UNIVERSITY OF CALIFORNIA

Los Angeles

Learning-based Optimization for Signal and Image Processing

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Mathematics

by

Jialin Liu

2020

ABSTRACT OF THE DISSERTATION

Learning-based Optimization for Signal and Image Processing

by

Jialin Liu
Doctor of Philosophy in Mathematics
University of California, Los Angeles, 2020
Professor Wotao Yin, Chair

Incorporating machine learning techniques into optimization problems and solvers attracts increasing attention. Given a particular type of optimization problem that needs to be solved repeatedly, machine learning techniques can find some features for this category of optimization and develop algorithms with excellent performance. This thesis deals with algorithms and convergence analysis in learning-based optimization in three aspects: learning dictionaries, learning optimization solvers and learning regularizers.

Learning dictionaries for sparse coding is significant for signal processing. Convolutional sparse coding is a form of sparse coding with a structured, translation invariant dictionary. Most convolutional dictionary learning algorithms to date operate in the batch mode, requiring simultaneous access to all training images during the learning process, which results in very high memory usage, and severely limits the training data size that can be used. I proposed two online convolutional dictionary learning algorithms that offered far better scaling of memory and computational cost than batch methods and provided a rigorous theoretical analysis of these methods.

Learning fast solvers for optimization is a rising research topic. In recent years, unfolding iterative algorithms as neural networks has become an empirical success in solving sparse recovery problems. However, its theoretical understanding is still immature, which prevents us from fully utilizing the power of neural networks. I studied unfolded ISTA (Iterative Shrinkage Thresholding Algorithm) for sparse signal recovery and established its convergence.

Based on the properties of parameters required by convergence, the model can be significantly simplified and, consequently, has much less training cost and better recovery performance.

Learning regularizers or priors improves the performance of optimization solvers, especially for signal and image processing tasks. Plug-and-play (PnP) is a non-convex framework that integrates modern priors, such as BM3D or deep learning-based denoisers, into ADMM or other proximal algorithms. Although PnP has been recently studied extensively with great empirical success, theoretical analysis addressing even the most basic question of convergence has been insufficient. In this thesis, the theoretical convergence of PnP-FBS and PnP-ADMM was established, without using diminishing stepsizes, under a certain Lipschitz condition on the denoisers. Furthermore, real spectral normalization was proposed for training deep learning-based denoisers to satisfy the proposed Lipschitz condition.

The dissertation of Jialin Liu is approved.

University of California, Los Angeles

2020

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First of all, I would like to express my deepest appreciation to my advisor Prof. Wotao Yin for his continuous support of my Ph.D. study and research. His insightful views on optimization, his way of thinking and patient guidance have helped me find my research direction and build my research taste during the past five years. This thesis would not have been possible without his help.

I am also grateful to my committee members, Prof. Ali H. Sayed, Prof. Lieven Vandenberghe, and Prof. Luminita A. Vese for their valuable advice and feedback during the thesis defense.

I am truly indebted to Dr. Brendt Wohlberg in Los Alamos National Lab and Prof. Zhangyang Wang in Texas A&M University for their mentorship. Dr. Wohlberg mentored me during my internship in Los Alamos and led me into the area of image processing. Prof. Wang brought me plenty of cutting-edge insights and knowledge in machine learning. It's always pleasant to work with them.

My research would not have been easy without my great collaborators: Xiaohan Chen, Dr. Yat-Tin Chow, Dr. Cristina Garcia-Cardona, Dr. Jian Kang, Dr. Wuchen Li, Dr. Ernest Ryu, and Sicheng Wang. The fruitful collaborations taught me a lot on the research and applications of mathematics. Furthermore, I am grateful for all the support I have had from my friends. They are: Dr. Hanqin Cai, Fei Feng, Qi Guo, Xuchen Han, Howard Heaton, Yanli Liu, Dr. Daniel Mckenzie, Yuejiao Sun, Baichuan Yuan, and Dr. Kun Yuan.

Last but not least, I want to thank my family for their unconditional love and support.

