RICE UNIVERSITY

Compressive Sensing for 3D Data Processing Tasks: Applications, Models and Algorithms

by

Chengbo Li

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Doctor of Philosophy

Approved, Thesis Committee:

Houston, Texas

April 2011

Abstract

Compressive Sensing for 3D Data Processing Tasks: Applications, Models and Algorithms

by

Chengbo Li

Compressive sensing (CS) is a novel sampling methodology representing a paradigm shift from conventional data acquisition schemes. The theory of compressive sensing ensures that under suitable conditions compressible signals or images can be reconstructed from far fewer samples or measurements than what are required by the Nyquist rate. So far in the literature, most works on CS concentrate on one-dimensional or two-dimensional data. However, besides involving far more data, three-dimensional (3D) data processing does have particularities that require the development of new techniques in order to make successful transitions from theoretical feasibilities to practical capacities. This thesis studies several issues arising from the applications of the CS methodology to some 3D image processing tasks. Two specific applications are hyperspectral imaging and video compression where 3D images are either directly unmixed or recovered as a whole from CS samples. The main issues include CS decoding models, preprocessing techniques and reconstruction algorithms, as well as CS encoding matrices in the case of video compression.

Our investigation involves three major parts. (1) Total variation (TV) regularization plays a central role in the decoding models studied in this thesis. To solve such models, we propose an efficient scheme to implement the classic augmented Lagrangian multiplier method and study its convergence properties. The resulting Matlab package TVAL3 is used to solve several models. Computational results show that, thanks to its low per-iteration complexity, the proposed algorithm is capable of handling realistic 3D image processing tasks. (2) Hyperspectral image processing typically demands heavy computational resources due to an enormous amount of data involved. We investigate low-complexity procedures to unmix, sometimes blindly, CS compressed hyperspectral data to directly obtain material signatures and their abundance fractions, bypassing the high-complexity task of reconstructing the image cube itself. (3) To overcome the "cliff effect" suffered by current video coding schemes, we explore a compressive video sampling framework to improve scalability with respect to channel capacities. We propose and study a novel *multi-resolution* CS encoding matrix, and a decoding model with a *TV-DCT* regularization function.

Extensive numerical results are presented, obtained from experiments that use not only synthetic data but also real data measured by hardware. The results establish feasibility and robustness, to various extent, of the proposed 3D data processing schemes, models and algorithms. There still remain many challenges to be further resolved in each area, but hopefully the progress made in this thesis will represent a useful first step towards meeting these challenges in the future.

Acknowledgements

I would like to express my deepest and sincerest gratitude to my academic advisor and also my spiritual mentor, Prof. Yin Zhang. His enthusiasm, profound knowledge, and upbeat personality have greatly influenced me in these four years. He has been helping me accumulate my research skills, tap into my full potential, as well as build up my confidence step by step in the course of researching. Without his wholehearted guidance, I might have already lost my interest in optimization, or even in research. I truly take pride in working with him.

I feel so grateful for Prof. Wotao Yin, who has led me to this CAAM family at Rice University since 2007. He has provided me tremendous help on both academic and living sides. I owe many thanks to him for his encouragement, patience, and guidance. Besides, his intelligence and humor have deeply impressed me. He is not only my mentor, but also my friend in life.

Prof. Kevin Kelly and Ting Sun, who are my collaborators in the ECE department of Rice University, have shared large quantities of data with me and helped me fully understand the mechanism of hardware they built like the single-pixel camera. It has been a great pleasure working with them and I look forward to the future collaboration in other areas.

Within these four years, two successful internship experiences tremendously enriched my life. I deeply appreciate my supervisors Dr. Hong Jiang in Bell Laboratories and Dr. Amit Chakraborty in Siemens Corporate Research for their instructions and praise for my work there. Besides, a profound discussion between Dr. Jiang and me inspired my research on video compression. I could not have made such rapid progress in the field of video coding without Dr. Jiang's encouragement and support.

Besides, I need to thank Prof. Richard Baraniuk who introduced me a treasured opportunity that I can continue projecting my professional strength after graduation; Prof. Richard Tapia who taught me that mathematicians could take on more than mathematics; Prof. William Symes who is one of my committee members and earnestly reviewed my thesis; Prof. Liliana Borcea who was my mentor during my first year at CAAM and helped me adapt the new environment; Daria Lawrence who reminded me about administrative procedures and important deadlines from time to time; Josh Bell who is one of my best friends in America and treated me just like one of his families; Chao Wang who is my soul mate and has been supportive through all these years. Meanwhile, I offer my regards and blessings to all of those professors and peers who have provided me knowledge and expertise during my undergraduate and graduate studies.

Last but certainly not least, I wish to dedicate this thesis to my grandparents and my parents for their selfless love and unconditional support over the years. No matter where I am and how far apart we are, you are the love of my life for eternity.

Contents

List of Figures

Chapter 1

Introduction

For many years, signal processing relies on the well-known Shannon sampling theorem [1], stating that the sampling rate must be at least twice as high as the highest frequency to avoid losing information while capturing the signal (the so-called Nyquist rate). In many applications, such as digital cameras, the Nyquist rate is too high to either store or transmit without making compression a necessity prior. In addition, increasing the sampling rate might be very costly in many other scenarios — medical scanners, high-speed analog-to-digital converters, and so forth.

In recent years, a new theory of compressive sensing — also known under the terminology of compressed sensing, compressive sampling, or CS — has drawn a lot of researchers' attention. It builds a fundamentally novel approach to data acquisition and compression which overcomes drawbacks of the traditional method. Nowadays, compressive sensing has been widely studied and applied to various fields, such as radar imaging [35], magnetic resonance imaging [36, 37, 38], analog-to-information conversion [39], sensor networks [40, 41] and even homeland security [42].

A new iterative CS solver — TVAL3 — has been proposed for 1D and 2D signal processing in the author's master thesis [9], and has been successfully applied to single-pixel cameras [32, 34]. TVAL3 is short for “TV minimization by augmented Lagrangian and alternating direction algorithms”. Its efficiency and robustness has been empirically investigated, but the theoretical convergence has not been established. In this thesis, the algorithm behind TVAL3 will be restated for more general cases and a proof of convergence will be studied and presented. After that, the thesis will move into the main part — high-dimensional data processing employing the CS theory and the general TVAL3 method. It would be inefficient to study the general case of the high-dimensional data without considering inherent structures and characteristics of different kinds. Therefore, two classes of 3D data processing problem will be addressed here — hyperspectral data unmixing and video compression.

The thesis is organized as follows: a review of compressive sensing, an introduction to the total variation, and the background of hyperspectral data unmixing and video compression will be covered in this chapter; Chapter 2 completes the general TVAL3 algorithm by extending it to a more general setting and establishing a convergence result; Chapter 3 and 4 describe in detail the compressive sensing and unmixing of hyperspectral data and the compressive video sensing framework, respectively; Chapter 5 concludes the thesis by iterating the main contributions and discussing the future work in the relevant fields of scientific research.

1.1 Compressive Sensing

In 2004, Donoho, Candès, Romberg and Tao conducted a series of in-depth research based on the discovery that a signal may still be recovered even though the number of data is deemed insufficient by Shannon’s criterion, and built the theory of compressive sensing [4, 3, 2]. To make the exact recovery possible from far fewer samples or measurements, CS theory counts on two principles: *sparsity* and *incoher-*

ence. Sparsity screens out the signal of interest, while incoherence restricts the sensing schema. Specifically, a large but sparse signal is encoded by a relatively small number of incoherent linear measurements, and the original signal can be reconstructed from the encoded sample by finding the sparsest signal from the solution set of a under-determined linear system. It has been proven that computing the sparsest solution directly (ℓ_0 minimization in mathematics) is NP-hard and generally requires prohibitive computations of exponential complexity [10]. However, the discovery of ℓ_0-ℓ_1 equivalence [8] averted solving NP-hard problems for compressive sensing.

Differing from ℓ_0-norm, which counts the number of nonzeros and is not a real norm literally, ℓ_1-norm measures the sum of magnitudes of all elements of a vector. The use of ℓ_1-norm as a sparsity-promotion function can be traced back decades. In 1986, for example, Santosa and Symes [13] introduced ℓ_1 minimization to reflection seismology, seeking a sparse reflection function which indicates significant variances between subsurface layers from bandlimited data. They appear to be the first to give a coherent mathematical argument behind using ℓ_1-norm for sparsity promotion, though it had been used by practitioners long before. In the next few years, Donoho and his colleague carried this brilliant idea further and explored some early results regarding ℓ_1 minimization and signal recovery [15, 16]. More work on ℓ_1 minimization under special setups was investigated in the early 2000s [22, 23, 24, 25].

Grounded on those early efforts, a major breakthrough was achieved by Candès, Romberg and Tao [3, 2], and Donoho [4] between 2004 and 2006, which theoretically proved ℓ_1 minimization is equivalent to ℓ_0 minimization under some conditions for signal reconstruction problems. Furthermore, they showed that a K-sparse signal (under some basis) could be exactly recovered from cK linear measurements using ℓ_1 minimization, where c is a constant. This new theory has significantly improved those earlier results on sparse recovery using ℓ_1. Here, the constant c directly decides the size of linear measurements. The introduction of the *restricted isometry property* (RIP) for matrices [5] — a key concept of compressive sensing — responded this question theoretically. Candès and Tao showed that if the measurement matrix satisfies the RIP to a certain degree, it is sufficient to guarantee the exact sparse signal recovery. It has been shown that Gaussian, Bernoulli and partial Fourier matrices with random permutations possess the RIP with high probability [3, 26], and become reasonable choices as the measurement or sensing matrix. For example, K-sparse signals of length N require only $cK\log(N/K) \ll N$ random Gaussian measurements for exact recovery. However, it is extremely difficult and sometimes impractical to verify the RIP property for most types of matrices. Is RIP truly an indispensable property for compressive sensing? For instance, measurement matrices A and GA in ℓ_1 minimization should retain exactly the same recoverability and stability as long as matrix G is square and nonsingular, but their RIP constant may vary a lot due to different choices of G. A non-RIP analysis, studied by Zhang, proved recoverability and stability theorems without the aid of RIP and claimed prior knowledge could never hurt, but possibly enhance the reconstruction via ℓ_1 minimization [7].

Other than ℓ_1 minimization methods (also known as Basis Pursuit [12, 27, 28]), greedy methods could also handle compressive sensing problems by iteratively computing the support of the signal. Generally speaking, a greedy method refers to the one following the metaheuristic of choosing the best immediate or local optimum at each stage and eventually expecting to find the global optimum. In 1993, Mallat and Zhang introduced Matching Pursuit (MP) [29], which is the prototypical greedy algorithm applied to compressive sensing. In recent years, a series of MP-based greedy methods have been proposed for compressive sensing, such as Orthogonal Matching Pursuit [30], Compressive Sampling Matching Pursuit [31], and so on. However, ℓ_1 minimization methods usually require fewer measurements than greedy algorithms and provide better stability. When noise exists or the signal is not exactly sparse, ℓ_1 minimization methods provide a much more stable solution and make the methods applicable to real world problems.

1.2 TV Regularization

Total variation (abbreviated TV) regularization can be regarded as a generalized ℓ_1 regularization in compressive sensing problems. Instead of assuming the signal is sparse, the premise of TV regularization is that the gradient of the underlying signal or image is sparse. In other words, total variation measures the discontinuities and the TV minimization seeks the solution with the sparsest gradient.

In the broad area of compressive sensing, TV minimization has attracted more and more research activities since recent research indicates that the use of TV regularization instead of the ℓ_1 term makes the reconstructed images sharper by preserving the edges or boundaries more accurately. In most cases, edges of the underlying image are more essential to characterize different properties than the smooth part. For example, in the realm of seismic imaging, detecting boundaries of distinct media play a key role in identifying the geological structure. This advantage of TV minimization stems from the property that it can recover not only sparse signals or images, but also dense staircase signals or piecewise constant images. Even though this result has only been theoretically proven under some special circumstances [2], it stands true on a much larger scale empirically.

The history of TV is long and rich, tracing back at least to 1881 when Jordan first introduced total variation for real-valued functions while studying the convergence of Fourier series [11]. After decades of research, it has been thoroughly investigated and widely used for the computation of discontinuous solutions of inverse problems (see

[19, 20, 21], for example). In 1992, Rudin, Osher and Fatemi [14] first introduced the concept total variation into image denoising problems. From then on, TV minimizing models have become one of the most popular and successful methodologies for image denoising [14, 43], deconvolution [47, 46] and restoration [49, 48], to cite just a few. Some constructive discussions on TV regularized problems have been reported by Chambolle *et al.* [50, 51].

In spite of those remarkable advantages of TV regularization, the properties of non-differentiability and non-linearity make TV minimization far less accessible and solvable computationally than ℓ_1 minimization. Geman and Yang [45] proposed a joint minimization method to solve half-quadratic models [44, 45]. Grounded on this work, Wang, Yang, Yin and Zhang proposed and studied a fast half-quadratic method to solve deconvolution and denoising problems with TV regularization [46] and further extended this method to image reconstruction [52] and multichannel image deconvolution problems [53, 54]. The two central ideas in this approach are "splitting" and "alternating". The key step is to introduce a so-called splitting variable to move the differentiation operator from inside the TV term to outside, thus enabling low-complexity subproblems in an alternating minimization setting. These ideas have been previously used in solving a number other problems, but their applications to TV regularized problems has resulted in algorithms significantly faster than the previous state-of-the-art algorithms in this area.

Even though this method is very efficient and effective, it restricts the measurement matrix to the partial Fourier matrix. Under a more general setting, Goldstein and Osher [56] added Bregman regularization [55] into this idea, producing the so-called split Bregman algorithm for TV regularized problems. This algorithm is equivalent to the classic alternating direction method of multipliers [58, 59] when only one inner iteration of split Bregman is performed. Around the same year, Li, Zhang and

Yin employed the same splitting and alternating direction idea on the classic augmented Lagrangian method [60, 61] and developed an efficient TV regularized solver — TVAL3 [9, 125]. This particular implementation also integrates a non-monotone line search [82] and Barzilai-Borwein steps [79] into it and results in a much faster algorithm. TVAL3 has been proposed and thoroughly studied in author's master thesis [9], and numerical evidences indicates that TVAL3 outperforms other TV solvers when solving compressive sensing problems, such as SOCP [48], ℓ_1-Magic [2, 3, 5], TwIST [86, 85] and NESTA [84]. However, its theoretical result of convergence has not been established until recently. In this thesis, algorithms of 3D data processing are extended from TVAL3, whose general descriptions as well as convergence proof will be revealed in Chapter 2.

1.3 3D Data Processing

Three-dimensional (3D) data processing has tremendous applications in today's world, such as in surveillance [93], exploitation [92], wireless communications [96], military intelligence [94], public entertainments [95], environmental monitoring [91], and so forth. However, some common bottlenecks or difficulties slow down the pace of development of 3D data processing. One of the main difficulties rises from the enormous volume of 3D data, which causes inconvenience of storing, transmitting and even processing. Therefore, it is critical to explore the inherence of data on different domains and develop effectual methods to reduce the volume of 3D data without losing the key information.

Compressive sensing has been widely recognized as a promising and effective acquisition method for 1D and 2D data processing. In this thesis, the author will explore two important classes of 3D data processing tasks — hyperspectral unmixing and video compression — grounded on the framework of compressive sensing. Both hyperspectral and video data can be regarded as a series of 2D images. Simply applying the compressive sensing idea on 2D images slice by slice could work to some extent, but is far from optimal or ideal situations. More sparsity and further compression can be obtained by properly utilizing inherent connections among those 2D slices. For example, video clips are usually continuous in time domain and the unchanged background in adjacent frames could be subtracted. This is one straightforward way to enhance the sparsity of video data. Moreover, advanced techniques or methods require further study on the nature of 3D data sets. More detailed introduction and review of hyperspectral and video data will be presented at the beginning of Chapters 3 and 4, respectively.

1.4 Organization

The thesis is organized as follows. Chapter 2 describes the TVAL3 algorithm in a general setting and establishes a theoretical convergence result for the algorithm. Chapter 3 focuses on the hyperspectral imaging and proposes new compressive sensing and unmixing schemes which can significantly reduce both the storage and computational complexity. Chapter 4 turns to the discussion of video compression for wireless communication and raises a novel multi-resolution framework based on the compressive video sensing. Both Chapter 3 and Chapter 4 contain descriptions and results of a number of numerical experiments to demonstrate the efficiency and effectiveness, as well as limitations, of proposed methods or framework. Lastly, Chapter 5 concludes the whole thesis and points out the future work of compressive sensing on 3D data processing.

Chapter 2

General TVAL3 Algorithm

The algorithm of TVAL3 has been proposed and numerically studied for TV regularized compressive sensing problems in author's master thesis [9]. Extensive numerical experiments have demonstrated its efficiency and high tolerance to noise. In this chapter, the methodology of TVAL3 will be described in a general case and convergence will be theoretically analyzed for the first time.

Starting with the review of the classic augmented Lagrangian method, this chapter will describe the development of the general TVAL3 algorithm step by step.

2.1 Review of Augmented Lagrangian Method

For constrained optimization, a fundamental class of methods is to seek the minimizer or maximizer by solving a sequence of unconstrained subproblems iteratively. The solutions of subproblems should converge to a minimizer or maximizer eventually. Back to 1943, Courant [57] proposed the quadratic penalty method, which could be viewed as the precursor to the augmented Lagrangian method. This method penalizes equality constraint violation by adding a multiple of the square of the constraint violation into the objective function and turns the constrained optimization problems to be unconstrained. Due to its simplicity and intuitive appeal, this approach has been used and studied comprehensively. However, it requires the penalty parameter to go to infinity to guarantee convergence, which may cause a deterioration in the numerical conditioning of the method. In 1969, Hestenes [60] and Powell [61] independently proposed the augmented Lagrangian method which, by introducing and adjusting Lagrangian multiplier estimates, no longer requires the penalty parameter to go to infinity for the method to converge.

2.1.1 Derivations and Basic Results

Let us begin with considering a general equality-constrained minimization problem

$$\min_x f(x), \quad \text{s.t. } h(x) = 0, \tag{2.1}$$

where h is a vector-valued function and both f and h_i for all i are differentiable. The first-order optimality conditions for (2.1) are

$$\begin{cases} \nabla_x \mathcal{L}(x, \lambda) = 0, \\ h(x) = 0, \end{cases} \tag{2.2}$$

where $\mathcal{L}(x, \lambda) = f(x) - \lambda^T h(x)$ is the Lagrangian function of (2.1). By optimization theory, conditions in (2.2) are necessary for optimality under some constraint qualifications. In addition, if problem (2.1) is a convex program, then they are also sufficient.

In light of the optimality conditions above, an optimum x^* to the original problem (2.1) is both a stationary point of the Lagrangian function and a feasible point of constraints, which means x^* solves

$$\min_x \mathcal{L}(x, \lambda), \quad \text{s.t. } h(x) = 0. \tag{2.3}$$

In fact, it is obvious that (2.1) is equivalent to (2.3) for any λ. According to the quadratic penalty method, a local minimizer x^* of (2.3) may be obtained by solving a series of unconstrained problems with the constraint violations penalized as follows:

$$\min_x \mathcal{L}_A(x, \lambda; \mu) = f(x) - \lambda^T h(x) + \frac{\mu}{2} h(x)^T h(x). \tag{2.4}$$

It follows the analysis of the penalty method that λ can be arbitrary but μ needs to go to infinity, which may cause a deterioration of the numerical conditioning and result in inaccuracy. The augmented Lagrangian method iteratively solves problem (2.4) above, but updates multiplier λ in a specific way, and still guarantee convergence to the minimizer of (2.1) without forcing penalty parameter μ to go to infinity. In that case, $\mathcal{L}_A(x, \lambda; \mu)$ is known as the *augmented Lagrangian function.*

Intuitively, the augmented Lagrangian function differs from the standard Lagrangian function by adding a square penalty term, and differs from the quadratic penalty function by the presence of the linear term involving the multiplier λ. Hence, the augmented Lagrangian method combines the advantages of the Lagrange multiplier and penalty techniques without having their respective drawbacks.

Specifically, the augmented Lagrangian method can be described as follows. Fixing the multiplier λ at the current estimate λ^k and the penalty parameter μ to $\mu^k > 0$ at the k-th iteration, we minimize the augmented Lagrangian function $\mathcal{L}_A(x, \lambda^k; \mu^k)$ with respect to x and denote the minimizer of current iterate as x^{k+1}. To update the multiplier estimates from iteration to iteration, Hestenes [60] and Powell [61] suggested the following update formula:

$$\lambda^{k+1} = \lambda^k - \mu^k h(x^{k+1}). \tag{2.5}$$

Bertsekas [71] proved one of the fundamental theorems to estimate the error bounds and also the rate of convergence. For convenience, $\|.\|$ refers to ℓ_2 norm hereafter. The theorem can be reiterated as follows:

Theorem 2.1.1 (Local Convergence)**.** *Let x^* be a strictly local optimum of* (2.1) *at which the gradients $\nabla h_i(x^*)$ are linearly independent, and $f, h \in C^2$ in an open neighborhood of x^*. Furthermore, x^* together with its associated Lagrangian multiplier λ^* satisfies*

$$z^T \nabla^2_{xx} \mathcal{L}(x^*, \lambda^*) z > 0,$$

for all $z \neq 0$ with $\nabla h_i(x^*)^T z = 0$ $\forall i$; i.e., the second-order sufficient conditions are satisfied for $\lambda = \lambda^*$. Choose $\bar{\mu} > 0$ so that $\nabla^2_{xx} \mathcal{L}_A(x^*, \lambda^*; \bar{\mu})$ is also positive definite. Then there exist positive constants δ, ϵ, and M such that the following claims hold:

1. *For all $(\lambda^k, \mu^k) \in \mathcal{D}$ where $\mathcal{D} \triangleq \{(\lambda, \mu) : \ \|\lambda - \lambda^*\| < \delta\mu, \ \mu \geq \bar{\mu}\}$, the problem*

$$\min_x \mathcal{L}_A(x, \lambda^k; \mu^k) \quad s.t. \ \|x - x^*\| = \epsilon$$

has a unique solution $x^k \triangleq x(\lambda^k, \mu^k)$. It satisfies

$$\|x^k - x^*\| \leq \frac{M}{\mu^k} \|\lambda^k - \lambda^*\|.$$

Moreover, function $x(\lambda, \mu)$ is continuously differentiable in the interior of $\mathcal{D}$.

2. *For all $(\lambda^k, \mu^k) \in \mathcal{D}$,*

$$\|\lambda^{k+1} - \lambda^*\| \leq \frac{M}{\mu^k}\|\lambda^k - \lambda^*\|,$$

   if λ^{k+1} is attained by (2.5)*.*

3. *For all $(\lambda^k, \mu^k) \in \mathcal{D}$, $\nabla^2_{xx}\mathcal{L}_A(x^k, \lambda^k; \mu^k)$ is positive definite and $\nabla h_i(x^k)$ are linearly independent.*

A detailed proof for local convergence theorem can be found in [71], pp. 108.

The local convergence theorem implies at least three features of the augmented Lagrangian method. First of all, the method converges in one iteration if $\lambda = \lambda^*$. Secondly, as long as μ^k satisfies $\frac{M}{\mu^k} < 1$ for any k, the error bounds in the theorem are able to guarantee that

$$\|\lambda^{k+1} - \lambda^*\| \quad < \quad \|\lambda^k - \lambda^*\|;$$

i.e., the multiplier estimates converge linearly. Hence, $\{x^k\}$ also converges linearly. Lastly, if μ^k goes to infinity, then

$$\lim_{k\to+\infty} \frac{\|\lambda^{k+1} - \lambda^*\|}{\|\lambda^k - \lambda^*\|} \quad = \quad 0;$$

i.e., the multiplier estimates converge superlinearly.

The augmented Lagrangian method requires solving an unconstrained minimization subproblem at each iteration, which could be overly expensive. Therefore, designing appropriate schemes to solve subproblems is one of the key issues when applying the augmented Lagrangian method.

Numerically, it is impossible to find an exact minimizer of unconstrained minimization subproblem at each iteration. For convex optimization, Rockafellar [63] proved the global convergence of the augmented Lagrangian method in the convex case for an arbitrary penalty parameter, without demanding an exact minimum at each iteration. In addition, the objective function f is no long assumed to be differentiable and the theorem still holds.

Theorem 2.1.2 (Global Convergence). *Suppose that*

1. *f is convex and h_i are linear constraints;*
2. *the feasible set $\{x : \ h(x) = 0\}$ is non-empty;*
3. *$\mu^k = \mu$ is constant for all k;*
4. *a sequence $\{\epsilon_k\}^\infty$ satisfies $0 \leq \epsilon_k \to 0$ and $\sum_i^\infty \sqrt{\epsilon_k} < \infty$;*
5. *at the k-th iteration, choose $x_{k+1} \in \{x : \ \|\nabla_x \mathcal{L}_A(x, \lambda^k; \mu^k)\| \leq \epsilon_k\}$ and update multiplier λ^{k+1} following* (2.5).

Then attained sequence $\{x^k\}$ converges to the global minimizer of (2.1).

A detailed proof for global convergence theorem can be found in [63], pp. 560–561.

This theorem confirms the global convergence under the condition of convexity and approximate solutions for unconstraint subproblems. The error tolerance for solving subproblems ensures the feasibility of this method in numerical analysis and scientific computation.

2.1.2 Operator Splitting

Now we consider some structured function f. A large number of problems in physics, mechanics, economics and mathematics consider $f(x) = f_1(Bx) + f_2(x)$, where both

f_1 and f_2 are convex, proper, lower semicontinuous functionals, and B is a linear operator. In the early 1980s, Glowinski *et al.* studied this type of problems in depth using the augmented Lagrangian and operator-splitting methods [68, 69, 70], which are also closely related to the time-dependent approach as can be seen in, e.g., [67]. We consider

$$\min_{x} \left\{ f_1(Bx) + f_2(x) \right\}, \quad \text{s.t. } Ax = b, \tag{2.6}$$

where f_1 may be non-differentiable. Let $w = Bx$, then (2.6) is clearly equivalent to

$$\min_{w,x} \left\{ f_1(w) + f_2(x) \right\}, \quad \text{s.t. } Ax = b,\ Bx = w. \tag{2.7}$$

With a new variable and the extra linear constraints, the objective of (2.6) has been split into two parts. The aim of splitting is to separate non-differentiable terms from other differentiable ones. Now (2.7) can be simply rewritten as

$$\min_{w,x} \left\{ f_1(w) + f_2(x) \right\}, \quad \text{s.t. } h(w,x) = 0, \tag{2.8}$$

where for simplicity the two linear constraints have been written into a single constraint.

The augmented Lagrangian function for (2.8) is

$$\mathcal{L}_A(w,x,\lambda;\mu) = f_1(w) + f_2(x) - \lambda^T h(w,x) + \frac{\mu}{2} h(w,x)^T h(w,x). \tag{2.9}$$

For fixed λ^k and μ^k, denote $f_1(w)$ as $\varphi(w)$ and other parts in $\mathcal{L}_A(w,x,\lambda^k;\mu^k)$ as

$\phi(w, x)$ which is differentiable. Then the augmented Lagrangian method solves

$$\min_{w,x} \{\varphi(w) + \phi(w, x)\} \tag{2.10}$$

at the k-th iteration and then update the multiplier. The multiplier-updating formula could be more general than the one suggested by Hestenes and Powell; that is,

$$\lambda^{k+1} = \lambda^k - \varsigma^k \mu^k h(x^{k+1}). \tag{2.11}$$

Provided that ς^k is selected from a closed interval in $(0, 2)$, the convergence of the augmented Lagrangian method is still guaranteed in the convex case analogous to Theorem 2.1.2 [63]. Considering problem (2.6) without constraints, Glowinski proved a stronger theorem for both finite and infinite dimensional settings [70].

Other than (2.11), Buys [62] and Tapia [64] have suggested two other multiplier update formulas (called Buys update and Tapia update respectively), both involving second-order information of $\mathcal{L}_A$. Tapia [65] and Byrd [66] have shown that both update formulas give quadratic convergence if one-step (for Tapia update) or two-step (for Buys update) Newton's method is applied to subproblems. However, the estimate of the second-order derivative and the use of Newton's step can be too expensive to compute at each iteration for large-scale problems.

Specifically, an implementation of the augmented Lagrangian method for (2.6) can be put into the following algorithmic framework:

Algorithm 2.1.1 (Augmented Lagrangian Method)**.**

```
*Initialize* $\mu^0$, $\lambda^0$, $0 < \alpha_0 \le \varsigma^0 \le \alpha_1 < 2$, *tolerance* $tol$, *and starting points* $w^0$, $x^0$.
**While** $\|\nabla \mathcal{L}(x^k, \lambda^k)\| > tol$ **Do**
    *Set* $w_0^{k+1} = w^k$ *and* $x_0^{k+1} = x^k$;
```

```
        Find a minimizer $(w^{k+1}, x^{k+1})$ of $\mathcal{L}_A(w, x, \lambda^k; \mu^k)$, starting from $w_0^{k+1}$ and
            $x_0^{k+1}$ and terminating when $\|\nabla_{(w,x)}\mathcal{L}_A(w^{k+1}, x^{k+1}, \lambda^k; \mu^k)\| \leq tol$;
        Update the multiplier using (2.11) to obtain $\lambda^{k+1}$;
        Choose the new penalty parameter $\mu^{k+1} \geq \mu^k$ and $\alpha_0 \leq \varsigma^{k+1} \leq \alpha_1$;
    End Do
```

To accommodate non-differentiable functions, let

$$\tilde{\nabla} g(u) = \underset{\xi \in \partial g(u)}{\operatorname{argmin}} \|\xi\|.$$

That is, $\tilde{\nabla} g(u)$ is the member of $\partial g(u)$ with the smallest ℓ_2 norm; and it is equivalent to the gradient of g if the functional is differentiable. In Algorithm 2.1.1, we will replace"∇" by "$\tilde{\nabla}$" whenever the objective function is non-differentiable.

In Algorithm 2.1.1, $\varsigma^k = 1$ appears to generally give the best convergence from our computational experience, but it is not necessarily the case for the choice of small μ^k. Concerning the choice of μ^k, it has been shown that larger μ^k results in faster asymptotic convergence rate. On the other hand, larger μ^k causes numerical conditioning problems in practice. Fortunately, the combined effect of all these factors is the fact that convergence of the augmented Lagrangian method is relatively insensitive to the choice of the penalty parameter in most cases. In practice, starting with a small μ^k and then increasing μ^k from iterate to iterate usually gives a faster convergence numerically than keeping μ^k fixed. This approach is also known as *parameter continuation.*

The augmented Lagrangian method has been successfully applied to different fields, such as constraint motion problems [75], seismic reflection tomography [76], and so forth. From a numerical perspective, the only nontrivial part in the use of Algorithm 2.1.1 is how to efficiently minimize the augmented Lagrangian function or equivalently (2.10) at each iteration. Taking into account the particular structure as in (2.10), a well-suited algorithm will be proposed and theoretically analyzed in the next section. Before that, another method of multipliers which has a close relation to the augmented Lagrangian method will be briefly reviewed.

2.1.3 A Discussion on Alternating Direction Methods

Extending the classic augmented Lagrangian method as described above, Glowinski *et al.* [58, 59] also suggested another slightly different way to handle (2.8) — the alternating direction method (abbreviated ADM). The common advantage of both methods includes the capability of handling the non-differentiability and side-constraints. Instead of requiring the exact minimizer of the augmented Lagrangian function (2.9) at each iteration, ADM only demands minimizers with respect to w and x respectively, and then update the multiplier. Specifically, at the k-th iteration, we compute

$$\begin{cases} x^{k+1} = \underset{x}{\operatorname{argmin}}\, \mathcal{L}_A(w^k, x, \lambda^k; \mu^k), \\ w^{k+1} = \underset{w}{\operatorname{argmin}}\, \mathcal{L}_A(w, x^{k+1}, \lambda^k; \mu^k), \\ \lambda^{k+1} = \lambda^k - \varsigma^k \mu^k h(w^{k+1}, x^{k+1}). \end{cases} \tag{2.12}$$

Contrary to the joint minimization as is done in the augment Lagrangian method, ADM uses the idea of alternating minimization to produce computationally more affordable iterations (2.12). Provided that

$$0 < \varsigma^k = \varsigma < \frac{1+\sqrt{5}}{2},$$

the theoretical convergence of ADM can be similarly guaranteed [70]. More results and analysis applying ADM to convex programming and variational inequalities can be found, for example, in [72, 73, 74].

ADM can potentially reduce the iteration-complexity of the algorithm by solving two simpler subproblems at each iteration, instead of directly minimizing the augmented Lagrangian function (2.9). In fact, under the assumption that f_2 is linear, Gabay and Mercier [59] also proved the convergence of ADM for

$$0 < \varsigma^k = \varsigma < 2.$$

However, the linear assumption is quite strict and most problems stemmed from signal processing or sparse optimization do not fall into this category.

Even though ADM seems more appealing than the classic augmented Lagrangian method, our general TVAL3 algorithm is still founded on the augmented Lagrangian method. First of all, on the problems of our interests ADM appears to be more sensitive to the choice of penalty parameters, whereas the augmented Lagrangian method is more robust. This is advantageous since the observation or data acquired by hardware in the field of signal processing are almost always noisy and a more robust method is favorable. Secondly, ADM requires separability of the objective function into exactly two blocks, and demands high-accuracy minimization for each block. ADM is most efficient if both subproblems can be accurately solved efficiently. However, it is not necessarily the case for the problems we solve in signal processing or sparse optimization. For example, in TV regularized minimization, one of those subproblems is usually quadratic minimization and that dominates the computation. Thus, without special structures, it can be too expensive to find a high-accuracy minimizer at each iteration. The general TVAL3 algorithm considered in this chapter handles the quadratic subproblems in an inexact manner (one aggressive step along the descent direction). The convergence of the general TVAL3 algorithm, founded on the framework of the augmented Lagrangian method, will be proved later in this chapter.

2.2 An Algorithm

A major concern while applying the augmented Lagrangian method for (2.10) is how to efficiently solve a series of unconstraint subproblems. Here we propose an alternating direction type method for minimizing the type of functions in (2.10).

2.2.1 Descriptions

Suppose $g : \mathbb{R}^n \to \mathbb{R}$ is continuous and bounded below, and has the following form:

$$g(u) \triangleq g(w, x) = \varphi(w) + \phi(w, x). \tag{2.13}$$

Furthermore, let us assume that ϕ is continuously differentiable and minimizing $g(w, x)$ with respect to w only is easy. Many optimization problems originated in compressive sensing, image denoising, deblurring and impainting fall into this category after introducing appropriate splitting variables and employing the augmented Lagrangian method or other penalty methods. An instance will be given in the next section and further discussions corresponding to this type will be involved in the following chapters.

The goal is to solve

$$\min_{w,x} g(w, x). \tag{2.14}$$

The proposed algorithm is based on an alternating direction scheme, as well as a nonmonotone line search [82] with Barzilai-Borwein [79] steps to accelerate convergence.

The Barzilai-Borwein (BB) method utilizes the previous two iterates to select step length and may achieve superlinear convergence under certain circumstances [79, 80]. For given w_k, applying BB method on minimizing $g(w_k, x)$ with respect to x leads to a step length

$$\bar{\alpha_k} \;=\; \frac{s_k^T s_k}{s_k^T y_k}, \tag{2.15}$$

or alternatively

$$\bar{\alpha_k} \;=\; \frac{s_k^T y_k}{y_k^T y_k}, \tag{2.16}$$

where $s_k = x_k - x_{k-1}$ and $y_k = \nabla_x g(w_k, x_k)^T - \nabla_x g(w_k, x_{k-1})^T$ (assuming g is differentiable w.r.t. x).

Starting with a BB step in (2.15) or (2.16), we utilize a nonmonotone line search algorithm (NLSA) to ensure convergence. The NLSA is an improved version of the Grippo, Lampariello and Lucidi nonmonotone line search [81]. Zhang and Hager [82] showed that the scheme was generally superior to previous schemes with either nonmonotone or monotone line search techniques, based on extensive numerical experiments. At each iteration, NLSA requires checking the so-called *nonmonotone Armijo condition*, which is

$$g(w_k, x_k + \alpha_k d_k) \le C_k + \delta \alpha_k \nabla_x g(w_k, x_k) d_k \tag{2.17}$$

where d_k is a descent direction and C_k is a weighted average of function values. More specifically, the algorithmic framework can be depicted as follows:

Algorithm 2.2.1 (Nonmonotone Alternating Direction).

```
Initialize $\zeta > 0$, $0 < \delta < 1 < \rho$, $0 \le \eta_{min} \le \eta_{max} \le 1$, tolerance tol,
    and starting points $w_0$, $x_0$. Set $Q_0 = 1$ and $C_0 = g(w_0, x_0)$.
While $\|\tilde{\nabla} g(w_k, x_k)\| > tol$ Do
    Let $d_k$ be a descent direction of $g(w_k, x)$ at $x_k$;
    Choose $\alpha_k = \bar{\alpha_k}\rho^{\theta_k}$ where $\bar{\alpha_k} > 0$ is the BB step and $\theta_k$ is the largest integer
        such that both the nonmonotone Armijo condition (2.17) and $\alpha_k \le \zeta$ hold;
    Set $x^{k+1} = x^k + \alpha_k d_k$;
    Choose $\eta_k \in [\eta_{min}, \eta_{max}]$ and set
        $Q_{k+1} = \eta_k Q_k + 1$, $C_{k+1} = (\eta_k Q_k C_k + g(w_k, x_{k+1}))/Q_{k+1}$;
    Set $w_{k+1} = \text{argmin}_w\, g(w, x_{k+1})$.
End Do
```

The nonmonotone Armijo condition could also been substituted by the nonmonotone Wolf conditions [82]. The choice of η_k controls the degree of nonmonotonicity. Specifically, if $\eta_k = 0$ for all k, the line search is monotone; if $\eta_k = 1$ for all k, C_k is the average value of objective function at (w_i, x_i) for $i = 1, 2, \ldots, k$. Therefore, the bigger η_k is, the more nonmonotone the scheme becomes. Besides, θ_k is not necessary to be positive. In practical implementations, starting from the BB step, we could increase or decrease the step length by forward or backward tracking until the nonmonotone Armijo condition satisfies.

Although Algorithm 2.2.1 takes a form of alternating direction method, it treats the two directions quite differently. One direction can be regarded as an "easy" direction, another a "hard" one. The proposed algorithm deviates from the two common alternating direction strategies: the classic alternating minimization or the popular block coordinate descent technique. Unlike the former, it does not require minimization of the objective function in the hard direction; and unlike the latter, it does not ask for a descent of function value at each iteration. This feature allows the algorithm to have inexpensive iterations and to take relatively large steps, while still possessing a convergence guarantee as will be shown. Indeed, computational evidence shows that this feature helps enhance the practical efficiency of the algorithm in a number of applications described later in this thesis.

2.2.2 Convergence Analysis

The convergence proof of Algorithm 2.2.1 has some similarities with the proof of NLSA shown in [82] and both proof follows the same path. However, NLSA only considers continuously differentiable functionals using gradient methods whereas Algorithm 2.2.1 takes into account non-differentiability of the objective function under the framework of alternating direction. For notational simplicity, define

$$g_k(\cdot) \quad \triangleq \quad g(w_k, \cdot). \tag{2.18}$$

The convergence proof requires the following two assumptions:

Assumption 2.2.1 (Direction Assumption)**.** *There exist $c_1 > 0$ and $c_2 > 0$ such that*

$$\begin{cases} \nabla g_k(x_k) d_k \leq -c_1 \|\nabla g_k(x_k)\|^2, \\ \| \, d_k \, \| \leq c_2 \|\nabla g_k(x_k)\|. \end{cases} \tag{2.19}$$

Assumption 2.2.2 (Lipschitz Condition)**.** *There exists $L > 0$, such that for any given x, $\tilde{x}$, and w,*

$$\|\nabla_x g(w, x) - \nabla_x g(w, \tilde{x})\| = \|\nabla_x \phi(w, x) - \nabla_x \phi(w, \tilde{x})\| \leq L\|x - \tilde{x}\|. \tag{2.20}$$

The direction assumption obviously holds if

$$d_k = -\nabla g_k(x_k)^T.$$

This choice leads to the simple steepest-descent step in Algorithm 2.2.1. The Lipschitz condition is widely assumed in the analysis of convergence of gradient methods. In this sense, Assumptions 2.2.1 and 2.2.2 are both reasonable.

