$. Similarly, the matrix $\left(\nabla_{\boldsymbol{\eta}} \boldsymbol{V}^T\right) \boldsymbol{V}$ is also skew-symmetric. A consequence of skew-symmetry is that the diagonal entries are zero.

- The matrix $\nabla_{\boldsymbol{\eta}} \boldsymbol{\Lambda}$ is diagonal.

- Recall the definitions of $\boldsymbol{E}^{ij}$ and $\boldsymbol{B}^{ij}$. One can simplify $\boldsymbol{U}^T \left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{D}_i\right) \boldsymbol{V} = \boldsymbol{E}^{ij}$. Refer to $(A.12)$,

$$\boldsymbol{E}^{ij} = \boldsymbol{U}^T \left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}\right) \boldsymbol{\Lambda} + \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{\Lambda} + \boldsymbol{\Lambda} \left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}^T\right) \boldsymbol{V}. \tag{A.13}$$

By the skew-symmetry of $\boldsymbol{U}^T \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{U}\right)$ and $\left(\nabla_{\boldsymbol{\eta}} \boldsymbol{V}^T\right) \boldsymbol{V}$, one has the diagonal elements of which are zeros. The r^{th} diagonal element of $(A.13)$ is

$$\left(\boldsymbol{E}^{ij}\right)_{rr} = \nabla_{\boldsymbol{B}^{ij}} \lambda_r.$$

Referring to *Proposition 1*,

$$\begin{aligned} \nabla \lambda_r &= \sum_{i,j} \boldsymbol{B}^{ij} \cdot \nabla_{\boldsymbol{B}^{ij}} \lambda_r = \sum_{i,j} \boldsymbol{B}^{ij} \cdot \left(\boldsymbol{E}^{ij}\right)_{rr} \\ &= \boldsymbol{B}^{rr} = \boldsymbol{U}_{:,r} \boldsymbol{V}_{:,r}^T. \end{aligned}$$

Next we compute

$$\nabla_{\boldsymbol{\eta}} \left(\nabla \lambda_r\right) = \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{U}_{:,r}\right) \boldsymbol{V}_{:,r}^T + \boldsymbol{U}_{:,r} \left(\nabla_{\boldsymbol{\eta}} \boldsymbol{V}_{:,r}^T\right)$$

for any given $\boldsymbol{\eta}$. From proposition 1 it is sufficient to compute $\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}_{:,r}$ and $\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}_{:,r}^T$. We also notice the skew-symmetry of $\boldsymbol{U}^T \left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}\right)$ and $\left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}^T\right) \boldsymbol{V}$, i.e., for any valid k,

$$\left(\boldsymbol{U}^T \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}\right)_{kr} = -\left(\boldsymbol{U}^T \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}\right)_{rk},$$

$$\left(\left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}^T\right) \boldsymbol{V}\right)_{kr} = -\left(\left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}^T\right) \boldsymbol{V}\right)_{rk}.$$

The off-diagonal elements of $(A.13)$ (i.e. $k \neq r$) give

$$\begin{aligned}
\left(\boldsymbol{E}^{ij}\right)_{rk} &= -\lambda_k \boldsymbol{U}_{:,k}^T \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}_{:,r} + \lambda_r \left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}_{:,r}^T\right) \boldsymbol{V}_{:,k} \text{ if } k < r, \\
\left(\boldsymbol{E}^{ij}\right)_{kr} &= \lambda_r \boldsymbol{U}_{:,k}^T \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}_{:,r} - \lambda_k \left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}_{:,r}^T\right) \boldsymbol{V}_{:,k} \text{ if } k < r, \\
\left(\boldsymbol{E}^{ij}\right)_{kr} &= \lambda_r \boldsymbol{U}_{:,k}^T \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}_{:,r} \text{ if } k > r.
\end{aligned}$$

The above linear equations give that

$$\boldsymbol{U}_{:,k}^T \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}_{:,r} = \begin{cases} \frac{\lambda_r (\boldsymbol{E}^{ij})_{kr} + \lambda_k (\boldsymbol{E}^{ij})_{rk}}{\lambda_r^2 - \lambda_k^2} & \text{if } k < r \\ (\boldsymbol{E}^{ij})_{rk} / \lambda_r & \text{if } k = r \\ 0 & \text{if } k > r \end{cases}$$

and

$$\left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}_{:,r}^T\right) \boldsymbol{V}_{:,k} = \begin{cases} \frac{\lambda_r (\boldsymbol{E}^{ij})_{rk} + \lambda_k (\boldsymbol{E}^{ij})_{kr}}{\lambda_r^2 - \lambda_k^2} & \text{if } k < r \\ 0 & \text{if } k = r \end{cases},$$

where the results for $k = r$ case are directly from the skew-symmetry of $\boldsymbol{U}^T \left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}\right)$ and

$\left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}^T\right) \boldsymbol{V}$. As a result, for any $\boldsymbol{\eta} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T = \sum_{i,j} \boldsymbol{S}_{ij}\boldsymbol{B}^{ij}$, we have

$$\begin{aligned}
\nabla_{\boldsymbol{\eta}} \boldsymbol{U}_{:,r} &= \sum_{i,j} \boldsymbol{S}_{ij} \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}_{:,r} \\
&= \sum_{i,j} \boldsymbol{S}_{ij} \sum_{k} \boldsymbol{U}_{:,k} \boldsymbol{U}_{:,k}^T \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}_{:,r} \\
&= \sum_{k} \boldsymbol{U}_{:,k} \sum_{i,j} \boldsymbol{S}_{ij} \boldsymbol{U}_{:,k}^T \nabla_{\boldsymbol{B}^{ij}} \boldsymbol{U}_{:,r} \\
&= \sum_{k=1}^{r-1} \boldsymbol{U}_{:,k} \frac{\lambda_r \boldsymbol{S}_{kr} + \lambda_k \boldsymbol{S}_{rk}}{\lambda_r^2 - \lambda_k^2} + \sum_{k=r+1}^{m} \boldsymbol{U}_{:,r} \boldsymbol{S}_{kr} / \lambda_r.
\end{aligned}$$

With similar derivation, we have

$$\begin{aligned}
\nabla_{\boldsymbol{\eta}} \boldsymbol{V}_{:,r}^T &= \sum_{i,j} \boldsymbol{S}_{ij} \sum_{k} \left(\nabla_{\boldsymbol{B}^{ij}} \boldsymbol{V}_{:,r}^T\right) \boldsymbol{V}_{:,k} \boldsymbol{V}_{:,k}^T \\
&= \sum_{k=1}^{r-1} \frac{\lambda_r \boldsymbol{S}_{rk} + \lambda_k \boldsymbol{S}_{kr}}{\lambda_r^2 - \lambda_k^2} \boldsymbol{V}_{:,k}^T.
\end{aligned}$$

Appendix B

Proofs of Propositions in Chapter 5

B.1 Derivation of the MSE closed form in Section 5.5

Denote the reconstruction MSE based on the least favorable distribution as

$$M_K(\alpha, \epsilon) \triangleq \mathbb{E}\left\{\|\eta_\alpha(\boldsymbol{y}) - \boldsymbol{x}\|_2^2\right\}.$$

Hereinafter we short $M_K(\alpha, \epsilon)$ to M_k and write it as two parts since they will be calculated separately,

$$M_K = \mathbb{E}\left\{\|\eta_\alpha(\boldsymbol{y}) - \boldsymbol{x}\|_2^2 \,|\, \|\boldsymbol{x}\|_2 \neq 0\right\} + \mathbb{E}\left\{\|\eta_\alpha(\boldsymbol{y}) - \boldsymbol{x}\|_2^2 \,|\, \|\boldsymbol{x}\|_2 = 0\right\}. \tag{B.1}$$

For the part when$\|\boldsymbol{x}\|_2 \neq 0$, because we assume the least favorable distribution in (5.4), we have$\forall i \in [K]$,$|\boldsymbol{x}_i| \to \infty$. Then

$$\begin{aligned}
\mathbb{E}\left\{\|\eta_\alpha(\boldsymbol{y}) - \boldsymbol{x}\|_2^2 \,|\, \|\boldsymbol{x}\|_2 \neq 0\right\} &= \mathbb{E}\left\{\left\|\frac{\|\boldsymbol{y}\| - \alpha}{\|\boldsymbol{y}\|}(\boldsymbol{x} + \boldsymbol{w}) - \boldsymbol{x}\right\|_2^2\right\} \\
&= \mathbb{E}\left\{\left\|\frac{-\alpha}{\|\boldsymbol{y}\|}\boldsymbol{x} + \frac{\|\boldsymbol{y}\| - \alpha}{\|\boldsymbol{y}\|}\boldsymbol{w}\right\|_2^2\right\} \\
&= \mathbb{E}\left\{\|-\alpha \cdot \mathbf{1} + \boldsymbol{w}\|_2^2\right\} \\
&= \alpha^2 + k.
\end{aligned}$$