VITA

2011–2015	B.S., Department of Automation, Tsinghua University, Beijing, China.
2015–2020	Teaching and Research Assistant, Department of Mathematics University of California, Los Angeles, Los Angeles, CA.
2016	Research Intern, Los Alamos National Lab, Los Alamos, NM.
2019	Research Intern, Alibaba US, DAMO Academy, Bellevue, WA.

PUBLICATIONS

J. Liu, X. Chen, Z. Wang, W. Yin, "ALISTA: Analytic Weights Are As Good As Learned Weights in LISTA." International Conference on Learning Representations (ICLR), 2019.

E. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, W. Yin, "Plug-and-Play Methods Provably Converge with Properly Trained Denoisers." International Conference on Machine Learning (ICML), 2019.

X. Chen, J. Liu, Z. Wang, W. Yin, "Theoretical Linear Convergence of Unfolded ISTA and its Practical Weights and Thresholds." Advances in Neural Information Processing Systems (NeurIPS), 2018.

J. Liu, W. Yin, W. Li, Y. T. Chow, "Multilevel Optimal Transport: a Fast Approximation of Wasserstein-1 distances." arXiv preprint arXiv:1810.00118 (2018).

J. Liu, C. Garcia-Cardona, B. Wohlberg, and W. Yin, “First- and Second-Order Methods for Online Convolutional Dictionary Learning.” SIAM Journal on Imaging Sciences (SIIMS), 2018.

J. Liu, C. Garcia-Cardona, B. Wohlberg, and W. Yin, “Online Convolutional Dictionary Learning.” IEEE International Conference on Image Processing (ICIP), 2017.

CHAPTER 1

Introduction

1.1 Background of learning-based optimization

Optimization is a field of studying and finding the minima or maxima of objective functions over given sets. Optimization problems are usually mathematical abstractions of making the optimal decisions from a set of candidates in the real world. In the area of signal processing and image processing, many problems can be described as optimization problems, such as signal restoration, image denoising, super-resolution, image recognization and classification, etc. Consequently, designing and solving optimization problems have become significant to signal and image processing.

The past decade has witnessed the extraordinary development of machine learning techniques that provides computer systems the ability to automatically learn from data and experience without being explicitly programmed for specific tasks. These learning-based methods have become overwhelmingly successful in some research areas such as computer vision and natural language processing. Meanwhile, learning-based methods also promote the development of optimization in two folds:

- Learning better optimization formulations and input data. Machine learning techniques can help us find patterns that human beings do not recognize, which helps us design optimization problems that are closer to the real world than those manually designed. Therefore, the mathematical solution will be closer to the optimal solution in the real world.

- Learning better optimization solvers.[1] Many practical optimization problems do not have a closed-form solution but are solvable by iterative algorithms. Thus, the convergence of an algorithm is a critical factor that makes us determine whether to use it in practice. Given a certain type of optimization problems that would be solved repeatedly, machine learning techniques can find some features for this category of optimization and develop new solvers with good performance that can be generalized to similar problems.

1.2 Motivation of this thesis

Plenty of works along the above two tracks have shown empirical success. However, theoretical analysis and understanding of this topic are still not enough to provide models or algorithms that are explainable and robust in practice. Specifically, the convergence theory, one of the core theories in optimization, is built based on the mathematical properties of both optimization problems and solvers. For example, for a convex minimization problem $\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x})$, the convergence of gradient descent algorithm $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha\nabla f(\mathbf{x}^{(k)})$ requires two assumptions: the gradient of the objective function $\nabla f(\mathbf{x})$ is Lipschitz continuous and the step size of gradient descent α is small enough. If the optimization problems and solvers are designed by machine learning, the above assumptions may not be satisfied and the convergence may not be guaranteed. This thesis focuses on several key open questions in learning-based optimization:

- To guarantee convergence, what assumptions should we make on learning-based optimization problems and solvers?
- To meet the assumptions, what regularizations or limitations should we make on the learning models?
- With everything above settled, what does the algorithm converge to?
- What's the convergence rate compared with classical solvers on classical problems?