To start with, the following lemma presents some basic properties and suggests the algorithm is well-defined.

Lemma 2.2.1. *If $\nabla g_k(x_k)d_k \leq 0$ holds for each k, then for the sequences generated by Algorithm 2.2.1, we have $g_k(x_k) \leq g_{k-1}(x_k) \leq C_k$ for each k and $\{C_k\}$ is monotone non-increasing. Moreover, if $\nabla g_k(x_k)d_k < 0$, step length $\alpha_k > 0$ always exists.*

Proof. Define real-valued function

$$D_k(t) = \frac{tC_{k-1} + g_{k-1}(x_k)}{t+1} \qquad \text{for } t \geq 0,$$

then

$$D_k'(t) = \frac{C_{k-1} - g_{k-1}(x_k)}{(t+1)^2} \qquad \text{for } t \geq 0.$$

Due to the nonmonotone Armijo condition (2.17) and $\nabla g_k(x_k)d_k \leq 0$, we have

$$C_{k-1} - g_{k-1}(x_k) \geq -\delta\alpha_{k-1}\nabla g_{k-1}(x_{k-1})d_{k-1} \geq 0.$$

Therefore, $D_k'(t) \geq 0$ holds for any $t \geq 0$, and then D_k is non-decreasing.
Since

$$D_k(0) = g_{k-1}(x_k) \quad \text{and} \quad D_k(\eta_{k-1}Q_{k-1}) = C_k,$$

we have

$$g_{k-1}(x_k) \le C_k \quad \text{for any } k.$$

As being described in Algorithm 2.2.1,

$$w_k = \operatorname*{argmin}_w g(w, x_k),$$

then we have

$$g(w_k, x_k) \le g(w_{k-1}, x_k).$$

Hence, $g_k(x_k) \le g_{k-1}(x_k) \le C_k$ holds for any k.

Furthermore,

$$C_{k+1} = \frac{(\eta_k Q_k C_k + g_k(x_{k+1}))}{Q_{k+1}} \le \frac{(\eta_k Q_k C_k + C_{k+1})}{Q_{k+1}},$$

i.e.,

$$(\eta_k Q_k + 1) C_{k+1} \le (\eta_k Q_k C_k + C_{k+1}),$$

i.e.,

$$C_{k+1} \le C_k.$$

Thus, $\{C_k\}$ is monotone non-increasing.

If C_k is replaced by $g_k(x_k)$ in (2.17), the nonmonotone Armijo condition becomes the standard Armijo condition. It is well-known that $\alpha_k > 0$ exists for the standard Armijo condition while $\nabla g_k(x_k) d_k < 0$ and g is bounded below (see [83] for example). Since $g_k(x_k) \le C_k$, it follows $\alpha_k > 0$ exists as well for the nonmonotone Armijo condition (2.17). $\square$

Defining A_k recursively by

$$A_k = \frac{1}{k+1}\sum_{i=0}^{k} g_k(x_k), \tag{2.21}$$

then by induction, it is easy to show that C_k is bounded above by A_k. Together with the facts that C_k is also bounded below by $g_k(x_k)$ and $\alpha_k > 0$ always exists, it is sufficient to claim that Algorithm 2.2.1 is well-defined.

In the next lemma, the lower bound of the step length generated by Algorithm 2.2.1 will be given in accordance with the final convergence proof.

Lemma 2.2.2. *Assuming $\nabla g_k(x_k)d_k \leq 0$ for any k and Lipschitz condition* (2.20) *holds with constant L, then*

$$\alpha_k \geq \min\left\{\frac{\zeta}{\rho}, \frac{2(1-\delta)}{L\rho}\frac{|\nabla g_k(x_k)d_k|}{\|d_k\|^2}\right\}. \tag{2.22}$$

Proof. It is noteworthy that $\rho > 1$ is required in Algorithm 2.2.1. If $\rho\alpha_k \geq \zeta$, then the lemma already holds.

Otherwise,

$$\rho\alpha_k = \bar{\alpha_k}\rho^{\theta_k+1} < \zeta,$$

which indicates that θ_k is not the largest integer to make the k-th step length less than ζ. According to Algorithm 2.2.1, θ_k must be the largest integer satisfying the nonmonotone Armijo condition (2.17), which leads to

$$g_k(x_k + \rho\alpha_k d_k) \geq C_k + \delta\rho\alpha_k\nabla g_k(x_k)d_k.$$

Lemma 2.2.1 showed $C_k \geq g_k(x_k)$, so

$$g_k(x_k + \rho\alpha_k d_k) \;\; \geq \;\; g_k(x_k) + \delta\rho\alpha_k \nabla g_k(x_k) d_k. \tag{2.23}$$

On the other hand, for $\alpha > 0$ we have

$$\int_0^\alpha \left(\nabla g_k(x_k + td_k) - \nabla g_k(x_k)\right) d_k \, dt = g_k(x_k + \alpha d_k) - g_k(x_k) - \alpha \nabla g_k(x_k) d_k.$$

Together with the Lipschitz condition, we get

$$\begin{aligned} g_k(x_k + \alpha d_k) &= g_k(x_k) + \alpha \nabla g_k(x_k) d_k + \int_0^\alpha \left(\nabla g_k(x_k + td_k) - \nabla g_k(x_k)\right) d_k \, dt \\ &\leq g_k(x_k) + \alpha \nabla g_k(x_k) d_k + \int_0^\alpha tL \|d_k\|^2 \, dt \\ &= g_k(x_k) + \alpha \nabla g_k(x_k) d_k + \frac{1}{2} L \alpha^2 \|d_k\|^2. \end{aligned}$$

Let $\alpha = \rho\alpha_k$, which gives

$$g_k(x_k + \rho\alpha_k d_k) \;\; \leq \;\; g_k(x_k) + \rho\alpha_k \nabla g_k(x_k) d_k + \frac{1}{2} L \rho^2 \alpha_k^2 \|d_k\|^2. \tag{2.24}$$

Compare (2.23) with (2.24), which implies

$$(\delta - 1) \nabla g_k(x_k) d_k \leq \frac{1}{2} L \rho \alpha_k \|d_k\|^2.$$

Since $\nabla g_k(x_k) d_k \leq 0$,

$$\alpha_k \geq \frac{2(1-\delta)}{L\rho} \frac{|\nabla g_k(x_k) d_k|}{\|d_k\|^2}.$$

Therefore, the step length α_k is bounded below by (2.22). □

With the aid of the above lower bound, we are able to establish the convergence of Algorithm 2.2.1:

Theorem 2.2.1 (Optimality Conditions)**.** *Suppose g is bounded below and both direction assumption* (2.19) *and Lipschitz condition* (2.20) *hold. Then the iterates $u_k \triangleq (w_k, x_k)$ generated by Algorithm 2.2.1 satisfies*

$$\lim_{k\to 0} \tilde{\nabla} g(u_k) = 0. \tag{2.25}$$

Proof. Since g is differentiable with respect to x, (2.25) is equivalent to

$$\begin{cases} \lim\limits_{k\to 0} \tilde{\nabla}_w g(w_k, x_k) = 0, \\ \lim\limits_{k\to 0} \nabla_x g(w_k, x_k) = 0. \end{cases} \tag{2.26}$$

The proof can be completed by showing two parts respectively.

First, due to the nature of Algorithm 2.2.1,

$$w_k = \operatorname*{argmin}_w g(w, x_k).$$

Then

$$0 \in \partial_w g(w_k, x_k),$$

which implies

$$\tilde{\nabla}_w g(w_k, x_k) = 0.$$

Next, let us show the second half grounded on the nonmonotone Armijo condition

$$g_k(x_k + \alpha_k d_k) \quad \leq \quad C_k + \delta \alpha_k \nabla g_k(x_k) d_k. \tag{2.27}$$

If $\rho\alpha_k < \zeta$, according to the lower bound of α_k given by Lemma 2.2.2 and direction assumption (2.19), we have

$$\begin{aligned} g_k(x_k+\alpha_k d_k) &\leq C_k - \delta\frac{2(1-\delta)}{L\rho}\frac{|\nabla g_k(x_k)d_k|^2}{\|d_k\|^2} \\ &\leq C_k - \frac{2\delta(1-\delta)}{L\rho}\frac{c_1^2\|\nabla g_k(x_k)\|^4}{c_2^2\|\nabla g_k(x_k)\|^2} \\ &= C_k - \left[\frac{2\delta(1-\delta)c_1^2}{L\rho c_2^2}\right]\|\nabla g_k(x_k)\|^2. \end{aligned}$$

On the other hand, if $\rho\alpha_k \geq \zeta$, this lower bound together with direction assumption (2.19) gives

$$\begin{aligned} g_k(x_k+\alpha_k d_k) &\leq C_k + \delta\alpha_k\nabla g_k(x_k)d_k \\ &\leq C_k - \delta\alpha_k c_1\|\nabla g_k(x_k)\|^2 \\ &\leq C_k - \frac{\delta\zeta c_1}{\rho}\|\nabla g_k(x_k)\|^2. \end{aligned}$$

Define constant

$$\tilde{\tau} = \min\left\{\frac{2\delta(1-\delta)c_1^2}{L\rho c_2^2}, \frac{\delta\zeta c_1}{\rho}\right\},$$

which leads to

$$g_k(x_k+\alpha_k d_k) \leq C_k - \tilde{\tau}\|\nabla g_k(x_k)\|^2. \tag{2.28}$$

Next we show that

$$\frac{1}{Q_k} \geq 1-\eta_{max}. \tag{2.29}$$

Obviously it follows $Q_0 = 1$ that

$$\frac{1}{Q_0} \geq 1-\eta_{max}.$$

Assuming that (2.29) also holds for $k = j$, then

$$\begin{aligned} Q_{j+1} &= \eta_j Q_j + 1 \\ &\leq \frac{\eta_j}{1-\eta_{max}} + 1 \\ &\leq \frac{\eta_{max}}{1-\eta_{max}} + 1 \\ &= \frac{1}{1-\eta_{max}}, \end{aligned}$$

which implies

$$\frac{1}{Q_{j+1}} \geq 1 - \eta_{max}.$$

By induction, we conclude that (2.29) holds for all k.

Thus, it follows from (2.28) and (2.29) that

$$\begin{aligned} C_k - C_{k+1} &= C_k - \frac{\eta_k Q_k C_k + g_k(x_{k+1})}{Q_{k+1}} \\ &= \frac{C_k(\eta_k Q_k + 1) - (\eta_k Q_k C_k + g_k(x_{k+1}))}{Q_{k+1}} \\ &= \frac{C_k - g_k(x_{k+1})}{Q_{k+1}} \\ &\geq \frac{\tilde{\tau}\|\nabla g_k(x_k)\|^2}{Q_{k+1}} \\ &\geq \tilde{\tau}(1-\eta_{max})\|\nabla g_k(x_k)\|^2. \end{aligned} \tag{2.30}$$

Since g is bounded below, $\{C_k\}$ is also bounded below. Besides, Lemma 2.2.1 illustrates $\{C_k\}$ is monotone non-increasing, so there exists $C^* \in \mathbb{R}$ such that

$$C_k \to C^*, \text{ as } k \to \infty.$$

Hence, we have

$$C_k - C_{k+1} \to 0, \text{ as } k \to \infty.$$

Combining this and (2.30), we get

$$\|\nabla g_k(x_k)\| \to 0;$$

i.e.,

$$\lim_{k\to 0} \tilde{\nabla}_x g(w_k, x_k) = 0.$$

Coupling two parts completes the proof of this theorem. □

With the aid of Theorem 2.2.1, we can further conclude the global convergence of Algorithm 2.2.1 under the assumption of strong convexity.

Corollary 2.2.1. *If g is jointly and strongly convex, then under the same assumptions as in Theorem 2.2.1, sequence (w_k, x_k) generated by Algorithm 2.2.1 converges to the unique minimizer (w^*, x^*) of unconstraint problem* (2.13)*.*

The proof is omitted here since it directly follows Theorem 2.2.1.

By this time, we have proposed an alternating direction type method with a non-monotone line search for a special class of unconstraint minimization problems, and fulfilled descriptions by thoroughly studying the convergence. TVAL3 — a combination of this algorithm and the classic augmented Lagrangian method — aiming at solving a more general class of both constraint and unconstraint problems will be depicted next.

2.3 General TVAL3 and One Instance

The general TVAL3 algorithm is built by means of a combination of the classic augmented Lagrangian method with an appropriate variable splitting (see Algorithm

2.1.1) and nonmonotone alternating direction method for subproblems (see Algorithm 2.2.1). More precisely, it implements the following algorithmic framework after variable splitting:

Algorithm 2.3.1 (General TVAL3)**.**

```
*Initialization.*
**While** $\|\tilde{\nabla}\mathcal{L}(x^k, \lambda^k)\| > tol$ **Do**
    *Set starting points* $w_0^{k+1} = w^k$ *and* $x_0^{k+1} = x^k$ *for the subproblem;*
    *Find minimizer* $w^{k+1}$ *and* $x^{k+1}$ *of* $\mathcal{L}_A(w, x, \lambda^k; \mu^k)$ *using Algorithm 2.2.1;*
    *Update the multiplier using* (2.11) *and non-decrease the penalty parameter;*
**End Do**
```

In fact, the purpose of variable splitting is to separate the non-differentiable part in order to easily find its closed-form solution while applying the general TVAL3 algorithm. In other words, the original non-differentiable problem is divided into two parts — separable non-differentiable part with explicit solution and differentiable part requiring heavy computation.

From previous analysis, the convergence of this method follows immediately. Theorem 2.1.2 has ensured the convergence of outer loop while Theorem 2.2.1 has provided the convergence of inner loop, which together indicates the convergence of the general TVAL3 method. The convergence rate is not deepened since it is not necessarily related to the practical efficiency of methods or algorithms. The convergence rate analyzes the relation between error and number of iterations, but neglects the complexity of each iteration. In the real world, the real cost relies on the multiplication of both. One advantage of the general TVAL3 method is its low cost at each iteration. Mostly it requires only two or three matrix-vector multiplications to process one inner iteration, which results in the significant decrease on overall computation.

2.3.1 Application to 2D TV Minimization

One instance is for solving the compressive sensing problem with total variation (TV) regularization:

$$\min_u TV(u) \triangleq \sum_i \|D_i u\|, \quad \text{s.t. } Au = b, \tag{2.31}$$

where $u \in \mathbb{R}^n$ or $u \in \mathbb{R}^{s\times t}$ with $s \cdot t = n$, $D_i u \in \mathbb{R}^2$ is the discrete gradient of u at pixel i, $A \in \mathbb{R}^{m\times n}$ $(m < n)$ is the measurement matrix, and $b \in \mathbb{R}^m$ is the observation of u via some linear measurements. The regularization term is called *isotropic* TV. If $\|.\|$ is replaced by 1-norm, then it is called *anisotropic* TV. With minor modifications, the following derivation for solving (2.31) is applicable for anisotropic TV as well.

In light of variable splitting, an equivalent variant of (2.31) is considered:

$$\min_{w_i,u} \sum_i \|w_i\|, \quad \text{s.t. } Au = b \text{ and } \ D_i u = w_i \text{ for all } i. \tag{2.32}$$

Its corresponding augmented Lagrangian function is

$$\begin{aligned}\mathcal{L}_A(w_i, u) \ &= \ \sum_i (\|w_i\| - \nu_i^T (D_i u - w_i) + \frac{\beta_i}{2}\|D_i u - w_i\|^2) \\ &\quad -\lambda^T (Au - b) + \frac{\mu}{2}\|Au - b\|^2,\end{aligned} \tag{2.33}$$

and then the subproblem at each iteration of TVAL3 becomes

$$\min_{w_i,u} \mathcal{L}_A(w_i, u). \tag{2.34}$$

At the k-th iteration, solving (2.34) with respect to w_i gives a closed-form solution since it is separable; i.e.,

$$w_{i,k+1} = \max\left\{\left\|D_i u_k - \frac{\nu_i}{\beta_i}\right\| - \frac{1}{\beta_i}, 0\right\} \frac{(D_i u_k - \nu_i/\beta_i)}{\|D_i u_k - \nu_i/\beta_i\|}, \tag{2.35}$$

where $0 \cdot (0/0) = 0$ is followed. This formula is used to be called *shrinkage* (see [46] for example). On the other hand, (2.33) is quadratic with respect to u and its gradient can be easily derived as

$$d_k(u) = \sum_i (\beta_i D_i^T (D_i u - w_{i,k+1}) - D_i^T \nu_i) + \mu A^T (Au - b) - A^T \lambda. \tag{2.36}$$

According to Algorithm 2.2.1, we only require one step of steepest descent with properly adjusted step length; i.e.;

$$u_{k+1} = u_k - \alpha_k d_k(u_k). \tag{2.37}$$

Therefore, the TVAL3 algorithm for TV regularized problems on compressive sensing has been obtained by incorporating (2.35), (2.36) and (2.37) into the general framework of Algorithm 2.3.1.

To demonstrate the efficiency of the TVAL3 implementation, it is compared to other state-of-the-art implementations of TV regularized methods, such as ℓ_1-Magic [2, 3, 5], TwIST [85, 86] and NESTA [84].

Experiments were performed on a Lenovo X301 laptop running Windows XP and MATLAB R2009a (32-bit) and equipped with a 1.4GHz Intel Core 2 Duo SU9400 and 2GB of DDR3 memory.

While running TVAL3, we uniformly set parameters $\eta = .9995$, $\rho = 5/3$, $\delta = 10^{-5}$ and $\zeta = 10^4$ presented in Algorithm 2.2.1, and initialized multipliers to 0 and fixed weights in front of multipliers at 1.6 presented in Algorithm 2.3.1. Additionally, the

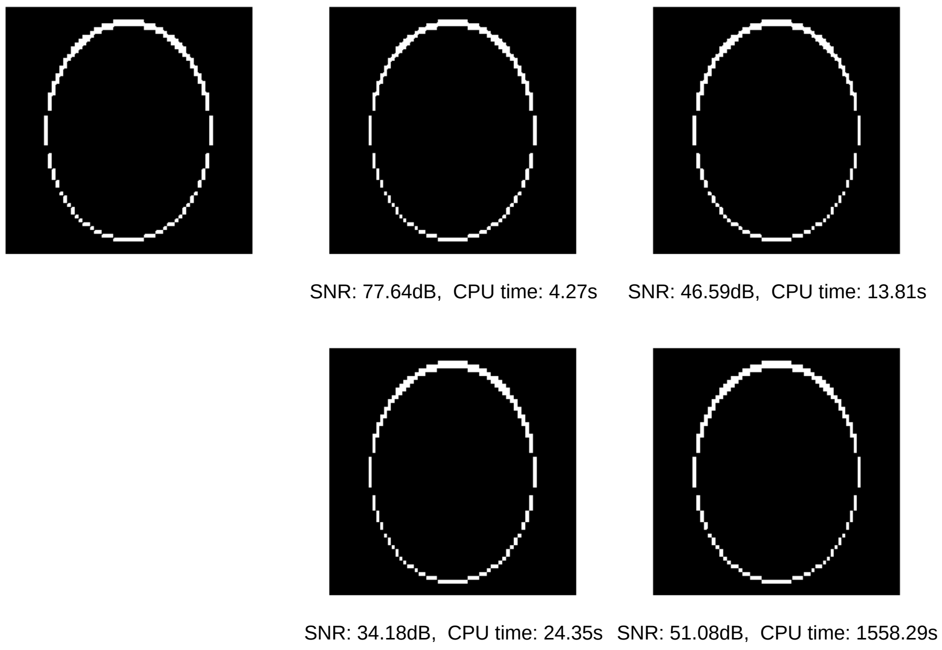


Figure 2.1: Recovered 64×64 phantom image from 30% orthonormal measurements without noise. **Top-left:** original image. **Top-middle:** reconstructed by TVAL3. **Top-right:** reconstructed by TwIST. **Bottom-middle:** reconstructed by NESTA. **Bottom-right:** reconstructed by ℓ_1-Magic.

values of penalty parameters might vary in a range of 2^5 to 2^9 according to distinct noise level and required accuracy.

In an effort to make comparisons fair, for other tested solvers mentioned above, we did tune their parameters and try to make them perform optimal or near optimal.

In the first test, a 64×64 phantom image is encoded by an orthonormal random matrix generated by QR factorization from a Gaussian random matrix. The images are recovered by TVAL3, TwIST, NESTA and ℓ_1-Magic respectively from 30% measurements without the additive noise. The quality of recovered images is measured by the signal-to-noise ratio (SNR), which is defined as the power ratio between a signal and the background noise. All parameters are tuned to achieve the best performance.

From Figure 2.1, we observe that TVAL3 achieves the highest-quality image

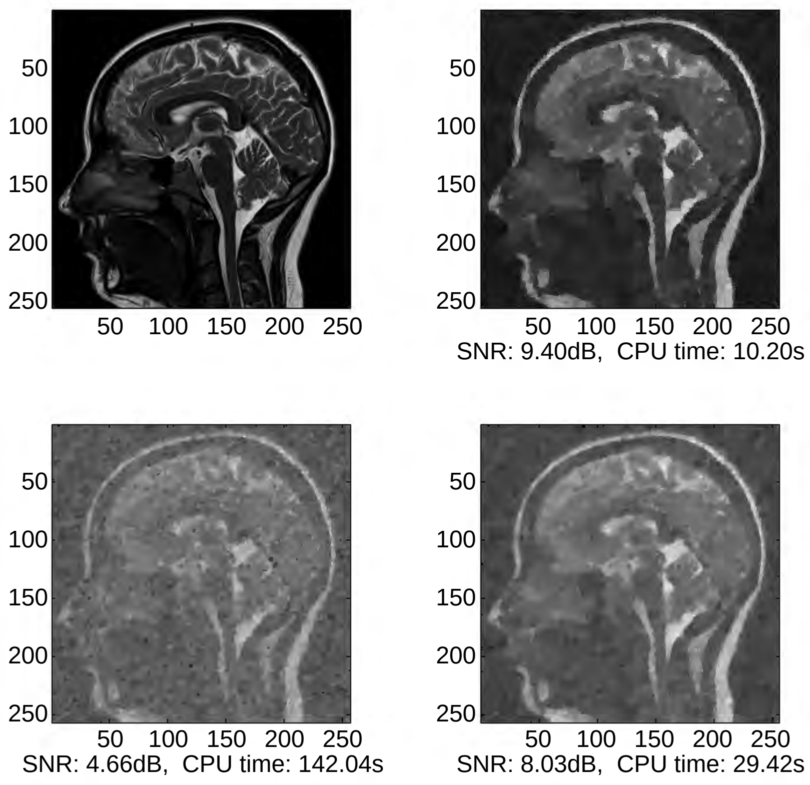

50
100
150
200
250
50 100 150 200 250
SNR: 9.40dB, CPU time: 10.20s
SNR: 4.66dB, CPU time: 142.04s
SNR: 8.03dB, CPU time: 29.42s

Figure 2.2: Recovered 256 × 256 MR brain image. Both the measurement rate and the noise level are 10%. **Top-left:** original image. **Top-right:** reconstructed by TVAL3. **Bottom-left:** reconstructed by TwIST. **Bottom-right:** reconstructed by NESTA.

(77.64dB) but requires the shortest running time (4.27 seconds). The second highest-quality image (51.08dB) is recovered by ℓ_1-Magic at the expense of the unacceptable running time (1558.29 seconds). TwIST and NESTA attain relatively medium-quality images (around 46.59dB and 34.18dB respectively) within reasonable running times (13.81 and 24.35 seconds respectively). This test suggests that TVAL3 is capable of high accuracy within an affordable running time, and outperforms other state-of-the-art implementations more or less.

Noise is inevitable in practice. The following test focuses on the performance of different implementations under the influence of Gaussian noise. Specifically, a 256 ×

256 MR brain image, which contains much more details than phantom, is encoded by a permutated sequency-ordered Walsh Hadamard matrix using fast transform. In order to investigate the robustness, we choose both noise level and measurement rate to be 10%. The above phantom test has indicated the ℓ_1-Magic is hardly applicable to large-scale problems due to its low efficiency, so only TVAL3, TwIST and NESTA are performed here.

From Figure 2.2, we can only recognize vague outline of the image recovered by TwIST even though the running time is longest. Nevertheless, the image recovered by either TVAL3 or NESTA is more subtle and preserves more details contained in the original brain image. In comparison with NESTA, TVAL3 achieves better accuracy (higher SNR) in shorter running time statistically, and provides higher contrast visually. For example, some gyri in the image recovered by TVAL3 are still distinguishable but this is not the case in images recovered by either TwIST or NESTA. Furthermore, the image recovered by NESTA is still noisy while the image recovered by TVAL3 is much cleaner. This implies the fact that TVAL3 is capable of better denoising effects than NESTA. Actually, this would be a desirable property when handling data with lots of noise, which will always be the case in practice.

Two tests are far less than enough to draw a solid conclusion. More numerical experiments and analysis with different flavors have been covered in [9], which revealed the comprehensive performance of TVAL3 on TV regularized problems.

With moderate modifications, TVAL3 is easily to extend to some other TV regularized models with extra requirements, for example, imposing nonnegativity constraints or dealing with complex signals/measurements. For the convenience of other researchers, it has been implemented in MATLAB aiming at solving various TV regularized models in the field of compressive sensing, and published at the following URL:

`http://www.caam.rice.edu/~optimization/L1/TVAL3/`.

Chapter 3

Hyperspectral Data Unmixing

In this chapter, we develop a hyperspectral unmixing scheme with the aid of compressive sensing. This scheme could recover the abundance and signatures straightly from the compressed data instead of the whole massive hyperspectral cube. In light of the general TVAL3 method discussed in Chapter 2, a effective and robust reconstruction algorithm is proposed and conscientiously investigated.

3.1 Introduction to Hyperspectral Imaging

By exploiting the wavelength composition of electromagnetic radiation (EMR), hyperspectral imaging collects and processes data from across the electromagnetic spectrum. Hyperspectral sensors capture information as a series of "images" over many contiguous spectral bands containing the visible, near-infrared and shortwave infrared spectral bands [98]. These images, generated from different bands, pile up and form a 3D hyperspectral cube for processing and further analysis. If each image can be viewed as a long vector, the hyperspectral cube will become a large matrix which is more easily accessible mathematically. Each column of the matrix records the information from the same spectral band and each row records the information at the same pixel. For much of the past decade, hyperspectral imaging has been actively researched and widely developed. It has matured into one of the most powerful and fast growing technologies. For example, the development of hyperspectral sensors and their corresponding software to analyze hyperspectral data has been regarded as a critical breakthrough in the field of remote sensing. Hyperspectral imaging has a wide range of applications in industry, agriculture and military, such as terrain classification, mineral detection and exploration [87, 88], pharmaceutical counterfeiting [89], environmental monitoring [91] and military surveillance [90].

The fundamental property of hyperspectral imaging which researchers want to obtain is *spectral reflectance*: the ratio of reflected energy to incident energy as a function of wavelength [97]. Reflectance varies with wavelength for most materials. These variations are evident and sometimes characteristic when comparing these spectral reflectance plots of different materials. Several libraries of reflectance spectra of natural and man-made materials are accessible for public use, such as ASTER Spectral Library [122] and USGS Spectral Library [123]. These libraries provide a source of reference spectra that helps the interpretation and analysis of hyperspectral images.

It is highly possible that more than one material contributes to an individual spectrum captured by the sensor, which leads to a composite or mixed spectrum. Typically, hyperspectral imaging is of spatially low resolution, in which each pixel, from a given spatial element of resolution and at a given spectral band, is a mixture of several different material substances, termed *endmembers*, each possessing a characteristic *hyperspectral signature* [99]. In general, endmembers imply those spectrally "pure" features, such as soil, vegetation, and so forth. In mineralogy, it refers to a mineral at the extreme end of a mineral series in terms of purity. For example, albite ($NaAlSi_3O_8$) and anorthite ($CaAl_2Si_2O_8$) are two endmembers in the plagioclase series of minerals.

If the endmember spectra or signatures are available beforehand, we can mathematically decompose each pixel's spectrum of a hyperspectral image to identify the relative abundance of each endmember component. This process is called *unmixing*. Linear unmixing is a simple spectral matching approach, whose underlying premise is that a relatively small number of common endmembers are involved in a scene, and most spectral variability in this scene can be attributed to spatial mixing of these endmember components in distinct proportions. In the linear model, interactions among distinct endmembers are assumed to be negligible [100], which is a plausible hypothesis in the realm of hyperspectral imaging. Frequently, the representative endmembers for a given scene are known *a priori* and their signatures can be obtained from a spectral library (e.g., ASTER [122] and USGS [123]) or codebook. On the other hand, when endmembers are unknown but the hyperspectral data is fully accessible, many algorithms exist for determining endmembers in a scene, including N-FINDR [102], PPI (pixel purity index) [101], VCA (vertex component analysis) [103], SGA (simplex growing algorithm) [104]; NMF-MVT (nonnegative matrix factorization minimum volume transform) [105], SISAL (simplex identification via split augmented Lagrangian) [106], MVSA (minimum volume simplex analysis) [108] and MVES (minimum-volume enclosing simplex) [107].

Because of the their enormous volume, it is particularly difficult to directly process and analyze hyperspectral data cubes in real time or near real time. On the other hand, hyperspectral data are highly compressible with two-fold compressibility:

1. each spatial image is compressible, and

2. the entire cube, when treated as a matrix, is of low rank.

To fully exploit such rich compressibility, a scheme is proposed in this chapter, which never requires to explicitly store or process a hyperspectral cube itself. In this scheme, data are acquired by means of compressive sensing (CS). As introduced in Chapter 1, the theory of CS shows that a sparse or compressible signal can be recovered from a relatively small number of linear measurements. In particular, the concept of the single pixel camera [32] can be extended to the acquisition of compressed hyperspectral data, which will be described and used while setting up the experiments. The main novelty of the scheme is in the decoding side where we combine data reconstruction and unmixing into a single step of much lower complexity. The proposed scheme is both computationally low-cost and memory-efficient. At this point, we start from the assumption that the involved endmember signatures are known and given, from which we then directly compute abundance fractions. For brevity, we will call the proposed procedure *compressive sensing and unmixing* or CSU scheme.

In fact, a prior information is not always accessible or precise. For example, the change of experimental environment may cause fluctuation of endmember reflectance and give rise to a slightly different signature from the one in the standard library. Without the aid of correct or complete a priori, the unmixing problem will become significantly more intractable. Later in this chapter, the CSU scheme is extended to *blind unmixing* where endmember signatures are not precisely known a priori.

3.2 Compressive Sensing and Unmixing Scheme

In this section, we propose and conduct a proof-of-concept study on a low-complexity, compressive sensing and unmixing (CSU) scheme, formulating a unmixing model based on total variation (TV) minimization, and developing an efficient algorithm to solve this model [109]. To validate the CSU scheme, experimental and numerical evidence will be provided in the next section. This proposed scheme directly unmixes compressively sensed data, bypassing the high-complexity step of reconstructing the hyperspectral cube itself. The effectiveness and efficiency of the proposed CSU scheme are demonstrates using both synthetic and hardware-measured data.

3.2.1 Problem Formulation

Let us introduce those necessary notations first. Suppose that in a given scene there exist n_e significant endmembers, with spectral signatures $w_i^T \in \mathbb{R}^{n_b}$, for $i = 1, \ldots, n_e$, where $n_b \geq n_e$ denotes the number of spectral bands. Let $x_i \in \mathbb{R}^{n_b}$ represent the hyperspectral data vector at the i-th pixel and $h_i^T \in \mathbb{R}^{n_e}$ represent the abundance fractions of the endmembers for any $i \in \{1, \ldots, n_p\}$, where n_p denotes the number of pixels. Furthermore, let $X = [x_1, \ldots, x_{n_p}]^T \in \mathbb{R}^{n_p \times n_b}$ denote a matrix representing the hyperspectral cube, $W = [w_1, \ldots, w_{n_e}]^T \in \mathbb{R}^{n_e \times n_b}$ the mixing matrix containing the endmember spectral signatures, and $H = [h_1, \ldots, h_{n_p}]^T \in \mathbb{R}^{n_p \times n_e}$ a matrix holding the respective abundance fractions. We use $A \in \mathbb{R}^{m \times n_p}$ to denote the measurement matrix in compressive sensing data acquisition, and $F \in \mathbb{R}^{m \times n_b}$ to denote the observation matrix, where $m < n_p$ is the number of samples for each spectral band. For convenience, $\mathbf{1}_s$ denotes the column vector of all ones with length s. In addition, we use $\langle \cdot, \cdot \rangle$ to denote the usual matrix inner product since the notation $(\cdot)^T(\cdot)$ for vector inner product would not correctly apply.

Assuming negligible interactions among endmembers, the hyperspectral vector x_i at the i-th pixel can be regarded as a linear combination of the endmember spectral signatures, and the weights are gathered in a nonnegative abundance vector h_i. Ideally, the components of h_i, representing abundance fractions, should sum up to unity; i.e., the hyperspectral vectors lie in the convex hull of endmember spectral signatures

[103]. In short, the data model has the form

$$X = HW, \qquad H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}, \qquad \text{and} \qquad H \geq 0. \tag{3.1}$$

However, in reality the sum-to-unity condition on H does not usually hold due to imprecisions and noise of various kinds. In our implementation, we imposed this condition on synthetic data, but skipped it for measured data.

Since each column of X represents a 2D image corresponding to a particular spectral band, we can collect the compressed hyperspectral data $F \in \mathbb{R}^{m \times n_b}$ by randomly sampling all the columns of X using the same measurement matrix $A \in \mathbb{R}^{m \times n_p}$, where $m < n_p$ is the number of samples for each column. Mathematically, the data acquisition model can be described as

$$AX = F. \tag{3.2}$$

Combining (3.1) and (3.2), we obtain constraints

$$AHW = F, \qquad H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}, \qquad \text{and} \qquad H \geq 0. \tag{3.3}$$

For now, we assume that the endmember spectral signatures in W are known, our goal is to find their abundance distributions (or fractions) in H, given the measurement matrix A and the compressed hyperspectral data F. In general, system (3.3) is not sufficient for determining H, necessitating the use of some prior knowledge about H in order to find it.

In compressive sensing, regularization by ℓ_1 minimization has been widely used. However, Chapter 1 has suggested shown that the use of TV regularization is empirically more advantageous on image problems such as deblurring, denoising and reconstruction, since it can better preserve edges or boundaries in images that are essential characteristics. TV regularization puts emphasis on sparsity in the gradient map of the image and is suitable when the gradient of the underlying image is sparse [2]. In our case, we make the assumption that the gradient of each image composed by abundance fractions for each endmember is mostly and approximately piecewise constant. This is reasonable in the sense that most applications of hyperspectral imaging focus on characteristics (or simply described as jumps) in a scenario instead of those smooth parts. Mathematically, we propose to recover the abundance matrix H by solving the following unmixing model:

$$\min_{H \in \mathbb{R}^{n_p \times n_e}} \sum_{j=1}^{n_e} \mathrm{TV}(He_j) \quad \text{s.t.} \quad AHW = F, \; H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}, \; H \geq 0, \tag{3.4}$$

where e_j is the j-th standard unit vector in $\mathbb{R}^{n_p}$,

$$\mathrm{TV}(He_j) \triangleq \sum_{i=1}^{n_p} \|D_i(He_j)\|, \tag{3.5}$$

$\|.\|$ is the 2-norm in $\mathbb{R}^2$ corresponding to the isotropic TV, and $D_i \in \mathbb{R}^{2\times n_p}$ denotes the discrete gradient operator at the i-th pixel, as described in Chapter 2. In stead of 2-norm, 1-norm is also applicable here corresponding to the anisotropic TV, which arouses quite similar analysis and derivation. Since the unmixing model directly uses compressed data F, we will call it a *compressed unmixing model.*

It is important to note that although H consists of several related images each corresponding to the distribution of abundance fractions of one material in a scene, these images generally do not share many common edges as in color images or some other vector-valued images. For example, a sudden decrease in one fraction can be compensated by an increase in another while all the rest fractions remain unchanged, indicating the occurrence of an edge in two but not all images in H. This phenomenon can be observed from the test cases in Section 3.3. Therefore, in our model (3.4), instead of applying a coupled TV regularization function for vector-valued images (see [17] and [18], for example), we simply use a sum of TV terms for individual scalar-valued images without coupling them in the TV regularization. It is possible that under certain conditions, the use of vector-valued TV is more appropriate, but this point is beyond the scope of this study. Nevertheless, the images in H are connected in the constraint $H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}$.

3.2.2 SVD Preprocessing

The size of the fidelity equation $AHW = F$ in (3.3) is $m \times n_b$ where m, although less than n_p in compressive sensing, can still be quite large, and n_b, the number of spectral bands, typically ranges from hundreds to thousands. Here a preprocessing procedure is proposed based on singular value decomposition of the observation matrix F, in order to decrease the size of the fidelity equations from $m \times n_b$ to $m \times n_e$. Since the number of endmembers n_e is typically up to two orders of magnitude smaller than n_b, the resulting reduction in complexity is significant, potentially enabling near-real-time processing speed. The proposed preprocessing procedure is based on the following result.

Theorem 3.2.1. *Let $A \in \mathbb{R}^{m\times n_p}$ and $W \in \mathbb{R}^{n_e\times n_b}$ be full-rank, and $F \in \mathbb{R}^{m\times n_b}$ be rank-n_e with $n_e < \min\{n_b, n_p, m\}$. Let $F = U_e\Sigma_e V_e^T$ be the economy-size singular value decomposition of F where $\Sigma_e \in \mathbb{R}^{n_e\times n_e}$ is diagonal and positive definite, $U_e \in \mathbb{R}^{m\times n_e}$ and $V_e \in \mathbb{R}^{n_b\times n_e}$ both have orthonormal columns. Assume that rank$(WV_e) = n_e$, then the two linear systems below for $H \in \mathbb{R}^{n_p\times n_e}$ have the same solution set; i.e.,* *the equivalence holds*

$$AHW = F \quad \Longleftrightarrow \quad AHWV_e = U_e\Sigma_e. \tag{3.6}$$

Proof. We show that the two linear system has an identical solution set. Denote the solution sets for the two system by $\mathscr{H}_1 = \{H : \ AHW = F\}$ and $\mathscr{H}_2 = \{H : AHWV_e = U_e\Sigma_e\}$, respectively, which are both subspaces. Given that $F = U_e\Sigma_e V_e^T$ and $V_e^T V_e = I$, it is obvious that $\mathscr{H}_1 \subseteq \mathscr{H}_2$. To show $\mathscr{H}_1 = \mathscr{H}_2$, it suffices to verify that the dimensions of the two are equal, i.e., $\dim(\mathscr{H}_1) = \dim(\mathscr{H}_2)$.

Let "vec" denote the operator that stacks the columns of a matrix to form a vector. By well-known properties of Kronecker product "$\otimes$", $AHW = F$ is equivalent to

$$(W^T \otimes A)\,\mathrm{vec}H = \mathrm{vec}F, \tag{3.7}$$

where $W^T \otimes A \in \mathbb{R}^{(n_b m)\times(n_e n_p)}$, and

$$\mathrm{rank}(W^T \otimes A) = \mathrm{rank}(W)\mathrm{rank}(A) = n_e m. \tag{3.8}$$

Similarly, $AHWV_e = U_e\Sigma_e$ is equivalent to

$$((WV_e)^T \otimes A)\,\mathrm{vec}H = \mathrm{vec}(U_e\Sigma_e), \tag{3.9}$$

where $(WV_e)^T \otimes A \in \mathbb{R}^{(n_e m)\times(n_e n_p)}$ and, under our assumption $\mathrm{rank}(WV_e) = n_e$,

$$\mathrm{rank}((WV_e)^T \otimes A) = \mathrm{rank}(WV_e)\mathrm{rank}(A) = n_e m. \tag{3.10}$$

Hence, $\mathrm{rank}(W^T \otimes A) = \mathrm{rank}((WV_e)^T \otimes A)$, which implies the solution sets of (3.7) and (3.9) have the same dimension; i.e., $\dim(\mathscr{H}_1) = \dim(\mathscr{H}_2)$. Since $\mathscr{H}_1 \subseteq \mathscr{H}_2$, we conclude that $\mathscr{H}_1 = \mathscr{H}_2$. $\square$

This proposition ensures that under a mild condition the matrices W and F in the fidelity equation $AHW = F$ can be replaced, without changing the solution set, by the much smaller matrices WV_e and $U_e\Sigma_e$, respectively, potentially leading to multi-order magnitude reductions in equation size.