For the part when $\|\boldsymbol{x}\|_2 = 0$, i.e., signal $\boldsymbol{x}$ are all zero and therefore $\boldsymbol{y} = \boldsymbol{w}$. Then

$$\begin{aligned}\mathbb{E}\left\{\|\eta_\alpha(\boldsymbol{y}) - \boldsymbol{x}\|_2^2 \,|\, \|\boldsymbol{x}\|_2 = 0\right\} &= \mathbb{E}\left\{\left\|\frac{\|\boldsymbol{w}\| - \alpha}{\|\boldsymbol{w}\|}\boldsymbol{w}\right\|_2^2\right\} \\ &= \mathbb{E}\left\{\|\|\boldsymbol{w}\| - \alpha\|_2^2\right\} \\ &= \mathbb{E}\left\{\|\boldsymbol{w}\|^2 - 2\alpha\|\boldsymbol{w}\| + \alpha^2\right\}.\end{aligned}$$

Let $z = \|\boldsymbol{w}\|$, then z^2 is χ^2 distributed with K degree of freedom. The probability density function:

$$f(Z) = \begin{cases} f\left(Z^2\right)\frac{\mathrm{d}z^2}{\mathrm{d}z} = c(k)\cdot\frac{1}{\sqrt{2\pi}}\cdot z^{K-1}\exp\left(-\frac{z^2}{2}\right) & z > \alpha \\ 0 & \text{otherwise}\end{cases}$$

where $c(K) = \sqrt{2\pi}\cdot\left(2^{\frac{K}{2}-1}\cdot\Gamma\left(\frac{K}{2}\right)\right)^{-1}$. $\Gamma(t)$ is the Gamma function having $\Gamma\left(\frac{1}{2}\right) = \sqrt{\pi}$, $\Gamma(1) = 1$ and $\Gamma(t+1) = t\Gamma(t)$. Now consider integral

$$\begin{aligned}\frac{1}{\sqrt{2\pi}}\int_\alpha^\infty \exp\left(-\frac{z^2}{2}\right)z^K\mathrm{d}z &= \frac{1}{\sqrt{2\pi}}\int_\alpha^\infty -z^{K-1}\mathrm{d}\exp\left(-\frac{z^2}{2}\right) \\ &= \frac{1}{\sqrt{2\pi}}\alpha^{K-1}\exp\left(-\frac{\alpha^2}{2}\right) + \frac{1}{\sqrt{2\pi}}\int_\alpha^\infty \exp\left(-\frac{z^2}{2}\right)\mathrm{d}z^{K-1} \\ &= \frac{1}{\sqrt{2\pi}}\alpha^{K-1}\exp\left(-\frac{\alpha^2}{2}\right) + \frac{(k-1)}{\sqrt{2\pi}}\int_\alpha^\infty \exp\left(-\frac{z^2}{2}\right)z^{K-2}\mathrm{d}z.\end{aligned}$$

Define functions $\Psi_\alpha(K) \triangleq \frac{1}{\sqrt{2\pi}}\int_\alpha^\infty \exp\left(-\frac{z^2}{2}\right)z^K\mathrm{d}z$, $\phi(\alpha) \triangleq \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{z^2}{2}\right)$ and $\Phi(\alpha) \triangleq \frac{1}{\sqrt{2\pi}}\int_{-\infty}^\alpha \exp\left(-\frac{z^2}{2}\right)\mathrm{d}z$. We have

$$\Psi_\alpha(K) = \begin{cases}\Phi(-\alpha) & K = 0 \\ \phi(\alpha) & K = 1 \\ \alpha^{K-1}\phi(\alpha) + (K-1)\Psi_\alpha(K-2) & K \geq 2\end{cases}$$

The second term of (B.1) has

$$\begin{aligned}\mathbb{E}\left\{\|\boldsymbol{w}\|^2 - 2\alpha\|\boldsymbol{w}\| + \alpha^2\right\} &= \int_\alpha^\infty f(z)\cdot\left(z^2 - 2\alpha z + \alpha^2\right)\mathrm{d}z \\ &= c(K)\int_\alpha^\infty \exp\left(-\frac{z^2}{2}\right)\left(z^{K+1} - 2\alpha z^K + \alpha^2 z^{K-1}\right)\mathrm{d}z \\ &= c(K)\cdot\left(\Psi_\alpha(K+1) - 2\alpha\Psi_\alpha(K) + \alpha^2\Psi_\alpha(K-1)\right).\end{aligned}$$

Merge the two cases, we obtain the form of the whole denoising MSE (per element of $\boldsymbol{x}$)

$$M_K = \frac{1}{K}\left(\epsilon\left(\alpha^2 + K\right) + (1-\epsilon)\cdot c(K)\cdot\left(\Psi_\alpha(K+1) - 2\alpha\Psi_\alpha(K) + \alpha^2\Psi_\alpha(K-1)\right)\right). \tag{B.2}$$

Note that this least favorable form holds for the scale invariance property, i.e., if now we change additive Gaussian noise $\boldsymbol{w}$ to i.i.d. random vector with $\mathcal{N}^{\sim}\left(0,\ \sigma^2\boldsymbol{I}\right)$, the resulting MSE is scaled by constant σ^2. This property is stated here and will be useful for deriving the worse case state evolution.

The above result gives

$$M_1 = \epsilon\left(\alpha^2+1\right) + (1-\epsilon)\left(2\left(1+\alpha^2\right)\Phi(-\alpha) - 2\alpha\phi(\alpha)\right).$$

This result consists with the one in [74]. Also we have

$$M_2 = \frac{1}{2}\epsilon\left(\alpha^2+2\right) + \frac{\sqrt{2\pi}}{2}(1-\epsilon)\left(2\phi(\alpha) - 2\alpha\Phi(-\alpha)\right),$$

$$M_3 = \frac{1}{3}\epsilon\left(\alpha^2+3\right) + \frac{2}{3}(1-\epsilon)\left(-\alpha\phi(\alpha) + \left(\alpha^2+3\right)\Phi(-\alpha)\right),$$

$$M_4 = \frac{1}{4}\epsilon\left(\alpha^2+4\right) + \frac{\sqrt{\pi}}{4\sqrt{2}}(1-\epsilon)\left(8\phi(\alpha) - 6\alpha\Phi(-\alpha)\right).$$

$$\vdots$$

The closed form of M_K's are functions of ϵ and α. Note that $\forall\epsilon\in(0,1)$, M_K is convex

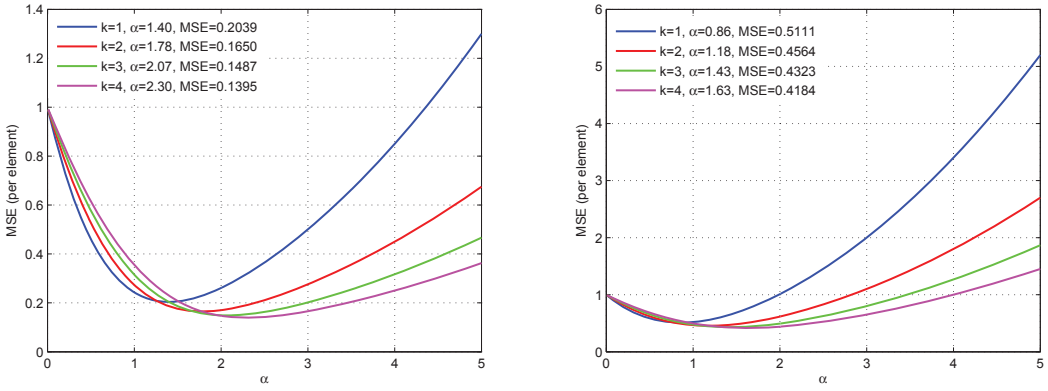

Figure B.1: The convexity of the minimax MSE of DCS AMP algorithm against parameter α. Left: $\epsilon = 0.05$; right: $\epsilon = 0.2$

respect to α. The proof is simply to check

$$\begin{aligned}\frac{\partial M_K}{\partial \alpha} &= \frac{2}{K}\left(\epsilon\alpha + (1-\epsilon)\cdot c\left(K\right)\cdot\left(\Psi_\alpha\left(K\right) - \alpha\Psi_\alpha\left(K-1\right)\right)\right) \\ &= \frac{2}{K}\epsilon\alpha + \frac{2}{K}\left(1-\epsilon\right)c\left(K\right)\frac{1}{\sqrt{2\pi}}\int_\alpha^\infty \exp\left(-\frac{z^2}{2}\right) z^{K-1}\left(z-\alpha\right)\mathrm{d}z,\end{aligned}$$

where on the right hand side the two terms are both monotonic increasing functions and the second is upper bounded. This implies there is only one intersection between function $\frac{\partial M_K}{\partial \alpha}$ and the α axis, which means that M_k is convex on α and admits a unique minimum. Therefore given the sparsity ϵ, we are able to find the optimal threshold α's to achieve the minimum MSE for different K. We denote this minimum MSE as

$$M_K^{\#}\left(\epsilon\right) \triangleq \min_{\alpha} M_K\left(\epsilon,\alpha\right).$$

We choose $\epsilon = 0.05$ and $\epsilon = 0.2$ and draw curves of α against M_K in Figure (B.1). In practice the α to achieve $M_K^{\#}$ can be numerically determined. In Figure (B.3) we print out the MSE curves choosing optimal α's, i.e., $M_K^{\#}$'s against the sparsity ϵ for $K = 1, 2, 3, 4$.