[1]By "solvers" in this thesis, we mean algorithms to solve a specific or a type of optimization problems.

- With the theories established and better understanding obtained, can we propose new algorithms with better performance?

In this dissertation, we try to move forward towards understanding the above questions on optimization problems of the following form:

$$\underset{\mathbf{x}\in\mathbb{R}^M}{\text{minimize}} \quad f(\mathbf{x}) + \gamma g(\mathbf{x}). \tag{1.1}$$

where the optimization variable $\mathbf{x} \in \mathbb{R}^M$ represents a signal, $f(\mathbf{x})$ measures data fidelity, $g(\mathbf{x})$, the *regularizer*, measures noisiness or complexity of the signal, and $\gamma \geq 0$ is a parameter representing the relative importance between f and g. Many modern signal and image processing problems such as total variation denoising, inpainting, and compressed sensing fall under this setup. *A priori* knowledge of the signal, such as that the signal should have small noise, is encoded in $g(\mathbf{x})$. So $g(\mathbf{x})$ is small if $\mathbf{x}$ has small noise or complexity. For example, g might be ℓ_1 norm $\|\mathbf{x}\|_1$, or the total variation $\|\nabla\mathbf{x}\|_1$, etc. *A posteriori* knowledge of the signal, such as noisy or partial measurements of the signal, is encoded in $f(\mathbf{x})$. So $f(\mathbf{x})$ is small if $\mathbf{x}$ agrees with the measurements.

1.3 Introduction to successive chapters

1.3.1 Learning dictionaries in linear measurements

In Chapter 2, we start with sparse coding, a special case of (1.1). *Sparse signal representation* aims to represent a given signal by a linear combination of only a few elements of a fixed set of signal components [MBP14]. For example, we can approximate an N-dimensional signal $\mathbf{b} \in \mathbb{R}^N$ as

$$\mathbf{b} \approx \mathbf{D}\mathbf{x} = \mathbf{d}_1 x_1 + \ldots + \mathbf{d}_M x_M \ ,$$

where

$$\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_M] \in \mathbb{R}^{N\times M}$$

is the *dictionary* with M *atoms* and

$$\mathbf{x} = [x_1, x_2, \cdots, x_M]^T \in \mathbb{R}^M$$

is the sparse representation. The problem of computing the sparse representation $\mathbf{x}$ given $\mathbf{b}$ and $\mathbf{D}$ is referred to as *sparse coding.* Among a variety of formulations of this problem, we focus on Basis Pursuit Denoising (BPDN) [CDS98]

$$\min_{\mathbf{x}\in\mathbb{R}^M} (1/2)\|\mathbf{D}\mathbf{x}-\mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1 \ . \tag{1.2}$$

With $f(\mathbf{x}) = (1/2)\|\mathbf{b}-\mathbf{D}\mathbf{x}\|^2$ and $g(\mathbf{x}) = \|\mathbf{x}\|_1$, problem (1.1) reduces to (1.2).

Sparse codings have been used in a wide variety of applications, including denoising [EA06, MBP14, KHL20], super-resolution [YWH10, ZXY15], classification [WYG09], and face recognition [WMM10]. A key issue when solving sparse coding problems as in (1.2) is how to choose the dictionary $\mathbf{D}$. Early work on sparse codings used a fixed basis [RBE10] such as wavelets [Mal99] or Discrete Cosine Transform (DCT) [HA07], but learned dictionaries can provide better performance [AEB06, EA06].

Dictionary learning aims to learn a good dictionary $\mathbf{D}$ for a given distribution of signals. If $\mathbf{b}$ is a random variable, the dictionary learning problem can be formulated as

$$\min_{\mathbf{D}\in\mathcal{C}} \mathbb{E}_{\mathbf{b}}\left\{ \min_{\mathbf{x}} \frac{1}{2}\|\mathbf{D}\mathbf{x}-\mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1 \right\} , \tag{1.3}$$

where $\mathcal{C} = \{\mathbf{D} \mid \|\mathbf{d}_m\|_2^2 \leq 1, \forall m\}$ is the constraint set, which is necessary to resolve the scaling ambiguity between $\mathbf{D}$ and $\mathbf{x}$.