Suppose that F is an observation matrix for a rank-n_e hyperspectral data matrix $\hat{X}$. Then $F = A\hat{H}\hat{W}$ for some full rank matrices $\hat{H} \in \mathbb{R}^{n_p \times n_e}$ and $\hat{W} \in \mathbb{R}^{n_e \times n_b}$. Clearly, the rows of $\hat{W}$ span the same space as the columns of V_e do. Therefore, the condition $\mathrm{rank}(WV_e) = n_e$ is equivalent to $\mathrm{rank}(W\hat{W}^T) = n_e$, which definitely holds for $W = \hat{W}$. It will also hold for a random W with high probability. Indeed, the condition $\mathrm{rank}(WV_e) = n_e$ is rather mild.

In practice, the observation matrix F usually contains model imprecisions or random noise, and hence is unlikely to be exactly rank n_e. In this case, truncating the SVD of F to rank-n_e is a sensible strategy, which will not only serve the dimension reduction purpose, but also a denoising purpose because the SVD truncation annihilates insignificant singular values of F likely caused by noise. Motivated by these considerations, the following SVD preprocessing procedure is suggested:

Algorithm 3.2.1 (SVD Preprocessing)**.**

```
Input $F$, $W$ and $n_e$.
Do the following:
    compute the rank-$n_e$ principal SVD: $F \approx U_e\Sigma_e V_e^T$;
    overwrite data: $W \leftarrow WV_e$ and $F \leftarrow U_e\Sigma_e$;
End Do
Output $F$ and $W$.
```

The idea of reduction is inspired by economy SVD decomposition and the solution set keeps the same while right-multiplying V_e on both sides. Then a nature question is followed up: is that possible to further reduce the dimension of the problem by left-multiplying U_e^T since SVD is symmetric in form? If so, observation $F \in \mathbb{R}^{m \times n_b}$ could be substituted for $\Sigma_e \in \mathbb{R}^{n_e \times n_e}$. Unfortunately, the answer is no and the following corollary elaborates the reason.

Corollary 3.2.1. *Suppose the same assumptions hold as Thereom 3.2.1. Then solving $AHW = F$ is not equivalent to solving $U_e^T AHW = \Sigma_e V_e^T$ with respect to H in general.*

Proof. Denote $\mathscr{H}_1 = \{H : \ AHW = F\}$ as before and $\mathscr{H}_3 = \{H : \ U_e^T AHW = \Lambda V_e^T\}$. Similar simple arguments can still give us $\mathscr{H}_1 \subseteq \mathscr{H}_3$, but the other direction would not stand any more.

As indicated in the proof of Theorem 3.2.1, $AHW = F$ is equivalent to

$$(W^T \otimes A)\text{vec}H = \text{vec}F, \tag{3.11}$$

and

$$\text{rank}(W^T \otimes A) = \text{rank}(W)\text{rank}(A) = n_e m.$$

Similarly, $U_e^T AHW = \Lambda V_e^T$ is equivalent to

$$(W^T \otimes (U_e^T A))\text{vec}H = \text{vec}(\Sigma_e V_e^T), \tag{3.12}$$

where $(W^T \otimes (U_e^T A)) \in \mathbb{R}^{(n_b n_e) \times (n_e n_p)}$.

Since both U and A are full-rank and $\text{rank}(U_e) < \text{rank}(A)$, we have

$$\text{rank}(W^T \otimes (U_e^T A)) = \text{rank}(W)\text{rank}(U_e^T A) = \text{rank}(W)\text{rank}(U_e) = n_e^2.$$

It follows $m > n_e$ that

$$\mathrm{rank}(W^T \otimes A) > \mathrm{rank}(W^T \otimes (U_e^T A)),$$

which implies

$$\dim(\mathscr{H}_1) \ll \dim(\mathscr{H}_3).$$

Recall that $\mathscr{H}_1 \subseteq \mathscr{H}_3$, so we can conclude that

$$\mathscr{H}_1 \subsetneqq \mathscr{H}_3.$$

Thus, solving $AHW = F$ is not equivalent to solving $U_e^T AHW = \Sigma_e V_e^T$ with respect to H. $\square$

This claim indicates that letting $A = U_e^T A$ and $F = \Sigma_e$ instead in the course of SVD preprocessing may enlarge the solution set and result in inequivalence, even though this action could further decrease the problem size and complexity.

3.2.3 Compressed Unmixing Algorithm

The computational experience indicates that, at least for the problems we tested so far, to obtain good solutions it suffices to solve a simplified compressed unmixing model that omits the nonnegativity of H,

$$\min_H \sum_{j=1}^{n_e} \mathrm{TV}(He_j) \ \text{ s.t. } \ AHW = F, \ H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}. \tag{3.13}$$

For simplicity, the algorithm is discussed based on the above model which was actually used in numerical experiments reported in the next section. In fact, in those experiments with hardware-measured data, the second constraint above is also omitted since it would not help in the presence of sizable system imprecisions and noise. On the other hand, the second constraint above plays a key role for blind unmixing, which will be involved later in this chapter. It should also be emphasized that in the compressed unmixing model (3.13), the matrices W and F are the output from the SVD preprocessing procedure. In particular, the size of the fidelity equation has been reduced to $m \times n_e$ from the original size $m \times n_b$, a factor of n_b/n_e reduction in size.

The main algorithm for the compressed unmixing model is based on the general TVAL3 Algorithm 2.3.1 proposed in Chapter 2. Minimizing the constraint model (3.13) is transferred into solving a series of unconstraint problems in virtue of the augmented Lagrangian method, and each unconstraint problem is separated into two tractable subproblems by introducing a new splitting variable.

To separate the discrete gradient operator from the non-differentiable TV term, we introduce splitting variables $v_{ij} = D_i(He_j)$ for $i = 1, \dots, n_p$ and $j = 1, \dots, n_e$. Then (3.13) is equivalent to

$$\min_{H, v_{ij}} \sum_{i,j} \|v_{ij}\| \quad \text{s.t.} \quad D_i(He_j) = v_{ij}, \ \forall\, i, j, \quad AHW = F, \ H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}. \tag{3.14}$$

The augmented Lagrangian function for (3.14) can be written as

$$\begin{aligned} \mathcal{L}_A(H, v_{ij}) \ \triangleq \ & \sum_{i,j} \left\{ \|v_{ij}\| - \lambda_{ij}^T (D_i(He_j) - v_{ij}) + \frac{\alpha}{2} \|D_i(He_j) - v_{ij}\|^2 \right\} \\ & - \langle \Pi, AHW - F \rangle + \frac{\beta}{2} \|AHW - F\|_F^2 - \nu^T (H\mathbf{1}_{n_e} - \mathbf{1}_{n_p}) \\ & + \frac{\gamma}{2} \|H\mathbf{1}_{n_e} - \mathbf{1}_{n_p}\|^2, \end{aligned} \tag{3.15}$$

where $\|.\|_F$ refers to Frobenius norm, λ_{ij}, Π, ν are multipliers of appropriate size, and $\alpha, \beta, \gamma > 0$ are penalty parameters corresponding to the three sets of constraints in (3.14), respectively. For brevity, we have omitted the multipliers in the argument list of $\mathcal{L}_A$.

The classic augmented Lagrangian method minimizes the augmented Lagrangian function $\mathcal{L}_A$ for fixed multipliers, and then updates multipliers as advised in (2.11) at each iteration. Specifically, in this case the multipliers are updated as follows. For all $1 \leq i \leq n_p$ and $1 \leq j \leq n_e$,

$$\begin{cases} \lambda_{ij} \;\leftarrow\; \lambda_{ij} - \varsigma\alpha(D_i(He_j) - v_{ij}), \\ \Pi \;\leftarrow\; \Pi - \varsigma\beta(AHW - F), \\ \nu \;\leftarrow\; \nu - \varsigma\gamma(H\mathbf{1}_{n_e} - \mathbf{1}_{n_p}). \end{cases} \tag{3.16}$$

To guarantee the convergence, ς should be selected in $(0, 2)$.

Under the framework of general TVAL3,Algorithm 2.2.1 is employed to minimize $\mathcal{L}_A(H, v_{ij})$ efficiently. The minimization problem with respect to v_{ij}'s is separable and has closed-form solutions v_{ij}^* according to the well-known shrinkage formula:

$$v_{ij}^* = \max\left\{\|\theta_{ij}\| - \frac{1}{\alpha}, 0\right\}\frac{\theta_{ij}}{\|\theta_{ij}\|}, \tag{3.17}$$

where

$$\theta_{ij} \triangleq D_i(He_j) - \frac{\lambda_{ij}}{\alpha}. \tag{3.18}$$

On the other hand, minimizing the augmented Lagrangian with respect to H can be excessively costly for large-sale problems, even if it is quadratic. Fortunately, according to Algorithm 2.2.1, all we need is to sufficiently decrease the augmented Lagrangian function by taking only one step of steepest descent on H from the current iterate; i.e.,

$$H \;\leftarrow\; H - \tau\,\mathcal{G}(H), \tag{3.19}$$

where $\mathcal{G}(H)$ denotes the gradient of $\mathcal{L}_A(H, v_{ij})$ with respect to H. The step length τ in (3.19) is chosen according to Algorithm 2.2.1.

If one intends to impose the nonnegativity of H in compressed unmixing model (3.13), the algorithm could be developed in the same way but replacing steepest descent (3.19) by one of projected gradient methods [77, 78]. The complexity would increase insignificantly by adding one more projection.

The derivation of $\mathcal{G}(H)$ is not abstruse but tedious. Let us tart with some propositions required in the course of derivation. The proof is quite basic and omitted here.

Proposition 3.2.1. *Suppose that the size of matrices X, Y and Λ are appropriate to form matrix multiplications $\langle \Lambda, XY \rangle$. Then*

$$\langle \Lambda, XY \rangle = tr(\Lambda^T XY) = tr(Y \Lambda^T X) = \langle \Lambda Y^T, X \rangle.$$

Furthermore,

$$\frac{\partial(\langle \Lambda, XY \rangle)}{\partial X} = \Lambda Y^T.$$

Here, "tr" denotes the trace of a matrix.

Proposition 3.2.2. *Suppose that the size of matrices X, Y, Z and Λ are appropriate to form matrix multiplications $X\Lambda^T Y \Lambda Z$. Then it follows Proposition 3.2.1 that*

$$\frac{\partial(tr(X\Lambda^T Y \Lambda Z))}{\partial \Lambda} = Y\Lambda Z X + Y^T \Lambda X^T Z^T.$$

Proposition 3.2.3. *Suppose that the size of matrices X, Y and Λ are appropriate to form matrix multiplications $X\Lambda Y$. Then*

$$vec(X\Lambda Y) = (Y \otimes X^T)^T vec(\Lambda) = (Y^T \otimes X) vec(\Lambda).$$

Furthermore,

$$\frac{\partial(X\Lambda Y)}{\partial\Lambda} = Y \otimes X^T.$$

Proposition 3.2.4. *Suppose that x and y are column vectors and Λ is matrix with appropriate size to form matrix multiplications $x^T\Lambda y$. Then it follows Proposition 3.2.3 that*

$$\frac{\partial(x^T\Lambda y)}{\partial\Lambda} = xy^T.$$

Except for Proposition 3.2.3, other three propositions derive the gradients of some scalar functions. Therefore, their gradients can be simply written in the form of matrix multiplications instead of the tensor or Kronecker products.

In order to derive $\mathcal{G}(H)$, the objective function $\mathcal{L}_A(H, v_{ij})$ is divided into four parts and the gradient with respect to H is obtained part by part. Define

$$\begin{aligned}
P_0(H) &= \sum_{i,j} \|v_{ij}\|, \\
P_1(H) &= \sum_{i,j} \left\{ -\lambda_{ij}^T(D_i(He_j) - v_{ij}) + \frac{\alpha}{2}\|D_i(He_j) - v_{ij}\|^2 \right\}, \\
P_2(H) &= -\langle \Pi, AHW - F\rangle + \frac{\beta}{2}\|AHW - F\|_F^2, \\
P_3(H) &= -\nu^T(H\mathbf{1}_{n_e} - \mathbf{1}_{n_p}) + \frac{\gamma}{2}\|H\mathbf{1}_{n_e} - \mathbf{1}_{n_p}\|^2.
\end{aligned}$$

Obviously, P_0 vanishes when computing the gradient with respect to H. According to Proposition 3.2.4,

$$\begin{aligned}
\frac{\partial\left(\lambda_{ij}^T(D_i(He_j) - v_{ij})\right)}{\partial H} &= \frac{\partial\left((\lambda_{ij}^T D_i)He_j\right)}{\partial H} \\
&= D_i^T\lambda_{ij}e_j^T, \qquad (3.20)
\end{aligned}$$

and according to Propositions 3.2.1 and 3.2.2,

$$\begin{aligned}
\frac{\partial\left(\alpha/2\|D_i(He_j)-v_{ij}\|^2\right)}{\partial H} &= \frac{\partial(\alpha/2\langle D_iHe_j-v_{ij}, D_iHe_j-v_{ij}\rangle)}{\partial H} \\
&= \frac{\partial(\alpha/2(\langle D_iHe_j, D_iHe_j\rangle - 2\langle v_{ij}, D_iHe_j\rangle + \langle v_{ij}, v_{ij}\rangle))}{\partial H} \\
&= \frac{\partial\left(\alpha/2\mathrm{tr}(e_j^TH^TD_i^TD_iHe_j)\right)}{\partial H} - \frac{\partial(\alpha\langle v_{ij}, D_iHe_j\rangle)}{\partial H} \\
&= \alpha D_i^TD_iHe_je_j^T - \alpha D_i^Tv_{ij}e_j^T \\
&= \alpha D_i^T(D_iHe_j-v_{ij})e_j^T \qquad (3.21)
\end{aligned}$$

It follows (3.20) and (3.21) that

$$\frac{\partial P_1}{\partial H} = \sum_j\sum_i\left\{-D_i^T\lambda_{ij}e_j^T + \alpha D_i^T(D_iHe_j-v_{ij})e_j^T\right\}. \qquad (3.22)$$

Applying Propositions 3.2.1, 3.2.2 and 3.2.4 as well, we have

$$\begin{aligned}
\frac{\partial\left(\langle\Pi, AHW-F\rangle\right)}{\partial H} &= \frac{\partial\left(\langle A^T\Pi, HW\rangle\right)}{\partial H} \\
&= A^T\Pi W^T, \qquad (3.23)
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial\left(\beta/2\|AHW-F\|_F^2\right)}{\partial H} &= \frac{\partial\left(\beta/2\langle AHW-F, AHW-F\rangle\right)}{\partial H} \\
&= \frac{\partial\left(\beta/2(\langle AHW, AHW\rangle - 2\langle F, AHW\rangle + \langle F, F\rangle)\right)}{\partial H} \\
&= \frac{\partial\left(\beta/2\mathrm{tr}(W^TH^TA^TAHW)\right)}{\partial H} - \frac{\partial(\beta\langle F, AHW\rangle)}{\partial H} \\
&= \beta A^TAHWW^T - \beta A^TFW^T \\
&= \beta A^T(AHW-F)W^T. \qquad (3.24)
\end{aligned}$$

Sequentially, combing (3.23) with (3.24), we get

$$\frac{\partial P_2}{\partial H} = -A^T \Pi W^T + \beta A^T (AHW - F) W^T. \tag{3.25}$$

Similar derivation can be applied to P_3, which gives

$$\frac{\partial P_3}{\partial H} = -\nu \mathbf{1}_{n_e}^T + \gamma (H \mathbf{1}_{n_e} - \mathbf{1}_{n_p}) \mathbf{1}_{n_e}^T. \tag{3.26}$$

Combining the results from (3.22), (3.25), and (3.26), we obtain the gradient of $\mathcal{L}_A(H, v_{ij})$ as follows:

$$\begin{aligned} \mathcal{G}(H) = & \sum_{i,j} \left\{ -D_i^T \lambda_{ij} e_j^T + \alpha D_i^T (D_i H e_j - v_{ij}) e_j^T \right\} - A^T \Pi W^T \\ & + \beta A^T (AHW - F) W^T - \nu \mathbf{1}_{n_e}^T + \gamma (H \mathbf{1}_{n_e} - \mathbf{1}_{n_p}) \mathbf{1}_{n_e}^T. \end{aligned} \tag{3.27}$$

Bringing (3.27) back to (3.19), we obtain the one-step update formula for H. Together with the shrinkage formula (3.17) for updating v_{ij}, the augmented Lagrangian function (3.15) can be efficiently minimized under the framework of Algorithm 2.2.1 — a modified alternating direction employed the nonmonotone line search and Barzilai-Borwein methods.

In summary, the algorithm for solving the compressed unmixing model (3.13) implements a combination of the SVD preprocessing (see Algorithm 3.2.1) and the general TVAL3 method (see Algorithm 2.3.1). More specifically, it can be depicted as follows:

Algorithm 3.2.2 (Compressed Unmixing)**.**

Input *data A, F and W.*

Preprocess F and W by Algorithm 3.2.1;

```
Initialize multipliers $\lambda_{ij}, \Pi, \nu$, penalty parameters $\alpha, \beta, \gamma > 0$, and other constants.
While "not converge" Do
    Update $v_{ij}$ and $H$ following Algorithm 2.2.1, with the aid of (3.17) and (3.19);
    Update multipliers $\lambda_{ij}, \Pi, \nu$ by formulas in (3.16) and non-decrease $\alpha, \beta, \gamma$;
End Do
Output $H$.
```

With the presence of nonnegativity $H \geq 0$ as in (3.4), a similar algorithm derivation would still apply, but an additional splitting variable would be necessary along with its own multiplier. This extra splitting would generally slow down the speed of convergence. Another alternative is to use a projected gradient method to enforce nonnegativity. However, our rather extensive computational experiments indicate that at least for the test cases we solved in this paper, enforcing the nonnegativity would not improve the resulting quality of solutions. Therefore, the reported results in this paper were all obtained using Algorithm 3 without enforcing nonnegativity on H.

The complexity of Algorithm 3.2.2 at each iteration is dominated by two matrix multiplications involving $W^T \otimes A$ and its transpose, respectively. Here W^T refers to the one being reduced size by SVD preprocessing and A could be implemented by some fast transform, so the algorithm is computationally low-cost as well as memory-efficient. In the next section, we will demonstrate the effectiveness of the algorithm in several sets of numerical experiments.

In Algorithm 3.2.2, the outer stopping criteria can be specified based on either relative change of variables or the optimality conditions of the compressed unmixing model (3.13). While the latter is more rigorous, it is also more costly. In experiments reported in the next section, we used relative change of variables in both outer and inner stopping criteria. Particular parameter settings and initial values will be specified in the course of descriptions of experiments.

3.3 Numerical Results on CSU Scheme

To demonstrate the feasibility, practicality and potential of the proposed CSU scheme, numerical results is presented based on applying the proposed CSU scheme to two types of data. Firstly, results are obtained on simulated or synthetic datasets. Then we provide results from much more realistic simulations where compressed hyperspectral data were directly measured by a hardware apparatus.

3.3.1 Setup of Experiments

We implemented the CSU scheme in a Matlab code which is still at an early stage of development. All numerical experiments reported in this paper were performed on a SONY VGN-FZ290 laptop running Windows 7 and MATLAB R2009b (32-bit), equipped with a 1.5GHz Intel Core 2 Duo CPU T5250 and 2GB of DDR2 memory.

In both types of experiments, we used randomized Walsh-Hadamard matrices as measurement matrices, A, considering that they permit fast transformation and easy hardware implementation. A Walsh-Hadamard matrix was randomized by choosing m random row from it and applying a random permutation to its columns.

In Algorithm 2.2.1 for handling subproblems, η was nailed to .9995 so as to attain high degree of nonmonotonicity; other constants δ, ρ and ζ were set to 10^{-4}, 5/3, and 10^4, respectively. In Algorithm 3.2.2, the multipliers λ_{ij}, Π and ν were always initialized to 0; the weight for updating multipliers ς was fixed at 1.6; the penalty parameters α, β and γ were selected from a range of 2^5 to 2^9, according to estimated noise levels. Despite of a lack of theoretical guidance, we observe that it is not particularly difficult to choose adequate values for these penalty parameters since the algorithm is not overly sensitive to such values as long as they fall into some appropriate but reasonably wide range. It takes a bit experience, and often a few trial-and-error attempts, to find good penalty parameter values, judged by observed speed of convergence or required computing times, for a given class of problems.

3.3.2 Experimental Results on Synthetic Data

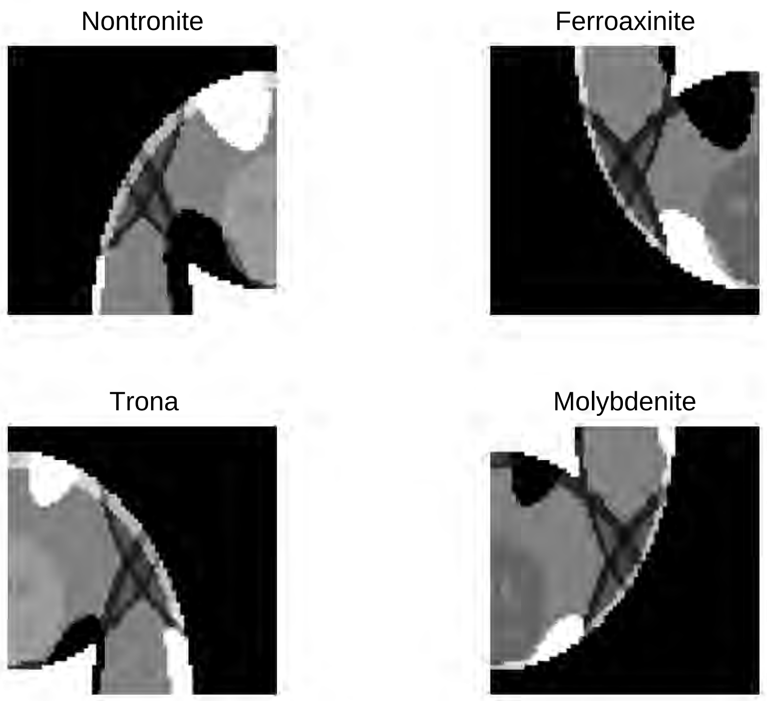


Figure 3.1: Synthetic abundance distributions.

In the first test, we generated compressed data according to data acquisition model (3.3). We selected 4 endmembers from the ASTER Spectral Library [122]: nontronite, ferroaxinite, trona and molybdenite, whose spectral signatures have been shown in Figure 3.2. A total of 211 bands were selected in the range of 0.4 to 2.5 micrometers. The distributions of abundance fractions corresponding to 4 endmembers were given

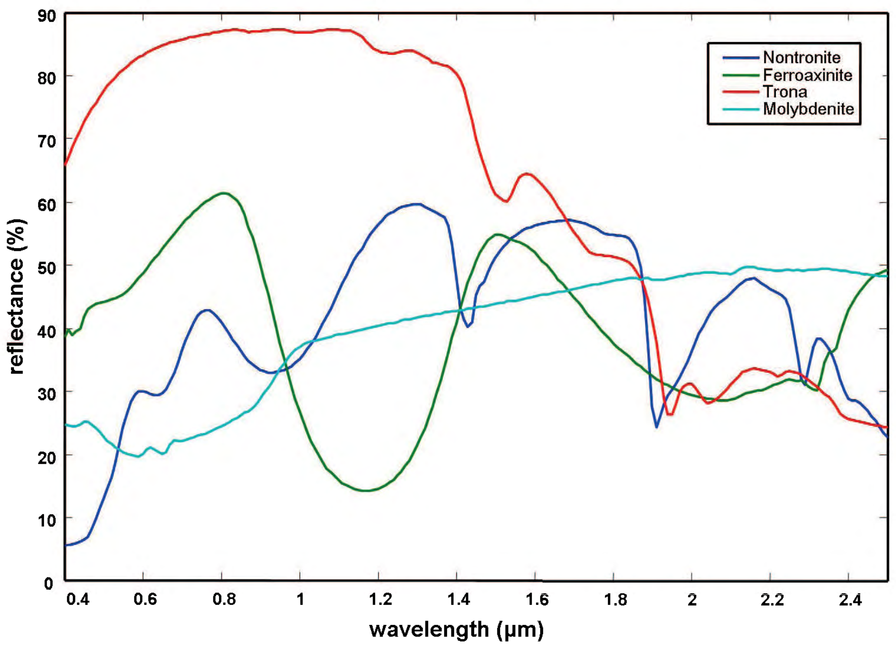


Figure 3.2: Endmember spectral signatures.

in Figure 3.1 with a spatial resolution of 64 × 64. Figures 3.1 and 3.2 gives the "true" H and W, respectively, from which we generated an observation $F = AHW$ in virtue of the measurement matrix A. In addition, to test the robustness of the CSU scheme, in some experiments we added zero-mean Gaussian random noise with standard derivation 0.8 to the observation matrix F.

In Figure 3.3, we plot relative errors in computed abundance fractions versus measurement rate of compressed data on 100 distinct testing points, with or without additive noise. The average elapsed time for these runs is less than 10 seconds. We observe that the CSU scheme attains relative error less than 1% when measurement rate is greater than 20% in both noisy and noise-free cases. This test empirically validates the feasibility of the proposed CSU scheme. More specifically, it shows that the abundance fraction can be unmixed directly from the compressed hyperspectral data by solving the proposed compressed unmixing model. The success on synthetic data sets has inspired us conducting further tests on larger and more realistic data.

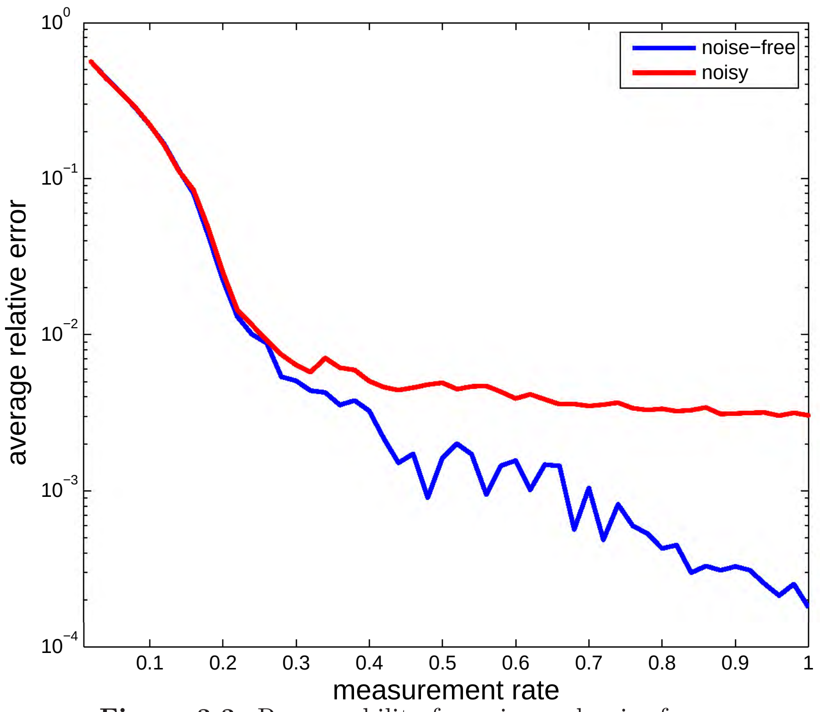


Figure 3.3: Recoverability for noisy and noise-free cases.

In the second test, we generated a compressed data matrix F by applying the data acquisition model (3.2) to the publicly available HYDICE Urban hyperspectral data [124], which contains 163 bands in a range from 0.4 to 2.5 micrometers, after some water absorption bands, each having a 307×307 resolution. According to the analysis of this Urban data cube in [110], there are 6 significant endmembers in the scene — road, metal, dirt, grass, tree and roof, as shown in Figure 3.4. The spectral signatures for these 6 selected endmembers are plotted in Figure 3.5.

Our computed unmixing result from 25% measurements are given in Figure 3.6, where six subfigures depict the computed distributions of abundance fractions for the six endmembers, respectively. It took about 215 seconds to run the algorithm. Qualitatively, we survey that the features in the original image such as roads, plants and buildings have been properly segmented by a visual comparison with Figure 3.4. For example, a hunk of roof marked by number 6 in Figure 3.4 appears prominently

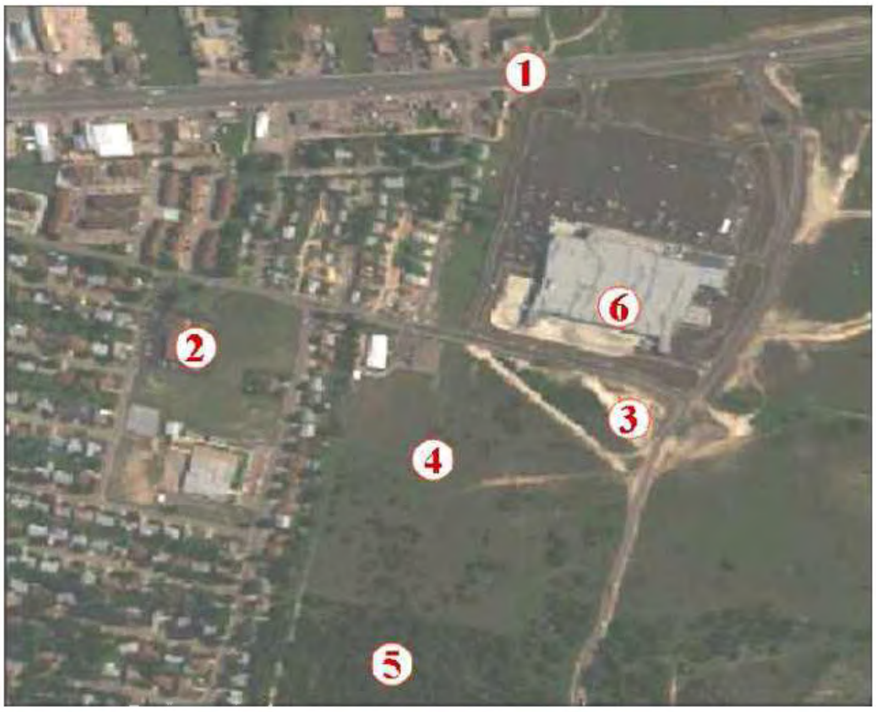


Figure 3.4: "Urban" image and endmember selection.

in the lower-right subfigure of Figure 3.6 for the abundance fractions of roof.

Figure 3.7 shows the least squares solution from directly solving $AHW = F$ for H with 100% data, which becomes an overdetermined linear system in this case. Comparing Figure 3.6 with Figure 3.7, we observe that the proposed CSU scheme, using 25% of the data, is capable of keeping important features and most details, even though the overall quality in computed abundance fractions by the CSU scheme is slightly lower than that of the least squares solution using 100% of the data.

3.3.3 Hardware Implementation

This section contains experimental results using hardware-measured data. Figure 3.8 shows the schematic of a compressing sensing hyperspectral imaging system based on a digital micro-mirror device (DMD). This system incorporates a micro-mirror array driven by pseudo-random patterns and one spectrometer. Similar to the single-pixel

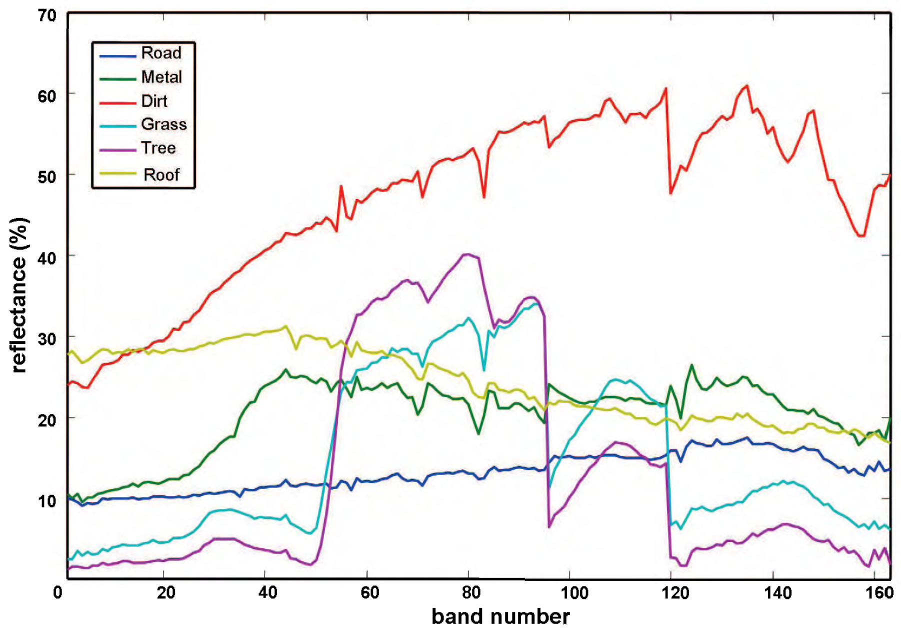


Figure 3.5: Spectral signatures with water absorption bands abandoned.

camera setup [32], it optically samples incoherent image measurements as dictated by the CS theory; then the Compressed Unmixing Algorithm 3.2.2 is applied to recover the the abundance fractions directly from the hardware-measured compressed data, bypassing the high-complexity of reconstructing the whole hyperspectral cube. The spectral information can be obtained from some library, codebook, or an independent experiment. If needed, the hyperspectral cube could be simply estimated by product of the estimated abundance fractions and endmember spectral signatures afterwards.

The spectrometer (on the right) we employed is a USB4000 by Ocean Optics which features a 3648-element linear array detector responsive from 200-1100 nm. The spectrometer and DMD (at the top) are synchronized to take data when the pseudo-random pattern switches. For each such a pattern, the measured data from the spectrometer is represented as a linear vector with the length of 3648. The target (at the bottom) is illuminated by two 35W daylight lamps from 45 degrees on both sides in order to achieve sufficiently uniform illumination.

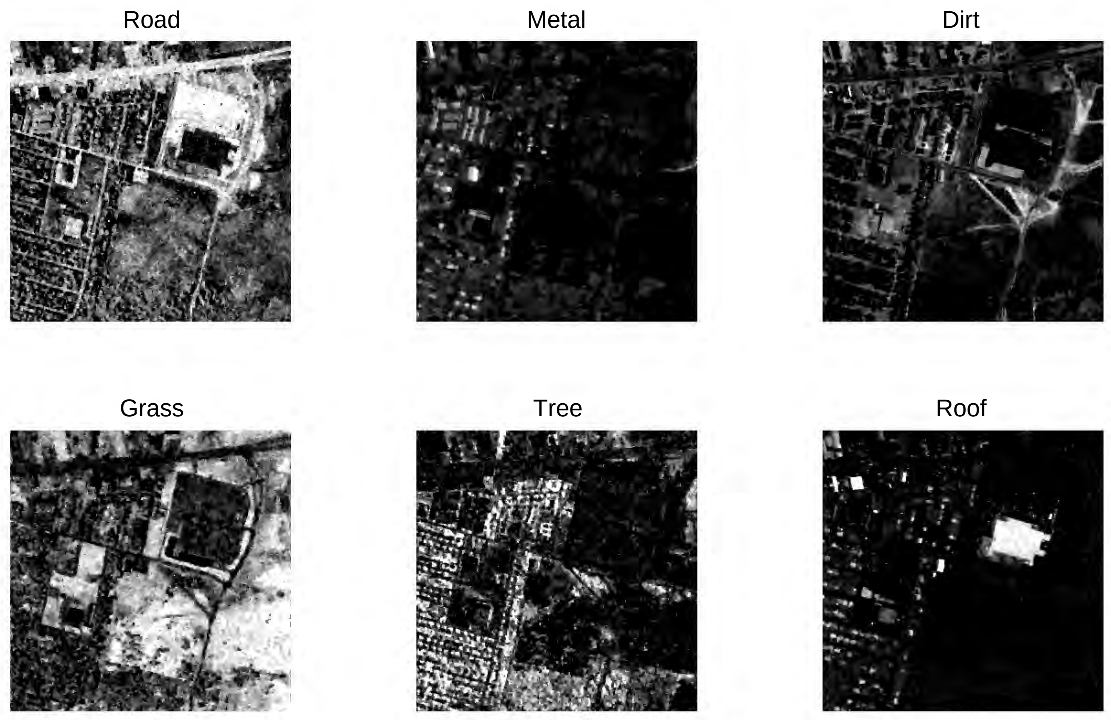


Figure 3.6: Estimated abundance: CS unmixing solution from 25% of measurements.

3.3.4 Experimental Results on Hardware-Measured Data

here we use compressed hyperspectral data collected by the hardware apparatus described above with the same type of measurement matrices A as in the previous experiments. Since the light shined on the object was distributed into over 3600 spectral bands, the intensity was significantly weakened in each channel, a relatively high level of noise became inevitable in the experiments. In many aspects, this represents a realistic and revealing setting to illustrate the concept of the proposed CSU scheme.

In the first test, our target image is an image of "color wheel", as shown in Figure 3.9, which is composed of various intensity levels of three colors: yellow, cyan and magenta. We selected 175 uniformly distributed bands in the range of 0.4 to 0.75 micrometers, and resolution at each band was 256×256. For convenience, we also

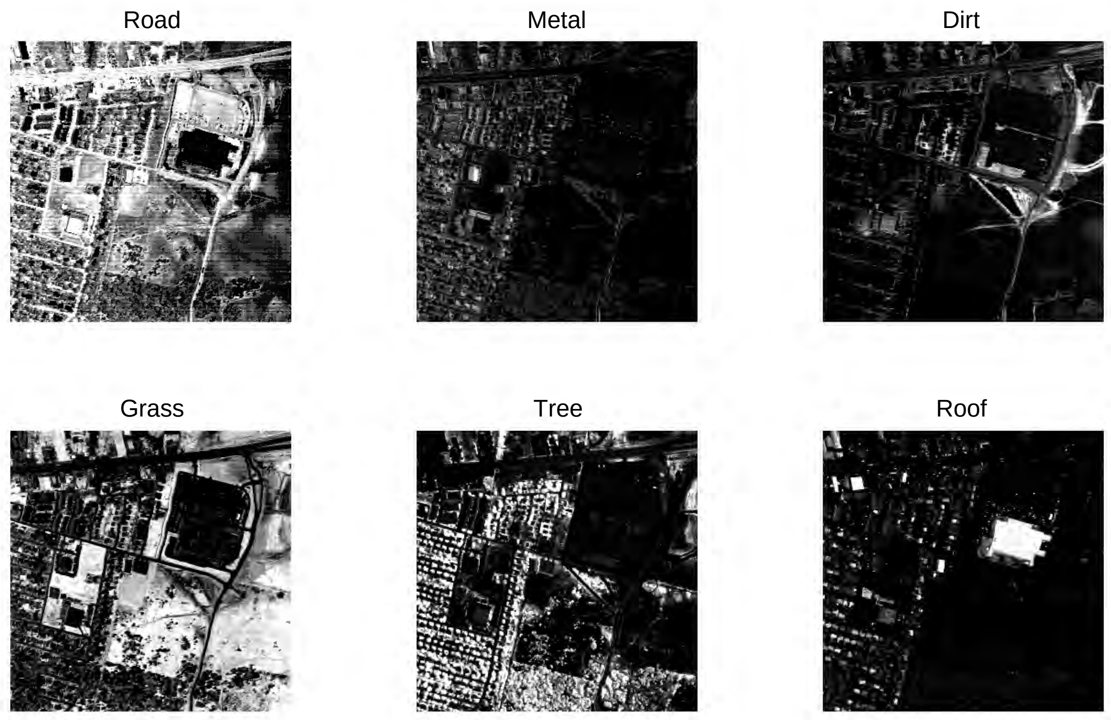


Figure 3.7: Estimated abundance: least squares solution.

chose yellow, cyan and magenta as the three endmembers, though different choices are certainly possible. In a separate experiment, we measured the spectral signatures for the three colors which are plotted in Figure 3.10. The parameters and initial values used in this test by Algorithm 3.2.2 are the same as those specified in Subsection 3.3.1.

The abundance fractions corresponding to the three endmembers were computed from 10% measured data, and are shown in Figure 3.11. The elapsed time to process the compressed unmixing was about 26 seconds. As we can see, our model and algorithm detected, quite accurately, the areas corresponding to each color at various levels of brightness.