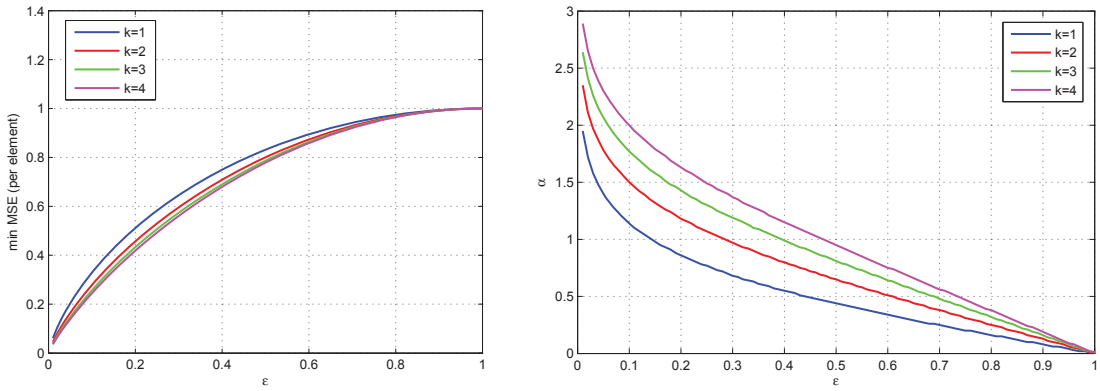

Figure B.2: Left: min MSE-ϵ; right: min $\alpha - \epsilon$

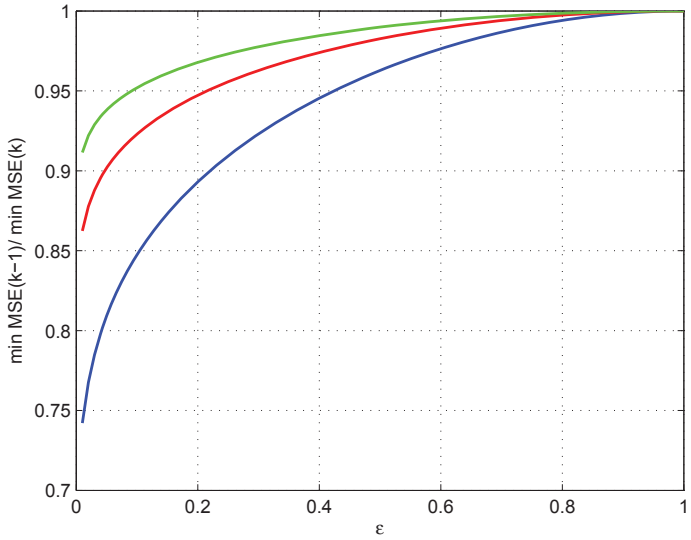

Figure B.3: Plots to show $M_{K-1}^{\#}/M_K^{\#}$ (both $M_{K-1}^{\#}$ and $M_K^{\#}$ choose their optimal α's) against ϵ.

B.2 Proof of Proposition 5.2

Proof. Assume K is even and approaches to an infinite large number (similar derivations can also be done by assuming k is odd). We compute

$$\begin{aligned}\lim_{K\to\infty}\frac{\partial\Psi_\alpha(K)}{\partial\alpha} &= -\alpha^K\phi(\alpha)+(K-1)\left(\alpha^{K-2}\phi(\alpha)+\frac{\partial\Psi_\alpha(K-2)}{\partial\alpha}\right)\\ &= -\alpha^K\phi(\alpha)+(K-1)\left(\alpha^{K-2}\phi(\alpha)+\left(\frac{\partial\alpha^{K-3}\phi(\alpha)+(K-3)\Psi_\alpha(K-4)}{\partial\alpha}\right)\right)\\ &= -\alpha^K\phi(\alpha)+(K-1)!!\cdot\left(\phi(\alpha)+\frac{\partial\Psi_\alpha(0)}{\partial\alpha}\right)\\ &= -\alpha^K\phi(\alpha).\end{aligned}$$

Use this result to compute

$$\begin{aligned}&\frac{\partial M(\rho,\delta)}{\partial\alpha}\\ &=\frac{1}{K}\left\{2\alpha\epsilon+(1-\epsilon)c(K)\left(\frac{\partial\Psi_\alpha(K+1)}{\partial\alpha}-2\alpha\frac{\partial\Psi_\alpha(K)}{\partial\alpha}-2\Psi_\alpha(K)+\alpha^2\frac{\partial\Psi_\alpha(K-1)}{\partial\alpha}+2\alpha\Psi_\alpha(K-1)\right)\right\}\\ &=\frac{1}{K}\left\{2\alpha\epsilon+(1-\epsilon)c(K)\left(-2\Psi_\alpha(K)+2\alpha\Psi_\alpha(K-1)\right)\right\}\\ &=0\end{aligned}$$

Then we have

$$\epsilon=\frac{c(K)\cdot(\Psi_\alpha(K)-\alpha\Psi_\alpha(K-1))}{\alpha+c(K)\cdot(\Psi_\alpha(K)-\alpha\Psi_\alpha(K-1))}. \tag{B.3}$$

We write out the general term formula of $\Psi_\alpha(K)$ and $\Psi_\alpha(K-1)$ using Taylor expansion. Since we assume K is even, then

$$\begin{aligned}\Psi_\alpha(K) &= \left(\alpha^{K-1}+(K-1)\alpha^{K-3}+(K-1)(K-3)\alpha^{K-5}+...+(K-1)!!\cdot\alpha\right)\cdot\phi(\alpha)\\ &\quad+(K-1)!!\cdot\Phi(-\alpha)\\ &= (K-1)!!\cdot\left(\phi(\alpha)\sum_{i=1}^{K/2}\frac{\alpha^{2i-1}}{(2i-1)!!}+\Phi(-\alpha)\right)\\ &\overset{(1)}{=} (K-1)!!\cdot\left(\phi(\alpha)\sqrt{2\pi}\left(\frac{1}{2}-\Phi(-\alpha)\right)\exp\left(\frac{\alpha^2}{2}\right)+\Phi(-\alpha)\right)\\ &= \frac{(K-1)!!}{2},\end{aligned}$$

and

$$\begin{aligned}\Psi_\alpha(K-1) &= \left(\alpha^{K-2} + (K-2)\alpha^{K-4} + (K-2)(K-4)\alpha^{K-6} + ... + (K-2)!!\cdot\alpha\right)\cdot\phi(\alpha) \\ &= (K-2)!!\cdot\phi(\alpha)\sum_{i=1}^{K/2}\frac{\alpha^{2i-2}}{(2i-2)!!} \\ &\overset{(2)}{=} (K-2)!!\cdot\phi(\alpha)\cdot\exp\left(\frac{\alpha^2}{2}\right) \\ &= \frac{(K-2)!!}{\sqrt{2\pi}}.\end{aligned}$$

Here (1) and (2) holds since $\lim_{K\to\infty}\sum_{i=0}^{K}\frac{\alpha^{2i}}{(2i)!!} = \exp\left(\frac{\alpha^2}{2}\right)$ and

$$\lim_{K\to\infty}\sum_{i=0}^{K}\frac{\alpha^{2i+1}}{(2i+1)!!} = \sqrt{2\pi}\left(\frac{1}{2} - \Phi(-\alpha)\right)\exp\left(\frac{\alpha^2}{2}\right).$$