Given a batch of signal samples $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_K\}$ for training, *batch dictionary learning methods* (e.g. [ERK99, EAH99, AEB06, XY16]) minimize an objective function such as

$$\min_{\mathbf{D}\in\mathcal{C},\mathbf{x}} \sum_{k=1}^{K} \left\{ \frac{1}{2}\|\mathbf{D}\mathbf{x}_k-\mathbf{b}_k\|_2^2 + \lambda\|\mathbf{x}_k\| \right\} .$$

These methods require simultaneous access to all the training samples during training.

In Chapter 2, we develop two effcient algorithms with convergence guarantee to solve the dictionary learning problem (1.3) under the following settings:

- The dictionary $\mathbf{D}$ is a convolutional operator. With this special structure, we can handle signal $\mathbf{b}$ with much larger size.

- The training samples are given in the process of training in a streaming way:

  $$\mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \cdots$$

  where $(1), (2), \cdots$ are training epoches. This *online learning* setting can significantly reduce the memory and time usage.

1.3.2 Learning fast solvers

In Chapter 3, we study the sparse coding problem (1.2) with a *given* dictionary $\mathbf{D}$ and develop fast solvers by *learning to optimize* (L2O), a recent proposed technique that gains rising attention. It learns optimization solvers and usually obtains good performance in signal (image) processing problems. In general, an iterative optimization solver for (1.1) can be written as

$$\mathbf{x}^{(k+1)} = \mathcal{T}_{f,g}(\mathbf{x}^{(k)}), \quad k = 0, 1, 2, \cdots \tag{1.4}$$

where $\mathcal{T}_{f,g}$ is an operator designed manually based on the optimization objectives f and g in problem (1.1). In L2O, the operator $\mathcal{T}_{f,g}$ is parameterized as $\tilde{\mathcal{T}}_{f,g}$:

$$\mathbf{x}^{(k+1)} = \tilde{\mathcal{T}}_{f,g}(\mathbf{x}^{(k)}; \theta^{(k)}), \quad k = 0, 1, 2, \cdots \tag{1.5}$$

where $\{\theta^{(k)}\}_k$ are parameters to determine. Recent machine learning techniques can help us find the parameters that lead to faster convergence and better recovery performance. For example, *unfolding algorithms*, one of the L2O techniques, unfolds the iterative algorithm (1.5) and truncates it into K iterations (a typical value of K is $10 \sim 20$):

$$\mathbf{x}^{(k+1)} = \tilde{\mathcal{T}}_{f,g}(\mathbf{x}^{(k)}; \theta^{(k)}), \quad k = 0, 1, 2, \cdots, K-1.$$

The last iterate $\mathbf{x}^{(K)}$ depends on $\mathbf{x}^{(0)}, \{\theta^{(k)}\}_{k=1}^{K-1}, f$ and g. In this thesis, we assume $\mathbf{x}^{(0)}$ is fixed. Thus, $\mathbf{x}^{(K)}$ can be written as $\mathbf{x}^{(K)}\big(\{\theta^{(k)}\}_{k=1}^{K-1}; f, g\big)$. Then the parameters $\{\theta^{(k)}\}_{k=1}^{K-1}$ are determined by solving

$$\min_{\{\theta^{(k)}\}_{k=1}^{K-1}} \mathbb{E}_{(f,g)\sim\mathcal{D}} \Big\| \mathbf{x}^{(K)}\big(\{\theta^{(k)}\}_{k=1}^{K-1}; f, g\big) - \mathbf{x}^*(f, g) \Big\|^2 \tag{1.6}$$

where $\mathcal{D}$ is a set consisting of some known instances of optimization objectives f, g and $\mathbf{x}^*$ is the target that we want the algorithm to converge to.