Figure 3.12 gives 4 slices of the computed hyperspectral cube, obtained by multiplying the estimated abundance fractions H with measured signatures W, corresponding to four different spectral bands or wavelengths. Specifically, "cyan" be-

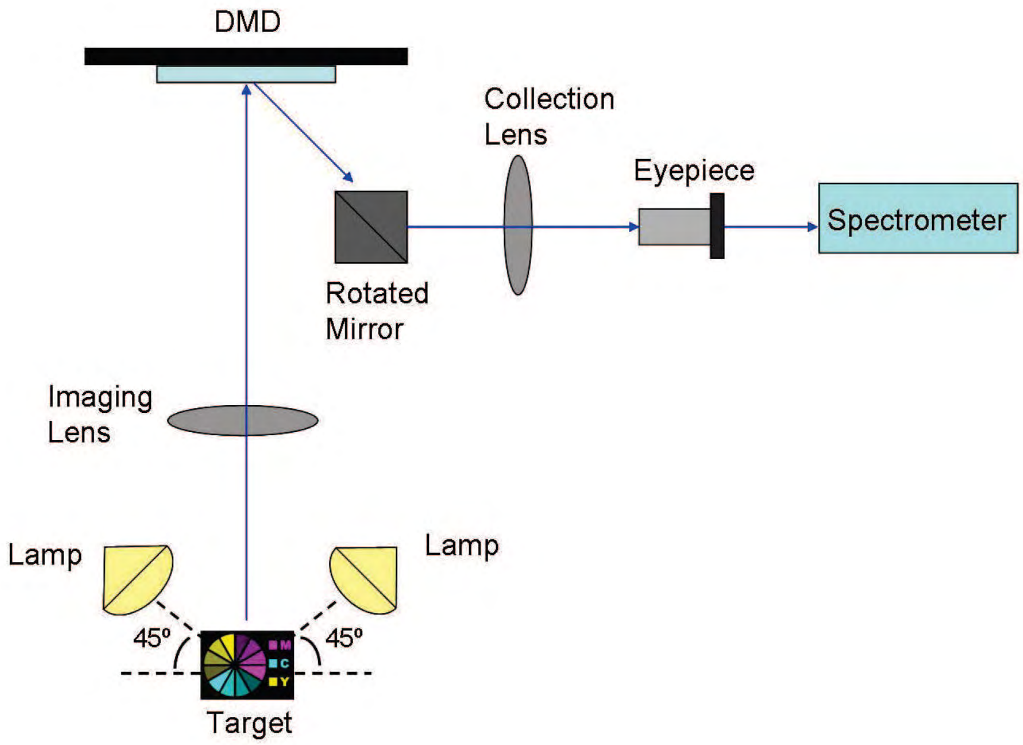


Figure 3.8: Single-pixel camera schematic for hyperspectral data acquisition.

comes bright around 490 $\sim$ 520 nm; "yellow" becomes bright around 520 $\sim$ 680 nm; "magenta" becomes bright around 680 nm. This result is consistent to the peak areas of three endmember spectral signatures plotted in Figure 3.10.

For comparison, Figure 3.13 gives four slices, corresponding to the same four spectral bands in Figure 3.12, of a computed hyperspectral cube that were computed from the same 10% of the measured dataset, one slice at a time, by the 2D TV solver TwIST [85, 86]. Similarly, Figure 3.14 contains corresponding results by the solver TVAL3 [125], and Figure 3.15 by NESTA [84]. In each of these three sets, the slices were reconstructed without using the information on the endmember signatures that leads to the (approximate) low-rankness in the CS compressed data matrix F.

It is evident that the results in Figure 3.12 are much cleaner and have stronger contrast than those in Figures 3.13, 3.14, and 3.15. This remarkable quality of the

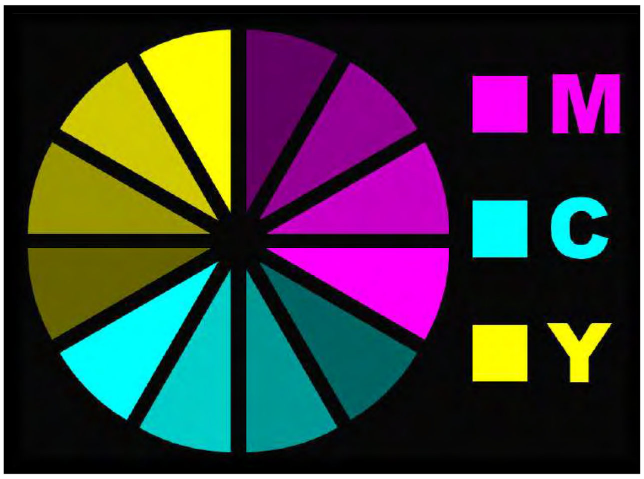


Figure 3.9: Target image "Color wheel".

CSU obtaind results should the consequence of two factors: 1) a proper exploitation of both low-rankness and sparsity in 3D hyperspectral data, 2) the denoising effects of SVD preprocessing. Besides the higher recovery quality, we mention that the amount of time required by the proposed CSU scheme to recover the hyperspectral cube (by first computing H then multiplying it with the signature matrix W) is more than one order of magnitude faster than those required by the state-of-the-art 2D TV solvers TwIST, TVAL3, and NESTA which perform the slice by slice recovery. This efficiency of the CSU scheme is the direct consequence of having to process a far less amount of data than the whole hyperspectral image cube.

In the second test, we kept all settings unchanged but the target image was replaced by a more complicated "subtractive color mixing", as shown in Figure 3.16. The three primary colors or endmembers here are still yellow, magenta and cyan. In subtractive mixing of color, the absence of color is white and the presence of all three primary colors is black corresponding to the central area of Figure 3.16. Similarly, 175 uniformly distributed bands were selected in the range of 0.4 to 0.75 micrometers, and resolution at each band was 256×256. The signatures of three endmembers have been plotted in Figure 3.10.

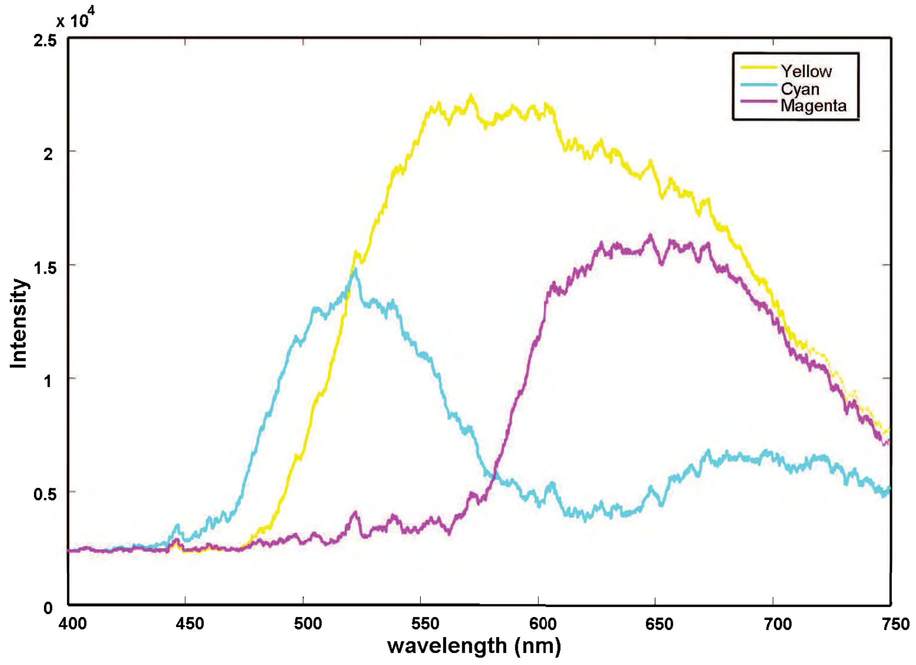


Figure 3.10: Measured spectral signatures of the three endmembers.

Figure 3.17 demonstrates the recovered abundance fractions based on 10% measured data with 32 seconds elapsed time. We observe that the area corresponding to pure colors has been accurately recognized, but the color-mixing area has not been perfectly distinguished. Specifically, the unmixing between yellow and cyan, and between magenta and cyan is evident. However, the unmixing between yellow and magenta is barely discernible in Figure 3.17. Ideally we should perceive 50/50 mixing of every two colors. The significant level of noise involved in measured data, which is also observable in slice-by-slice recovery results in Figures 3.19, 3.20 and 3.21, was most likely responsible for this loss of interpretability. Hopefully, a more accurate unmixing could be achieved by reducing the noise in the process of collecting data, with an improved hardware setup.

From the perspective of recovered cube, the proposed scheme (Figure 3.18) was also compared with 2D slice-by-slice recovery scheme using TwIST (Figure 3.19), TVAL3 (Figure 3.20), and NESTA (Figure 3.21), respectively. Similar to the results

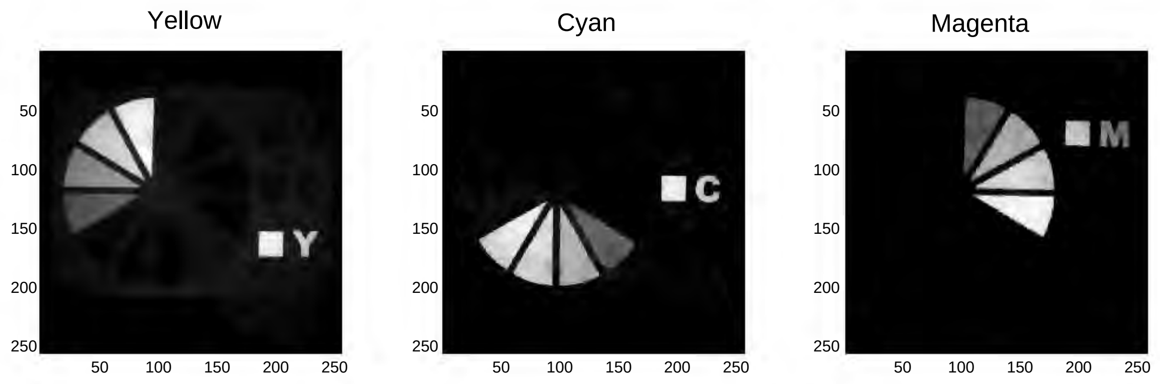


Figure 3.11: Estimated abundance: CS unmixing solution from 10% measurements.

in the first test, we can observe the remarkable improvements in both quality and efficiency using the proposed CSU scheme.

So far the CSU scheme has been well demonstrated in terms of its feasibility, efficiency and robustness. The numerical experiments have indicated that the scheme is promising and worth further investigations. It is certainly desirable to extend this work to more practical situations where knowledge about endmember spectral signatures are either very rough, highly incomplete, or partially missing, leading to a much more difficult task of compressive sensing and blind unmixing. In particular, the optimization models for this task become non-convex. However, some recent successes in solving non-convex matrix factorization models, such as [111] on matrix completion, offer hopes for us to conduct further research along this direction.

3.4 Extension to CS Blind Unmixing

The previous two sections proposed and studied a new unmixing framework using compressive sensing, consisting of a data acquisition method, hardware setup, and a CS unmixing scheme directly working on the compressed observations. The numerical results have shown its promises in the application of large-scale hyperspectral unmix-

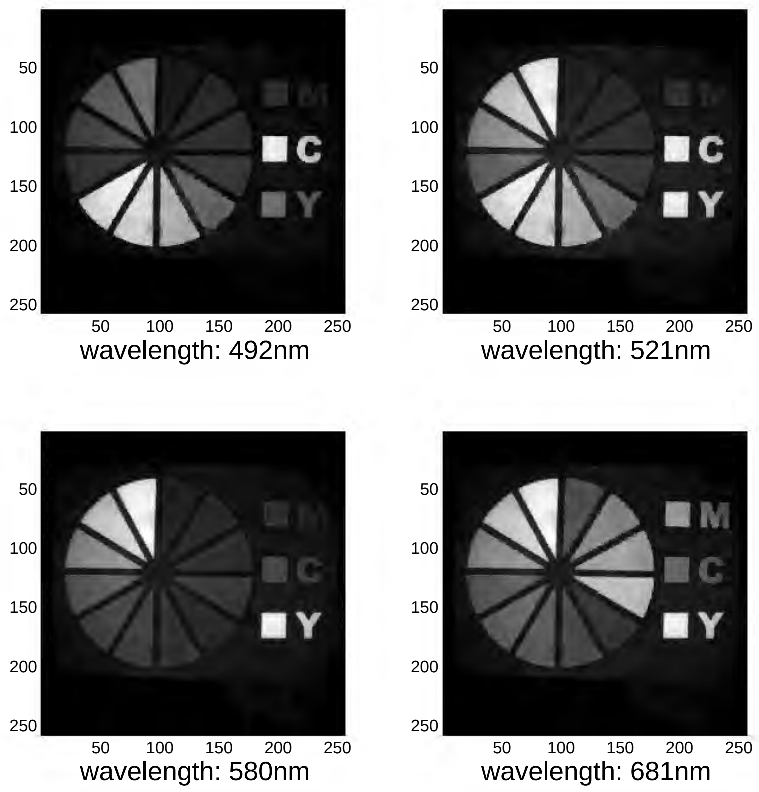


Figure 3.12: Four slices computed by the proposed approach.

ing. The experiments covered scenarios both with and without noise, demonstrating the robustness of CSU scheme as well as its remarkable denoising effects in existence of noise.

So far we have only considered cases where spectral signatures are given with a high accuracy. Usually the spectral signatures of endmembers can be obtained from three sources: libraries, codebooks and measured by hardware. The quality of hardware-measured spectra may be influenced by many factors, such as the precision of the equipment and experience of the personnel involved. All these factors could inevitably introduce noise into the measured spectra. Signatures extracted from a library or codebook are commonly considered as noise-free, but measured spectral data can still vary with measurement conditions and environments. In practice, obtaining

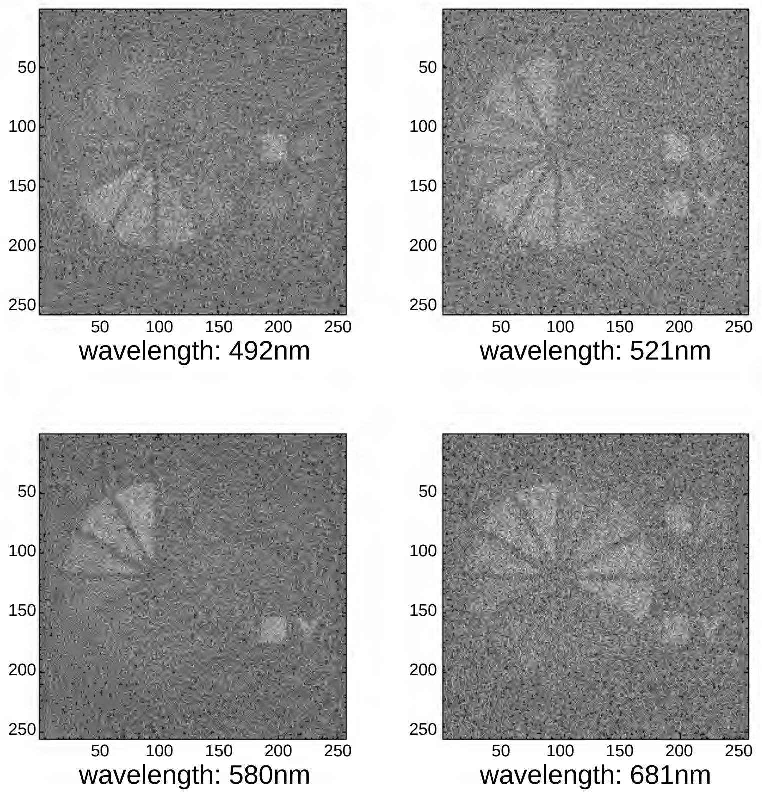


Figure 3.13: Four slices computed slice-by-slice using 2D TV algorithm TwIST.

exact signatures are unrealistic.With minor noise in signatures, CSU scheme can still estimate the corresponding abundance approximately. However, if noise level in spectral signatures is too high and stays uncorrected, the CSU scheme would generally fail. It is desirable to broaden the scope of CSU scheme to the situations where signatures are corrected at the same time of unmixing (named *CS blind unmixing*).

Under the same assumption of negligible interactions among endmembers, the data acquisition model (3.3) is still valid. To extend CSU scheme to blind unmixing, we consider the folllowing *compressed blind unmixing model* with respect to both W and H:

$$\min_{W,H} \sum_{j=1}^{n_e} \mathrm{TV}(He_j) \quad \text{s.t.} \quad AHW = F,\ H\mathbf{1}_{n_e} = \mathbf{1}_{n_p},\ H \geq 0, \tag{3.28}$$

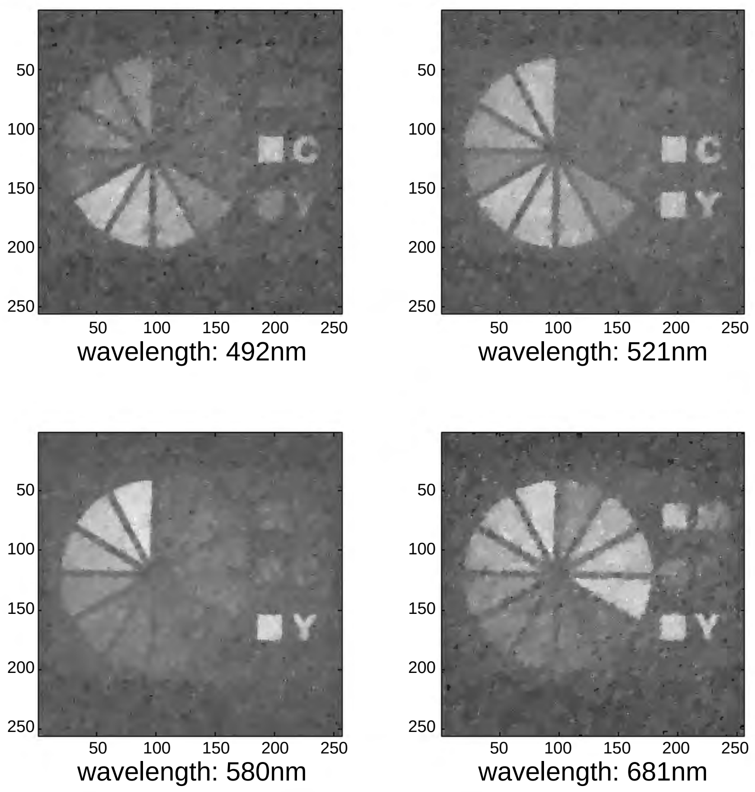


Figure 3.14: Four slices computed slice-by-slice using 2D TV algorithm TVAL3.

or the simplified model:

$$\min_{W,H} \sum_{j=1}^{n_e} \mathrm{TV}(He_j) \quad \text{s.t.} \quad AHW = F,\ H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}. \tag{3.29}$$

We will derive an algorithm for (3.29), which can be easily extended to the case with the extra nonnegativity constraint. On the surface, (3.29) appears to be quite close to the compressed unmixing model (3.13). Although the same objective function is used in both models, (3.13) is convex with a linear constraint whereas (3.29) becomes non-convex with a non-linear equality constraint.

Non-convex optimization is usually difficult to solve for a global minimum due to the existence of many local minima. The recent research on matrix completion [111]

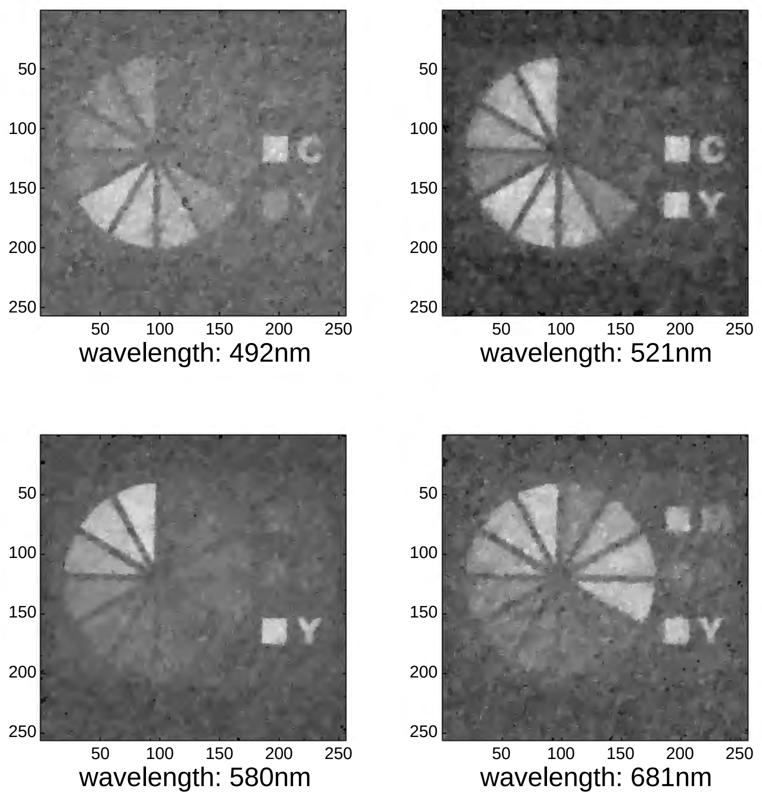


Figure 3.15: Four slices computed slice-by-slice using 2D TV algorithm NESTA.

proposed an ADM-type method on a non-convex minimization problem, which was shown to be effective and efficient. The purpose of matrix completion is to fill in the missing entries of a low rank matrix. In that paper, the matrix completion problem was reformulated using low-rank approximation, which is also one way to interpret the hyperspectral data model

$$X = WH.$$

Alternating direction method (ADM) is an extension of the classic augmented Lagrangian method and has been reviewed in Chapter 2. ADM minimizes the augmented Lagrangian function of the underlying problem with respect to each variable, and then update the multiplier. Unlike the augmented Lagrangian method, ADM

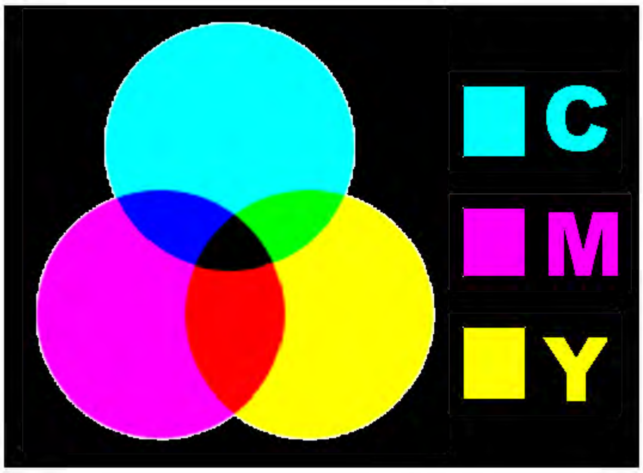


Figure 3.16: Target image "Subtractive color mixing".

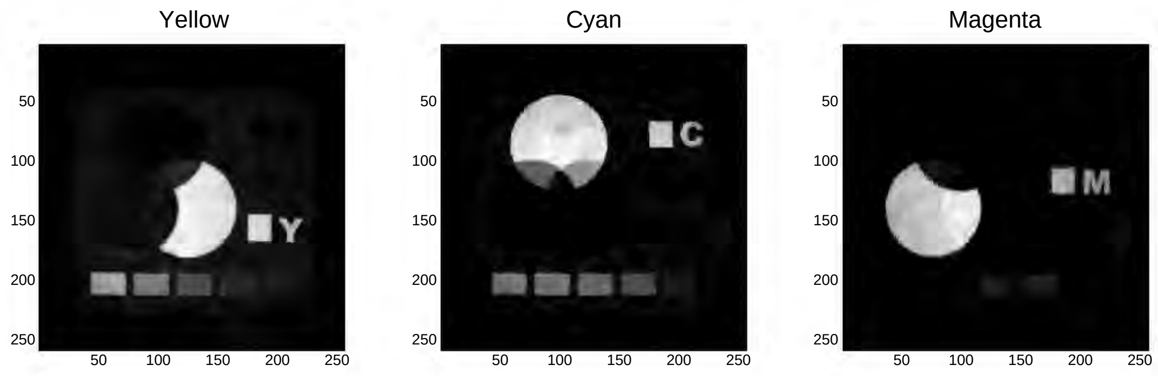


Figure 3.17: Estimated abundance: CS unmixing solution from 10% measurements.

avoids minimizing the augmented Lagrangian function exactly.

It follows the idea of ADM that for fixed Π we consider

$$\min_{W,H\in\Omega} \mathcal{Q}_A(W,H) \triangleq \sum_{j=1}^{n_e} \mathrm{TV}(He_j) - \langle \Pi, AHW - F\rangle + \frac{\beta}{2}\|AHW - F\|_F^2, \qquad (3.30)$$

where $\Omega = \{H \in \mathbb{R}^{n_p \times n_e} : H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}\}$, and $\mathcal{Q}_A(W,H)$ is non-convex and non-differentiable. Fortunately, $\mathcal{Q}_A(W,H)$ is convex with respect to each separate variable W and H and the global minimizer corresponding to either of them can be computed.

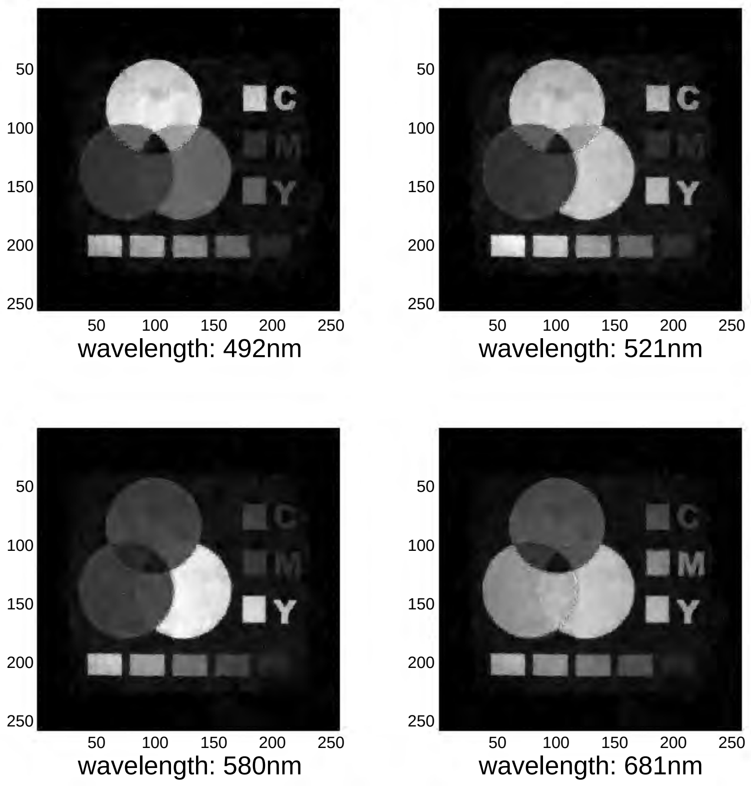


Figure 3.18: Four slices computed by the proposed approach.

To be specific, for fixed H, minimizing $\mathcal{Q}_A(W,H)$ is equivalent to

$$\min_{W} -\langle \Pi, AHW - F\rangle + \frac{\beta}{2}\|AHW - F\|_F^2, \tag{3.31}$$

which is differentiable and quadratic. In fact, (3.31) can be written as

$$\min_{W} \frac{\beta}{2}\left\|AHW - F - \frac{\Pi}{\beta}\right\|_F^2, \tag{3.32}$$

Since $AH \in \mathbb{R}^{m\times n_e}$ with $m > n_e$, this system is over-determined. Simple derivation gives the following closed-form minimizer of (3.32):

$$W^* = (AH)^{\dagger}(F + \Pi/\beta), \tag{3.33}$$

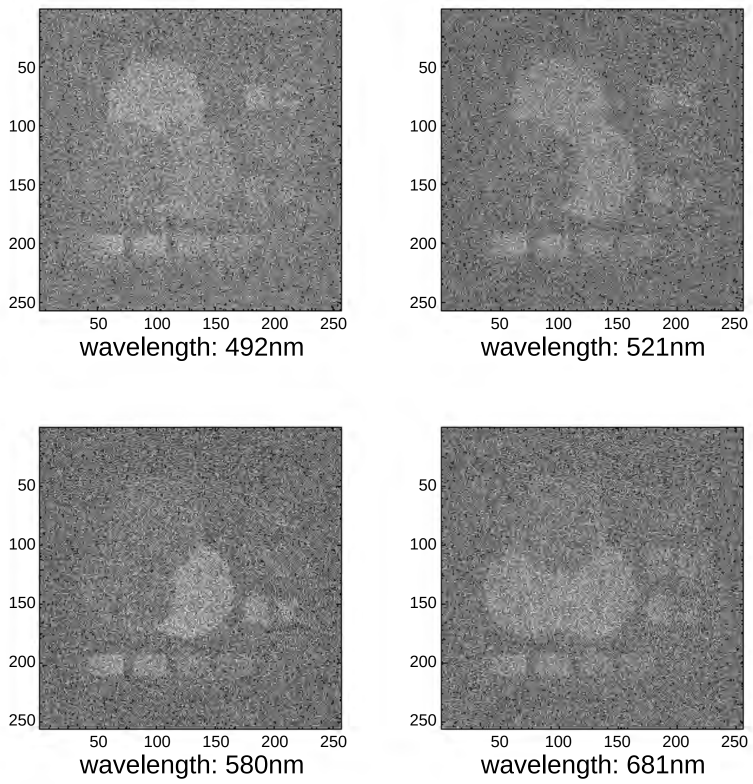


Figure 3.19: Four slices computed slice-by-slice using 2D TV algorithm TwIST.

where $S^\dagger$ denotes the Moore-Penrose pseudoinverse of a given matrix S, which is a generalization of the inverse matrix and defined as the unique matrix satisfying

$$SS^\dagger S = S,\ \ S^\dagger SS^\dagger = S^\dagger,\quad (S^\dagger S)^T = S^\dagger S,\quad \text{and } (SS^\dagger)^T = SS^\dagger.$$

On the other hand, for fixed W, minimizing $\mathcal{Q}_A$ on $H \in \Omega$ is relatively difficult. Mathematically, we want to solve the subproblem

$$\min_{H\in\Omega} \mathcal{Q}_A(W,H) \qquad \text{s.t.}\quad H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}, \tag{3.34}$$

In Section 3.2.3, the Compressed Unmixing Algorithm 3.2.2 — a particular implementation of the general TVAL3 method — has been proposed and investigated on

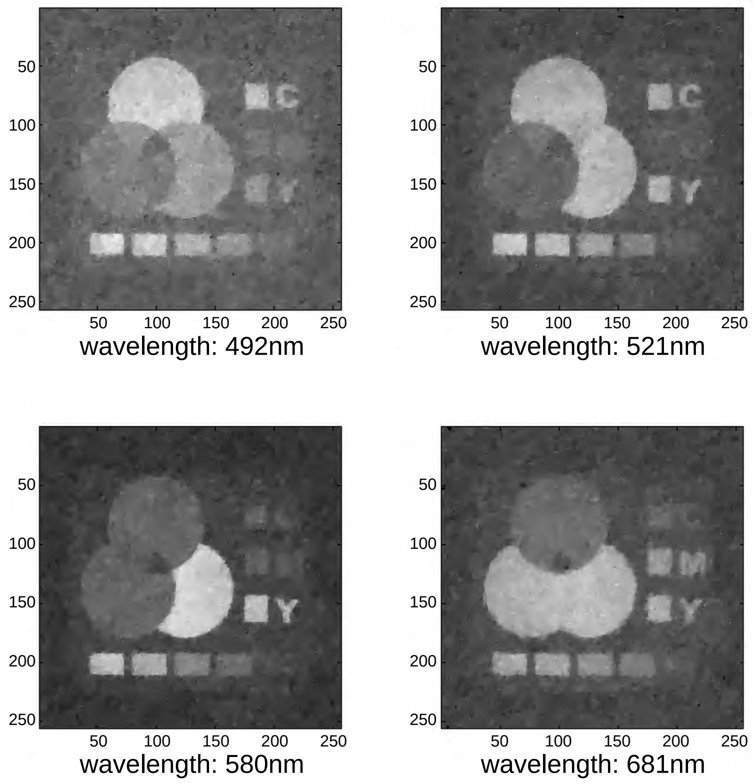


Figure 3.20: Four slices computed slice-by-slice using 2D TV algorithm TVAL3.

solving an almost identical problem (3.13). Specifically, by introducing new splitting variables, (3.34) is equivalent to

$$\begin{aligned} &\min_{H,v_{ij}} \sum_{i,j} \|v_{ij}\| - \langle \Pi, AHW - F \rangle + \frac{\beta}{2}\|AHW - F\|_F^2, \\ &\text{s.t.} \quad D_i(He_j) = v_{ij},\ \forall\, i,j; \quad H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}. \end{aligned} \tag{3.35}$$

The augmented Lagrangian function of (3.35) possesses exactly the same form as (3.15), except that Π and β are given and fixed here. Therefore, under mild modifications, the Compressed Unmixing Algorithm 3.2.2 can be applied to obtain the minimizer of (3.35). The modifications are two-fold:

1. the SVD preprocessing is not applicable any more, and

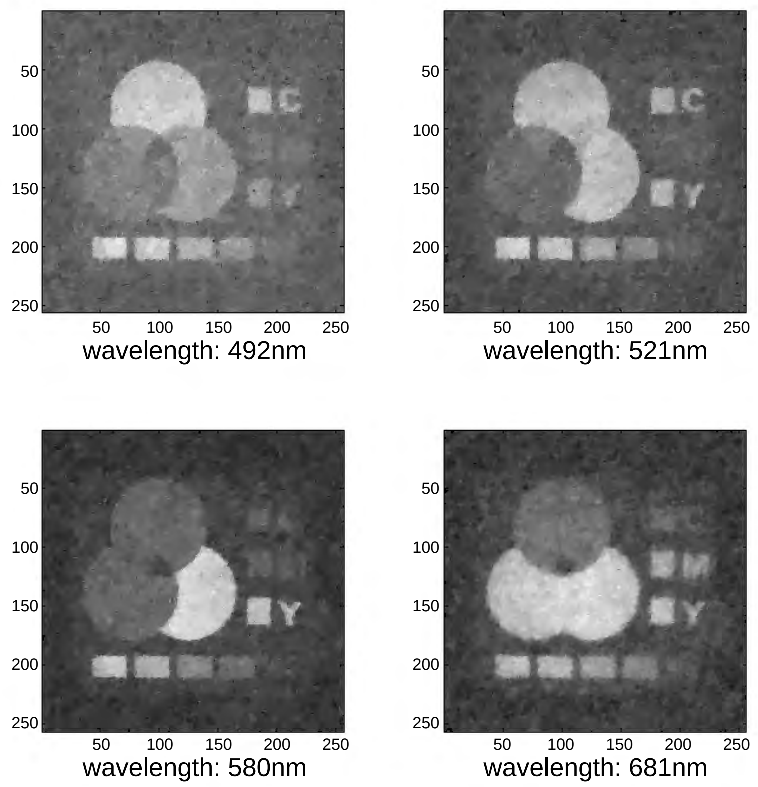


Figure 3.21: Four slices computed slice-by-slice using 2D TV algorithm NESTA.

2. there is no need to update Π and β.

After having obtained the minimizers of two subproblems, we can update the multiplier following (2.12); i.e.,

$$\Pi \leftarrow \Pi - \varsigma\beta(AHW - F). \tag{3.36}$$

So far the necessary subroutines have been derived in accordance to (2.12), and the ADM-type method can be employed to handle the compressed blind unmixing model (3.29). Moreover, to deal with the situation of selecting the most likely signatures from a large group of candidates, we propose the following preprocessing and postprocessing procedures:

Algorithm 3.4.1 (Selection Preprocessing)**.**

```
Input $F$ and $W = [w_1, \dots, w_{n_e}]^T$, and choose the threshold $\varepsilon$.
Compute the economy SVD: $F = U\Sigma V^T$, where $\Sigma = diag(\sigma_1, \dots, \sigma_{n_b})$,
    and $V = [v_1, \dots, v_{n_b}]$.
Determine $s \le n_e$ to satisfy $\sigma_s \ge \varepsilon$ and $\sigma_{s+1} < \varepsilon$.
Choose $\{\ell_1, \dots, \ell_s\} \subset \{1, \dots, n_e\}$, such that $\{v_1, \dots, v_s\}$ and $\{w_{\ell_1}, \dots, w_{\ell_s}\}$
    span the same space (or as close as possible).
Overwrite data: $n_e \leftarrow s$ and $W \leftarrow [w_{\ell_1}, \dots, w_{\ell_s}]^T$.
Output $n_e$ and $W$.
```

Algorithm 3.4.2 (Selection Postprocessing)**.**

```
Input $W = [w_1, \dots, w_{n_e}]^T$ and $H = [\hat{h}_1, \dots, \hat{h}_{n_e}]$, and choose the threshold $\epsilon$.
For $j = 1, \dots, n_e$ Do
    If $\|\hat{h}_j\| < \epsilon \|H\|_F / \sqrt{n_e}$ Do
        Decrease $n_e$ by 1, and delete $\hat{h}_j$ and $w_j^T$ from $H$ and $W$, respectively;
    End Do
End Do
Output $W$ and $H$.
```

The selection preprocessing procedure is based upon the fact that the rank of compressed observation F should be equal to the number of effective endmembers in the absence of noise. Moreover, suppose that the rank-n_e principal SVD is applied on F; i.e.,

$$F = U_e \Sigma_e V_e^T.$$

Then the space spanned by V_e should be identical to the space spanned by signatures of the effective endmembers. When noise exists, similar claims are also valid with a carefully chosen threshold. On the other hand, the selection postprocessing procedure is built on the fact that the abundance fractions corresponding to the non-existing endmembers should be close to zero or significantly smaller than others. Ideally, either one of two suggested procedures could successfully screen out the effective endmembers. However, the threshold criteria may be difficult to select under the influence of noise. A combination of both procedures as described in Algorithms 3.4.1 and 3.4.2 is recommended to enhance the accuracy of the selection. This is particularly important when developing a robust compressed blind unmixing method and algorithm.

In conclusion, the ADM algorithmic framework for solving the compressed blind unmixing model (3.29) can be depicted as follows:

Algorithm 3.4.3 (Compressed Blind Unmixing)**.**

```
Input data $A$ and $F$, and initial guess $W_0$.
Preprocess $W_0$ and overwrite $n_e$ by Algorithm 3.4.1.
Scale $W_0$ to an appropriate region.
Initialize the multiplier $\Pi = 0$, the penalty parameter $\beta > 0$, the selection threshold $\epsilon$,
    and other constants.
While "not converge" Do
    Compute the minimizer $H^{k+1}$ of $\mathcal{Q}_A(W^k, H)$ on $\Omega$ based on Algorithm 3.2.2;
    Compute the minimizer $W^{k+1}$ of $\mathcal{Q}_A(W, H^{k+1})$ by the formula in (3.33);
    Update multiplier $\Pi$ according to (3.36) and update $\beta$;
End Do
Postprocess $W$ and $H$ by Algorithm 3.4.2.
Output $W$ and $H$.
```

Several remarks are made here to further clarify the description of CS blind unmixing scheme and Algorithm 3.4.3:

Remark 1. *Since the non-convexity of the problem, the initial guess should be somewhat "close" to the global minimizer to avoid being stuck at a local minimum. An initial guess without any valid priori information, such as a random or zero matrix, would fail the blind unmixing. Fortunately, some prior information is usually available in practice, even if it might be incomplete, corrupted by severe noise, or deformed by environments.*

Remark 2. *As indicated in Section 3.2.3, the linear constraint* $H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}$ *is sometimes negligible in the course of CSU scheme, since* W *is given and assumed to be accurate. However, it seems helpful to impose* $H\mathbf{1}_{n_e} = \mathbf{1}_{n_p}$ *in the blind unmixing model* (3.29)*, otherwise* $(\varrho W^*, H^*/\varrho)$ *may give smaller objective function value than the minimizer* (W^*, H^*)*, for any* $\varrho > 1$.

Remark 3. *The algorithmic framework remains almost the same even if imposing the nonnegativity on* H*. Particularly, projected gradient [77, 78] can be used when updating* H*. However, imposing the nonnegativity would further complicate the non-convex model and slow down convergence. Presumably, as a form of prior information, taking nonnegativity into account may be beneficial in case that other forms of prior information are insufficient. Our computational experience indicates, however, that at least for the scenarios we tested so far, nonnegativity has not been an critical factor in those experiments.*

Remark 4. *In Algorithm 3.4.3, properly scaling the initial guess* W_0 *before the loop would make the initial guess closer to the global optimum, and then improve the* *accuracy. A simple way to do the scaling is as follows:*

$$\begin{cases} \tilde{H} = \underset{H}{\operatorname{argmin}}\, \mathcal{Q}_A(W_0, H), \\ \kappa \simeq avg(\tilde{H}), \\ W_0 = \kappa W_0, \end{cases} \tag{3.37}$$

where avg *represents the average of all entries.*

Numerical results are reported in the next section for different scenarios of blind unmixing.