Recall that $c(K) = \sqrt{2\pi}\cdot\left(2^{\frac{K}{2}-1}\cdot\Gamma\left(\frac{K}{2}\right)\right)^{-1} = \frac{\sqrt{2\pi}}{(K-2)!!}$. Substitute $c(K)$, $\Psi_\alpha(K)$ and $\Psi_\alpha(K-1)$ into (B.3), we get

$$\begin{aligned}c(K)\cdot(\Psi_\alpha(K) - \alpha\Psi_\alpha(K-1)) &= \frac{\sqrt{2\pi}}{(K-2)!!}\cdot\left(\frac{(K-1)!!}{2} - \frac{(K-2)!!}{\sqrt{2\pi}}\cdot\alpha\right) \\ &= \sqrt{\frac{\pi}{2}}\cdot\frac{(K-1)!!}{(K-2)!!} - \alpha \\ &\overset{(3)}{=} \sqrt{K-1} - \alpha \\ &\simeq \sqrt{K} - \alpha.\end{aligned}$$

Here (3) holds since $\frac{(K-2)!!}{(K-1)!!} \simeq \sqrt{\frac{\pi}{2(K-1)}}\left\{1 - \frac{1}{4(K-1)} + \frac{1}{32(K-1)^2} - ...\right\}$ for the odd number $(K-1)\to\infty$. Then

$$\epsilon = 1 - \frac{\alpha}{\sqrt{K}}.$$

For the state evolution we have equation $K\delta = M_k(\epsilon, \alpha)$, then

$$\begin{aligned} K\delta &= M_K(\epsilon, \alpha) \\ &= \epsilon\left(\alpha^2 + K\right) + (1-\epsilon)\cdot c(K)\cdot\left(\Psi_\alpha(K+1) - 2\alpha\Psi_\alpha(K) + \alpha^2\Psi_\alpha(K-1)\right) \\ &\overset{(4)}{=} \epsilon\left(\alpha^2 + K\right) + \frac{c(K)}{c(K+1)}\alpha\epsilon - \alpha^2\epsilon \\ &\overset{(5)}{=} \epsilon\left(\alpha^2 + K\right) + \sqrt{K}\alpha\epsilon - \alpha^2\epsilon \\ &= \epsilon\sqrt{K}\left(\alpha + \sqrt{K}\right). \end{aligned}$$

Here (4) and (5) hold since $-2\alpha\epsilon = (1-\epsilon)\cdot c(K)\cdot(-2\Psi_\alpha(K) + 2\alpha\Psi_\alpha(K-1))$ and $\lim_{n\to\infty}\frac{\Gamma(n+a)}{\Gamma(n)} = n^a$. Hence we proved that

$$\delta = \frac{\epsilon}{\rho} = 1 - \frac{\alpha^2}{K}.$$

□

B.3 Derivation of MSE $M_K^{\#}$ in Section 5.2

The expected MSE from (5.12) with the signal prior can be separated into two expectations,

$$\begin{aligned} M_K^{\#} &= \mathbb{E}\left\{\left\|\eta\left(\boldsymbol{X}+\boldsymbol{W};\, \boldsymbol{\Lambda}^{-1}, \alpha\right) - \boldsymbol{X}\right\|_2^2\right\} \\ &= \epsilon\mathbb{E}\left\{\left\|\eta\left(\boldsymbol{X}+\boldsymbol{W},\, \boldsymbol{\Lambda}^{-1}, \alpha\right) - \boldsymbol{X}\right\|_2^2\right\}\delta_{\|\boldsymbol{X}\|_2=0} \qquad \text{(B.4)} \\ &\quad + (1-\epsilon)\,\mathbb{E}\left\{\left\|\eta\left(\boldsymbol{X}+\boldsymbol{W},\, \boldsymbol{\Lambda}^{-1}, \alpha\right) - \boldsymbol{X}\right\|_2^2\right\}\delta_{\|\boldsymbol{X}\|_2=\infty} \qquad \text{(B.5)} \end{aligned}$$

For term (B.4) where $\boldsymbol{X} = 0$, we have $\boldsymbol{Y} = \boldsymbol{W}$. Then

$$\begin{aligned}
&\mathbb{E}\left\{\left\|\eta\left(\boldsymbol{X}+\boldsymbol{W},\, \boldsymbol{\Lambda}^{-1}, \alpha\right) - \boldsymbol{X}\right\|_2^2\right\} \\
=&\mathbb{E}\left\{\left\|\left(\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{W}\right\|_2 - \alpha\right)_+ \cdot \frac{\boldsymbol{W}}{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{W}\right\|_2}\right\|_2^2\right\} \\
=&\sum_{k=1}^{K}\mathbb{E}\left\{\left(\sqrt{\sum_{l=1}^{K}\sigma_l^{-2}W_l^2} - \alpha\right)_+^2 \cdot \frac{\sigma_k^{-2}W_k^2}{\sum_{l=1}^{K}\sigma_l^{-2}W_l^2}\right\}\sigma_k^2 \\
=&\sum_{k=1}^{K}\frac{1}{K}\mathbb{E}\left\{\left(\sqrt{\sum_{l=1}^{K}\sigma_l^{-2}W_l^2} - \alpha\right)_+^2\right\}\sigma_k^2. \qquad \text{(B.6)}
\end{aligned}$$

The last equal sign holds because of the isotropy of the random vector $\boldsymbol{\Lambda}^{-1}\boldsymbol{W}$.

For term (B.5) where as $\|\boldsymbol{X}\|_2 \to \infty$ and $\|\boldsymbol{W}\|_2 / \|\boldsymbol{X}\|_2 \to 0$, we have $\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{Y}\right\|_2 / \left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{X}\right\|_2 \to 1$. Then

$$\begin{aligned}
&\mathbb{E}\left\{\left\|\eta\left(\boldsymbol{X}+\boldsymbol{W},\, \boldsymbol{\Lambda}^{-1}, \alpha\right) - \boldsymbol{X}\right\|_2^2\right\} \\
=&\mathbb{E}\left\{\left\|\frac{-\alpha}{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{X}\right\|_2}\boldsymbol{X} + \frac{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{X}\right\|_2 - \alpha}{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{X}\right\|_2}\boldsymbol{W}\right\|_2^2\right\} \\
=&\mathbb{E}\left\{\sum_{k=1}^{K}\left(\frac{-\alpha X_k}{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{X}\right\|_2} + \frac{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{X}\right\|_2 - \alpha}{\left\|\boldsymbol{\Lambda}^{-1}\boldsymbol{X}\right\|_2}W_k\right)^2\right\} \\
=&\alpha^2\mathbb{E}\left\{\frac{\sum_{k=1}^{K}X_k^2}{\sum_{k=1}^{K}\sigma_k^{-2}X_k^2}\right\} + \sum_{k=1}^{K}\sigma_k^2 \\
=&\frac{K\alpha^2}{\sum_{k=1}^{K}\sigma_k^{-2}} + \sum_{k=1}^{K}\sigma_k^2 \qquad \text{(B.7)}
\end{aligned}$$

The last equal sign holds in (B.7) because $X_k = \delta_{\pm\infty}$ and $\boldsymbol{X}$ is isotropic, therefore $X_k^2 = X_l^2$.

B.4 Derivation of the two special cases in Section 5.5

Proof. of the equivalence between the double integral in the first special case and the multiple integral (5.18) as $\sigma_2^2 = ... = \sigma_k^2$. We focus on integrand x_2, ..., x_K in (5.18),

denote

$$\begin{aligned}\tilde{I}_{1,K-1} &\triangleq \int \frac{c_1 \prod_{l\neq 1} f_G\left(x_l;0,1\right)}{c_2+\prod_{l\neq 1} f_G\left(x_l;0,\frac{1-R_l}{R_l}\right)} d\boldsymbol{x}_{l\backslash 1} \\ &= \int \frac{c_1 \prod_{l\neq 1} \exp\left(-\frac{x_l^2}{2}\right)}{c_2+c_3\prod_{l\neq 1} \exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right)} d\boldsymbol{x}_{l\backslash 1},\end{aligned}$$

where c_1, c_2 are constants and $\boldsymbol{x}_{l\backslash 1}$ means the multiple integral is of $l=2,3,...,K$. Denote $z=\sqrt{\sum_{l=2}^{K} x_l^2}$ and treat $I_{1,K}$ as a function of z, then

$$\tilde{I}_{1,K-1}\mathrm{d}z = \frac{c_1 \exp\left(-\frac{z^2}{2}\right)}{c_2+c_3 \exp\left(-\frac{R_l}{1-R_l}\frac{z^2}{2}\right)} \int_{\nu} \mathrm{d}\boldsymbol{x}_{l\backslash 1}, \tag{B.8}$$

where ν is that elemental shell volume at $z\left(x_2,\ ...,\ x_K\right)$. Term $\int_{\nu}\mathrm{d}\boldsymbol{x}_{l\backslash 1}$ can be seen as the area an $(K-2)$-sphere [54] with radius z which is

$$\int_{\nu}\mathrm{d}\boldsymbol{x}_{l\backslash 1} = \frac{\left(K-1\right)z^{K-2}\pi^{\frac{k-1}{2}}}{\Gamma\left(\frac{K+1}{2}\right)}. \tag{B.9}$$