Problem (1.6) can be solved approximately by recent developed machine learning platforms (TensorFlow [AAB15], PyTorch [PGM19], etc.). The process of solving (1.6) is called *training* and the data set $\mathcal{D}$ is called the *training set.* With (1.6) solved, the obtained solver $\mathbf{x}^{(k+1)} = \tilde{\mathcal{T}}_{f,g}(\mathbf{x}^{(k)}; \theta^{(k)})$ is named as a *learned solver* in this thesis. A learned solver can be generalized well to optimization problems similar with those in the training set. [GL10, SBS15, WCZ16, WLH16, WLC16, WYC16, SLX16, BSR17, ZG18a, AO18, ZDD18, ITW18]

In Chapter 3, we develop algorithms that are a result of training the unfolded iterations of Iterative Shrinkage-Thresholding Algorithm (ISTA). Some convergence theories are first established for the learned solvers and, based on the theories, the learned solvers are interpretable and capable of being simplified to reduce the training cost.

1.3.3 Learning regularizers

In Chapter 4, we consider the generic problem (1.1) and learn the regularizer g in the framework of *Plug-and-Play.*

First we describe Plug-and-Play here. Plug-and-Play is built based on first-order iterative methods, which are often used to solve optimization problem (1.1), and ADMM is one such method:

$$\begin{aligned}
\mathbf{x}^{(k+1)} &= \arg\min_{\mathbf{x}\in\mathbb{R}^M} \left\{\sigma^2 g(\mathbf{x}) + (1/2)\|\mathbf{x} - (\mathbf{y}^{(k)} - \mathbf{u}^{(k)})\|^2\right\} \\
\mathbf{y}^{(k+1)} &= \arg\min_{\mathbf{y}\in\mathbb{R}^M} \left\{\alpha f(\mathbf{y}) + (1/2)\|y - (\mathbf{x}^{(k+1)} + \mathbf{u}^{(k)})\|^2\right\} \\
\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \mathbf{x}^{(k+1)} - \mathbf{y}^{(k+1)}
\end{aligned}$$

with $\sigma^2 = \alpha\gamma$. Given a function h on $\mathbb{R}^M$ and $\alpha > 0$, define the proximal operator of h as

$$\mathrm{Prox}_{\alpha h}(z) = \arg\min_{\mathbf{x}\in\mathbb{R}^M} \left\{\alpha h(\mathbf{x}) + (1/2)\|\mathbf{x} - \mathbf{z}\|^2\right\},$$

which is well-defined if h is proper, closed, and convex. Now we can equivalently write ADMM as

$$\begin{aligned}
\mathbf{x}^{(k+1)} &= \mathrm{Prox}_{\sigma^2 g}(\mathbf{y}^{(k)} - \mathbf{u}^{(k)}) \\
\mathbf{y}^{(k+1)} &= \mathrm{Prox}_{\alpha f}(\mathbf{x}^{(k+1)} + \mathbf{u}^{(k)}) \\
\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \mathbf{x}^{(k+1)} - \mathbf{y}^{(k+1)}.
\end{aligned}$$

We can interpret the subroutine $\mathrm{Prox}_{\sigma^2 g} : \mathbb{R}^M \to \mathbb{R}^M$ as a denoiser (regularizer)[2], i.e.,

$$\mathrm{Prox}_{\sigma^2 g} : \text{noisy image} \mapsto \text{less noisy image}$$

(For example, if σ is the noise level and $g(\mathbf{x})$ is the total variation (TV) norm, then $\mathrm{Prox}_{\sigma^2 g}$ is the standard Rudin–Osher–Fatemi (ROF) model [ROF92].) We can think of $\mathrm{Prox}_{\alpha f} : \mathbb{R}^M \to \mathbb{R}^M$ as a mapping enforcing consistency with measured data, i.e.,

$$\mathrm{Prox}_{\alpha f} : \text{less consistent} \mapsto \text{more consistent with data}$$

More precisely speaking, for any $\mathbf{x} \in \mathbb{R}^M$ we have

$$g(\mathrm{Prox}_{\sigma^2 g}(\mathbf{x})) \leq g(\mathbf{x}), \qquad f(\mathrm{Prox}_{\alpha f}(\mathbf{x})) \leq f(\mathbf{x}).$$

However, some state-of-the-art image denoisers with great empirical performance do not originate from optimization problems. Such examples include non-local means (NLM) [BCM05], Block-matching and 3D filtering (BM3D) [DFK07], and *learning-based* models like convolutional neural networks (CNN) [ZZC17]. Nevertheless, such a denoiser $H_\sigma : \mathbb{R}^M \to \mathbb{R}^M$ still has the interpretation

$$H_\sigma : \text{noisy image} \mapsto \text{less noisy image}$$

where $\sigma \geq 0$ is a noise parameter. Larger values of σ correspond to more aggressive denoising.