3.5 Experiments for CS Blind Unmixing

The test platform is the same as the one described in Section 3.3.1 — a SONY VGN-FZ290 laptop running Windows 7 and MATLAB R2009b (32-bit), equipped with a 1.5GHz Intel Core 2 Duo CPU T5250 and 2GB of DDR2 memory.

In these experiments, randomized Walsh-Hadamard matrices were used as measurement matrices to take advantages of fast transformation and easy hardware implementation. The nonnegativity constraint was left out unless noted otherwise.

In Algorithm 3.2.2, the multiplier Π was initialized to 0; the weight for updating multipliers ς was fixed at 1.6; the selection thresholds ε and ϵ were both set to $1e-3$; the penalty parameter β was selected from a range of 2^5 to 2^9, according to estimated noise levels. The settings of other parameters for solving subproblems could be found in Section 3.3.1. As mentioned before, even though the algorithm is not overly sensitive to the choice of penalty parameters, it still takes a few trial-and-error attempts to find suitable values for a fast convergence.

3.5.1 Denoising Tests

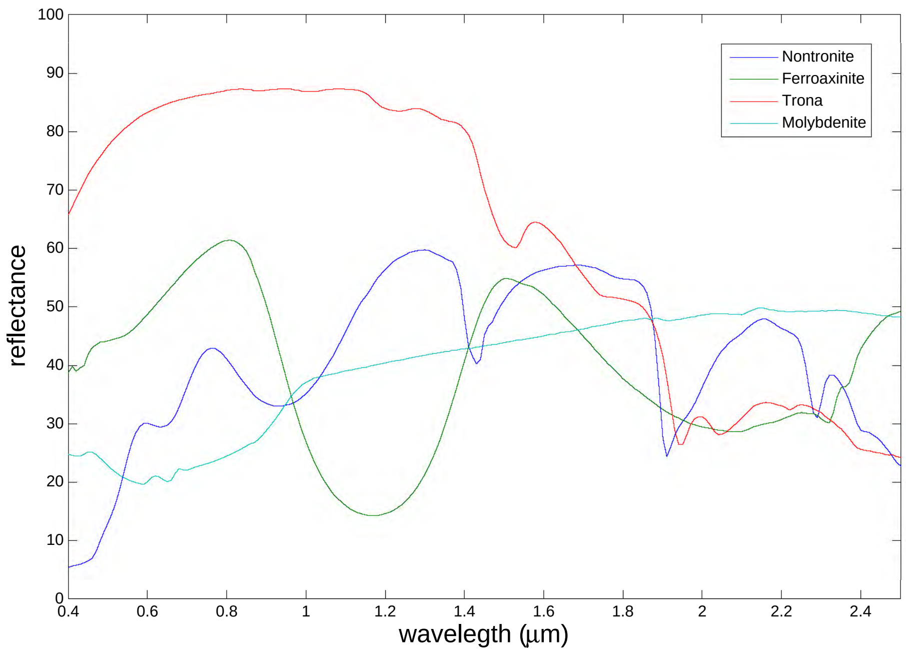


Figure 3.22: Endmember spectral signatures.

For ease of comparison, all tests demonstrated here were extended from the first synthetic test in Section 3.3.2. Four materials including nontronite, ferroaxinite, trona and molybdenite were selected and their signatures were obtained from ASTER spectral library as shown in Figure 3.22. The corresponding abundance fractions were constructed as shown in Figure 3.23. The endmembers were mixed by the multiplication of the signatures and the abundance, giving the hyperspectral cube which is used to synthetically acquire the compressed observation. In real world, compressed data would be collected directly from a scene and no hyperspectral cube would be needed by our approach. Constructing the hyperspectral cube in such synthetic experiments provides ground truths and helps the assessment of computed results. Six slices extracted from the original cube are exhibited in Figure 3.24, each corresponding to an image of a specific wavelength. The spatial resolution was

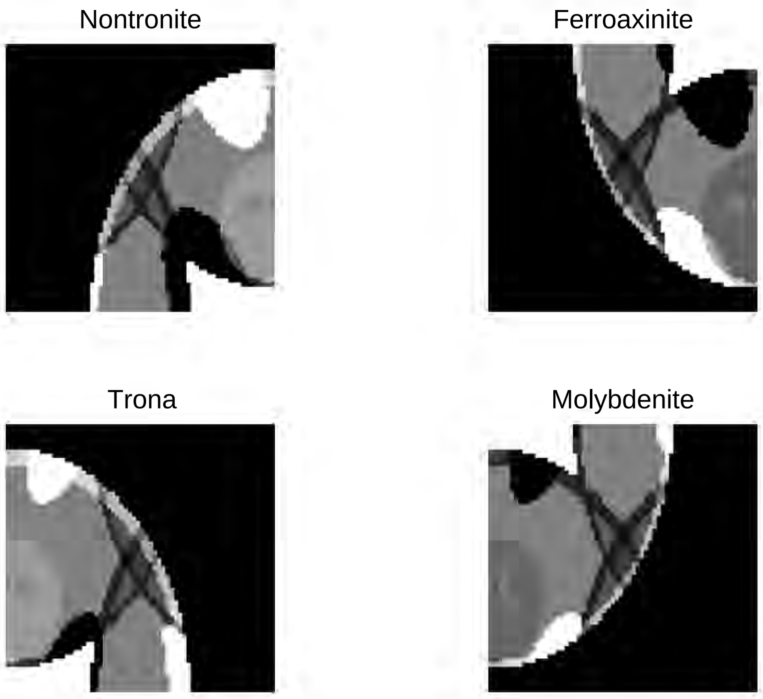


Figure 3.23: Synthetic abundance distributions.

set to 64 × 64; the number of total bands was set to 211 which were uniformly distributed between 0.4 and 2.5 micrometers according to the library specifications for the signatures.

The purpose of tests here is to study the impact of distinct types of noise to the proposed compressed blind umixing method. Specifically, Gaussian, periodic and impulsive noise are tested in the experiments. The noise-free compressed observation is always generated by $F = AHW$, whose size depends on the number of measurements taken. In particular, all tests were done using 20% measurements and the detailed results of each one were reported in Table 3.1.

In the first test, we demonstrate the robustness of Algorithm 3.4.3 under Gaussian noise. The signatures of four endmembers were corrupted by zero-mean Gaussian random noise with standard derivation 22.7 and then input as the initial guess. From Figure 3.25(a), we can see that the signatures have been severely impaired and the

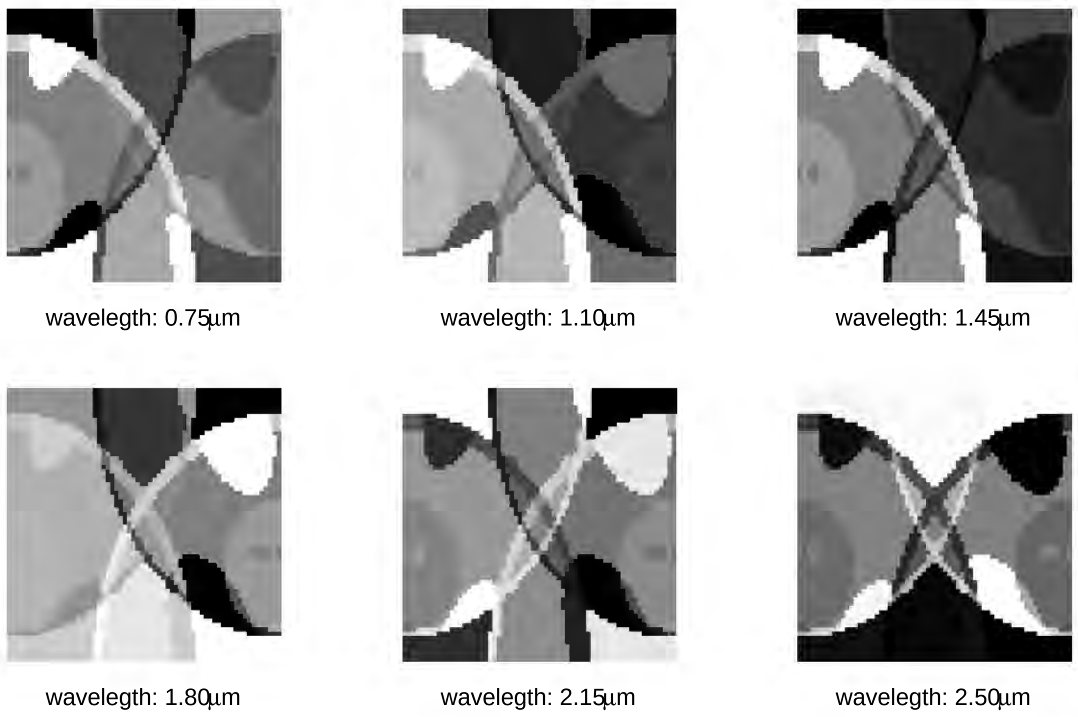


Figure 3.24: Hyperspectral imaging under specific wavelengths.

corresponding endmembers are hardly recognizable from these corrupted signatures. Figure 3.25(b) shows the plot of recovered signatures by Algorithm 3.4.3. A significant improvement is unmistakable in contrast to the initial guess in Figure 3.25(a). Clearly, the four endmembers could be easily identified from the recovered signatures. Numerically, the relative error in the signatures has been decreased from 43.8% to 3.6%, while the induced hyperspectral cube has a 2.39% relative error.

Gaussian noise is unavoidable in the course of collecting data in practice. In hyperspectral data acquisition, a spectrometer is usually employed to distribute the light into hundreds or even thousands of channels, and the power in each channel is dissipated by orders. In this case, Gaussian noise becomes even more significant, and a robust algorithm capable of denoising is advantageous.

In the second test, we consider the impact of periodic noise. The noise was constructed using a sinusoidal function with a period of 0.422 and an amplitude of 22.7. The corrupted signatures are shown in Figure 3.26(a). Starting with the corrupted signatures as initial guess, we applied Algorithm 3.4.3 to remove the periodic noise as well as to unmix the compressed data. Figure 3.26(b) shows the recovered signatures, from which the endmembers could be recognized despite some minor deviations from the true signatures plotted in Figure 3.22. Numerically, the relative error in the signatures decreased from 32.7% to 4.1%, and led to an estimated hyperspectral cube with only 0.98% relative error.

Another type of possible noise is impulsive noise, which refers to total loss of signature values in some bands. Unlike either Gaussian or periodic noise, impulsive noise is discrete and the true values are substituted, with some probability, by random values from a noise source. Visually the outbursts caused by impulsive noise are observable as discrete high-contrast points, which explains why it is also called *pepper-and-salt noise*. In the test with impulsive noise, the probability of noise occurrence was set at 25%, and impulsive noise was imposed uniformly random at positions. The corrupted signatures were plotted in Figure 3.27(a) as the starting point for Algorithm 3.4.3. Similarly, the recovered signatures are depicted in Figure 3.27(b), with a 6.5% relative error. In addition, the resulted hyperspectral cube has a less than 1% relative error.

In the forth test, we still imposed impulsive noise on the signatures of four endmembers with 25% probability of occurrence. However, we kept the same distance between every two adjacent corrupted positions in stead of randomly selecting positions, and all four signatures were corrupted at the same positions, as shown in Figure 3.28(a). We could observe a similar successful reconstruction as indicated in Figure 3.28(b). Specifically, the relative error of signatures dropped from 33.9% to 3.8%, which leads to a 1% relative error in hyperspectral cube.

Table 3.1: Numerical results of denoising tests

	$\frac{m}{n}$	$\frac{\|W_0-W^*\|}{\|W^*\|}$	$\frac{\|W_{est}-W^*\|}{\|W^*\|}$	$\frac{\|H_{est}-H^*\|}{\|H^*\|}$	$\frac{\|X_{est}-X^*\|}{\|X^*\|}$	$t-t_0$
Test 1 (Gaussian)	20%	43.802%	3.624%	15.416%	2.390%	53.68s
Test 2 (periodic)	20%	32.710%	4.133%	19.574%	0.980%	57.53s
Test 3 (impulsive)	20%	30.266%	6.517%	19.995%	0.987%	60.91s
Test 4 (impulsive)	20%	33.926%	3.798%	8.044%	1.009%	64.22s

These numerical results, as reported in Table 3.1, suggest that the proposed compressed blind unmixing algorithm could remove severe noise of these four kinds from poor signature estimation. Overall speaking, recovered results by Algorithm 3.4.3 preserved most, if not all, critical characteristics of the true signatures such as waves and spikes, from just 20% measurements. These experimental results on a variety of noise scenarios substantiate the claim that the proposed scheme is able to handle noisy compressed data directly acquired by hardware and perform blind unmixing.

3.5.2 Further Scenario Tests

Other than noise, a number of factors may result in biased or even misleading prior information. In this part, we test robustness of proposed compressed blind unmixing method by simulating several scenarios. The basic setup of testing is the same as what we described at the beginning of Section 3.5.1: nontronite, ferroaxinite, trona and molybdenite, whose signatures were plotted in Figure 3.22, were chosen as endmembers and their corresponding abundance fractions were constructed as shown in Figure 3.23. Statistics of computational results are given in Table 3.2.

In some cases, signatures may be measured not as reflectance but as energy or intensity instead. This may alter the scale of the signatures. We start with a simple test to demonstrate that the compressed blind unmixing algorithm can restore the correct scale in signatures. In our test, the signatures of four endmembers were amplified by tenfold and then input as the initial guess W_0 into Algorithm 3.4.3. Figure 3.29 compares the recovered signatures shown in (b) with the initial guess shown in (a). Compared to the true one shown in Figure 3.22, we observe that the amplitude has been successfully restored. The relative error of the recovered signatures is 0.18%. The estimated hyperspectral cube, generated by the recovered H and W, has a relative error of 0.15%.

This test confirms that the proposed compressed blind unmixing method can effectively deal with different scales in measured signatures. This would become particularly appealing for real data acquisition, since acquiring the compressed data and measuring the signatures of participated endmembers are normally carried out by two independent experiments under different experimental conditions. In such circumstances, differences in scales are likely to occur.

In the second test, we simulated the scenario that the number of endmembers was not supplied but we were aware of the range of candidates which might be present in a scene. In unmixing, we tried to find the contributing endmembers with the aid of the preprocessing and postprocessing described in Algorithms 3.4.1 and 3.4.2, both of which have been incorporated into the Compressed Blind Unmixing Algorithms 3.4.3. More specifically, four other materials were picked from the library including chalcopyrite, talc, illite and lepidolite. Their signatures together with the signatures of nontronite, ferroaxinite, trona and molybdenite were drawn in Figure 3.30(a). These eight endmembers were viewed as candidates for Algorithm 3.4.3. Figure 3.30(b) shows that only four out of the eight candidates have been selected as contributing endmembers. Further comparison with Figure 3.22 indicates that the selections are correct.

The following two experiments studied the scenarios where a given set of signatures contains one or two wrong endmembers.

In the third test, we considered the case where one endmember out of four is wrong. More specifically, the presence of nontronite was incorrectly replaced by illite. As illustrated in Figure 3.31(a), the signature of illite together with signatures of the other three original endmembers were provided as initial guess to Algorithm 3.4.3. It is notable from the plot that the signature of illite is totally unrelated to the one of nontronite, which implies that priori information on nontronite is missing. From Figure 3.31(b), we observe that the recovered signatures matched the characteristics of nontronite, ferroaxinite, trona and molybdenite (see Figure 3.22) very well whereas the signature of illite has vanished.

This result indicates that the compressed blind unmixing method is capable of retrieving one missing signature. However, the method may fail if too much information is missing.

Next we repeated the last test but assuming two endmembers were misidentified, which further increased the difficulty of blind unmixing. Figure 3.32(a) plots the signatures of illite, ferroaxinite, trona and lepidolite which were fed as initial guess instead of the original four endmembers. From the plot of the recovered signatures shown in 3.32(b), we can barely recognize the missing two endmembers — nontronite and molybdenite, which implies the failure of blind unmixing. Even if we imposed nonnegativity for extra information, the unmixing result still hardly improved. The relative error of recovered signatures remains over 30% but the relative error of induced hyperspectral cube drops to less than 3%. This result reveals that the reason for this failure in this test is not the lack of enough measurements, but the non-convexity of the problem.

Table 3.2: Numerical results of scenario tests

	$\frac{m}{n}$	$\frac{\|W_0-W^*\|}{\|W^*\|}$	$\frac{\|W_{est}-W^*\|}{\|W^*\|}$	$\frac{\|H_{est}-H^*\|}{\|H^*\|}$	$\frac{\|X_{est}-X^*\|}{\|X^*\|}$	$t-t_0$
Test 1	20%	900%	0.176%	0.987%	0.153%	58.29s
Test 2	20%	123.816%	0.647%	3.721%	0.290%	66.29s
Test 3	20%	21.972%	6.849%	25.170%	5.028%	69.53s
Test 4	20%	38.965%	30.190%	116.405%	2.965%	79.60s

3.5.3 Remarks on Compressed Blind Unmixing

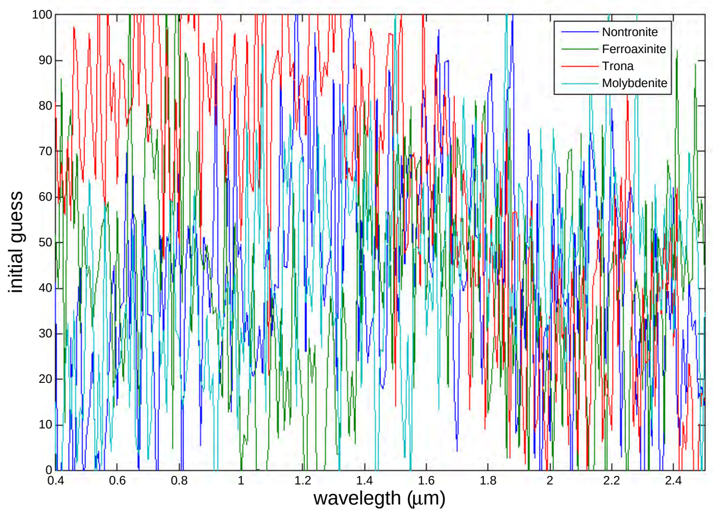
From our experience, the number of measurements used for reconstruction is not necessarily correlated with the success of blind unmixing. Since the blind unmixing problem is non-convex, the success of unmixing depends heavily on the information content of initial guess. Generally, with poor initial guess increasing the number of measurements can reduce the error in estimated hyperspectral cube, but not help find better signatures or corresponding abundances. Due to non-uniqueness in matrix factorization, it is likely that both signatures and abundances are completely wrong but their product gives a well-estimated hyperspectral cube (see results in Section 3.5.2).

Our extensive experiments have shown the effectiveness and robustness of the proposed compressed blind unmixing method. The algorithm can not only remove several types of noise but also correct wrong scales. It can also remove extra endmembers not present on a scene. Moreover, the algorithm can handle the cases where one endmember has been totally misidentified. However, we should by no means minimize the difficulty of blind unmixing. The success of our approach relies on the availability of sufficient prior information on endmember signatures. The "blind unmixing" approach is not totally blind.

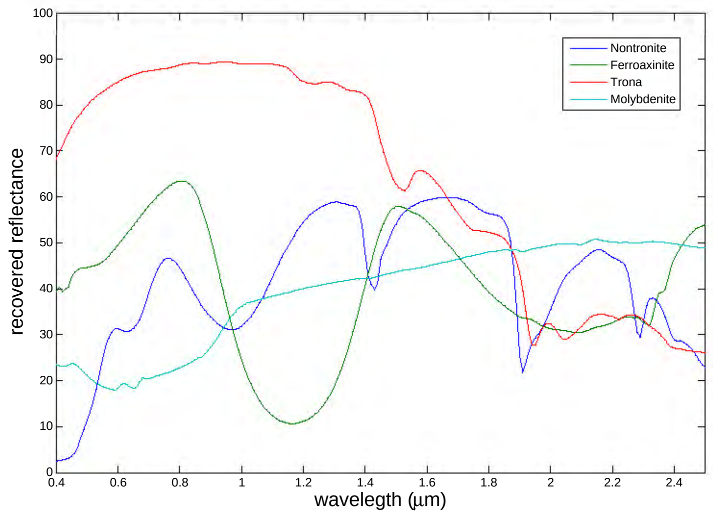
On the other hand, hyperspectral cubes can usually be well estimated by the compressed blind unmixing method with or without quality prior knowledge, as long as the number of measurements is sufficient in the sense of compressive sensing. The induced cube can be analyzed by other methods or techniques, which may or may not require optimization.

3.6 Conclusion

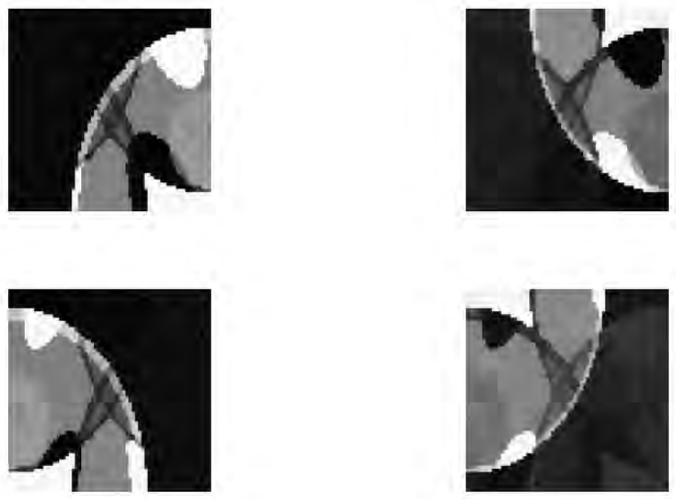
In a nutshell, the proposed scheme has been empirically, and rather convincingly, validated using both synthetic data and measured data acquired by a hardware device similar to the single-pixel camera [32]. The numerical results clearly demonstrate that compressively acquired data of size ranging from 10% to 25% of the full size can produce satisfactory results highly agreeable with the "ground truth". The remarkable denoising effects against various types have been justified by solid evidence. The process speed achieved so far, which can certainly be further improved, seems to fall within a promising range. Furthermore, both the framework and the algorithm have been successfully extended to the case of blindly unmixing the compressed hyperspectral data.



(a) Initial guess of signatures.

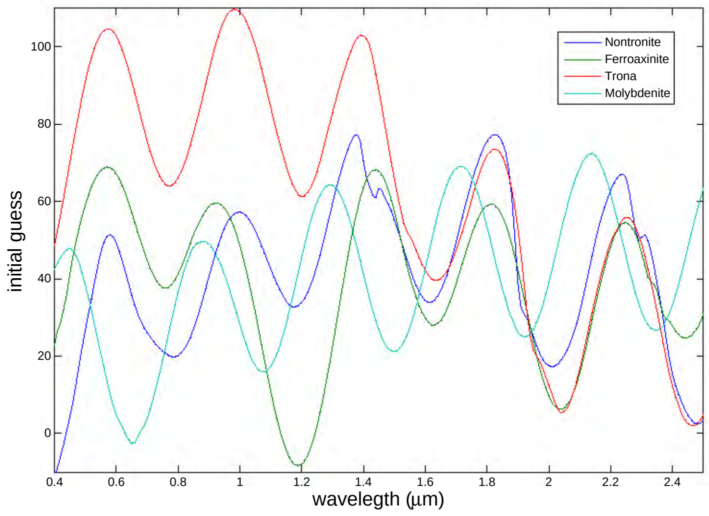


(b) Recovered signatures.

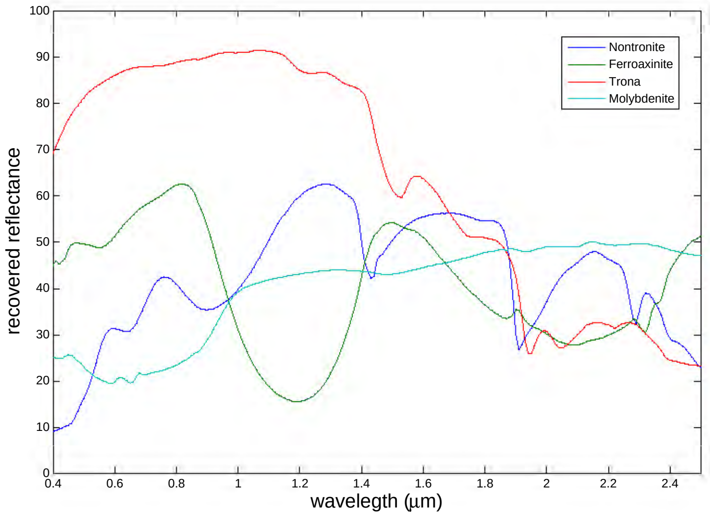


(c) Recovered abundance distributions.

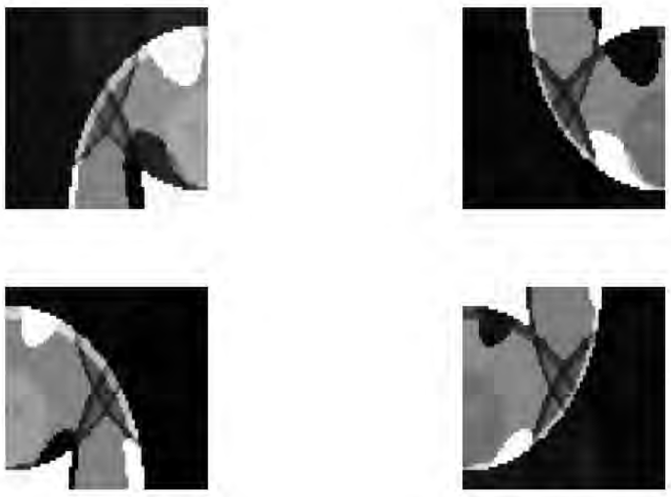
Figure 3.25: Removing the Gaussian noise involved in endmembers.



(a) Initial guess of signatures.

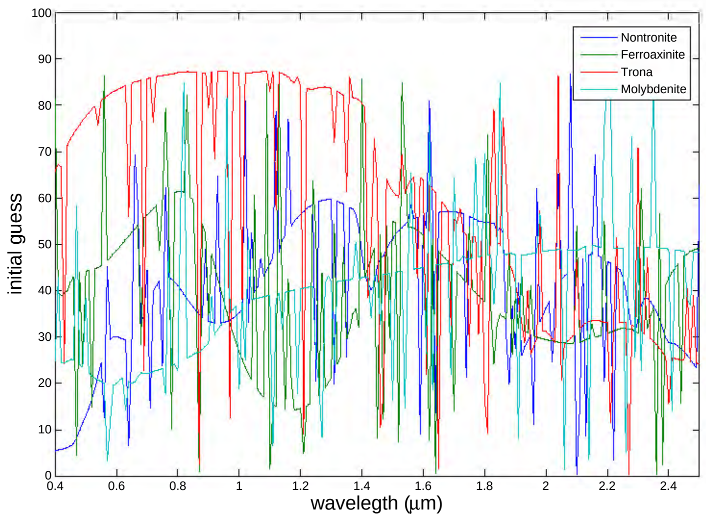


(b) Recovered signatures.

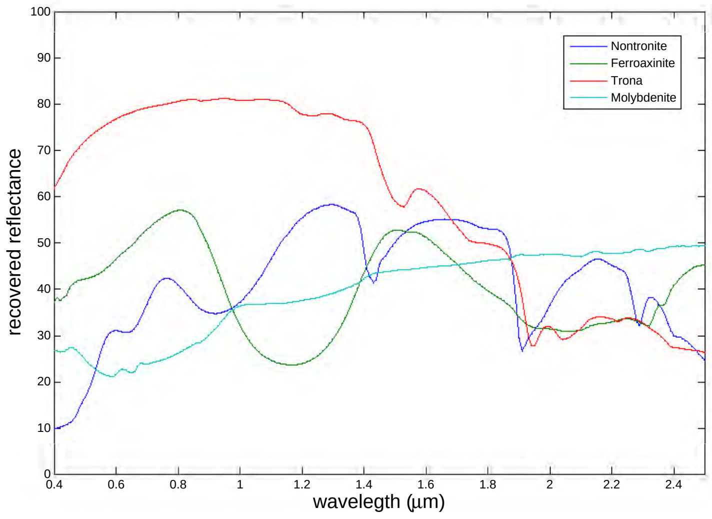


(c) Recovered abundance distributions.

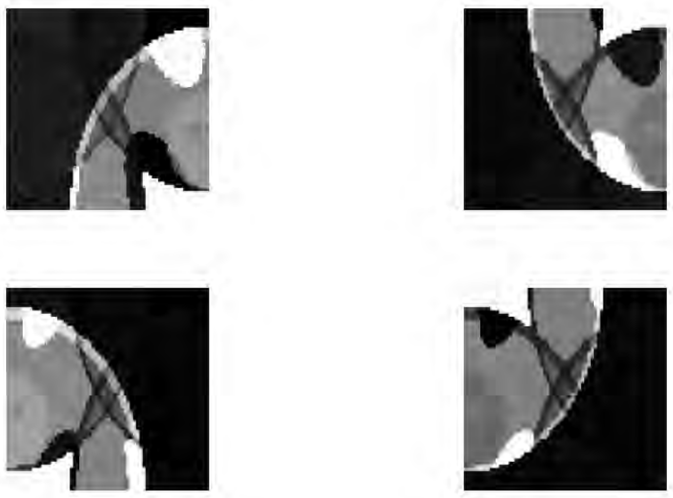
Figure 3.26: Removing the periodic noise involved in endmembers.



(a) Initial guess of signatures.

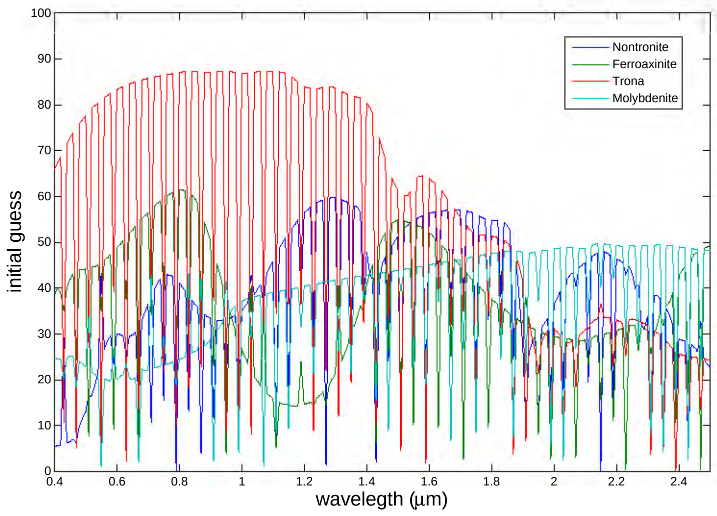


(b) Recovered signatures.

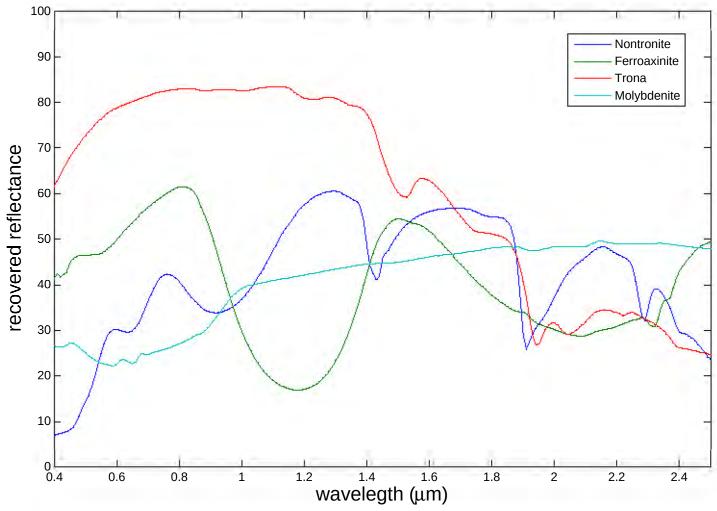


(c) Recovered abundance distributions.

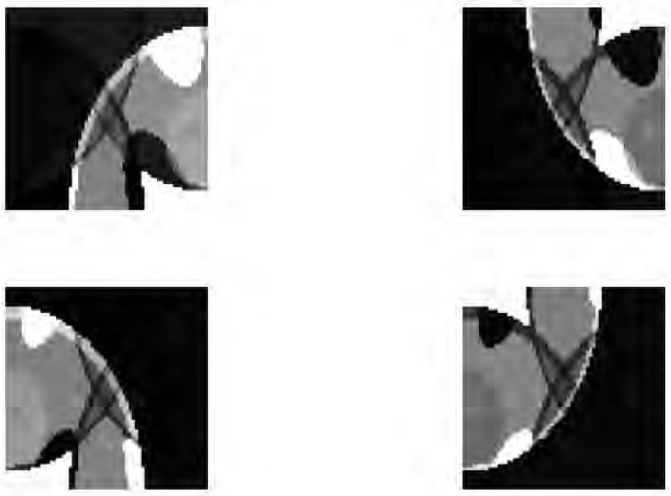
Figure 3.27: Removing the impulsive noise involved in endmembers (random positions corrupted).



(a) Initial guess of signatures.

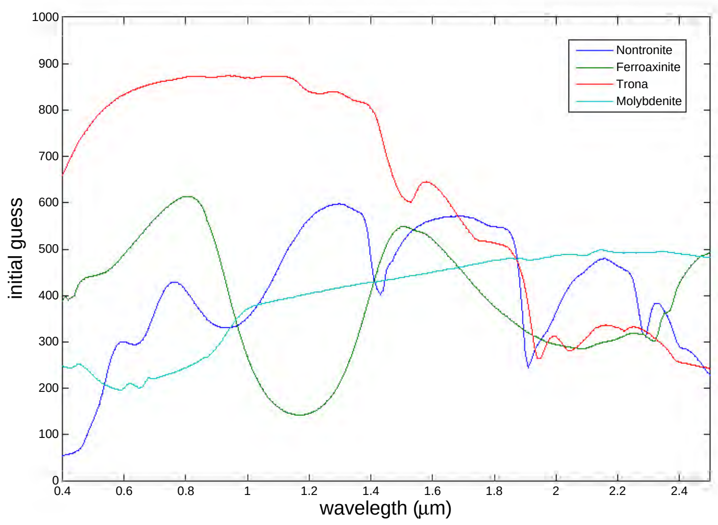


(b) Recovered signatures.

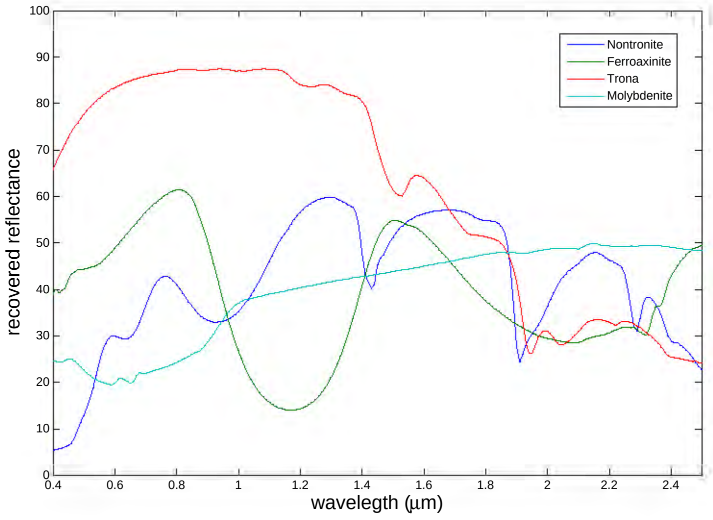


(c) Recovered abundance distributions.

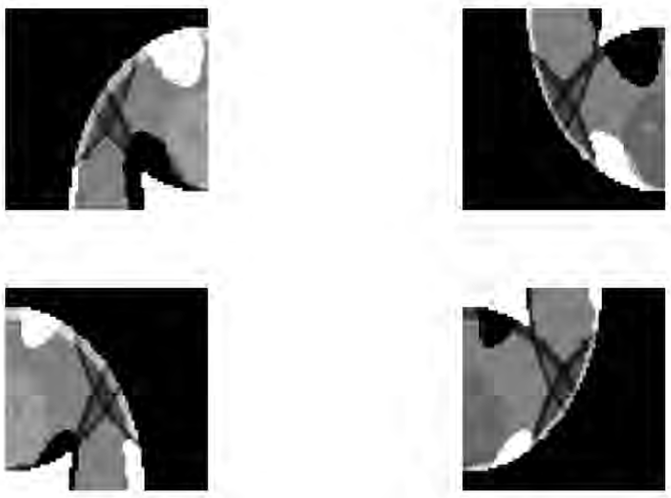
Figure 3.28: Removing the impulsive noise involved in endmembers (same positions corrupted).



(a) Initial guess of signatures.

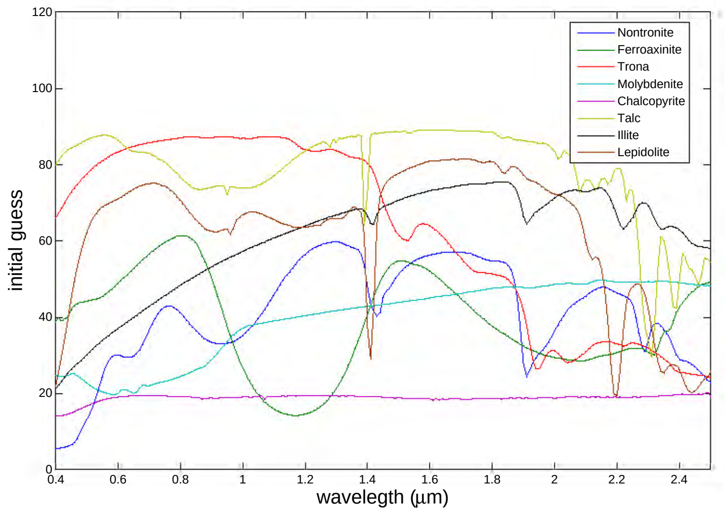


(b) Recovered signatures.

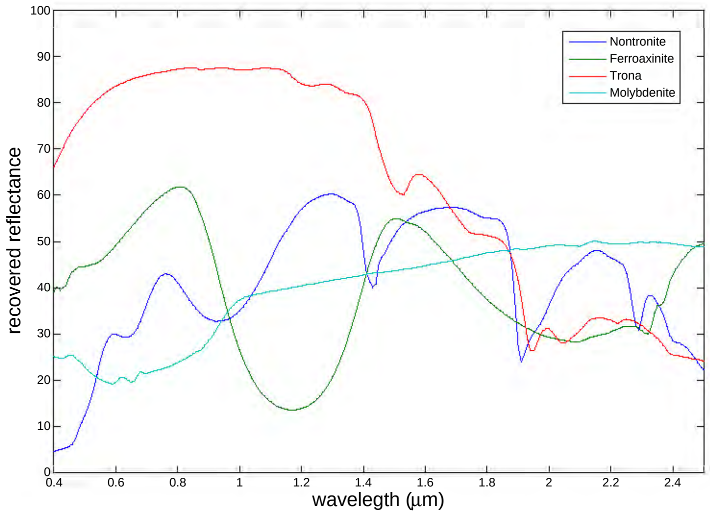


(c) Recovered abundance distributions.

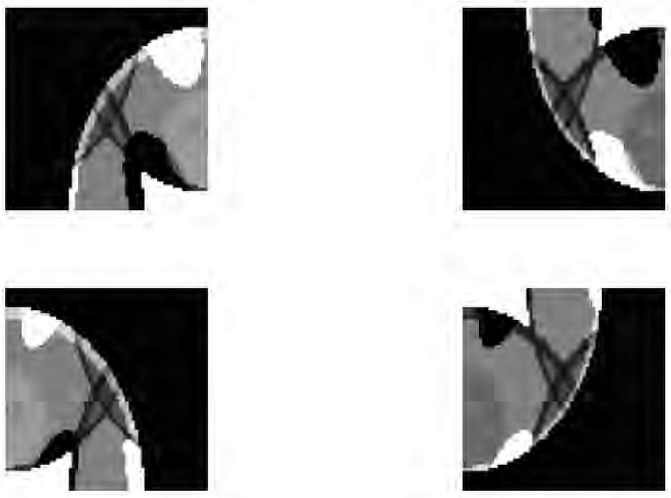
Figure 3.29: Correcting the wrong scale involved in endmembers.