Substituting (B.8), (B.9) into (5.18) and noticing that $\Gamma(k+1)=k\Gamma(k)$, it yields:

$$\begin{aligned}I_{k,K} &= \int \tilde{I}_{1,K-1}\mathrm{d}z \\ &= \iint \frac{f_G\left(x_1;0,1\right)\frac{2^{-\frac{K-3}{2}}z^{K-2}\exp\left(-\frac{z^2}{2}\right)}{\Gamma\left(\frac{K-1}{2}\right)}x_1^2}{1+\frac{1-\epsilon}{\epsilon\prod_{l\neq 1}\sqrt{R_l}} f_G\left(x_1;0,1\right)\frac{2^{-\frac{K-3}{2}}z^{K-2}\exp\left(-\frac{R_l}{1-R_l}\frac{z^2}{2}\right)}{\left(\frac{1-R_l}{R_l}\right)^{\frac{K-1}{2}}\Gamma\left(\frac{K-1}{2}\right)}}\mathrm{d}x_1\mathrm{d}z.\end{aligned}$$

Use the similar analysis to prove the equivalence between the double integral (5.23) and the multiple integral (5.18) as $\sigma_1^2=...=\sigma_k^2$. Note that here vector $\boldsymbol{x}$ is isotropic, we

can replace term x_l^2 by z^2/K and have

$$\begin{aligned} I_{k,K} &= \int \tilde{I}_{1,K} \mathrm{d}z \\ &= \int \frac{\frac{2^{-\frac{K-2}{2}} z^{K+1} \exp\left(-\frac{z^2}{2}\right)}{K\Gamma\left(\frac{K}{2}\right)}}{1 + \frac{1-\epsilon}{\epsilon \prod_l \sqrt{R_l}} \frac{2^{-\frac{K-2}{2}} z^{K-1} \exp\left(-\frac{R_l}{1-R_l}\frac{z^2}{2}\right)}{\left(\frac{1-R_l}{R_l}\right)^{\frac{K}{2}} \Gamma\left(\frac{K}{2}\right)}} \mathrm{d}z \\ &= \frac{1}{K} \int \frac{f_\chi\left(z; 1, K\right) z^2}{1 + \frac{1-\epsilon}{\epsilon \prod_l \sqrt{R_l}} f_\chi\left(z; \frac{1-R_l}{R_l}, K\right)} \mathrm{d}z \end{aligned} \tag{B.10}$$

$$f_\chi\left(z; \sigma^2, k\right) = \frac{2^{1-\frac{k}{2}} z^{k-1} e^{-\frac{z^2}{2\sigma^2}}}{\sigma^k \Gamma\left(k/2\right)}.$$

Substitute (B.10) into (5.17). With the number of signal blocks increase to $K \to \infty$, we have

$$\lim_{K\to\infty} R_k \cdot I_{k,K} = 0.$$

Therefore,

$$\lim_{K\to\infty} M_{k,K} = \epsilon\left(1 - R_k \cdot I_{k,K}\right) = \epsilon.$$

□

B.5 Proof of Lemma 5.5

Proof. To prove $\frac{\partial M_{k,K}}{\partial R_l} < 0$, we equivalently prove $\frac{\partial I_{k,K}}{\partial \sigma_l^2} < 0$. Note that the partial derivative

$$\begin{aligned} &\frac{\partial}{\partial \sigma_l^2} \frac{1}{\sqrt{1-R_l}} \exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right) \\ =&\exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right)\left(x_l^2 - \frac{1}{1-R_l}\right)\frac{R_l^{\frac{3}{2}}}{2\sigma_l}. \end{aligned}$$

And the order of the integrals and derivative are interchangeable as long as the integrand is a smooth function. Then we have

$$\begin{aligned}
&\frac{\partial I_{k,K}}{\partial \sigma_l^2} \\
=&\int \frac{\partial I_{k,K}}{\partial \sigma_l^2} \int \frac{\prod_i f_G\left(x_i;0,1\right) x_k^2}{1+\frac{1-\epsilon}{\epsilon}\prod_i \frac{1}{\sqrt{1-R_i}} \exp\left(-\frac{R_i}{1-R_i}\frac{x_i^2}{2}\right)} \mathrm{d}x_l \mathrm{d}\boldsymbol{x}_{j\neq l} \\
=&\iint \frac{\partial I_{k,K}}{\partial \sigma_l^2} \frac{\prod_i f_G\left(x_i;0,1\right) x_k^2}{1+\frac{1-\epsilon}{\epsilon}\prod_i \frac{1}{\sqrt{1-R_i}} \exp\left(-\frac{R_i}{1-R_i}\frac{x_i^2}{2}\right)} \mathrm{d}x_l \mathrm{d}\boldsymbol{x}_{j\neq l} \\
=&\iint \frac{-c_1 \cdot \exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right)\left(x_l^2-\frac{1}{1-R_l}\right) R_l^{\frac{3}{2}} f_G\left(x_l;0,1\right)}{2\sigma_l\left(1+\frac{1-\epsilon}{\epsilon}\prod_i \frac{1}{\sqrt{1-R_i}} \exp\left(-\frac{R_i}{1-R_i}\frac{x_i^2}{2}\right)\right)^2} \mathrm{d}x_l \mathrm{d}\boldsymbol{x}_{j\neq l} \\
<&\int \frac{-c_1 \int \exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right)\left(x_l^2-\frac{1}{1-R_l}\right) R_l^{\frac{3}{2}} f_G\left(x_l;0,1\right) \mathrm{d}x_l}{2\sigma_l\left(1+\frac{c_2}{\sqrt{1-R_l}} \exp\left(-\frac{1}{2\sigma_l^2}\right)\right)^2} \mathrm{d}\boldsymbol{x}_{j\neq l} \qquad \text{(B.11)} \\
=&0,
\end{aligned}$$

where c_1, c_2 are functions of $(\boldsymbol{x}_{j\neq l}, \boldsymbol{R}_{j\neq l})$ and are always positive.The inequality (B.11) holds since $\forall x_l \in (-\infty, \infty)$,

$$\exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right) < \exp\left(-\frac{1}{2\sigma_l^2}\right).$$

□

B.6 Derivation of the MMSE estimator in Section 5.5

The joint probability of $\boldsymbol{X}$ and $\boldsymbol{Y}$,

$$\begin{aligned}
p_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{x},\boldsymbol{y}\right) =& p_{\boldsymbol{X}}\left(\boldsymbol{x}\right) \cdot p_{\boldsymbol{W}}\left(\boldsymbol{w}=\boldsymbol{y}-\boldsymbol{x}\right) \\
=& \left[\left(1-\epsilon\right)\delta_{\boldsymbol{x}=\boldsymbol{0}} + \epsilon p_G\left(\boldsymbol{x};\boldsymbol{0},\boldsymbol{\Sigma}_X\right)\right] \cdot f_G\left(\boldsymbol{w}=\boldsymbol{y}-\boldsymbol{x};\boldsymbol{0},\boldsymbol{\Sigma}_W\right) \\
=& \left(1-\epsilon\right)\left|2\pi\boldsymbol{\Sigma}_W\right|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\right)\delta_{\boldsymbol{x}=\boldsymbol{0}} \\
&+\epsilon\left|4\pi^2\boldsymbol{\Sigma}_W\boldsymbol{\Sigma}_X\right|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\boldsymbol{x}^T\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)\boldsymbol{x} + \boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\right).
\end{aligned}$$