Is it possible to use such denoisers for a broader range of imaging problems, even though we cannot directly set up an optimization problem? To address this question, [VBW13]

[2]In this thesis, we focus on regularizers of denoising and use "denoiser" to refer regularizer.

proposed *Plug-and-Play ADMM (PnP-ADMM)*, which simply replaces the proximal operator $\mathrm{Prox}_{\sigma^2 g}$ with the denoiser H_σ:

$$\begin{aligned}
\mathbf{x}^{(k+1)} &= H_\sigma(\mathbf{y}^{(k)} - \mathbf{u}^{(k)}) \\
\mathbf{y}^{(k+1)} &= \mathrm{Prox}_{\alpha f}(\mathbf{x}^{(k+1)} + \mathbf{u}^{(k)}) \\
\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \mathbf{x}^{(k+1)} - \mathbf{y}^{(k+1)}.
\end{aligned} \tag{1.7}$$

Surprisingly and remarkably, this ad-hoc method exhibited great empirical success, and spurred much follow-up work. The empirical success of Plug-and-Play (PnP) naturally leads us to ask theoretical questions: When does PnP converge and what denoisers can we use? Past theoretical analysis has been insufficient.

In Chapter 4, we study the convergence of PnP. An assumption on the denoiser (or regularizer) H_σ is proposed for convergence, and we develop a training method to enforce learning-based denoisers meet the assumption.

CHAPTER 2

Learning Dictionaries

In this chapter, we focus on dictionary learning (introduced in Chapter 1)

$$\min_{\mathbf{D}\in\mathcal{C}} \mathbb{E}_{\mathbf{b}}\left\{ \min_{\mathbf{x}} \frac{1}{2}\|\mathbf{D}\mathbf{x}-\mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1 \right\} , \tag{2.1}$$

with a *convolutional structure* in the settings of *online learning.* Section 2.1 describes convolutional dictionary learning and its computational issues; Section 2.2 introduces notation and necssary math tools; In Sections 2.3 and 2.4, we propose two online convolutional dictionary learning algorithms with convergence guarantee; Section 2.5 provides some numerical results; Section 2.6 concludes this chapter.

2.1 Convolutional Sparse Coding

Convolutional Sparse Coding (CSC) [LS99, ZKT10] [Woh16d, Sec. II], a highly structured sparse representation model, has recently attracted increasing attention for a variety of imaging inverse problems [GZX15, LCW16, ZP16, QJ16, Woh16c, ZP17]. CSC aims to represent a given signal $\mathbf{b}\in\mathbb{R}^N$ as a sum of convolutions,

$$\mathbf{b} \approx \mathbf{d}_1 * \mathbf{x}_1 + \ldots + \mathbf{d}_M * \mathbf{x}_M \ , \tag{2.2}$$

where dictionary atoms $\{\mathbf{d}_m\}_{m=1}^M$ are linear filters and the representation $\{\mathbf{x}_m\}_{m=1}^M$ is a set of *coefficient maps*, each map $\mathbf{x}_m$ having the same size N as the signal $\mathbf{b}$. Since we implement the convolutions in the frequency domain for computational efficiency, it is convenient to adopt *circular boundary conditions* for the convolution operation.