(a) Initial guess of signatures.

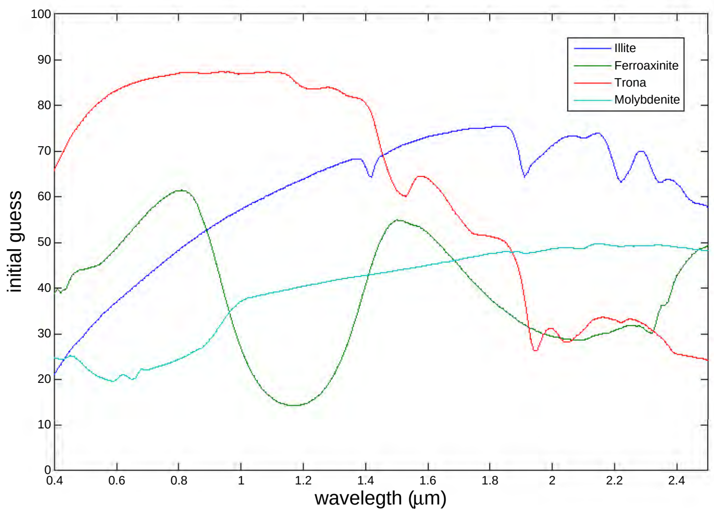


(b) Recovered signatures.

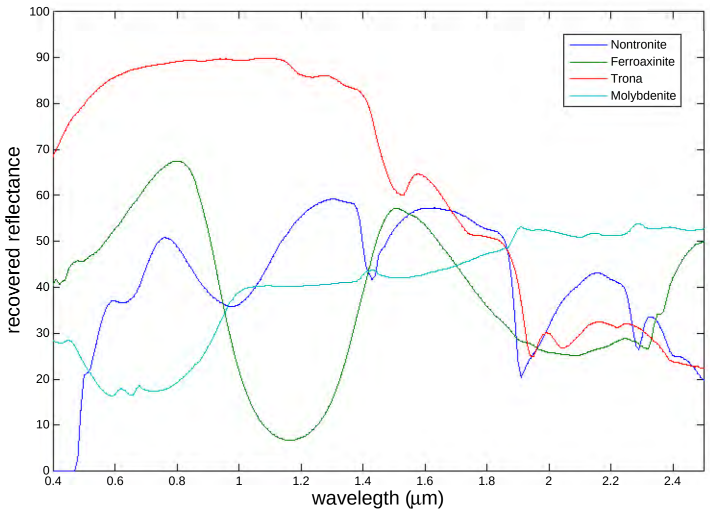


(c) Recovered abundance distributions.

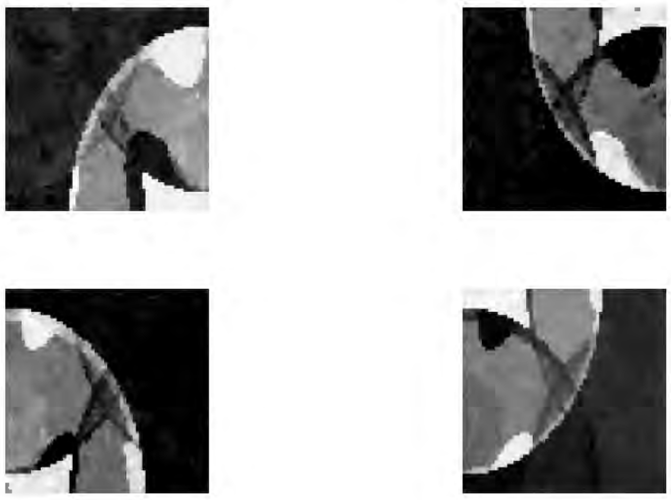
Figure 3.30: Selecting endmembers from candidates.



(a) Initial guess of signatures.

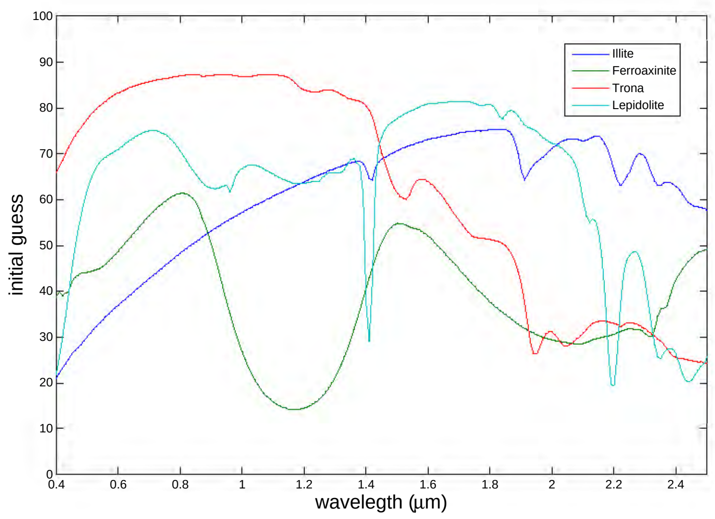


(b) Recovered signatures.

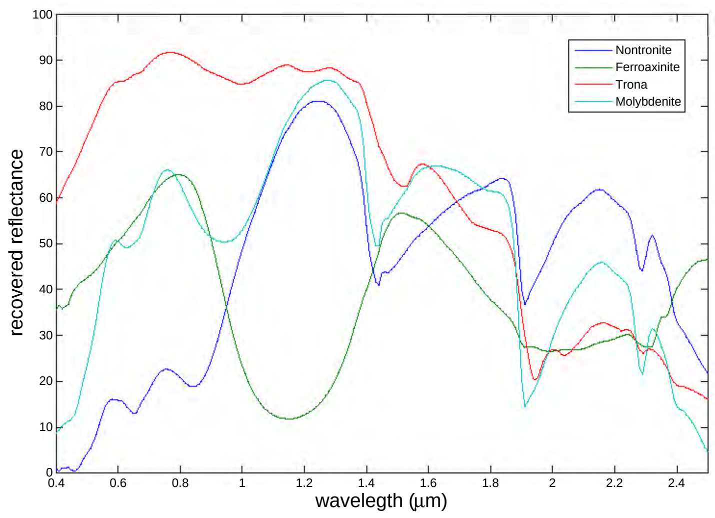


(c) Recovered abundance distributions.

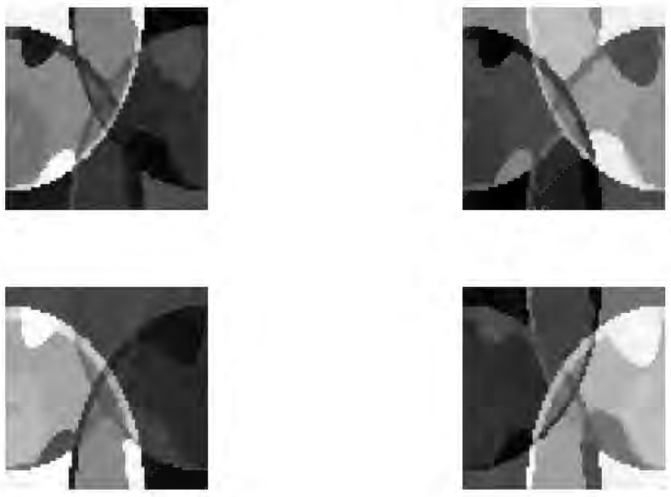
Figure 3.31: Unmixing from one endmember missing.



(a) Initial guess of signatures.



(b) Recovered signatures.



(c) Recovered abundance distributions.

Figure 3.32: Unmixing from two endmembers missing.

Chapter 4

Scalable Video Coding

The merits of compressive sensing have been demonstrated in many fields including the hyperspectral imaging covered in Chapter 3. It is well known that the sparsity plays a key role in compressive sensing. By exploring the temporal redundancy among adjacent frames, video data should be well sparsified. In this chapter, a framework of *compressive video sensing* (abbreviated CVS) is discussed and a new way to construct the sensing matrices is proposed, which together lead to a novel scheme of scalable video coding. An algorithm based on the general TVAL3 method is suggested for the purpose of decoding.

4.1 Introduction

Generally speaking, video is composed by a sequence of still images in order to depict scenes in motion, and each still image is known as a *frame*. It includes recording, processing, storing, transmitting, and reconstructing a series of frames. Two fundamental concepts of a video are *frame rate* and *display resolution*: frame rate refers to the number of frames per second, which ranges from 20 to 30 normally; display resolution refers to the number of pixels in each dimension, and may vary over a wide range due to various applications.

In video distribution applications, a video source may be transmitted to multiple clients with different characteristics. The clients in the video network have different channel capacities, different display resolutions, and different computing resources. For example, a video source may be transmitted through a network to a high-end computer with a high resolution monitor in a residential home, and at the same time, to a mobile device with a low resolution screen and a battery powered CPU. In other words, the quality of video which a client device can render is constrained by the channel conditions as well as the client's computing capabilities. Optimal video quality is achieved if the encoder is tuned specifically for those constraints. However, in typical video networks a video source is transmitted to multiple clients with different characteristics. Furthermore, the properties of the channels, especially wireless ones, may change over time, sometimes abruptly. Therefore, it is desirable to have a *scalable video coding scheme* — one in which the source is encoded once, the same code is used for transmission to all clients and each client can decode it to achieve a near-optimal viewing experience, subject to its own constraints, as depicted in Figure 4.1.

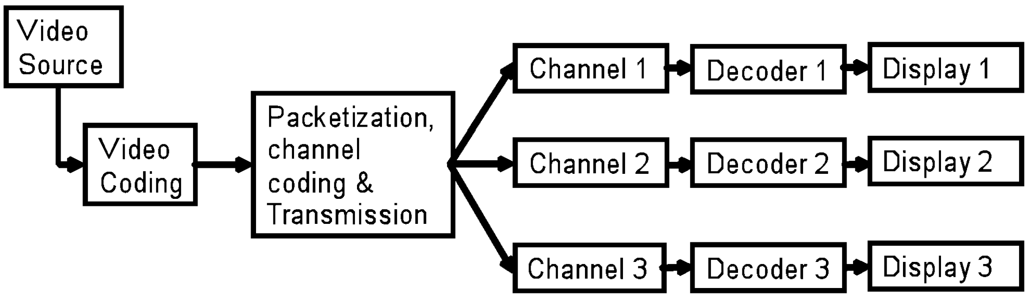


Figure 4.1: Diagram of a video network.

Standard video coders such as AVC [113] are not scalable. An encoded video can be decoded only at one resolution, with a fixed complexity (not considering post-processing such as resizing, or enhancement, after decoding). If the channel capacity is lower than encoded data rate, or if the decoding complexity exceeds the client's capability, video decoding would fail. On the other hand, if the channel capacity exceeds the encoded bit rate and the client has ample computational resources and a high resolution display, there is no way to get a better viewing experience which is commensurate with the available resources. This phenomenon is known as the *cliff effect* — no video is available unless some threshold is met and no improvement is achievable with extra capacities.

Scalable video coding (SVC) [114] introduces some scalability by encoding video into ordered layers, where each higher layer provides a refinement to the encoding by the lower layers. In this way, the resolution or quality of the decoded video increases progressively while higher layers are added. However, the loss of a lower layer in the transmission makes the higher layers useless, even when they are received error-free. Since decoding of lower layers is a prerequisite for the decoding of higher ones, the lower layers need a higher level of protection in transmission. In broadcast applications each layer may be transmitted as a separate stream, with different protection levels, e.g. by using hierarchical modulation [115]. In point to point applications a feedback from the client usually implies how many layers to be transmitted. In these situations, providing a different protection to each layer is usually difficult and inefficient; hence all layers get the same protection, which severely limits the scalability. Practical limitation of SVC necessitates a relatively small number of layers (due to compression efficiency considerations) which makes the scalability achieved by SVC rather limited. Typically one can maintain the same video quality by using a more complex encoder/decoder while reducing the bit rate, or vice versa. However, increasing the number of decoded SVC layers increases both the data rate and the decoder complexity. Furthermore, SVC features no scalability in resolution — either spatial or temporal. There have been attempts to introduce scalability in decoding resolution [116] and to achieve scalability by joint source and channel coding [121]. However, the problem of scalability of video coding is far from being solved.

In the wake of recent advances in compressive sensing as reviewed in Chapter 1, video coding using compressive measurements is rapidly emerging [33, 118]. Compressive video sensing represents the video as a set of independent measurements, each of which carries an equal amount of information about the source. The video may be decoded from any subset with a minimum number of the equally important measurements. Ideally, the reconstruction is exact as long as the number of measurements exceeds a theoretical bound. However, most video clips are not strictly sparse under any basis and exact recovery is unrealistic. Usually, the quality of rendering improves as more measurements are added [6], but is independent of the choice of specific measurements. Therefore, the transmitted data rate may be continuously adapted to the channel capacity. Furthermore, special protection is unnecessary for measurements in the sense that if one is lost in transmission it may be replaced by any other measurement. In addition, decoding is no longer the simple inverse process of encoding, but done by solving an optimization problem. Therefore, compressive sensing offers the scalability desired in video network [119, 118], allowing each client to receive as many measurements as its channel capacity allows and to use the best decoding method which its computational resources can sustain. As networks conditions change the decoded video quality may change, but the cliff effect vanishes.

In Section 4.2, we propose a framework for video coding using compressive measurements in which an encoded video is scalable with the channel capacity and with decoding complexity. Several algorithms, with different complexities, can be employed or extended for decoding, and we focus on the one implementing a TVAL3 method.

The quality of the decoded video depends only on the number of measurements and not on the particular measurements used. Therefore, if network conditions dictate a smaller number of measurements, we get graceful degradation with few objectionable artifacts. The effectiveness of compressive sensing is based on representing the signal at hand in a domain in which it is sparse. By exploring temporal redundancy, we present a novel sparse representation for video — spatial Total Variation (TV) combined with temporal Discrete Cosine Transform (DCT), TV$_S$-DCT$_T$, or simply TV-DCT, and show its effectiveness in simulations [96].

In order to meet the needs of large high-definition television screens, video has to be encoded at a high resolution. However, if the receiver is a small mobile device, the rendered resolution may be much smaller. For such small devices it is undesirable to decode the video at the original resolution and then down-sample it to the display resolution. In order to get a decent decoded quality in a high resolution, one needs many more measurements than would be necessary for a low resolution, and in a wireless network that amount of measurements may exceed the channel capacity. Furthermore, high resolution decoding requires more computational resources. In Section 4.3, a novel way to build sensing matrices by means of Kronecker products is suggested to achieve scalability. Built on this technique, a scalable resolution coding method is described [112], which allows encoding at a high resolution and direct decoding at a lower resolution.

Multi-resolution should not be confused with format conversion. Often a source video need to be converted into the client's display format, e.g. from VGA to CIF or from cinema frame rate to PAL or NSTC. These conversions are at non-integer but fixed ratios and they are performed after the decoding by a dedicated graphic processor. These issues are not discussed here. This chapter is concerned with reducing the video resolution as a part of the decoding process, in order to overcome the limited bandwidth or limited computational resources.

4.2 Compressive Video Sensing

A source video consists of a sequence of frames, each comprising three arrays of pixel values for the three color components (either YUV or RGB). To simplify the notation and description we only consider a single color component, say luminance, making it a black and white video source. The other two components may be treated similarly. In this setting a pixel position is represented by a three dimensional index vector $\mathbf{i} = (i_1, i_2, i_3)$, where i_1, i_2, i_3 are vertical and horizontal offsets, and the frame number, respectively. For higher efficiency, the video source is divided into volumes, which are contiguous, non-intersecting sets of pixels, and compressive sensing encoding is performed independently on each volume. For simplicity we assume that each volume is a cube of pixels of size $\mathbf{n} = (n_1, n_2, n_3)$ which occupies pixel positions $\{\mathbf{i} : \mathbf{o} \leq \mathbf{i} < \mathbf{o} + \mathbf{n}\}$, where $\mathbf{o}$ is the top-left-oldest corner of the cube and inequality signs between index vectors indicate inequality for each component. The total number of pixels in a volume is denoted by $n \triangleq n_1 n_2 n_3$. This framework can be easily extended to volumes of non-regular shape, which could be utilized, for example, for the purpose of motion estimation and compensation.

Except for dealing with boundaries of adjacent volumes, the processing of each volume is independent of the others and we can assume that $\mathbf{o}$ is the origin. Thus the cube is defined by its pixel values $P \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_3}$, where $P_{\mathbf{i}}$ is the pixel value at position $\mathbf{i} \in \mathbb{Z}^{n_1} \times \mathbb{Z}^{n_2} \times \mathbb{Z}^{n_3}$ where $\mathbb{Z}^n$ denotes the set $\{1, \ldots, n\}$.

4.2.1 Encoding Using Compressive Sensing

Compressive sensing encoding, as depicted in Chapter 1, is a straightforward linear operation. Nevertheless, some caution is necessary in dealing with the conversion of the 3D volume into the 1D signal from which the measurements are taken.

Let $\mathcal{T}: \ \mathbb{Z} \to \mathbb{Z}^{n_1} \times \mathbb{Z}^{n_2} \times \mathbb{Z}^{n_3}$ be a one-to-one mapping of the 1D indices $1, \ldots, n$ into 3D pixel indices. Let $\mathcal{S}: \ \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_3} \to \mathbb{R}^n$ be a mapping from the volume into a 1D signal performing a column-by-column scan of a cube. Mathematically,

$$\mathcal{S}(P)_i = P_{\mathcal{T}(i)}. \tag{4.1}$$

Furthermore, let

$$x = \mathcal{S}(P), \tag{4.2}$$

and then the measurement vector y is given by

$$y_i = \sum_{j=1}^{n} a_{ij} P_{\mathcal{T}(j)}, \qquad \text{for } 1 \leq i \leq m, \tag{4.3}$$

or equivalently, by

$$y = Ax. \tag{4.4}$$

This process is illustrated in Figure 4.2.

As indicated in Chapter 1, the measurement matrix A should be incoherent with the sparsity basis of the video volume. A large number of matrices meet this requirement, including randomly generated ones. Considering the advantages of easy implementation on hardware, fast transformation, and satisfactory recoverability, randomly permutated Walsh-Hadamard (abbreviated WH) matrices are used in our tests.

It is noteworthy that if n is sufficiently large, generated measurements behave

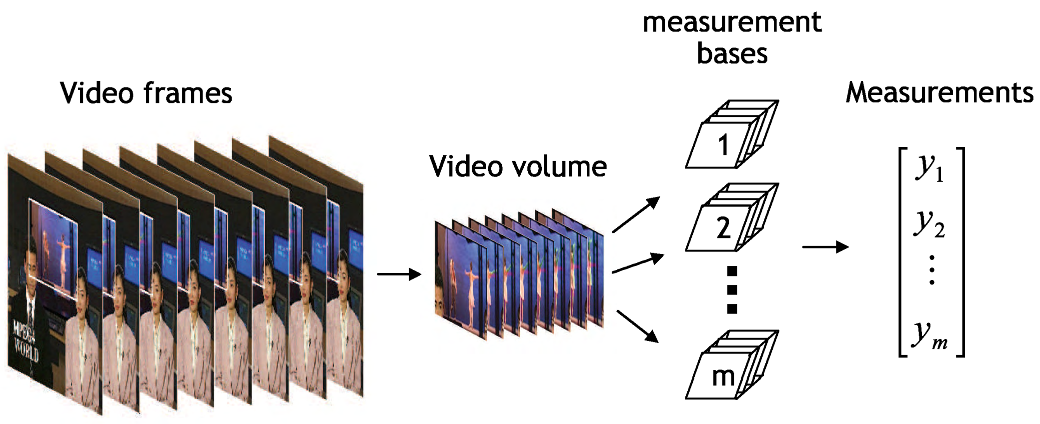


Figure 4.2: Video coding using compressive sensing.

similarly to uncorrelated Gaussian random variables. We consider the entries of a random matrix to be independent random variables, and therefore, each measurement is a weighted sum of these random variables with weights being the pixel values of the volume. According to the central limit theorem in statistics, the sum has approximately the Gaussian distribution if the number of terms in the sum is large enough. Since the rows of A are uncorrelated, the measurements are uncorrelated as well. This distribution of the measurements enables efficient encoding and quantization for transmission.

4.2.2 TV-DCT Method for Decoding

Traditional video coding methods, such as MPEG-1 and MPEG-2, decode a video by inverting the encoding process. However, it is no longer the case for compressive video sensing. The task of the decoder becomes to estimate x from the underdetermined system (4.4) given the measurement vector y and sensing matrix A, afterwards the pixel values in the video volume can be obtained by back-projecting $P^* = \mathcal{S}^{-1}(x^*)$, where x^* is the approximation of true x. In light of compressive sensing, x^* can be found by solving the constrained minimization problem

$$\min_x \Phi(x), \quad \text{s.t. } Ax = y, \tag{4.5}$$

where $\Phi(x)$ stands for a regularization term. Due to quantization at the encoder or network errors, the measurements received by the decoder may contain noise; hence forcing the equality constraint would be unnecessary. Instead, we consider an unconstraint minimization problem to recover x^*:

$$\min_x \left\{ \Phi(x) + \frac{\mu}{2}\|Ax - y\|^2 \right\}, \tag{4.6}$$

where the second term penalizes the violation of the fidelity term with a balancing parameter μ.

In compressive sensing, the regularization term is in the form of the following:

$$\Phi(x) = \|Lx\|_1, \tag{4.7}$$

where L is a linear transformation which yields the coefficients of a sparse representation of x according to the selected sparsity basis. Typical choice of L leads to the extensively used ℓ_1 or TV regularization, which has been introduced in Chapter 1. Especially in the realm of imaging, the merits of TV regularization has been indicated in former chapters, as well as some other literature. The transformation suggested here for video decoding consists of a temporal DCT followed by spatial total variation.

Typically there is a high correlation between pixels in the same spatial position of different frames. For example, the still background in a video clip signifies the same pixel value in the same position of distinct frames. We apply the temporal DCT to all pixels which share the same spatial position. Former notation gives $\mathbf{i} = \mathcal{T}(i)$, and then

$$\begin{aligned}\tilde{x}_i &= \tilde{P}_{\mathbf{i}} \\ &= \sum_{j=1}^{n_3} P_{i_1,i_2,j} \cos\left[\frac{\pi}{n_3-1}\left(j-\frac{1}{2}\right)\right] \\ &= \sum_{j=1}^{n_3} x_{T^{-1}([T(i)_1,T(i)_2,j])} \cos\left[\frac{\pi}{n_3-1}\left(j-\frac{1}{2}\right)\right]. \end{aligned} \tag{4.8}$$

After the DCT transform, the first frame represents the component which is unchanged over time and subsequent frames represent components of increasing time variability. If there are no temporal changes in the video volume, as the example mentioned above, only the first transformed frame is non-zero. Therefore, the temporal DCT generally enhances the sparsity or compressibility of the video source.

If each frame consists of a small number of regions with uniform pixel values, then it could be sparsely represented by the boundaries between the regions and the pixel differences. This information can be extracted by applying a gradient operator

$$(\nabla x)_i \triangleq \left[P_{T(i)+[1,0,0]} - P_{T(i)}, P_{T(i)+[0,1,0]} - P_{T(i)}\right], \tag{4.9}$$

with carefully chosen boundary conditions, such as periodic or Neumann. Real DCT-transformed video frames may only be approximated by such models, hence their gradient is only nearly sparse. However, as explained before, this may suffice for adequate reconstruction.

Similar to the regular TV used in former chapters, the spatial total variation (abbreviated TV_s) is defined by

$$\mathrm{TV}_s(x) = \sum_{i=1}^{n} \|(\nabla x)_i\|_p, \tag{4.10}$$

where $p = 2$ corresponds to the *isotropic* TV_s and $p = 1$ corresponds to the *anisotropic* TV_s. Total variation has been widely and successfully used in image processing [14, 6, 46], and recently in hyperspectral imaging as investigated in Chapter 3. In the process of decoding, spatial TV is applied to the temporal DCT of the signal; hence

$$\Phi(x) \triangleq \mathrm{TV}_s(\mathrm{DCT}_t(x)) = \sum_{i=1}^{n} \|(\nabla\tilde{x})_i\|_p, \tag{4.11}$$

with $\tilde{x}$ defined in (4.8). Intuitively, the proposed TV-DCT regularization is illustrated in Figure 4.3.

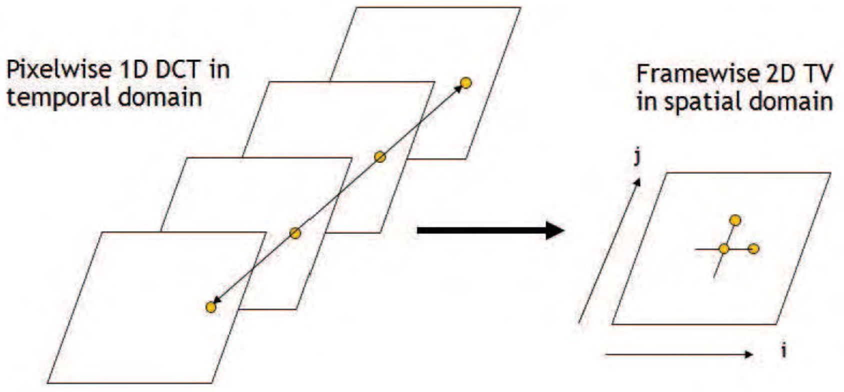

Pixelwise 1D DCT in temporal domain
Framewise 2D TV in spatial domain
j
i

Figure 4.3: TV-DCT regularization.

For simplicity, let $T_i \in \mathbb{R}^{2\times n}$ denote the linear transformation satisfying

$$T_i x = (\nabla\tilde{x})_i, \tag{4.12}$$

for $1 \le i \le n$. Then the decoding model (4.5) becomes

$$\min_x \sum_{i=1}^{n} \|T_i x\|_p, \quad \text{s.t. } Ax = y. \tag{4.13}$$

It is noteworthy that (4.13) is the same type as the instance we discussed in Section 2.3.1 and can be efficiently solved by means of the general TVAL3 algorithm. As a reminder, the augmented Lagrangian function of (4.13) is split into two parts by introducing a set of new variables $w_i = T_i x$ for $1 \leq i \leq n$. One part is separable with closed-form minimizer whereas the other part is quadratic. The general TVAL3 method employs the alternating direction in an inexact way to minimize the augmented Lagrangian function, and attains the global minimizer of (4.13) eventually. In the presence of noise involved in measurements, one can employ the same algorithm to solve the unconstraint model (4.6) with TV-DCT regularization.

So far we have proposed a so-called TV-DCT method to decode a video stream. A general TVAL3 algorithm is suggested to handle the model with TV-DCT regularization to achieve scalability on rendering quality and complexity. Furthermore, compressive video sensing has the virtue of simple encoding without requiring special protection and open-ended decoding — the decoding algorithm is not limited to the suggested one but a variety of others. A better reconstruction algorithm potentially developed in the future would lead to a stronger performance of TV-DCT method.

4.3 Multi-Resolution Scheme

An advantage of using a random-like matrix, such as a permutated Walsh-Hadamard matrix, as the measurement matrix is that the measurements of the video are equally important, so that the quality of the reconstructed video only depends on the number of measurements available, but independent of the availability of any particular measurement. This is the property that makes the coding inherently scalable. Besides, there is still more to be desired. Suppose the measurements described in Section 4.2.1 are transmitted, and due to a low channel capacity, only very few measurements are correctly received at the decoder. The number of received measurements may be too small to reconstruct a video of the original resolution with an acceptable quality. It is possible to use the received measurements to reconstruct a video of the original resolution, and then resize it to a lower resolution, but the quality of the downsized video is inherently limited by that of the reconstructed video, although the smaller size of the downsized video may make some undesirable artifacts less obvious. Therefore, an alternative method is proposed in the following in which a video of lower resolution is reconstructed directly using those few measurements that are correctly received.

Suppose $\tilde{x} = Rx, R \in \mathbb{R}^{r\times n}, r < n$ is a signal of length r which, in some sense, is an approximation of $x \in \mathbb{R}^n$. R is the reduction matrix and it may be based on interpolation, filtering, or any other well known techniques of rate reduction [120]. The low resolution may be spatial (fewer pixels per frame), temporal (lower frame rate) or a combination of both. Low resolution decoding refers to estimating $\tilde{x}$ from available measurements y directly, that is, without recoverying x first. Typically low resolution decoding is attempted either in order to reduce the number of measurements needed for reliable decoding or in order to reduce the required amount of computation. Before showing how low resolution decoding may be done, let us first examine if and when the number of required measurements can be decreased. The reduction of required computation will become clear along with further discussions.

4.3.1 Theoretical basis of Low Resolution Reconstruction

Based on the theory of compressive sensing [2], the theoretical bound of measurements required for reconstructing a signal is

$$m_0 = c_\varepsilon k \log n,$$

where c_ε is some constant corresponding to the given accuracy ε and k is the sparsity of x. After reducing the resolution from n to r, the theoretical minimum becomes

$$\tilde{m}_0 = \tilde{c}_\varepsilon \tilde{k} \log r,$$

where $\tilde{k}$ is the sparsity of the low resolution signal. Replacing n by $r < n$ reduces the bound but the impact is small because of the logarithm. The key of having $\tilde{m}_0$ smaller than m_0 is to have $\tilde{k} \leq k$. However, this is by no means guaranteed in general, if assuming no connection between the sparsity basis and the rate reduction procedure. The following sufficient condition ensures that rate reduction will at least maintain the same level of sparsity:

Proposition 4.3.1. *Let $b^1, \ldots, b^n$ be a sparsity basis in $\mathbb{R}^n$ and let $\tilde{C} = \{Rb^1, \ldots, Rb^n\} - \{0\}$ be the set of non-zero vectors in the low resolution image of the sparsity basis vectors. If the vectors in $\tilde{C}$ are linearly independent, then $\tilde{k} \leq k$.*

Proof. Due to the linear independency, $\tilde{C}$ can be extended into a basis $\tilde{B} \supset \tilde{C}$ in the low resolution space. Let x be a k-sparse vector with a representation in the sparsity basis; i.e.,

$$x = \sum_{j=1}^{k} \alpha_j b^{i_j}.$$

Then

$$Rx = \sum_{j=1}^{k} \alpha_j R b^{i_j}.$$

If we eliminate zero summands in the right hand side, if any, we get a representation of Rx by no more than k different basis vectors. In other words,

$$\tilde{k} \leq k.$$

□

At the beginning of Section 4.2.2, we posed two decoding models to handle both noise-free and noisy situations respectively. The unconstraint model (4.6) is adopted here because the gap between an approximation of lower resolution and the signal of original resolution is mostly unavoidable. The penalty term in the unconstraint model may prohibit the occurrence of overfitting. In order to solve the minimization problem (4.6) for a lower resolution signal $\tilde{x}$ decoding, we need to replace the original signal x by introducing an expansion matrix $E \in \mathbb{R}^{n\times r}$ which is designed so that $E\tilde{x}$ would be a good approximation of x. Then (4.6) becomes

$$\min_{\tilde{x}} \left\{ \Phi(\tilde{x}) + \frac{\mu}{2}\|AE\tilde{x} - y\|^2 \right\}. \tag{4.14}$$

This is also equivalent to

$$\min_{\tilde{x}} \left\{ \Phi(\tilde{x}) + \frac{\mu}{2}\|Ax - y\|^2 + \frac{\mu}{2}\|A(E\tilde{x} - x)\|^2 \right\}, \quad \text{s.t. } \tilde{x} = Rx. \tag{4.15}$$

Thus a noise induced penalty term, $\|A(E\tilde{x} - x)\|^2$, arises in the minimization. In the reconstructed video of lower resolution, the error caused by this noise is proportional to the noise level [3], which is proportional to error associated with the reduction to lower resolution. Therefore, the detrimental effect of this noise term on (4.14) is at the same level as the unavoidable degradation of a low resolution.

Note that the sufficient condition has been defined in term of the reduction matrix R, yet it is E, not R which appears in (4.14). However, R is constrained by E because of the requirement $RE = I$. A well constructed reduction matrix R has the full rank, and therefore, the expansion matrix E can be obtained, for example, from the onesided inverse of the reduction matrix R, as

$$E = R^T(RR^T)^{-1}. \tag{4.16}$$

4.3.2 Illustration of Low Resolution Reconstruction

Equations (4.4) and (4.14) constitute a basis for video coding in which one encoding fits a variety of channels and display resolutions in accordance with specific requirements. This is illustrated in Figure 4.4(a).

In Figure 4.4(a), the source video is encoded using a random measurement matrix. The encoded video is transmitted, for example, in a broadcast system, and the correctly received measurements are used to reconstruct video of a desired resolution by using an appropriate expansion matrix E. More precisely, decoder $i \in \{1, 2, 3\}$ with channel capacity C_i may use an expansion matrix E_i to reconstruct a video of certain resolution by substituting $E = E_i$ in (4.14).

An alternative, but unfavorable, encoding and transmission scheme is shown in Figure 4.4(b). In Figure 4.4(b), to transmit the source video to decoder $i \in \{1, 2, 3\}$ with channel capacity C_i, the source video is first down-sized to a resolution suitable for the display of the i-th decoder by using a reduction matrix R_i. The down-sized video is encoded using a random matrix A_i. The compressive measurements are transmitted and the correctly received measurements are used to reconstruct the video of the same resolution as the down-sized video by substituting $A = A_i$ in (4.4) and (4.6). Clearly, the system in Figure 4.4(a) is more preferable in the sense that no separate encoding is demanded and the same piece of measurements is capable of decoding at various resolutions. More importantly, Figure 4.4(a) indicates no need of a prior knowledge on decoding side.

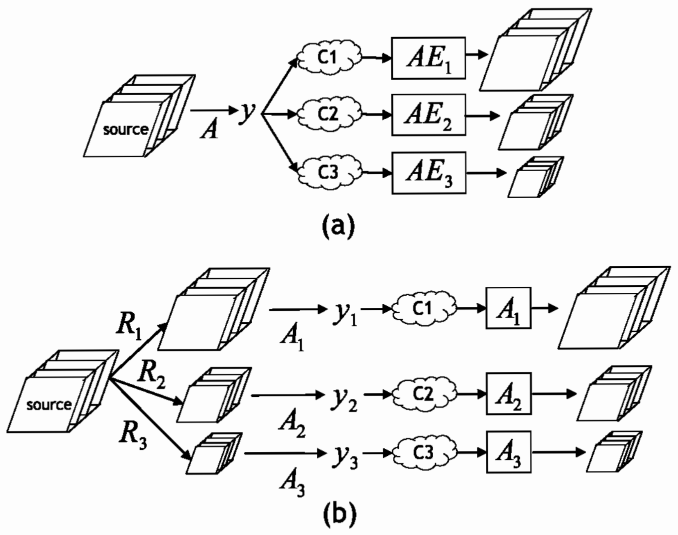


Figure 4.4: Flowchart of two schemes: encoding, transmission and reconstruction.

4.3.3 A Novel Idea to Build Scalable Sensing Matrices

The implementation of the framework described above may result in a very high complexity because of the evaluation of $AE\tilde{x}$. Unless the matrices are constructed with some special structures, either the complexity of $AE\tilde{x}$ is proportional to that of the original resolution if the computation is performed as $A(E\tilde{x})$, or a large memory (to store the matrix AE) and a matrix-matrix multiplication are required if the computation is performed as $(AE)\tilde{x}$. Therefore, it is highly desirable to simplify the computation of $AE\tilde{x}$ for video applications due to the limited resources available at decoders. In this part, a new way to construct sensing matrices is proposed by means of Kronecker products. The special structure of this type provides much lower complexity as well as multi-resolution reconstruction. The discussion here is restricted to lowering the spatial resolution.

For any two matrices $A = [a_{ij}]_{p\times q}$ and $B = [b_{ij}]_{r\times l}$, the *Kronecker product* of A

and B is defined as

$$A \otimes B \;=\; \begin{bmatrix} a_{11}B & a_{12}B & \ldots & a_{1q}B \\ a_{21}B & a_{22}B & \ldots & a_{2q}B \\ \vdots & \vdots & \vdots & \vdots \\ a_{p1}B & a_{p2}B & \ldots & a_{pq}B \end{bmatrix}_{pr \times ql}.$$

A fundamental property of Kronecker product is the following:

Proposition 4.3.2. *Suppose that $A \in \mathbb{R}^{m\times n}$ is given by*

$$A = A_1 \otimes A_2,$$

where $A_1 \in \mathbb{R}^{(m/p)\times(n/q)}$ and $A_2 \in \mathbb{R}^{p\times q}$. m and n are chosen such that m and n are divisible by p and q, respectively. Then matrix-vector multiplication can be computed by

$$\begin{cases} Ax = vec(A_2 mtx(x) A_1^T), \\ A^T y = vec(A_2^T mtx(y) A_1), \end{cases} \tag{4.17}$$

where "vec" represents the operator that stacks the columns of a matrix into a vector, and "mtx" represents the inverse operator of "vec".

Generally speaking, the complexity of computing Ax for $A \in \mathbb{R}^{m\times n}$ is $2mn$. In light of Proposition 4.3.2, the complexity decreases to $2(p+m/p)n$, due to the structure of Kronecker product. Besides, the physical storage of A can be reduced from mn down to $pq+mn/pq$. In addition, if both A_1 and A_2 are implemented using some fast transforms this proposition points out a way to efficiently evaluate Ax without acquiring A explicitly.

Another well-known property of Kronecker product is the following:

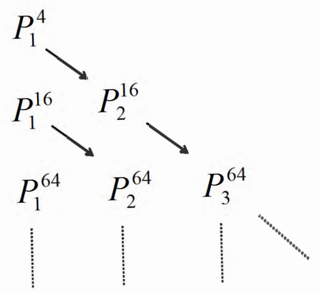


Figure 4.5: Recursive construction of vectorized permutation matrices.

Proposition 4.3.3. *Suppose that A_1, A_2, A_3 and A_4 are matrices with proper sizes. Then*

$$(A_1 \otimes A_2)(A_3 \otimes A_4) = (A_1A_3) \otimes (A_2A_4). \tag{4.18}$$

After going through these preliminaries, we elaborate on the construction of new sensing matrices suitable for multi-resolution schemes. Specifically, the measurement matrix A comes from the Kronecker products of small sensing matrices and structured permutation matrices. First, a predetermined number of decoding resolutions is specified. Each resolution will be called a “level”. Then A is constructed for the specified number of decoding levels. The video is encoded by the compressive measurements of video cubes using this A. The same measurements may be used to reconstruct a video of any one of the resolution levels. The process of building the measurement matrix A can be divided into four steps:

1. Specify an integer $k > 0$ as the encoding level, which determines the lowest resolution a video can be reconstructed from the encoded video. Specifically, the encoded video of the spatial resolution $n_1 \times n_2$ can be decoded to one of the resolutions $(n_1/2^l) \times (n_2/2^l)$, for $l = 0, \dots, k$.

   For the convenience of description, we assume the dimensions $n_1/2^l$ and $n_2/2^l$ are always integers. For non-divisible cases, the scheme is still applicable with mild modifications.

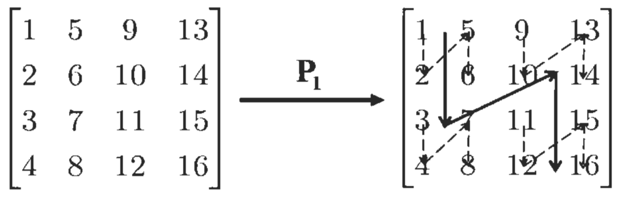


Figure 4.6: Demo of the initial permutation matrix: perform permutations on a 4×4 block.

2. Construct a series of permutation matrices $P_1^n, P_2^n, \ldots, P_k^n$, named *block-wise vectorized permutations*, by recursion:

$$P_i^n = P_{i-1}^{n/4} \otimes I_4, \text{ for } i = 2, 3, \ldots, k \tag{4.19}$$

   where $P_i^s \in \mathbb{R}^{s \times s}$ and I_4 represents the 4×4 identity matrix. This recursive reconstruction is illustrated in Figure 4.5.