Then we have the MMSE estimator,

$$\begin{aligned}
\hat{X}_i &= \mathbb{E}_{\boldsymbol{X}}\left[X_i|\boldsymbol{Y}=\boldsymbol{y}\right] \\
&= \frac{\int x_i \cdot p_{X,Y}\left(\boldsymbol{x},\boldsymbol{y}\right)\mathrm{d}\boldsymbol{x}}{\int p_{X,Y}\left(\boldsymbol{x},\boldsymbol{y}\right)\mathrm{d}\boldsymbol{x}} \\
&= \frac{\int x_i \exp\left(-\frac{1}{2}\boldsymbol{x}^T\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)\boldsymbol{x}+\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\boldsymbol{x}-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\right)\mathrm{d}\boldsymbol{x}}{\frac{1-\epsilon}{\epsilon}\left|2\pi\boldsymbol{\Sigma}_X\right|^{\frac{1}{2}}\exp\left(-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\right)+\int\exp\left(-\frac{1}{2}\boldsymbol{x}^T\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)\boldsymbol{x}+\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\boldsymbol{x}-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\right)\mathrm{d}\boldsymbol{x}} \\
&= \frac{\left(\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)^{-1}\left(\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\right)\right)_i}{\frac{1-\epsilon}{\epsilon}\left|\boldsymbol{\Sigma}_X\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)\right|^{\frac{1}{2}}\exp\left(-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)^{-1}\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\right)+1}.
\end{aligned}$$

The covariance matrix

$$\begin{aligned}
\boldsymbol{\Sigma}_{\hat{X}} &= \mathbb{E}_{\boldsymbol{Y}}\left[\mathbb{E}_{\boldsymbol{X}}\left[\boldsymbol{X}|\boldsymbol{Y}=\boldsymbol{y}\right]\mathbb{E}_{\boldsymbol{X}}\left[\boldsymbol{X}|\boldsymbol{Y}=\boldsymbol{y}\right]^T\right] \\
&= \int \frac{\epsilon^2\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)^{-1}\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)^{-1}f_G^2\left(\boldsymbol{y};\boldsymbol{0},\boldsymbol{\Sigma}_X+\boldsymbol{\Sigma}_W\right)}{\epsilon f_G\left(\boldsymbol{y};\boldsymbol{0},\boldsymbol{\Sigma}_X+\boldsymbol{\Sigma}_W\right)\left(\frac{1-\epsilon}{\epsilon}\left|\boldsymbol{\Sigma}_W\left(\boldsymbol{\Sigma}_X+\boldsymbol{\Sigma}_W\right)^{-1}\right|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)^{-1}\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\right)+1\right)}\mathrm{d}\boldsymbol{y} \\
&= \int \frac{\epsilon\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)^{-1}\boldsymbol{\Sigma}_W^{-1}\boldsymbol{y}\boldsymbol{y}^T\boldsymbol{\Sigma}_W^{-1}\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)^{-1}f_G\left(\boldsymbol{y};\boldsymbol{0},\boldsymbol{\Sigma}_X+\boldsymbol{\Sigma}_W\right)}{\frac{1-\epsilon}{\epsilon}\left|2\pi\left(\boldsymbol{\Sigma}_X+\boldsymbol{\Sigma}_W\right)\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)\boldsymbol{\Sigma}_W\right|^{\frac{1}{2}}f_G\left(\boldsymbol{y};\boldsymbol{0},\boldsymbol{\Sigma}_W\left(\boldsymbol{\Sigma}_X^{-1}+\boldsymbol{\Sigma}_W^{-1}\right)\boldsymbol{\Sigma}_W\right)+1}\mathrm{d}\boldsymbol{y}.
\end{aligned}$$

Directly from the above result, we have

$$\begin{aligned}
\boldsymbol{\Sigma}_Z &= \mathbb{E}_{\boldsymbol{Y}}\left[\left(\mathbb{E}_{\boldsymbol{X}}\left[\boldsymbol{X}|\boldsymbol{Y}=\boldsymbol{y}\right]-\boldsymbol{X}\right)\left(\mathbb{E}_{\boldsymbol{X}}\left[\boldsymbol{X}|\boldsymbol{Y}=\boldsymbol{y}\right]-\boldsymbol{X}\right)^T\right] \\
&= -\boldsymbol{\Sigma}_{\hat{X}}+\epsilon\boldsymbol{\Sigma}_X,
\end{aligned}$$

where the diagonal elements of $\boldsymbol{\Sigma}_Z$ is the MSE associated to the MMSE estimator $\hat{\boldsymbol{X}}$.

B.7 Numerial Accuracy of the integral in Section 5.8

In the theoretical phase transition curves, we need to compute integral 5.18. Due the computer tools at hand does not support double integral on interval $[-\infty,\infty]$, we instead

look for an appropriate interval for the integral. Notice that

$$\begin{aligned} I_{k,K}(\boldsymbol{R},\epsilon) &= \int \frac{\prod_l f_G(x_l;0,1)\, x_k^2}{1+\frac{1-\epsilon}{\epsilon}\prod_l \frac{1}{\sqrt{1-R_l}}\exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right)}\mathrm{d}\boldsymbol{x} \\ &\leq \int \frac{\prod_l f_G(x_l;0,1)\, x_k^2}{\frac{1-\epsilon}{\epsilon}\prod_l \frac{1}{\sqrt{1-R_l}}\exp\left(-\frac{R_l}{1-R_l}\frac{x_l^2}{2}\right)}\mathrm{d}\boldsymbol{x} \\ &= \frac{\epsilon}{1-\epsilon}\prod_l (1-R_l)\int f_G(x_l;0,1-R_l)\, x_k^2 \mathrm{d}\boldsymbol{x} \end{aligned}$$

Use the 6σ rules of the Gaussian distribution. Choose the integral interval as $[-6\sigma_k,\ 6\sigma_k]$, where $\sigma_k^2 = 1-R_k$ is the variance of the Guassian function at hand. The cdf shall be larger than $1-2\times10^{-9}$. Thus we can compute

$$\begin{aligned} &\int f_G(x_l;0,1-R_l)\, x_k^2 \mathrm{d}\boldsymbol{x} \\ >&\int_{-6\sigma}^{6\sigma} f_G(x_k;0,1-R_k)\, x_k^2 \mathrm{d}x_k \cdot \prod_{l\neq k}\int_{-6\sigma}^{6\sigma} f_G(x_l;0,1-R_l)\,\mathrm{d}x_l \\ >&\int_{-6\sigma}^{6\sigma} f_G(x_k;0,1-R_k)\, x_k^2 \mathrm{d}x_k \cdot \left(1-2\times10^{-9}\right)^{K-1} \\ =&\left[1-\frac{12\phi(6)}{2\Phi(6)-1}\right]\cdot\left(1-2\times10^{-9}\right)^{K-1}(1-R_k) \\ =&(1-7.3\times10^{-8})\cdot\left(1-2\times10^{-9}\right)^{K-1}(1-R_k)\,. \end{aligned}$$

Therefore we choose $[-6\sigma_k,\ 6\sigma_k]$ as the integral interval and the calculation tolerance shall be less than $\frac{\epsilon}{1-\epsilon}(1-7.3\times10^{-8})\cdot\left(1-2\times10^{-9}\right)^{K-1}\prod_l(1-R_l)$.

B.8 The information dimension of theoretical phase transition in Section 5.7

Use the Renyi information dimension definition $d(X)=\lim_{m\to\infty}\frac{H(\lfloor mX\rfloor)}{\log m}$, where X is a real-valued random variable, $m\in\mathbb{N}$, floor function $\lfloor\cdot\rfloor$ is taken component-wise. Theorem 1 in [111] shows that when X has a discrete-continuous mixed distribution, e.g., Bernoulli-Gaussian distribution, where $H(\lfloor X\rfloor)<\infty$, then $d(X)=\epsilon$.

Now consider multi-variable $\boldsymbol{X}\in\mathbb{R}^K$ with joint pdf (5.13) and $\boldsymbol{\Sigma}=\boldsymbol{I}$, thus written in

the following

$$p_{\boldsymbol{X}}\left(\boldsymbol{x};\epsilon,\mathbf{0},\boldsymbol{I}\right)=\left(1-\epsilon\right)\delta_{\boldsymbol{x}=\mathbf{0}}+\epsilon f_{G}\left(\boldsymbol{x};\mathbf{0},\boldsymbol{I}\right).$$

Using Theorem 1 in [50], we define matrix $\boldsymbol{B}=\epsilon\boldsymbol{I}+\left(1-\epsilon\right)\mathbf{0}$ which corresponds to our joint pdf and have

$$d\left(\boldsymbol{X}\right)=\mathbb{E}\left(\operatorname{rank}\left(\boldsymbol{B}\right)\right)=K\epsilon.$$

Therefore we have $d\left(X_{k}|\boldsymbol{X}_{l\neq k}\right)=d\left(\boldsymbol{X}\right)-d\left(\boldsymbol{X}_{l\neq k}\right)=\epsilon$.