Given $\{\mathbf{d}_m\}$ and $\mathbf{b}$, the maps $\{\mathbf{x}_m\}$ can be obtained by solving the Convolutional Basis

Pursuit DeNoising (CBPDN) ℓ_1-minimization problem

$$\min_{\{\mathbf{x}_m\}} \frac{1}{2}\Big\| \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{x}_m - \mathbf{b}\Big\|_2^2 + \lambda \sum_{m=1}^{M} \|\mathbf{x}_m\|_1 \ . \tag{2.3}$$

The corresponding dictionary learning problem is called *Convolutional Dictionary Learning (CDL)*. Specifically, given a set of K training signals $\{\mathbf{b}_k\}_{k=1}^K$, CDL is implemented via minimization of the function

$$\min_{\{\mathbf{d}_m\},\{\mathbf{x}_{k,m}\}} \frac{1}{2}\sum_{k=1}^{K} \Big\| \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{x}_{k,m} - \mathbf{b}_k\Big\|_2^2 + \lambda \sum_{k=1}^{K}\sum_{m=1}^{M} \|\mathbf{x}_{k,m}\|_1$$
$$\text{subject to} \|\mathbf{d}_m\|_2 \leq 1, \ \forall m \in \{1, \dots, M\} \ , \tag{2.4}$$

where the coefficient maps $\mathbf{x}_{k,m}$, $k \in \{1, \dots, K\}$, $m \in \{1, \dots, M\}$, represent $\mathbf{b}_k$, and the norm constraint avoids the scaling ambiguity between $\mathbf{d}_m$ and $\mathbf{x}_{k,m}$.

Most current CDL algorithms [BEL13, HHW15, GZX15, Woh16d, vv16, Woh16a, GW17, CF17] are batch learning methods that alternatively minimize over $\{\mathbf{x}_{k,m}\}$ and $\{\mathbf{d}_m\}$, dealing with the entire training set at each iteration. When K is large, the $\mathbf{d}_m$ update subproblem is computationally expensive, e.g. the single step complexity and memory usage are both $\mathcal{O}(KMN \log(N))$ for one of the current state-of-the-art methods [vv16, GW17]. For example, for a medium-sized problem with $K = 40, N = 256 \times 256, M = 64$, we have $KMN \log(N) \approx 10^9$, which is computationally very expensive.

2.2 Preliminaries

Here we introduce our notation. The signal is denoted by $\mathbf{b} \in \mathbb{R}^N$, and the dictionaries by $\mathbf{d} = (\mathbf{d}_1 \ \mathbf{d}_2 \ \dots \ \mathbf{d}_M)^T \in \mathbb{R}^{MD}$, where the dictionary kernels (or filters) are $\mathbf{d}_m \in \mathbb{R}^D$. The coefficient maps are denoted by $\mathbf{x} = (\mathbf{x}_1 \ \mathbf{x}_2 \ \dots \ \mathbf{x}_M)^T \in \mathbb{R}^{MN}$, where $\mathbf{x}_m \in \mathbb{R}^N$ is the coefficient map corresponding to $\mathbf{d}_m$. In addition to the *vector form*, $\mathbf{x}$, of the coefficient maps, we define an *operator form* $\mathbf{X}$. First we define a linear operator $\mathbf{X}_m$ on $\mathbf{d}_m$ such that $\mathbf{X}_m\mathbf{d}_m = \mathbf{d}_m * \mathbf{x}_m$ and let $\mathbf{X} \triangleq \big(\mathbf{X}_1 \ \mathbf{X}_2 \cdots \mathbf{X}_M\big)$. Then, we have

$$\mathbf{X}\mathbf{d} \triangleq \sum_{m=1}^{M} \mathbf{X}_m \mathbf{d}_m = \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{x}_m \approx \mathbf{b} \ . \tag{2.5}$$

Hence, $\mathbf{X} : \mathbb{R}^{MD} \to \mathbb{R}^N$, a linear operator defined from the dictionary space to the signal space, is the operator form of $\mathbf{x}$.