   For all $s = 4, 16, 64, \ldots,$ P_1^s is a permutation matrix following a particular build-up scheme based on 2×2 blocks. For example, P_1^{16} is the permutation matrix that works in the way illustrated in Figure 4.6. In other words, let $u = [1, 2, 3, 4, 5, 6, 7, 8, \ldots, 13, 14, 15, 16]^T$ be the column vector formed by concatenating the columns of the matrix on the left hand side of Figure 4.6. Then $P_1^{16}u = [1, 2, 5, 6, 3, 4, 7, 8, \ldots, 11, 12, 15, 16]^T$. In general, for a square matrix U of dimension n, u is the column vector formed by concatenating the columns of the matrix U, and $P_1^n u$ is the column vector formed by first dividing the matrix U into blocks of four elements (2×2 blocks), and then, concatenating the columns of each 2×2 block followed by concatenating all these 2×2 blocks column by column.

   From this point on, we will omit the superscript of P_i^s for simplicity. Its size can be determined by properly forming the matrix product.

3. Select a series of small sensing matrices $A_0 \in \mathbb{R}^{m_0 \times (n/4^k)}$ and $A_i \in \mathbb{R}^{m_i \times 4}$ for

$1 \leq i \leq k$ which satisfy

$$\prod_{i=0}^{k} m_i = m. \tag{4.20}$$

The choice of $m_0, m_1, \ldots, m_k$ is definitely not unique based on (4.20). One feasible way is to choose m_0 close to $n/4^k$ or as large as possible, and meanwhile choose $4 \geq m_1 \geq \cdots \geq m_k \geq 1$. The underlying reason is that m_0 decides the number of valid measurements corresponding to the lowest resolution, and bigger m_0 gives higher quality of reconstruction at the lowest resolution. Similar argument explains the reason for choosing $m_1 \geq \cdots \geq m_k$. The upper bounds of $m_0, m_1, \ldots, m_k$ are chosen to avoid overcompleteness. Here we just pointed out one strategy, and other strategies require further research along this road.

4. Let

$$Q_k = P_k P_{k-1} \cdots P_1, \tag{4.21}$$

and

$$A = (A_0 \otimes A_1 \otimes \cdots \otimes A_k) Q_k, \tag{4.22}$$

which gives the measurement matrix scalable for multi-resolution purpose.

On the encoding side, it follows from (4.4) and (4.22) that

$$y = (A_0 \otimes A_1 \otimes \cdots \otimes A_k) Q_k x. \tag{4.23}$$

Since Q_k acts as a permutation matrix, $Q_k x$ can be regarded as a reordering of x. For any $0 \leq \tilde{k} < k$, the Kronecker products in (4.23) can be decomposed into two parts:

$$\begin{cases} A' = A_0 \otimes \cdots \otimes A_{\tilde{k}}, \\ A'' = A_{\tilde{k}+1} \otimes \cdots \otimes A_k, \end{cases} \tag{4.24}$$

whose orders are $m' \times n'$ and $m'' \times n''$ respectively. We partition the vectors $Q_k x$ and y into segments x^i for $i = 1, \ldots, n'$ and y^i for $i = 1, \ldots, m'$ of lengths n'' and m'' respectively; i.e.,

$$\begin{cases} x^i_j = (Q_k x)_{in''+j} \quad \text{for} \quad j = 1, \ldots, n'', \\ y^i_j = y_{im''+j} \quad \text{for} \quad j = 1, \ldots, m''. \end{cases} \tag{4.25}$$

Then (4.23) is equivalent to

$$\begin{bmatrix} (y^1)^T \\ \vdots \\ (y^{m''})^T \end{bmatrix} = A' \begin{bmatrix} (A''x^1)^T \\ \vdots \\ (A''x^{n''})^T \end{bmatrix} \tag{4.26}$$

If we consider each x^i as representing a local neighborhood in the signal, then (4.26) implies that the measurement can be performed in two stages:

1. first local measurements are computed on each locality x^i separately, using the sensing matrix A'';

2. then the final measurement vector is generated by applying A' to the local measurements.

This structure explains why Kronecker products can carry out low resolution decoding, where each local neighborhood needs to be replaced by a low resolution approximation.

If we want each x^i to represent a local neighborhood in the video volume, the index mapping $\mathcal{T}$, as defined in Section 4.2.1, has to map the indices of each segment to neighboring voxels. More specifically, if the video cube is partitioned into n' sub-cubes of n'' pixels, then one has to map the indices of each sub-cube onto the indices of one segment x^i as illustrated in Figure 4.7. However, x is a vector originally composed by

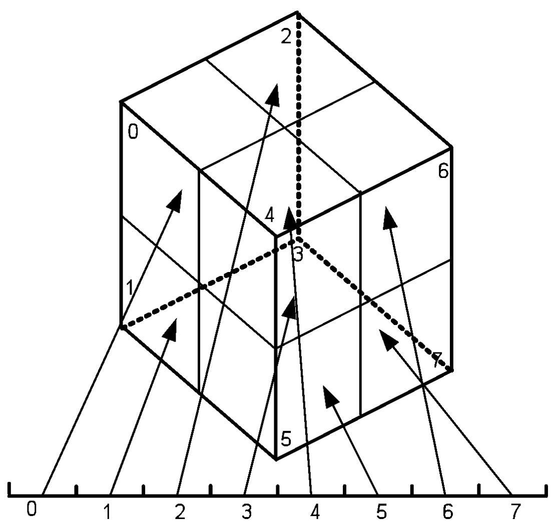


Figure 4.7: Diagram of the mapping $\mathcal{T}$: a segment is mapped into the indices of a cube such that each sub-segment is mapped into the indices of a sub-cube.

stacking pixel values of a video cube column by column, and then, frame by frame. Thus we need to reorder x to cluster pixels according to neighborhoods. This is done by constructing permutation matrix Q_k and performing it on x first.

Next, let us now look at the computational aspects of rate reduction from a decoding perspective. Minimizing (4.14) is an iterative process which requires repeated calculation of $AE\tilde{x}$. Typically calculating AE once and storing it in memory is neither efficient nor practical because of the large sizes of these matrices. In the following we show how the multi-resolution structure can greatly simplify this calculation. Some notations will be introduced before getting into details.

Level l $(l \leq k)$ *decoding* refers to the resolution of the reconstructed video cube being $(n_1/2^l) \times (n_2/2^l)$ and $U_l \in \mathbb{R}^{(n_1/2^l)\times(n_2/2^l)\times r}$ denotes the level l resolution approximation of a video cube U. In other words, U_l is the video having a resolution of $(n_1/2^l) \times (n_2/2^l)$ reconstructed from the compressive measurements made from the original video cube U of the resolution $n_1 \times n_2$. Vectors x and x_l represent the vectorizations of U and U_l, respectively, by concatenating the pixels of video cubes column by column, and then, frame by frame. For convenience, $\mathbf{1}_{s\times t}$ represents a $s \times t$ matrix whose entries are 1 everywhere. The second dimension of subscript t can be omitted if $t = 1$. For instance, $\mathbf{1}_4$ denotes $[1, 1, 1, 1]^T$ while I_4 denotes a 4×4 identity matrix. Furthermore, $B^{\circ j}$ denotes the j-degree power of Kronecker products, i.e.,

$$B^{\circ j} = \underbrace{B \otimes \cdots \otimes B}_{j}. \tag{4.27}$$

Given a low resolution representation U_l, one straightforward way to approximate U of original resolution is to copy each pixel to its neighborhoods; i.e.,

$$U \simeq U_l \otimes \mathbf{1}_{2^l \times 2^l}, \tag{4.28}$$

Due to the way of constructing permutation matrices $\{P_i\}_1^k$, (4.28) implies

$$P_l \cdots P_1 x \simeq x_l \otimes \mathbf{1}_{4^l} = x_l \otimes \mathbf{1}_4^{\circ l}. \tag{4.29}$$

Therefore, we can define the expansion matrix E for level l ($l \leq k$) decoding as follows:

$$Ex_l = P_1^T \cdots P_l^T (x_l \otimes \mathbf{1}_4^{\circ l}). \tag{4.30}$$

Then, we have

$$AEx_l = A(P_1^T \cdots P_l^T (x_l \otimes \mathbf{1}_4^{\circ l})). \tag{4.31}$$

Given $P_k = P_{k-l} \otimes I_4^{\circ l}$, it follows from (4.31), (4.22) and (4.21) that

$$\begin{aligned}
AEx_l &= (A_0 \otimes A_1 \otimes \cdots \otimes A_k) \cdot Q_k(P_1^T \cdots P_l^T(x_l \otimes \mathbf{1}_4^{\circ l})) \\
&= (A_0 \otimes A_1 \otimes \cdots \otimes A_k) \cdot P_k \cdots P_{l+1}(x_l \otimes \mathbf{1}_4^{\circ l}) \\
&= (A_0 \otimes A_1 \otimes \cdots \otimes A_k)((P_{k-l} \otimes I_4^{\circ l}) \cdots (P_1 \otimes I_4^{\circ l})(x_l \otimes \mathbf{1}_4^{\circ l})) \\
&= (A_0 \otimes A_1 \otimes \cdots \otimes A_k)((P_{k-l} \cdots P_1 x_l) \otimes (I_4^{\circ l} \cdots I_4^{\circ l} \mathbf{1}_4^{\circ l})) \\
&= ((A_0 \otimes \cdots \otimes A_{k-l}) \otimes A_{k-l+1} \cdots \otimes A_k) \cdot ((P_{k-l} \cdots P_1 x_l) \otimes \mathbf{1}_4^{\circ l}) \\
&= ((A_0 \otimes \cdots \otimes A_{k-l}) P_{k-l} \cdots P_1 x_l) \otimes (A_{k-l+1} \mathbf{1}_4) \cdots \otimes (A_k \mathbf{1}_4).
\end{aligned}$$

Let $\Delta_k^l = (A_0 \otimes \cdots \otimes A_{k-l}) P_{k-l} \cdots P_1$ and $a_j = A_j \mathbf{1}_4$ for $1 \leq j \leq k$. Then the above derivations can be simply written as

$$AEx_l = (\Delta_k^l x_l) \otimes a_{k-l+1} \otimes \cdots \otimes a_k. \tag{4.32}$$

Noting that the size of Δ_k^l is $\prod_{i=0}^{k-l} m_i \times (n/4^l)$ which is proportional to the decoding resolution instead of the encoding one, and the size of a_j for $1 \leq j \leq k$ is m_j which is chosen to be no larger than 4, it implies that a significant reduction on evaluating AEx_l in terms of either computation or storage. More specifically, both computation and storage directly correspond to the decoding resolution. The lower resolution one demands, the less resources the decoder need. Furthermore, if Δ_k^l involves some fast transform or efficient implementation as a result of Proposition 4.3.2, the reduction could be even more remarkable.

It follows (4.32) that the minimization problem (4.14) is equivalent to the following *level l decoding model*:

$$\min_{x_l} \Phi(x_l) + \frac{\mu}{2} \|(\Delta_k^l x_l) \otimes a_{k-l+1} \otimes \cdots \otimes a_k - y\|_2^2. \tag{4.33}$$

The low resolution video cube x_l can be obtained by solving the minimization problem (4.33). In light of former discussions on decoding models, a TV-DCT regularization is adopted here to help obtain a desirable solution.

As indicated in Chapters 2 and 3, the general TVAL3 method is quite efficient and robust for TV regularized minimization. Within the framework of the general TVAL3, the decoding model (4.33) can be solved similarly to other TV regularized problems as described before. In more specific terms, the Kronecker products in the fidelity term lead to a different gradient in form while employing Algorithm 2.3.1 and minimizing the corresponding augmented Lagrangian function. Besides this minor modification, the previously derived algorithm is completely applicable.

The complexity of this algorithm is dominated by two matrix-vector multiplications at each iteration, which is proportional to the size of Δ_k^l, or, equivalently, decoding resolution of a video clip. Therefore, the proposed multi-resolution scheme using structured sensing matrix is able to reduce the computational costs as the decoding resolution decreases. The structure of this type of sensing matrices is the key to achieve scalability, but also decreases the degree of randomness at the same time. This trade-off may result in the degradation of reconstruction at the original resolution compared to using totally random sensing matrices without structures. In short, both the complexity and elapsed time are scalable with the resolution of the reconstructed video by means of the proposed scheme, at the cost of less randomness. The full scalability of this multi-resolution scheme will be exhibited by examples in the next section.

4.4 Numerical Experiments

Simulations are performed on two formats of video clips, CIF (352×288) and HD 1080p (1920×1080), with a frame rate of 30 frames per second (fsp). In the simulations, a source video is encoded with the compressive measurements as described previously by using a permutated Walsh-Hadamard matrix as the measurement matrix. Each video volume consists of 8 full CIF frames, i.e., the total number of pixels in each video volume is either $n = 352 \times 288 \times 8 = 811,008$ for CIF or $n = 1920 \times 1080 \times 8 = 16,588,800$ for HD 1080p. Recovered video quality is measured using *peak signal-to-noise ratio*, abbreviated PSNR. A higher PSNR normally indicates that the reconstruction is of higher quality.

The parameter settings for the decoding algorithm were similar to what we described in Section 2.3.1 and we avoid restatement herein.

The results will be presented from two aspects. First, we focus on analysis of the TV-DCT method, the recoverability and the impact of Gaussian noise and quantization. Graceful degradation can be observed from these results. Secondly, we demonstrate the multi-resolution scheme and show the significant improvement as compared to the traditional compressive sampling methods or 3D DCT transform.

4.4.1 Graceful Degradation of TV-DCT Method

In this part, tests were primarily conducted on two standard CIF test video clips, *News.cif* and *Container.cif*, including demonstration of recoverability, comparison of various regularizations, and degradation caused by noise or quantization. Figure 4.8 shows one frame of each clip.

The first test studies the recoverability of compressive video sensing, or particularly TV-DCT method. Figure 4.9 shows the recovered quality of two test videos

Figure 4.8: CIF test videos: Frames from (a) *News* and (b) *Container*.

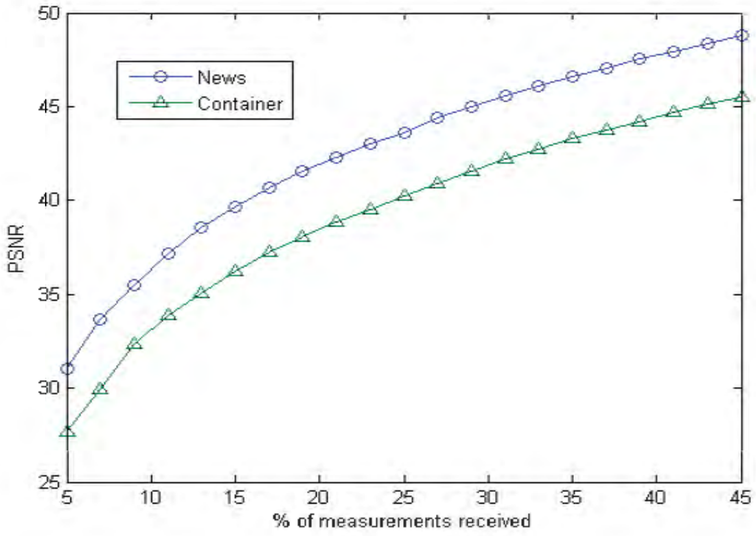
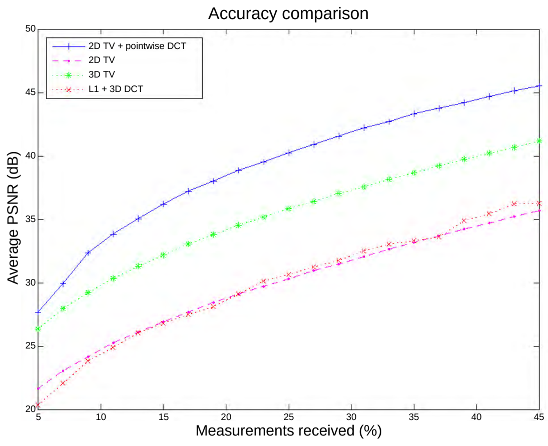


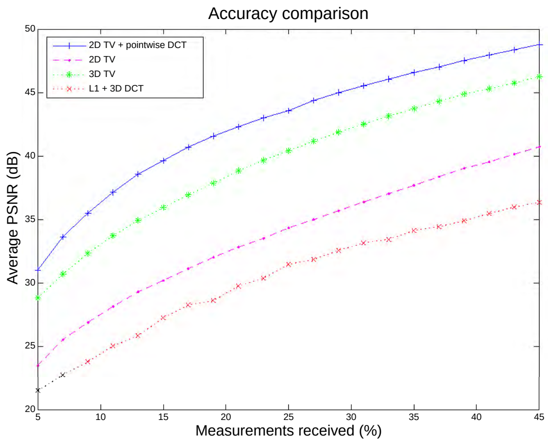
Figure 4.9: Recoverability for the noise-free case.

using different number of measurements, m, received as a percentage of n. Specifically, the PSNRs of the video clips reconstructed by TV-DCT method are measured as a function of the percentage of the measurements received (m/n).

It is clear from Figure 4.9 that the quality of video is progressively increasing as the number of correctly received measurements increases. This demonstrates that compressive video sensing is scalable with the channel capacity. It provides graceful degradation and avoids the cliff effect found in the traditional video coding methods such as MPEG2 or H.264. This advantage would become notably preferable in the realm of wireless communications.



(a) *Container*



(b) *News*

Figure 4.10: PSNR comparison using different regularizations.

Within the framework of compressive video sensing, several types of regularizations could be adopted other than the TV-DCT one. The proposed TV-DCT regularization has been theoretically discussed in Section 4.2.2 and claimed as an appropriate decoding method for video compression and transmission. In order to empirically confirm this argument, three commonly used regularizations involving TV or ℓ_1 are described and plugged into (4.5) in comparison to TV-CT method. For each amount of the correctly received measurements, four different regularizations are used to reconstruct the same video clip, and then the PSNRs of reconstructed videos are reported and compared. To fairly compare the results, reconstruction algorithms for different regularizations were all implemented in accordance with the general TVAL3 algorithm. The definitions of four regularization terms are given below.

1. 2D TV+pointwise DCT (TV-DCT): This is the method proposed and described in Section 4.2.2.

2. 2D TV: This is the method minimizing the sum-up of 2D TV of every frame. In other words, the video cube is treated as individual frames, and no temporal relations are explored. Mathematically, let X denote the original 3D cube of x before vectorization, and 2D TV regularization is defined as

$$\Phi(x) = \sum_{i,j,k} \left(|X_{i+1,j,k} - X_{i,j,k}| + |X_{i,j+1,k} - X_{i,j,k}| \right).$$

3. 3D TV: This is the method minimizing the 3D TV of the whole cube, assuming the sparsity of gradient in both spatial and temporal directions. Mathematically, under the same notation, 3D TV regularization is defined as

$$\Phi(x) = \sum_{i,j,k} \left(|X_{i+1,j,k} - X_{i,j,k}| + |X_{i,j+1,k} - X_{i,j,k}| + |X_{i,j,k+1} - X_{i,j,k}| \right).$$

4. ℓ_1+3D DCT: This is the method minimizing the ℓ_1-norm of the 3D DCT coefficients of the source cube. That is, a 3D DCT transform is performed on the video cube, and the ℓ_1-norm of the resulting coefficients is minimized. Mathematically, let $DCT_3(x)$ denote the vectorization of 3D DCT of the cube X, ℓ_1

regularization under 3D DCT basis is defined as

$$\Phi(x) = \|DCT_3(x)\|_1.$$

In all four models, the same permutated Walsh-Hadamard matrix is used as the sensing matrix, and the received measurements used for reconstruction are randomly chosen. The PSNR values are reported in Figure 4.10.

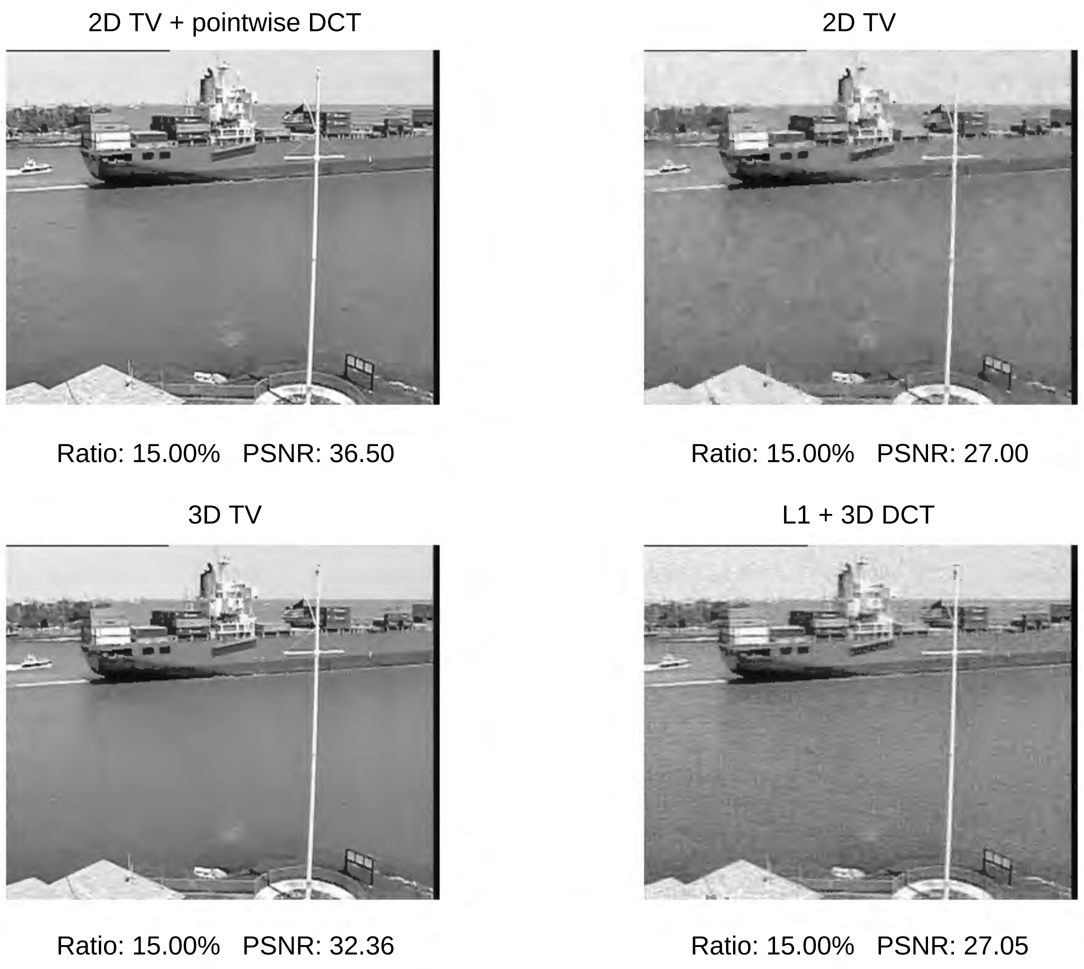

2D TV + pointwise DCT
2D TV
Ratio: 15.00% PSNR: 36.50
Ratio: 15.00% PSNR: 27.00
3D TV
L1 + 3D DCT
Ratio: 15.00% PSNR: 32.36
Ratio: 15.00% PSNR: 27.05

Figure 4.11: A typical frame from recovered clips *Container*.

Two observations can be made from these results. First the compressive sampling methods — TV-DCT, 2D TV, 3D TV and ℓ_1+3D DCT — all have the scalability

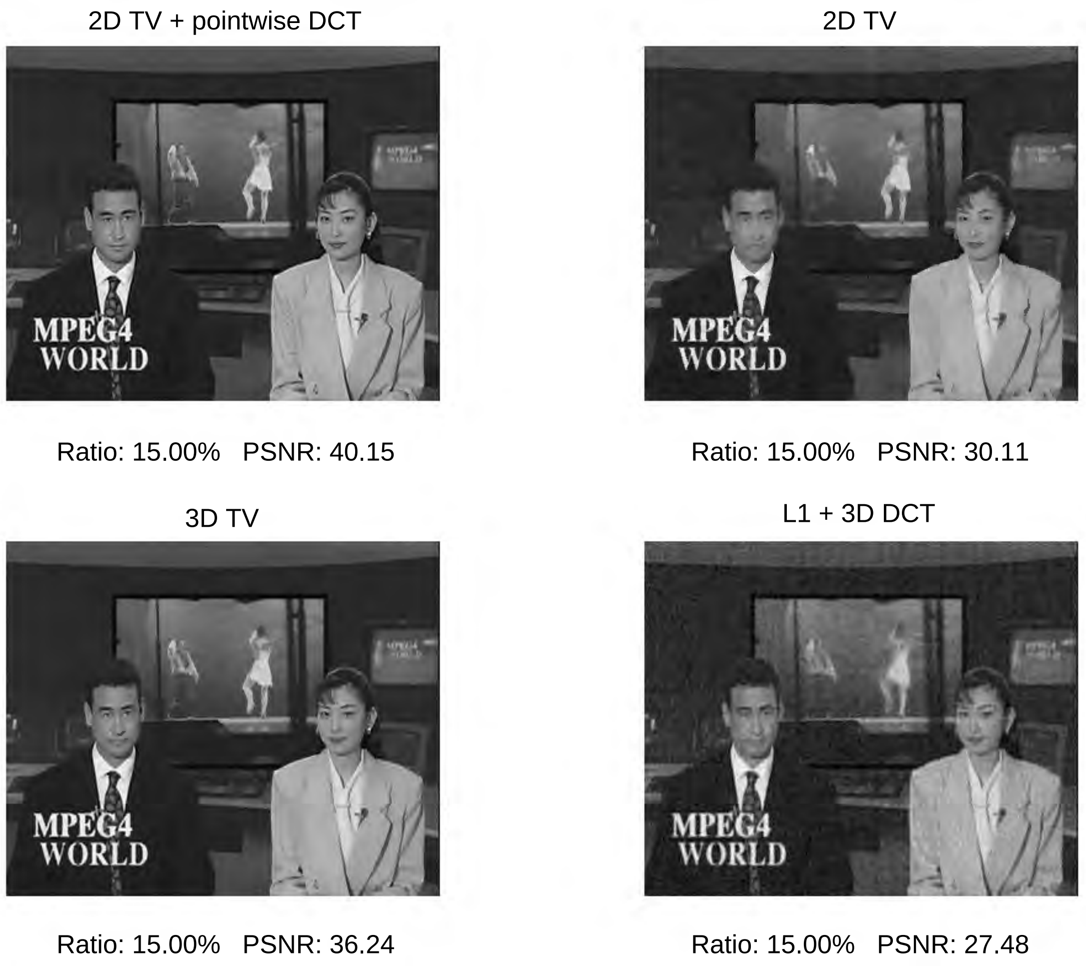
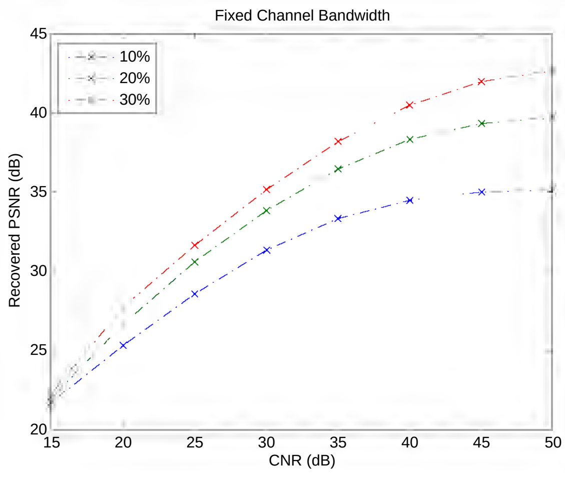
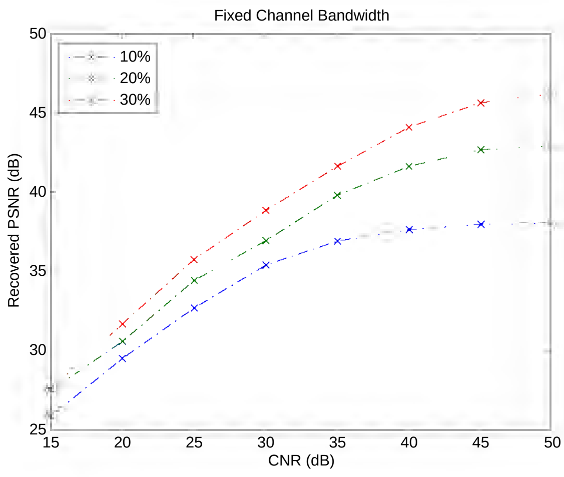

2D TV + pointwise DCT
2D TV
Ratio: 15.00%   PSNR: 40.15
Ratio: 15.00%   PSNR: 30.11
3D TV
L1 + 3D DCT
Ratio: 15.00%   PSNR: 36.24
Ratio: 15.00%   PSNR: 27.48

Figure 4.12: A typical frame from recovered clips *News*.

property desired in wireless transmission, namely, the PSNR of the reconstructed video increases progressively with the number of measurements received. This conforms to the result implied by the first test. The second observation is that the TV minimization based methods are more superior to the ℓ_1 minimization under 3D DCT basis. Among all methods simulated, TV-DCT method which minimizes 2D total variation of frames composed of pointwise DCT coefficients is clearly better over other three. A typical frame in the recovered videos for both *News* and *Container* clips for each of the four tested methods is shown in Figure 4.12 and Figure 4.11, respectively. The ones on the upper left which were reconstructed by TV-DCT method obviously showed stronger contrast (observable in Figure 4.12) and preserved more details (observable in Figure 4.11). Numerically TV-DCT method could provide higher PSNR than the other three from the perspective of distinct regularizations. Due to a major improvement on recovered quality in contrast with other conventional regularizations, TV-DCT method is adopted in the following tests.



(a) *Container*



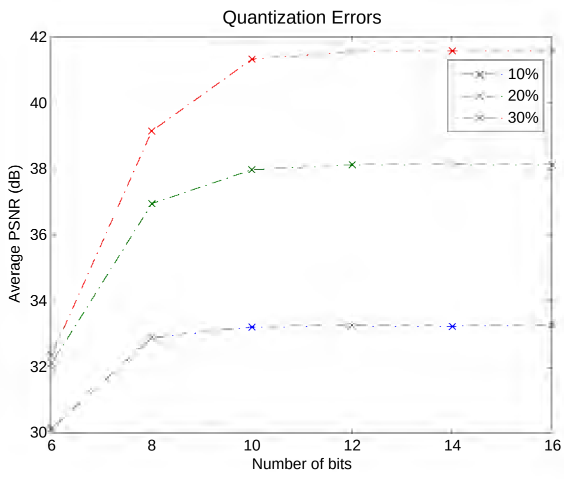
(b) *News*

Figure 4.13: PSNR as a function of additive Gaussian noise (CNR).

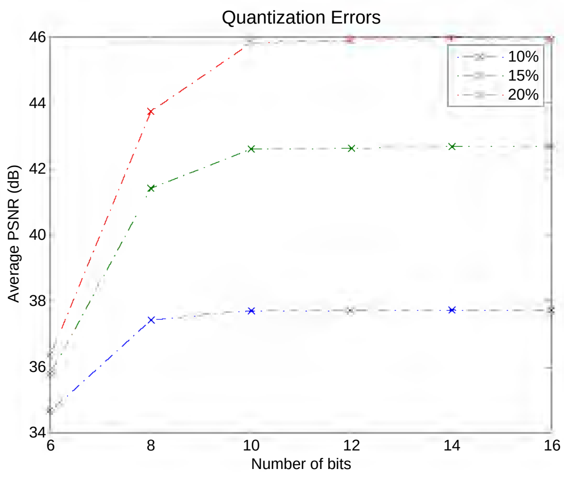
Next, simulations are performed when channel noise is present. In the simulation, the selected measurements are injected with Gaussian noise, and the resulting noisy measurements are used for reconstruction. In other words, the reconstruction is performed by minimizing (4.6) with y replaced by $\hat{y} = y + n$, where $\hat{y}$ is the received measurements with additive Gaussian noise n. The PSNR of the reconstructed video as a function of the noise level which is measured by *carrier-to-noise ratio*, is shown for different percentages of measurements used in reconstruction, as indicated in Figure 4.13. The carrier-to-noise ratio, often written CNR, is a measure of the received carrier strength relative to the strength of the received noise. A higher CNR indicates better quality of reception, and generally higher communications accuracy and reliability.

The results demonstrate that our model and reconstruction algorithm are reliable even when noise is present in measurements. Furthermore, for any given amount of noise the quality of video may be improved by using more measurements.

In some broadcasting schemes the number of received measurements is the same for all channels and the quality of the reconstructed video is determined solely by the channel's noise level [121]. These results demonstrate that our coding method is scalable with the transmission channel conditions as well.

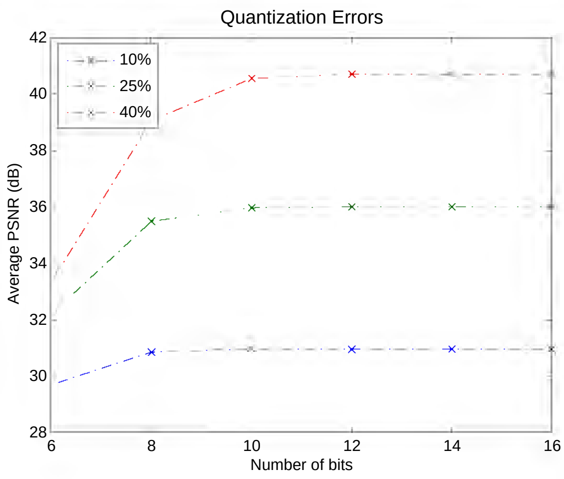


(a) *Container*

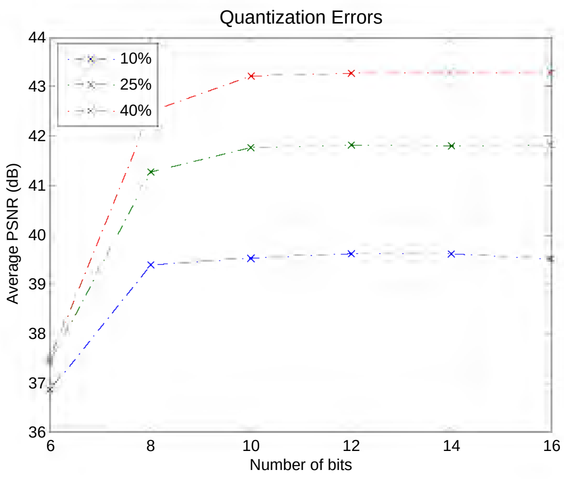


(b) *News*

Figure 4.14: Impact of quantization on CIF videos.



(a) *Life*



(b) *Rush_Hour*

Figure 4.15: Impact of quantization on HD videos.

Finally, we examine the impact of quantization on the performance of the reconstruction. Quantization is a lossy compression technique achieved by representing the real-value measurements by integers, or equivalently, a number of bits. For wireless networks, it is a series of bits instead of real numbers to be transmitted and received over the air. In the following tests, each received measurement is quantized to bits before being used in the reconstruction. For the sake of higher accuracy, we suggest to double the number of bits for measurements of magnitude exceeding a threshold in the quantization. The threshold decides the ratio between the number of double-bits quantized measurements and the total number of received measurements. To balance the tradeoff between the average number of bits for each measurement and recovered quality, this ratio is chosen to be around $1 : 10,000$. The PSNR of the reconstructed video is measured as a function of the average number of bits in the quantization. In addition to two CIF video clips, we ran the tests on two video clips of HD format, *Life.1080p* and *Rush_hour.1080p*, to indicate these quantization results are typical and not limited by the video format or resolution. One frame of each HD video is displayed in Figure 4.16(a) or Figure 4.17(a). The results, for different percentages of measurements used in reconstruction, are shown in Figure 4.14 and Figure 4.15.

It can be concluded that the reconstruction is not sensitive to quantization. There is no observable degradation above $8-10$ bits per measurement on average. Furthermore, for a given quantization level, the quality may be improved by increasing the number of measurements used in the reconstruction.

4.4.2 Scalability of Multi-Resolution Scheme

The multi-scale coding method described previously is implemented in simulations using an encoding matrix that is capable of providing four levels of decoded resolution. Those small sensing matrices for the construction of A are extracted from

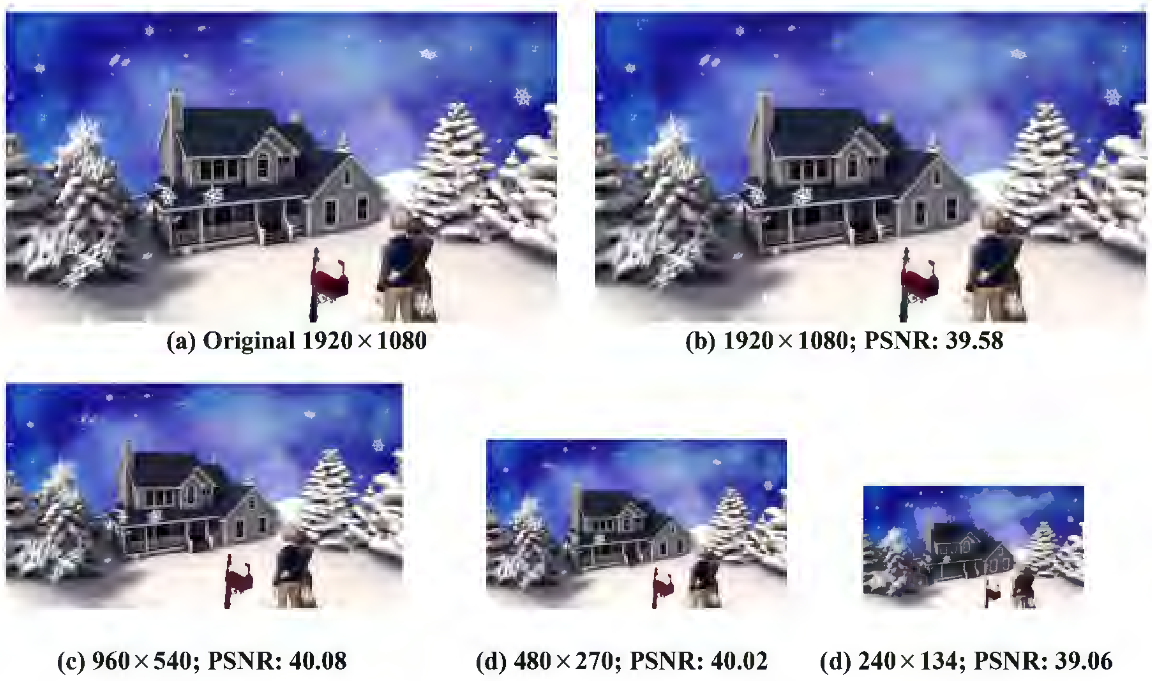
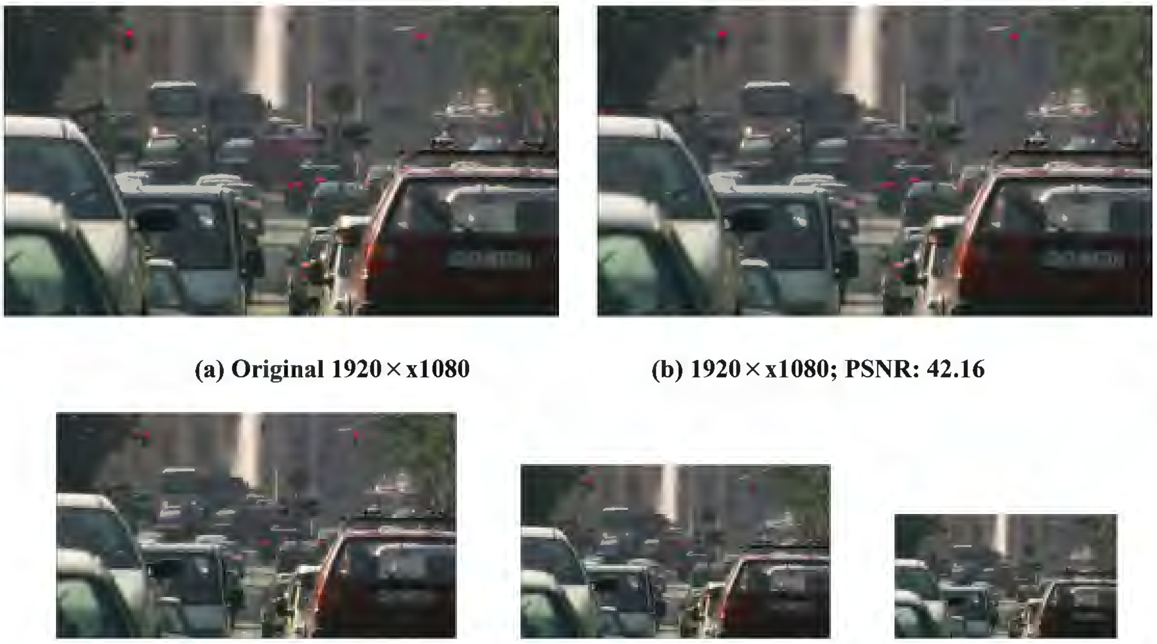


Figure 4.16: Reconstruction at different resolutions for HD video clip *Life*.

the permutated Walsh-Hadamard matrices. Results for two standard HD test video sequences will be presented, and they are *Life.1080p* and *Rush_hour.1080p*. Similar to former settings, the two source video sequences have frame rate of 30 fsp. For each source video, the same measurement matrix as described previously is used to encode the video. Each video volume consists of 8 entire frames.