Bibliography

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

[2] B. Adcock, Purdue Univ, A. Hansen, C. Poon, and B. Roman. Breaking the coherence barrier: asymptotic incoherence and asymptotic sparsity in compressed sensing. Technical report, 2013.

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

[4] R.G. Baraniuk, V. Cevher, M.F. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Trans. on Inf. Theory*, 56(4):1982–2001, 2010.

[5] Dror Baron, Marco F. Duarte, Michael B. Wakin, Shriram Sarvotham, and Richard G. Baraniuk. Distributed compressive sensing. *ArXiv:0901.3403*, 2009.

[6] Mohsen Bayati and Andrea Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *ArXiv:1001.3448*, 2010.

[7] Mohsen Bayati and Andrea Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Trans. Inf. Theory*, 57(2):764–785, 2011.

[8] Mohsen Bayati and Andrea Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Trans. Inf. Theory*, 57(2):764–785, 2011.

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

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

[11] T. Blumensath and M. E. Davies. Compressed sensing and source separation. *Int. Conf. Independent Component Analysis and Signal Separation*, 2007.

[12] T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. *ArXiv:0805.0510*, 2008.

[13] T. Blumensath and M. E. Davies. Sampling theorems for signals from the union of finite-dimensional linear subspaces. *IEEE Transactions on Information Theory*, 55(4):1872–1882, April 2009.

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

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

[16] M. Bronstein, M. Zibulevsky, and Y. Zeevi. Sparse ica for blind separation of transmitted and reflected images. *Intl. J. Imaging Science and Technology*, 15:84–91, 2005.

[17] T. Tony Cai. Adaptive wavelet estimation: A block thresholding and oracle inequality approach. *Ann. Statist*, 27:898–924, 1998.

[18] E. Candes and T. Tao. Decoding by linear programming. *IEEE Trans. Inf. Theory*, 51(12):4203–4215, 2005.

[19] E. J. Candes and D. L. Donoho. Curvelets – a surprisingly effective nonadaptive representation for objects with edges. 2000.

[20] J. F. Cardoso and A. Souloumiac. Blind beamforming for non-Gaussian signals. volume 140, pages 362–370, 1993.

[21] Jie Chen and X. Huo. Theoretical results on sparse representations of multiple-measurement vectors. *IEEE Trans. Signal Process.*, 54(12):4634–4643, Dec 2006.

[22] S. Chen, D. L. Donoho, and A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1998.

[23] Shane F. Cotter, Bhaskar D. Rao, Kjersti Engan, and Kenneth Kreutz-Delgado. Sparse solutions to linear inverse problems with multiple measurement vectors. *IEEE Trans. Signal Process.*, 53(7):2477–2488, 2005.

[24] W. Dai and O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Trans. Inf. Theory*, 55(5):2230–2249, May 2009.

[25] W. Dai, T. Xu, and W. Wang. Simultaneous codeword optimization (SimCO) for dictionary update and learning. *IEEE Trans. Signal Process.*, 60(12):6340–6353, Dec. 2012.

[26] Wei Dai and O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Trans. Inf. Theory*, 55(5):2230–2249, May 2009.

[27] Wei Dai, Mona A. Sheikh, Olgica Milenkovic, and Richard G. Baraniuk. Compressive sensing dna microarrays. *Eurosip J. on Bioinfo. and Systems Biology*, 2009.

[28] Mark A. Davenport and Michael B. Wakin. Analysis of orthogonal matching pursuit using the restricted isometry property. Technical report, 2009.

[29] Thong T. Do, Yi Chen, Dzung T. Nguyen, Nam Nguyen, Lu Gan, and Trac D. Tran. Distributed compressed video sensing. In *ICIP*, pages 1393–1396. IEEE, 2009.

[30] D. L. Donoho, A. Maleki, and A. Montanari. Message-passing algorithms for compressed sensing. *Proc. Nat. Acad. Sci. U.S.A.*, 106(45):18914–18919, 2009.

[31] D. L. Donoho, A. Maleki, and A. Montanari. Message-passing algorithms for compressed sensing. *Proc. Nat. Acad. Sci. U.S.A.*, 106(45):18914–18919, 2009.

[32] David L. Donoho. Compressed sensing. *IEEE Trans. Inf. Theory*, 52:1289–1306, 2006.

[33] David L. Donoho and Iain M. Johnstone. Minimax risk over l p-balls for l q-error, 1994.

[34] David L. Donoho, Arian Maleki, and Andrea Montanari. The noise-sensitivity phase transition in compressed sensing. *IEEE Trans. on Inf. Theory*, 57(10):6920–6941, 2011.

[35] D.L. Donoho, I. Johnstone, and A. Montanari. Accurate prediction of phase transitions in compressed sensing via a connection to minimax denoising. *IEEE Trans. on Inf. Theory*, 59(6):3396–3433, 2013.

[36] D.L. Donoho, I. Johnstone, and A. Montanari. Accurate prediction of phase transitions in compressed sensing via a connection to minimax denoising. *IEEE Trans. Inf. Theory*, 59(6):3396–3433, 2013.

[37] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 20:303–353, 1999.

[38] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

[39] Michael Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer Publishing Company, Incorporated, 1st edition, 2010.

[40] M. Eladl. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer Publishing Company, Incorporated, 1st edition, 2010.

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

[42] K. Engan, K. Skretting, and J. Husoy. Family of iterative ls-based dictionary learning algorithms, ils-dla, for sparse signal representation. *Digital Signal Process.*, 17(1):32–49, 2007.

[43] 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.

[44] Q. Geng, H. Wang, and J. Wright. On the local correctness of l1 minimization for dictionary learning. *ArXiv:1101.5672*, 2011.

[45] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 2nd edition, 1989.

[46] I. F. Gorodnitsky, J. S. George, and B. D. Rao. Neuromagnetic source imaging with focuss: a recursive weighted minimum norm algorithm. *Electroencephalography and Clinical Neurophysiology*, 95:231–251, 1995.

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

[48] R. Gribonval and K. Schnass. Dictionary identification - sparse matrix-factorisation via l1-minimisation. *ArXiv:0904.4774*, 2009.

[49] S. Haghighatshoar. Multi terminal probabilistic compressed sensing. pages 221–225, June 2014.

[50] Saeid Haghighatshoar and Emmanuel Abbe. Polarization of the renyi information dimension for single and multi terminal analog compression. *ArXiv:1301.6388*, 2013.

[51] Peter Hall, Gerard Kerkyacharian, and Dominique Picard. Block threshold rules for curve estimation using kernel and wavelet methods. *The Annals of Statistics*, 26(3):pp. 922–942, 1998.

[52] Jarvis Haupt, Robert Nowak, and Rui Castro. Adaptive sensing for sparse signal recovery. In *13th Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop*, pages 702–707, Marco Island, Florida, 2009.

[53] Lihan He and Lawrence Carin. Exploiting structure in wavelet-based bayesian compressive sensing. *IEEE Trans. on Signal Process.*, 57(9):3488–3497, 2009.

[54] D.W. Henderson and E. Moura. *Experiencing Geometry: On Plane and Sphere*. Prentice Hall, 1996.

[55] Matthew A. Herman and Thomas Strohmer. High-resolution radar via compressed sensing. *Trans. Sig. Proc.*, 57(6):2275–2284, June 2009.

[56] F. B. Hildebrand. *Advanced Calculus for Applications*. Prentice-Hall, 1976.

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

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

[59] Joseph Mitola III and Gerald Q. Maguire Jr. Cognitive radio: making software radios more personal. *IEEE Personal Commun.*, 6(4):13–18, 1999.

[60] R. Jenatton, R. Gribonval, and F. Bach. Local stability and robustness of sparse dictionary learning in the presence of noise. *ArXiv:1210.0685*, 2012.

[61] Shihao Ji, David B. Dunson, and Lawrence Carin. Multitask compressive sensing. *IEEE Trans. Signal Process.*, 57(1):92–106, 2009.

[62] 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.

[63] Jongmin Kim, Woohyuk Chang, Bangchul Jung, Dror Baron, and Jong Chul Ye. Belief propagation for joint sparse recovery. *ArXiv:1102.3289*, 2011.

[64] David Koslicki, Simon Foucart, and Gail Rosen. Quikr: a method for rapid reconstruction of bacterial communities via compressive sensing. *Bioinformatics*, 29(17):2096–2102, 2013.

[65] M. Ledoux. *The concentration of measure phenomenon*. Mathematical Surveys and Monographs, 2001.