Decoding of four resolutions was performed: 1920×1080 (the original resolution), 960×540, 480×270 and 240×134. A different amount of measurements are used in the reconstructions of video with a different resolution. More specifically, reconstructed videos of 4 distinct resolutions were decoding from 35%, 9%, 2% and .5% of measurements, respectively. Figure 4.16 and Figure 4.17 show the typical results of multi-resolution scheme.

Normally, compressive sensing is not capable of any reasonable reconstruction with only .5% or even 2% of measurements since it is much lower than theoretical lower



(a) Original 1920 × x1080 (b) 1920 × x1080; PSNR: 42.16

(c) 960 × x540; PSNR: 43.93 (d) 480 × x270; PSNR: 44.37 (d) 240 × x134; PSNR: 44.61

Figure 4.17: Reconstruction at different resolutions for HD video clip *Rush_hour*.

bound. However, Figure 4.16 and Figure 4.17 suggest that the new proposed multi-resolution scheme can significantly reduce the lower bound of number of measurements for successful recovery at lower resolutions. This scheme may also be considered as a general sensing matrix construction scheme and applied to other applications which require fewer measurements or have limited bandwidth.

The complexity of reconstruction is scalable with the decoded resolution. This is evident from the elapsed time it takes to decode the video of different resolutions. When the average time it takes to decode a video of original resolution is normalized to 1, the average times it takes to decode videos at lower resolution are 0.22, 0.055 and 0.016, respectively.

Next, the accuracy of reconstruction will be measured by calculating PSNR in the reconstructed video. In order to measure the PSNR, a reconstructed video must be compared with an original video of the same resolution. To accomplish this, reference

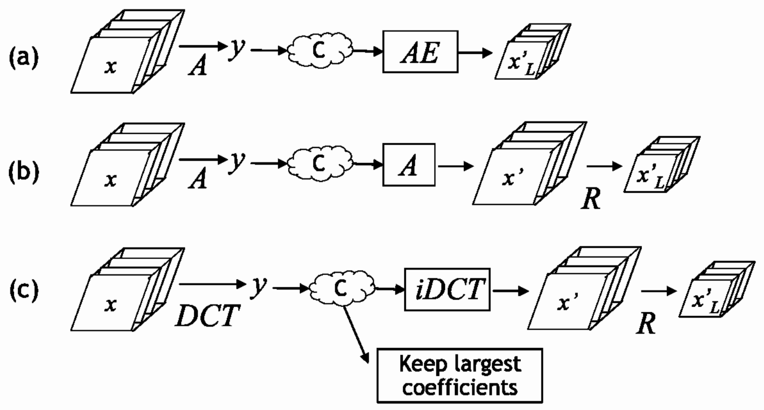
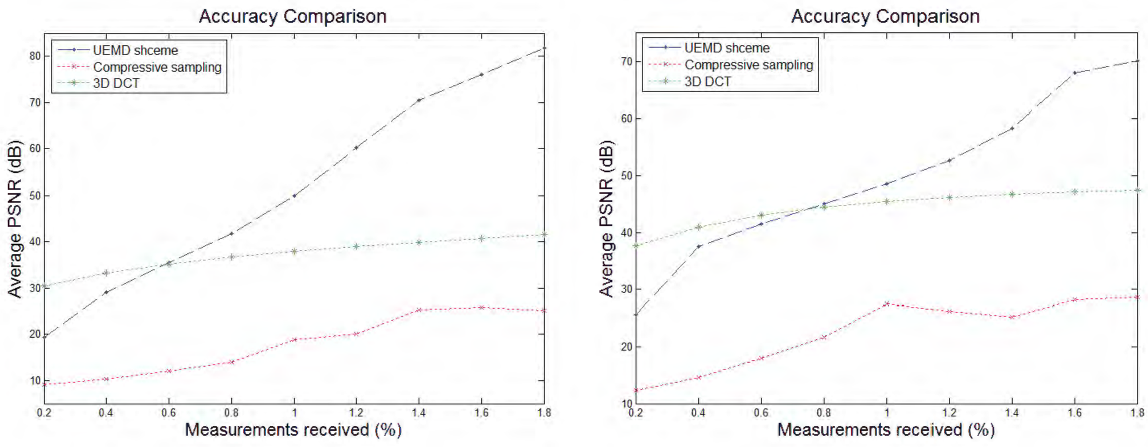


Figure 4.18: Three methods used for low-resolution reconstruction: (a) multi-resolution scheme, (b) conventional compressive video sensing reconstruction followed by resizing to lower resolution and (c) 3D DCT transform followed by resizing to lower resolution.

videos are made by properly interpolating from the source video using MATLAB function *imresize*. The decoded video has the same resolution as the reference video. Finally, the PSNR of the decoded video as compared to the reference video is measured and reported. In order to exhibit the improvement gained by multi-resolution scheme, it is compared to other two methods in the simulations and the PSNRs of decoded videos from the three methods will be evaluated. The sketches of these methods are illustrated in Figure 4.18.

The first method is the proposed multi-resolution scheme as shown in Figure 4.18(a). The lower resolution decoded video x_L is obtained directly as part of reconstruction from correctly received measurements y by solving 4.33. The second, shown in Figure 4.18(b), is a conventional compressive sensing reconstruction. The measurement matrix A is a permutated Walsh-Hadmard matrix. The correctly received measurements y are used to reconstruct a video x' of the same resolution as the source video by solving (4.5). Then the reconstructed video is resized to the lower resolution x_L by downsampling. The last, shown in Figure 4.18(c), is utilizing the 3D DCT transform. The source video x is encoded by 3D DCT transform on a video volume. The DCT coefficients are transmitted. The correctly received y are the largest coefficients of DCT transform. In other words, the coefficients are sorted in descending order according to their amplitudes. For example, if 10% coefficients are received, it is assumed that the first 10% of the sorted coefficients (the largest 10% in amplitudes) are received correctly. This, of course, places a huge advantage to the third method, because in the compressive sensing methods of Figure 4.18(a) and Figure 4.18(b), the correctly received measurements are randomly chosen and no particular protection is needed. In all methods, the PSNR is calculated by comparing x_L with the reference video x_R which is the resized frames of the original video. The PSNR values as a function of the percentage of measurements received for the video clip *Life* and *Rush_hour* are plotted in Figure 4.19, respectively.



(a) *Life* (b) *Rush_Hour*

Figure 4.19: PSNR comparison for low-resolution reconstruction: the source video x is 1080p and the decoded video x_L is at lower resolution of 240×134.

In Figure 4.19, the source video clips *Life* and *Rush_hour* of 1080p resolution are encoded and transmitted as previously described. The decoded videos have a lower resolution of 240×134, which have approximately 1/8 as many pixels as the source videos have in both horizontal and vertical directions. The dashed blue curve is the PSNR for the multi-resolution scheme and the red curve with crosses is the PSNR for the conventional compressive sensing reconstruction, and the green curve with squares is the PSNR for 3D DCT transform.

The results in Figure 4.19 show that the method proposed in Section 4.3 has much higher accuracy on average than the traditional compressive sensing method in which a video of the original resolution is reconstructed and then resized to a lower resolution. Compared to 3D DCT transform, our proposed method has relatively lower quality at the beginning, which is reasonable since it first receives the largest 3D DCT coefficients and then receives those relative small ones. Those large measurements contain more information than the small ones, which put this method advantageous at the very beginning. However, the obvious drawback of this method is that we cannot afford to lose the large measurements, which ineluctably generates the redundancy in reality. Overall, the newly proposed multi-resolution scheme outperforms both methods after approximately 0.8% of measurements, though it may vary a little case by case.

4.5 Discussions

The purpose of this chapter is to bring scalability into video coding from different aspects by means of compressive sensing. We first discussed the framework of compressive video sensing and proposed a TV-DCT method for decoding a compressed video. Then a multi-resolution scheme was theoretically analyzed and implemented in light of a new way to construct sensing matrices using Kronecker products. The multi-resolution scheme is capable of the same encoding but various decoding according to available resources. By means of this scheme, low resolution reconstruction becomes possible and preferable even if the measurements are severely deficient for regular reconstruction of the original resolution. The proposed sensing matrix construction method is not limited within the field of video compression, but should be regarded as a general CS technique to build sensing matrices in order for multiscale or multilayer reconstruction. However, the tradeoff of multiscale or multilayer reconstruction using this type of sensing matrices is the slightly decreased recoverability of reconstructing at original resolution in contrast with using non-structured sensing matrices. To effectively and efficiently solve several decoding models such as (4.5), (4.6), (4.14) and (4.33), we employed the general TVAL3 algorithm which has been thoroughly analyzed and widely applied.

To be brief, these work is mainly concerned with compressive video coding that is scalable with channel capacity, which is of particular interests in wireless broadcast since the mobile clients experience vastly different channel conditions. We have shown that compressive video sensing is a promising technique for video transmission. However, there are challenges remaining to be resolved before this technique becomes practical. First, further work is needed for compressive video sensing to achieve a high compression ratio, comparable to mature techniques such as H.264. To that end, motion estimation and motion compensation need to be integrated with compressive video sensing. Secondly, in spite of convenient encoding process, the complexity of reconstruction in compressive video sensing is still too high for real time decoding in practical systems with today's technology. Development of more efficient optimization algorithms and advancement of hardware technology will help make real time decoding feasible. By demonstrating the desirable properties of compressive video sensing that are fundamentally lacking in the traditional video coding, we hope that a motivation is created to address the remaining issues in the future research.

Chapter 5

Conclusions and Remarks

In recent years, compressive sensing has been intensively investigated from different perspectives and applied to applications in diverse areas. This thesis is centered around developing efficient algorithms mainly for TV and ℓ_1 minimizations and extending the idea of compressive sensing to the field of 3D data processing which traditionally requires tremendous computation and resources.

5.1 Contributions

The major algorithmic contributions of this thesis are three-fold:

1. a general TVAL3 algorithm has been both computationally and theoretically studied which is capable of solving many TV minimization problems effectively;

2. a new hyperspectral data sensing and unmixing scheme has been presented, successfully bypassing the high complexity of recovering the whole hyperspectral cube;

3. a framework of compressive video sensing using TV-DCT regularization has

been proposed, and a novel multi-resolution scheme has been devised which achieves high scalability in many aspects.

The TVAL3 algorithm was first introduced in the author's master thesis with a description limited to 1D or 2D TV regularized minimization and without a theoretical convergence analysis. In Chapter 2, we extend the algorithmic framework to a general form of linearly constraint minimization problems. Specifically, the general TVAL3 method is based on the classic augmented Lagrangian multiplier method after a proper variable splitting and an alternating direction approach. To accelerate the convergence, a nonmonotone line search procedure with Barzilai-Borwein initial step is employed in the alternating steps. The convergence of this method has been proven. This work provides a springboard for further analysis on 3D data processing.

In the aspect of 3D data processing, this thesis focuses on hyperspectral imaging and video compression. The work in hyperspectral imaging in Chapter 3 is a proof-of-concept study on a compressive sensing and (blind) unmixing scheme for hyperspectral data processing that does not require forming or storing any full-size data cube. This scheme consists of three major steps:

1. data acquisition by compressive sensing,
2. data preprocessing using SVD, and
3. data unmixing by solving a new unmixing model with total variation regularization on abundance distributions.

In the first-stage study covered in Sections 3.2 and 3.3, we considered the situation where the spectral signatures of the endmembers are either precisely or approximately known. After performing the SVD preprocessing, data sizes to be processed become much smaller and independent of the number of spectral bands. An efficient algorithm is constructed to unmix the corresponding abundance fractions under the framework of the general TVAL3 method, while the signatures are fixed.

In the second-stage study covered in Sections 3.4 and 3.5, we considered blind unmixing where both abundance fractions and signatures are recovered at the same time. An ADM type-method has been proposed to handle the non-convex compressed blind unmixing model. Due to the non-convexity of the problem, some prior information is necessary to avoid local minima. Rather extensive numerical experiments have been conducted to demonstrate the feasibility and efficiency of the proposed approach, using both synthetic data and hardware-measured data. Experimental and computational evidences obtained from this study indicate that the proposed scheme possesses a high potential in real-world applications.

In addition to hyperspectral imaging, we researched on another type of 3D data processing — video coding, where traditional coding methods such as MPEG2 or H.264 could no longer offer the scalability desired by today's network. In Chapter 4, a framework for video coding using compressive sensing is studied. In the framework, a source video is divided into video volumes, and random measurements are taken using a random sensing matrix. The video is reconstructed by minimizing the spatial total variation of the temporal DCT coefficients, or abbreviated TV-DCT regularization. In the general TVAL3 framework, the TV-DCT regularized problem for video decoding could be handled in a similar way as the TV regularized problem for 2D image processing.

Furthermore, taking advantage of Kronecker product, we have presented a new way to construct the sensing matrices. This type of sensing matrices implements a novel multi-resolution scheme in which the same encoded video stream may be used to reconstruct videos of distinct resolutions with distinct complexities. Unlike those methods based on downsampling or downsizing, the multi-resolution scheme reconstructs the video of lower resolution directly from measurements of the source video at the original resolution. This scheme is especially applicable when the received measurements are much fewer than the minimal requirement to reconstruct the original video clip or when the receiver has a limited capacity. Numerical experiments have shown that a HD video clip encoded using this scheme can be decoded at a lower resolution from as little as .5% of total measurements which used to result in failures of decoding. In virtue of these progresses, video coding using compressive video sensing has exhibited many characteristics such as graceful degradation and multi-resolution decoding that are desirable for transmission in a video network. In particular, the coded video is highly scalable with channel capacity, decoding complexity, display resolution and quality. Moreover, the new sensing matrix construction method can be useful beyond the realm of video coding. It represents a multilayer sensing paradigm, which allows different resolutions with compatible complexity.

5.2 Remarks and Future Work

This thesis has addressed several topics involving compressive sensing and 3D data processing. It is important to discuss the issue of applicability related to some of the models and algorithms proposed and studied in the thesis.

The TVAL3 method is presented for a general form of linearly constraint minimization problems. However, its efficiency relies on how easy the non-smooth subproblems can be solved, and how effective the smooth subproblem can be attacked by the non-monotone gradient descent scheme. In all the calculations performed in the thesis, the non-smooth subproblems are solved by closed-form formulas, while the smooth ones are quadratic minimization problems. From our experience, the TVAL3 method is particularly efficient and robust for situations when data are noisy and, as a result, only a moderate accuracy is required.

TVAL3 does not require users to supply or tune many algorithmic parameters with possibly one exception — the penalty parameter for linear equality constraints. When noise level varies, the penalty parameter value need to vary in order to maintain near-optimal performance. It takes a little experience, and sometimes a few trial-and-error attempts, to select good parameter values. This issue of penalty parameter selection certainly deserves further investigations.

The proposed compressive sensing and unmixing scheme requires a sufficient amount of endmember spectral signature information in order to successfully recover the corresponding abundance distributions. Our blind unmixing scheme, which solves a non-convex minimization model, cannot be totally blind in the following sense. Synthetic data simulations have suggested that the scheme can succeed under various scenarios where the endmember spectral signature information are either severely corrupted, deformed, or partially missing. On the other hand, the numerical procedure would usually get stuck at a local minimum when it is started from a set of totally random guesses for spectral signatures.

As for the compressive sensing scheme for video compression, despite its various advantages discussed in the thesis, the scheme requires solving a minimization problem for decoding, which is still too costly to be practical under today's technological conditions. Another primary drawback is that the compression ratio of the proposed scheme may not be as high as that of MPEG2 or H.264 in general. The compression ratio could be improved by taking motion estimation and compensation into the account of the framework at the cost of reduced scalability in capacity. There remain some fundamental difficulties and tradeoffs that require a great deal of further research.

Bibliography

[1] C. E. Shannon, *Communication in the presence of noise*, Proc. Institute of Radio Engineers, vol. 37, no. 1, pp. 10–21, Jan. 1949. Reprint as classic paper in: Proc. IEEE, vol. 86, no. 2, Feb. 1998.

[2] E. Candès, J. Romberg and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Trans. Inform. Theory, vol. 52, no. 2, pp. 489–509, 2006.

[3] E. Candès and T. Tao, *Near optimal signal recovery from random projections: Universal encoding strategies?*, IEEE Trans. on Inform. Theory, vol. 52, no. 12, pp. 5406–5425, 2006.

[4] D. Donoho, *Compressed sensing*, IEEE Transactions on Information Theory, vol. 52, no. 4, pp. 1289–1306, 2006.

[5] E. Candès and T. Tao, *Decoding by linear programming*, IEEE Trans. Inform. Theory, vol. 51, no. 12, pp. 4203–4215, 2005.

[6] J. Romberg, *Imaging via compressive sampling*, IEEE Signal Processing Magazine, vol. 25, no. 2, pp. 14-20, March 2008.

[7] Y. Zhang, *On theory of compressive sensing via ℓ_1-minimization: Simple derivations and extensions*, CAAM Technical Report TR08-11, Department of Computational and Applied Mathematics, Rice University, July 2008.

[8] D. Donoho, *Neighborly polytopes and sparse solution of underdetermined linear equations*, IEEE. Trans. Info. Theory, 2006.

[9] C. Li, *An Efficient Algorithm for Total Variation Regularization with Applications to the Single Pixel Camera and Compressive Sensing*, Mater Thesis, Computational and Applied Mathematics, Rice University, 2009.

[10] B. K. Natarajan, *Sparse approximate solutions to linear systems*, SIAM Journal on Computing, vol. 24, pp. 227–234, 1995.

[11] C. Jordan, *Sur la série de Fourier*, Comptes rendus hebdomadaires des séances de l'Académie des sciences, vol. 92, pp. 228–230, 1881.

[12] J. F. Claerbout and F. Muir, *Robust modeling with erratic data*, Geophys. Mag., vol. 38, no. 5, pp. 826–844, Oct. 1973.

[13] F. Santosa and W. W. Symes, *Linear inversion of band-limited reflection seismograms*, SIAM J. Sci. Statist. Comput., vol. 7, no. 4, pp. 1307–1330, 1986.

[14] L. Rudin, S. Osher and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Physica D, pp. 259–268, 1992.

[15] D. L. Donoho and P. B. Stark, *Uncertainty principles and signal recovery*, SIAM J. Appl. Math., vol. 49, pp. 906–931, 1989.

[16] D. L. Donoho and B. F. Logan, *Signal recovery and the large sieve*, SIAM J. Appl. Math., vol. 52, pp. 577–591, 1992.

[17] A. Bonnet, *On the regularity of edges in image segmentation*, Ann. Inst. H. Poincaré Anal. Non Linéaire, vol. 13, pp. 485–528, 1996.

[18] P. Blomgren and T. Chan, *Color TV: Total variation methods for restoration of vector-valued images*, IEEE Transactions on Image Processing, vol. 7, no. 3, pp. 304–309, 1998.

[19] R. Acar and C. Vogel, *Analysis of bounded variation penalty methods for ill-posed problems*, Inverse Problems, vol. 10, pp. 1217–1229, 1994.

[20] S. Teboul, L. Blanc-Feraud, G. Aubert, and M. Barlaud, *Variational approach for edge-preserving regularization using coupled PDE's*, IEEE Trans. Image Processing, vol. 7, pp. 387–397, 1998.

[21] R. Luce and S. Perez, *Parameter identification for an elliptic partial differential equation with distributed noisy data*, Inverse Problems, vol. 15, pp. 291–307, 1999.

[22] D. L. Donoho and X. Huo, *Uncertainty principles and ideal atomic decomposition*, IEEE Trans. Inform. Theory, vol. 47, pp. 2845–2862, 2001.

[23] M. Elad and A. M. Bruckstein, *A generalized uncertainty principle and sparse representation in pairs of $\mathbb{R}^N$ bases*, IEEE Trans. Inform. Theory, vol. 48, pp. 2558–2567, 2002.

[24] R. Gribonval and M. Nielsen, *Sparse representations in unions of bases*, IEEE Trans. Inform. Theory, vol. 49, pp. 3320–3325, 2003.

[25] J. J. Fuchs, *On sparse representations in arbitrary redundant bases*, IEEE Trans. Inform. Theory, vol. 50, pp. 1341–1344, 2004.

[26] H. Rauhut, *Stability results for random sampling of sparse trigonometric polynomials*, IEEE Trans. Inform. Theory, vol. 54, pp. 5661–5670, 2008.

[27] S. S. Chen, *Basis Pursuit*, PhD thesis, Stanford University, Department of Statistics, 1995.

[28] S. S. Chen, D. L. Donoho and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM Review, vol. 43, no. 1, pp. 129–159, 2001.

[29] S. G. Mallat and Z. Zhang, *Matching pursuits with time-frequency dictionaries*, IEEE Transactions on Signal Processing, vol. 41, no. 12, pp. 3397–3415, 1993.

[30] J. Tropp and A. Gilbert, *Signal recovery from random measurements via orthogonal matching pursuit*, IEEE Trans. Info. Theory, vol. 53, pp. 4655–4666, Dec 2007.

[31] D. Needell and J. A. Tropp, *Cosamp: Iterative signal recovery from incomplete and inaccurate samples*, Applied and Computational Harmonic Analysis, vol. 26, pp. 301–321, 2009.

[32] D. Takhar, J. N. Laska, M. B. Wakin, M. F. Duarte, D. Baron, S. Sarvotham, K. F. Kelly and R. G. Baraniuk, *A new compressive imaging camera architecture using optical-domain compression*, Computational Imaging IV, vol. 6065, pp. 43–52, Jan. 2006.

[33] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly and R. Baraniuk, *Compressive imaging for video representation and coding*, Proc. Picture Coding Symposium (PCS), Beijing, China, April 2006.

[34] T. Sun, C. Li, Y. Zhang and K. F. Kelly, *Infrared Imaging by a Single Photodiode based on Compressive Sensing*, accepted by Applied Physics Letters, 2009.

[35] T. Strohmer and M. Hermann, *Compressed Sensing Radar*, IEEE Proc. Int. Conf. Acoustic, Speech, and Signal Processing, 2008, pp. 1509–1512, 2008.

[36] M. Lustig, D.L. Donoho and J.M. Pauly, *Rapid MR imaging with compressed sensing and randomly under-sampled 3DFT trajectories*, in Proc. 14th Ann. Meeting ISMRM, Seattle, WA, May 2006.

[37] M. Lustig, D. Donoho, J. Pauly, *Sparse MRI: The application of compressed sensing for rapid MR imaging* Magnetic Resonance in Medicine, vol. 58 no. 6, pp. 1182–1195, 2007.

[38] T. Chang, L. He and T. Fang, *MR image reconstruction from sparse radial samples using bregman iteration*, ISMRM, 2006.

[39] J. Laska, S. Kirolos, M. Duarte, T. Ragheb, R. Baraniuk and Y. Massoud, *Theory and implementaion of an analog-to-information converter using random demodulation*, In Proceedings of the IEEE International Symposium on Circuites and Systems (ISCAS), New Orleans, Louisiana, 2007.

[40] M. F. Duarte, S. Sarvotham, D. Baron, M. B. Wakin and R. G. Baraniuk, *Distributed compressed sensing of jointly sparse signals*, in 39th Asilomar Conference on Signals, Systems and Computers, pp. 1537–1541, 2005.

[41] J. Haupt, W. U. Bajwa, M. Rabbat and R. Nowak, *Compressed Sensing for Networked Data*, IEEE Signal Processing, vol. 25, no. 2, pp. 92–101, March 2008.

[42] W. Chan, M. Moravec, R. Baraniuk and D. Mittleman, *Compressive sensing applied to homeland security*, IEEE Sensors Applications Symposium, Atlanta, GA, USA, Feb. 2008.

[43] A. Chambolle, *An algorithm for total variation minimization and applications*, Journal of Mathematical Imaging and Vision, vol. 20, 89–97, Jan. 2004.

[44] D. Geman and G. Reynolds, *Constrained restoration and the recovery of discontinuities*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 3, pp. 367–383, 1992.

[45] D. Geman and C. Yang, *Nonlinear image recovery with half-quadratic regularization*, IEEE Transactions on Image Processing, vol. 4, no. 7, pp. 932–946, 1995.

[46] Y. Wang, J. Yang, W. Yin and Y. Zhang, *A new alternating minimization algorithm for total variation image reconstruction*, SIAM J. Imag. Sci., vol. 1, no. 4, pp. 248–272, 2008.

[47] T. Chan and C. K. Wong, *Total variation blind deconvolution*, IEEE Trans. Image Processing, vol. 7, no. 3, pp. 370–375, 1998.

[48] D. Goldfarb and W. Yin, *Second-order cone programming methods for total variation based image restoration*, SIAM Journal on Scientific Computing, vol. 27, no. 2, pp. 622–645, 2005.

[49] Y. Boykov, O. Veksler and R. Zabih, *Fast approximate energy minimization via graph cuts*, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 23, no. 11, pp. 1222–1239, 2001.

[50] A. Chambolle and P. L. Lions, *Image recovery via total variation minimization and related problems*, Numer. Math., vol. 76, pp. 167–188, 1997.

[51] T. F. Chan, S. Esedoglu, F. Park and A. Yip, *Recent developments in total variation image restoration*, CAM Report 05-01, Department of Mathematics, UCLA, 2004.

[52] J. Yang, Y. Zhang and W. Yin, *A fast TVL1-L2 minimization algorithm for signal reconstruction from partial fourier data*, Tech. Report 08-27, CAAM, Rice University. Submitted to J-STSP, 2008.

[53] J. Yang, W. Yin, Y. Zhang and Y. Wang, *A fast algorithm for edge-preserving variational multichannel image restoration*, SIAM J. Imaging Sciences, vol. 2, no. 2, pp. 569–592, 2009.

[54] J. Yang, Y. Zhang and W. Yin, *An efficient TVL1 algorithm for deblurring of multichannel images corrupted by impulsive noise*, SIAM Journal on Scientific Computing, vol. 31, no. 4, pp. 2842–2865, 2009.

[55] W. Yin, S. Osher, D. Goldfarb and J. Darbon, *Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing*, Siam J. Imaging Science, 1, 142–168, 2008.

[56] T. Goldstein and S. Osher, *The split Bregman algorithm for ℓ_1 regularized problems*, UCLA CAM report 08-29, SIAM. J. Image Sci, 2, 323–343, 2009.

[57] R. Courant, *Variational methods for the solution of problems with equilibrium and vibration*, Bull. Amer. Math. Soc., vol. 49, pp. 1–23, 1943.

[58] R. Glowinski and A. Marrocco, *Sur l'approximation par éléments finis d'ordre un et la résolution par pénalisation-dualité d'une classe de problèmes de Dirichlet nonlinéaires*, C. R. Acad. Sci. Paris, 278A, 1649–1652, 1974 (in French).

[59] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximations*, Comp. Math. Appl., vol. 2, pp. 17–40, 1976.

[60] M. R. Hestenes, *Multiplier and gradient methods*, Journal of Optimization Theory and Applications, vol. 4, pp. 303–320, and in Computing Methods in Optimization Problems, 2 (Eds L.A. Zadeh, L.W. Neustadt and A.V. Balakrishnan), Academic Press, New York, 1969.

[61] M. J. D. Powell, *A Method for Nonlinear Constraints in Minimization Problems*, Optimization (Ed. R. Fletcher), Academic Press, London, New York, pp. 283–298, 1969.

[62] J. D. Buys, *Dual Algorithms for Constrained Optimization*, Rijksuniversiteit de Leiden, The Netherlands, PhD Thesis, 1972.

[63] R. T. Rockafellar, *The multiplier method of Hestenes and Powell applied to convex programming*, Journal of Optimization Theory and Applications, vol. 12, no. 6, pp. 555–562, 1973.

[64] R. A. Tapia, *Newton's method for problems with equality constraints*, SIAM Journal on Numerical Analysis, vol. 11, pp. 174–196, 1974.

[65] R. A. Tapia, *Diagonalized multiplier methods and quasi-Newton methods for constrained optimization*, Journal of Optimization Theory and Applications, vol. 22, no. 2, pp. 135–194, 1977.

[66] R. H. Byrd, *Local convergence of the diagonalized method of multipliers*, Journal of Optimization Theory and Applications, vol. 26, no. 4, pp. 483–498, 1978.

[67] P. L. Lions and B. Mercier, *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer. Anal., vol. 16, pp. 964–979, 1979.

[68] M. Fortin and R. Glowinski, *Méthodes de Lagrangien Augmenté*, Application à la résolution numérique de problèmes aux limites, Dunod-Bordas, Paris, 1982 (in French).

[69] M. Fortin and R. Glowinski, *Augmented Lagrangian Methods: Application to the Numerical Solution of Boundary Value Problems*, North-Holland, Amsterdam, 1983.

[70] R. Glowinski, *Numerical Methods for Nonlinear Variational Problems*, Springer-Verlag, New York, Berlin, Heidelberg, Tokyo, 1984.

[71] D. Bertsekas, *Constraint Optimization and Lagrange Multiplier Methods*, Academic Press, New York, 1982.

[72] P. Tseng, *Applications of a splitting algorithm to decomposition in convex programming and variational inequalities*, SIAM J. Control Optim., vol. 29, no. 1, pp. 119–138, 1991.

[73] B. He and H. Yang, *Some convergence properties of a method of multipliers for linearly constrained monotone variational inequalities*, Oper. Res. Lett., vol. 23, pp. 151–161, 1998.

[74] B. He, L. Liao, D. Han and H. Yang, *A new inexact alternating directions method for monotone variational inequalities*, Math. Progam., Ser. A, vol. 92, pp. 103–118, 2002.

[75] R. Glowinski, M. Holmstrom, *Constrained motion problems with applications by nonlinear programming methods*, Surveys on Math. for Industry, vol. 5, pp. 75–108, 1995.

[76] F. Delbos, J. Ch. Gilbert, R. Glowinski and D. Sinoquet, *Constrained optimization in seismic re ection tomography: a Gauss-Newton augmented Lagrangian approach*, Geophysical Journal International, vol. 164, pp. 670–684, 2006.

[77] G.P. McCormick and R.A. Tapia, *The gradient projection method under mild differentiability conditions*, SIAM J. Contr., vol. 10, pp. 93–98, 1972.

[78] P. H. Calamai and J. J. More, *Projected gradient methods for linearly constrained problems*, Mathematical Programming, vol. 39, pp. 93–116, 1987.

[79] J. Barzilai and J. M. Borwein,*Two-point step size gradient methods*, IMA J. Numer. Anal., vol. 8, pp. 141–148, 1988.

[80] M. Raydan, *Convergence Properties of the Barzilai and Borwein Gradient Method*, Ph.D. thesis, Department of Mathematical Sciences, Rice University, Houston, TX, U.S.A., 1991.

[81] L. Grippo, F. Lampariello and S. Lucidi, *A nonmonotone line search technique for Newton's method*, SIAM J. Numer. Anal., vol. 23, pp. 707–716, 1986.

[82] H. Zhang and W. W. Hager, *A nonmonotone line search technique and its application to unconstrained optimization*, SIAM J. Optim., vol. 14, pp. 1043–1056, 2004.

[83] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Verlag, New York, NY, 1999.

[84] S. Becker, J. Bobin and E. Candès, *NESTA: A fast and accurate first-order method for sparse recovery*, Technical Report, California Institute of Technology, April 2009.

[85] J. Bioucas-Dias and M. Figueiredo, *Two-step algorithms for linear inverse problems with non-quadratic regularization*, IEEE International Conference on Image Processing–ICIP2007, San Antonio, TX, USA, September 2007.

[86] J. Bioucas-Dias and M. Figueiredo, *A new TwIST: Two-step iterative thresholding algorithm for image restoration*, IEEE Trans. Imag. Process., vol. 16, no. 12, pp. 2992–3004, 2007.

[87] D. Manolakis, C. Siracusa and G. Shaw, *Hyperspectral subpixel target detection using linear mixing model*, IEEE Trans. Geosci. Remote Sensing, vol. 39, pp. 1392–1409, 2001.

[88] C. Chang and D. Heinz, *Subpixel spectral detection for remotely sensed images*, IEEE Trans. Geosci. Remote Sensing, vol. 38, 1144–1159, 2000.

[89] M. B. Lopes, J. C. Wolff, J. M. Bioucas-Dias, M. A. T. Figueiredo, *Near-infrared hyperspectral unmixing based on a minimum volume criterion for fast and accurate chemometric characterization of counterfeit tablets*, Analytical Chemistry, vol. 82, pp. 1462–1469, 2010.

[90] S. M. Chai, A. Gentile, W. E. Lugo-Beauchamp, J. L. Cruz-Rivera and D. S. Wills, *Hyper-spectral image processing applications on the SIMD pixel processor for the digital battlefield*, IEEE Workshop on Computer Vision Beyond the Visible Spectrum: Method and Applications, pp. 130–138, 1999.

[91] Li and A. H. Strahler, *Geometric-optical bidirectional reflectance modeling of the discrete-crown vegetation canopy: effect of crown shape and mutual shadowing*, IEEE Trans. Geosci. Remote Sens. vol. 30, pp. 276–292, 1992.

[92] J. M. Ellis, H. H. Davis and J. A. Zamudio, *Exploring for onshore oil seeps with hyperspectral imaging*, Oil and Gas Journal, pp. 49–56, 2001.

[93] K. W. Bowyer, K. Chang and P. Flynn, *A survey of approaches and challenges in 3D and multi-modal 3D+2D face recognition*, Computer Vision and Image Understanding, vol. 101, pp. 1–15, 2006.

[94] C. Witte, W. Armbruster and K. Jager, *Automatic generation of 3D models from real multisensor data*, 11th International Conference on Information Fusion, Cologne, Germany, 2008.

[95] L. D. Shefer and F. T. Marchese, *A system for real-time transcoding and delivery of video to smartphones*, International Conference on Information Visualisation, Los Alamitos, CA, USA, 2010.

[96] C. Li, H. Jiang, P. Wilford and Y. Zhang, *Video coding using compressive sensing for wireless communications*, IEEE Wireless Communications and Networking Conference (WCNC2011), Canjun, Mexico, March 2011.

[97] R. B. Smith, *Introduction to Hyperspectral Imaging*, MicroImages, Inc., July 2006.

[98] G. Vane, R. Green, T. Chrien, H. Enmark, E. Hansen and W. Porter, *The airborne visible/infrared imaging spectrometer (AVIRIS)*, Rem. Sens. of the Environ., vol. 44, pp. 127–143, 1993.

[99] T. Lillesand, R. Kiefer and J. Chipman, *Remote Sensing and Image Interpretation*, John Wiley & Sons, Inc., fifth edition, 2004.

[100] R. Clark and T. Roush, *Reflectance spectroscopy: Quantitative analysis techniques for remote sensing applications*, J. of Geophysical Research, vol. 89, pp. 6329–6340, 1984.

[101] J. Boardman, *Automating spectral unmixing of AVIRIS data using convex geometry concepts*, in JPL Pub.93-26, AVIRIS Workshop, vol. 1, pp. 11–14, 1993.

[102] M. E. Winter, *N-FINDR: an algorithm for fast autonomous spectral endmember determination in hyperspectral data*, in Proc. of the SPIE conference on Imaging Spectrometry V, vol. 3753, pp. 266–275, 1999.

[103] J. Nascimento and J. Bioucas-Dias, *Does independent component analysis play a role in unmixing hyperspectral data?*, IEEE Transactions on Geoscience and Remote Sensing, vol. 43, pp. 175–187, 2005.

[104] C. Chang, C. Wu, W. Liu and Y. Ouyang, *A new growing method for simplex-based endmember extraction algorithm*, IEEE Transactions on Geoscience and Remote Sensing, vol. 44, no. 10, pp. 2804–2819, 2006.

[105] L. Zhang, X. Tao, B. Wang and J. Zhang, *A new scheme for decomposition of mixed pixels based on nonnegative matrix factorization*, IEEE Internationla Geoscience and Remote sensing Symposium, pp. 1759–1762, 2007.

[106] J. Bioucas-Dias, *A variable splitting augmented Lagrangian approach to linear spectral unmixing*, In First IEEE Workshop on Hyperspectral Imaging and Signal Processing: Evolution in Remote Sensing, Grenoble, France, 2009.

[107] C. Chi T. Chan and W. Ma, *A convex analysis based minimum-volume enclosing simplex algorithm for hyperspectral unmixing*, in IEEE International Conference in Acoustics, Speech and Signal Porcessing-ICASSP2009, Taiwan, 2009.

[108] J. Li and J. Bioucas-Dias, *Minimum volume simplex analysis: a fast algorithm to unmix hyperspectral data*, in IEEE International Geoscience and Remote sensing Symposium -IGARSS2008, Boston, 2008.

[109] C. Li, T. Sun, K. Kelly and Y. Zhang, *A compressive sensing and unmixing scheme for hyperspectral data processing*, accepted by IEEE Transactions on Image Processing, July 2011.

[110] Z. Guo, T. Wittman and S. Osher, *L1 unmixing and its application to hyperspectral image enhancement*, UCLA CAM report, March 2009.

[111] Z. Wen, W. Yin and Y. Zhang, *Solving a Low-Rank Factorization Model for Matrix Completion by a Non-linear Successive Over-Relaxation Algorithm*, CAAM Technical Report TR10-07, Department of Computational and Applied Mathematics, Rice University, March 2010.

[112] H. Jiang, C. Li, R. Haimi-Cohen, P. Wilford and Y. Zhang, *Scalable video coding using compressive sensing*, accepted by Bell Labs Technical Journal, July 2011.

[113] J.-B. Lee and H. Kalva, *The VC-1 and H.264 Video Compression Standards for Broadband Video Services*, Springer, 2008.

[114] H. Schwarz, D. Marpe and T. Wiegand, *Overview of the scalable video coding extension of H.264/AVC*, IEEE Trans. Circuits and Systems for Video Tech., vol. 17, no. 9, pp. 1103-1120, Sept. 2007.

[115] H. Jiang and P. Wilford, *A Hierarchical Modulation for Upgrading Digital Broadcast Systems*, IEEE Ttrans. Broadcasting, vol. 51, no. 2, pp. 223-229, June 2005.

[116] C. Li 1, H. Xiong, J. Zou, T. Chen, *A Unified QoS Optimization for Scalable Video Multirate Multicast over Hybrid Coded Network*, 2010 IEEE International Conference on Communications (ICC), pp. 23-27, May 2010.

[117] S. Jakubczak, H. Rahul and D. Katabi, *SoftCast: One video to serve all wireless receivers*, Computer Science and Artificial Intelligence Laboratory Technical Report, MIT-CSAIL-TR-2009-005, MIT, Feb, 2009.

[118] J. Prades-Nebot, Y. Ma and T. Huang, *Distributed Video Coding using Compressive Sampling*, 2009 Picture Coding Symposium, PCS 2009, pp. 1-4, 2009.

[119] T. Do, Y. Chen, D. Nguyen, N. Nguyen, L. Gan and T. Tran, *Distributed compressed video sensing*, 16th IEEE International Conference on Image Processing (ICIP), pp. 1393-1396, 2009.

[120] R.E. Crochiere and L.R. Rabiner, *Mulirate Digital Signal Procesing*, Prentice Hall, 1983.

[121] S. Jakubczak, H. Rahul and D. Katabi, *SoftCast: One video to serve all wireless receivers*, Computer Science and Artificial Intelligence Laboratory Technical Report, MIT-CSAIL-TR-2009-005, MIT, Feb, 2009.

[122] NASA, *http://speclib.jpl.nasa.gov*.

[123] USGS Spectroscopy Lab,
http://speclab.cr.usgs.gov/spectral.lib04/spectral-lib04.html.

[124] US Army Corps of Engineers,
http://www.agc.army.mil/research/products/Hypercube/.

[125] C. Li, Y. Zhang and W. Yin,
http://www.caam.rice.edu/~optimization/L1/TVAL3/.