[66] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In B. Schölkopf, J.C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 801–808. MIT Press, 2007.

[67] Michael S. Lewicki and Terrence J. Sejnowski. Learning overcomplete representations. *Neural Comput.*, 12(2):337–365, February 2000.

[68] J. Mairal, F. Bach, J.Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res.*, 11:19–60, March 2010.

[69] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *Trans. Img. Proc.*, 17(1):53–69, January 2008.

[70] A. Maleki, L. Anitori, Z. Yang, and R.G. Baraniuk. Asymptotic analysis of complex lasso via complex approximate message passing (camp). *IEEE Trans. Inf. Theory*, 59(7):4290–4308, July 2013.

[71] S. Mallat and Z. Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Trans. on Signal Process.*, 41:3397–3415, 1993.

[72] I. Maravic and M. Vetterli. Sampling and reconstruction of signals with finite rate of innovation in the presence of noise. *IEEE Trans. Signal Process.*, 53(8):2788–2805, Aug 2005.

[73] Lukas Meier, Sara van de Geer, and Peter Bühlmann. The group lasso for logistic regression. *J. of the Royal Stat. Society. Series B*, 70(1):53–71, 2008.

[74] Andrea Montanari. Graphical models concepts in compressed sensing. *ArXiv:1011.4328*, 2010.

[75] Joao F. C. Mota, Joao M. F. Xavier, Pedro M. Q. Aguiar, and Markus Puschel. Distributed basis pursuit. *IEEE Trans. Signal Process.*, 60(4):1942–1956, 2012.

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

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

[78] Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311 – 3325, 1997.

[79] T. Papadopoulo and M. Lourakis. Estimating the jacobian of the singular value decomposition: Theory and applications. In *In Proc. European Conf. on Computer Vision, ECCVï¿œï¿œ00*, pages 554–570. Springer, 2000.

[80] F. Parvaresh and B. Hassibi. Explicit measurements with almost optimal thresholds for compressed sensing. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 3853–3856, March 2008.

[81] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Ann. Asilomar Conf. on Signals, Systems, and Computers*, pages 40–44, 1993.

[82] K. B. Petersen and M. S. Pedersen. The matrix cookbook. nov 2012.

[83] G. Peyré, J. Fadili, and J-L. Starck. Learning adapted dictionaries for geometry and texture separation. In *Proceedings of SPIE Wavelet XII*, volume 6701, page 67011T, 2007.

[84] Sundeep Rangan. Generalized approximate message passing for estimation with random linear mixing. *ArXiV:1010.5141*, 2010.

[85] Sundeep Rangan. Generalized approximate message passing for estimation with random linear mixing. *IEEE Int. Sym. on Inf. Theory*, pages 2168–2172, 2011.

[86] Sundeep Rangan, Philip Schinter, Erwin Riegler, Alyson Fletcher, and Volkan Cevher. Fixed points of generalized approximate message passing with arbitrary matrices. *IEEE Int. Sym. on Inf. Theory*, pages 664–668, 2013.

[87] Volker Roth and Bernd Fischer. The group-lasso for generalized linear models: uniqueness of solutions and efficient, 2008.

[88] M. Seeger. Bayesian inference and optimal design for the sparse linear model. *J. of Machine Learning Research*, 9:759–813, 2008.

[89] I. W. Selesnick and A. F. Abdelnour. Symmetric wavelet tight frames with two generators. *Applied and Computational Harmonic Analysis*, pages 211–225, 2004.

[90] J. W. Silverstein. The smallest eigenvalue of a large dimensional wishart matrix. *Ann. Probab.*, 1985.

[91] K. Skretting and K. Engan. Recursive least squares dictionary learning algorithm. *IEEE Trans. on Signal Process.*, 58(4):2121–2130, Apr 2010.

[92] S. Som, L.C. Potter, and P. Schniter. On approximate message passing for reconstruction of non-uniformly sparse signals. *IEEE National Aerospace and Electronics Conf.*, pages 223–229, 2010.

[93] 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.

[94] Mihailo Stojnic. Block-length dependent thresholds in block-sparse compressed sensing. *ArXiv:0907.3679*, 2009.

[95] Mihailo Stojnic, Farzad Parvaresh, and Babak Hassibi. On the reconstruction of block-sparse signals with an optimal number of measurements. *ArXiv:0804.0041*, 2008.

[96] Dennis Sundman, Saikat Chatterjee, and Mikael Skoglund. Distributed greedy pursuit algorithms. *ArXiv:1306.6815*, 2013.

[97] Dennis Sundman, Dave Zachariah, Saikat Chatterjee, and Mikael Skoglund. Distributed predictive subspace pursuit. In *ICASSP*, pages 4633–4637. IEEE, 2013.

[98] Armeen Taeb, Arian Maleki, Christoph Studer, and Richard G. Baraniuk. Maximin analysis of message passing algorithms for recovering block sparse signals. *ArXiv:1303.2389*, 2013.

[99] E. Tanczos and R Castro. Adaptive sensing for estimation of structured sparse signals. *ArXiv:1311.7118*, 2013.

[100] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

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

[102] Joel A. Tropp. Algorithms for simultaneous sparse approximation. part ii: Convex relaxation. *IEEE Trans. Signal Process.*, 86(3):589–602, March 2006.

[103] Joel A. Tropp, Anna C. Gilbert, and Martin J. Strauss. Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. *IEEE Trans. Signal Process.*, 86(3):572–588, March 2006.

[104] George Tzagkarakis, Dimitris Milioris, and Panagiotis Tsakalides. Multiple-measurement bayesian compressed sensing using gsm priors for doa estimation. In *ICASSP*, pages 2610–2613. IEEE, 2010.

[105] S. S. Vasanawala, M. J. Murphy, M. T. Alley, P. Lai, Kurt Keutzer, John M. Pauly, and Michael Lustig. Practical parallel imaging compressed sensing mri: Summary of two years of experience in accelerating body mri of pediatric patients. In *ISBI*, pages 1039–1043. IEEE, 2011.

[106] N. Vaswani and Wei Lu. Modified-cs: Modifying compressive sensing for problems with partially known support. In *IEEE International Symposium on Information Theory*, pages 488–492, Seoul, Korea, 2009.

[107] E. Vincent, R. Gribonval, and C. Fevotte. Performance measurement in blind audio source separation. *IEEE Trans. on Audio, Speech and Language Process.*, 14(4):1462–1469, 2006.

[108] M.B. Wakin, J.N. Laska, M.F. Duarte, D. Baron, S. Sarvotham, D. Takhar, K.F. Kelly, and R.G. Baraniuk. An architecture for compressive imaging. In *Image Processing, 2006 IEEE International Conference on*, pages 1273–1276, Oct 2006.

[109] D. Wei and A. O. Hero. Multistage Adaptive Estimation of Sparse Signals. *IEEE Journal of Selected Topics in Signal Processing*, 7:783–796, October 2013.

[110] Thakshila Wimalajeewa and Pramod K. Varshney. Cooperative sparsity pattern recovery in distributed networks via distributed-omp. In *ICASSP*, pages 5288–5292. IEEE, 2013.

[111] Yihong Wu and Sergio Verdu. Optimal phase transitions in compressed sensing. *IEEE Trans. Inf. Theory*, 58(10):6241–6263, Oct 2012.

[112] T. Xu, W. Wang, and W. Dai. Sparse coding with adaptive dictionary learning for underdetermined blind speech separation. *Speech Communication*, 55(3):432–450, 2013.

[113] Weiyu Xu and B. Hassibi. Efficient compressive sensing with deterministic guarantees using expander graphs. In *Information Theory Workshop, 2007. ITW '07. IEEE*, pages 414–419, Sept 2007.

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

[115] Allen Y. Yang, Michael Gastpar, Ruzena Bajcsy, and Shankar S. Sastry. Distributed sensor perception via sparse representation. *Proceedings of the IEEE*, 98(6):1077–1088, 2010.

[116] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

[117] Wenbo Zhang, Cong Ma, Weiliang Wang, Yu Liu, and Lin Zhang. Side information based orthogonal matching pursuit in distributed compressed sensing. In *Network Infrastructure and Digital Content, 2010 2nd IEEE International Conference on*, pages 80–84, Sept 2010.

[118] X. Zhao, T. Xu, G. Zhou, W. Wang, and W. Dai. Joint image separation and dictionary learning. *Int. Conf. Digital Signal Processing*, 2013.

[119] Justin Ziniel and Philip Schniter. Efficient high-dimensional inference in the multiple measurement vector problem. *IEEE Trans. Signal Process.*, 61(2):340–354, 2013.