2.2.1 Problem settings

To introduce online algorithms, we reformulate (2.4) into a more general form. Usually, the signal is sampled from a large training set, but we consider the training signal $\mathbf{b}$ as a random variable following the distribution $\mathbf{b} \sim P(\mathbf{b})$. Our goal is to optimize the dictionary $\mathbf{d}$. Given $\mathbf{b}$, the loss function l to evaluate $\mathbf{d}, \mathbf{x}$ is defined as

$$l(\mathbf{d}, \mathbf{x}; \mathbf{b}) = (1/2)\|\mathbf{X}\mathbf{d} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1 \; . \tag{2.6}$$

Given $\mathbf{b}$, the loss function f to evaluate $\mathbf{d}$ and the corresponding minimizer are respectively,

$$f(\mathbf{d}; \mathbf{b}) \triangleq \min_{\mathbf{x}} l(\mathbf{d}, \mathbf{x}; \mathbf{b}) \quad \text{and} \quad \mathbf{x}^*(\mathbf{d}; \mathbf{b}) \triangleq \arg\min_{\mathbf{x}} l(\mathbf{d}, \mathbf{x}; \mathbf{b}) \; . \tag{2.7}$$

A general CDL problem can be formulated as

$$\min_{\mathbf{d} \in \mathcal{C}} \mathbb{E}_{\mathbf{b}}[f(\mathbf{d}; \mathbf{b})] \; , \tag{2.8}$$

where $\mathcal{C}$ is the constraint set of $\mathcal{C} = \{\mathbf{d} \mid \|\mathbf{d}_m\|^2 \leq 1, \forall m\}$.

2.2.2 Two online frameworks

Now we consider the CDL problem (2.8) when the training signals $\mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \cdots, \mathbf{b}^{(t)}, \cdots$ arrive in a streaming fashion. Inspired by online methods for standard dictionary learning problems, we propose two online frameworks for CDL problem (2.8). One is a first order method based on Projected Stochastic Gradient Descent (SGD) [WYY10, MBP12, AE08]:

$$\mathbf{d}^{(t)} = \mathrm{Proj}_{\mathcal{C}}\Big(\mathbf{d}^{(t-1)} - \eta^{(t)} \nabla f\big(\mathbf{d}^{(t-1)}; \mathbf{b}^{(t)}\big)\Big) \; . \tag{2.9}$$

The other is a second order method, which is inspired by least squares estimator for dictionary learning [MBP09, SE10, SPL11, ZJD12, SG14, ZKY15, KWB12]. A naive least squares estimator can be written as

$$\mathbf{d}^{(t)} = \arg\min_{\mathbf{d} \in \mathcal{C}} \Big\{ \underbrace{\min_{\mathbf{x}} \ell(\mathbf{d}, \mathbf{x}, \mathbf{b}^{(1)}) + \cdots + \min_{\mathbf{x}} \ell(\mathbf{d}, \mathbf{x}, \mathbf{b}^{(t)})}_{\text{Objective function on training samples } F^{(t)}(\mathbf{d})} \Big\} \; .$$

This is not practical because the inner minimizer of $\mathbf{x}$ depends on $\mathbf{d}$, which is unknown. To solve this problem, we can fix $\mathbf{d}$ when we minimize over $\mathbf{x}$, i.e.

$$\mathbf{x}^{(t)} = \arg\min_{\mathbf{x}} \ell(\mathbf{d}^{(t-1)}, \mathbf{x}; \mathbf{b}^{(t)}). \tag{2.10a}$$

$$\mathbf{d}^{(t)} = \arg\min_{\mathbf{d}\in\mathcal{C}} \Big\{ \underbrace{\ell(\mathbf{d}, \mathbf{x}^{(1)}, \mathbf{b}^{(1)}) + \cdots + \ell(\mathbf{d}, \mathbf{x}^{(t)}, \mathbf{b}^{(t)})}_{\text{Surrogate function } \mathcal{F}^{(t)}(\mathbf{d})} \Big\} . \tag{2.10b}$$

Direct application of these methods to the CDL problem is very computationally expensive, but we propose a number of techniques to reduce the time and memory usage. The details are discussed in Sections 2.3 and 2.4 respectively.

2.2.3 Techniques to calculate operator X

Before introducing our algorithms for (2.8), we consider a basic problem and two computational techniques that are used in this section as well as in Sections 2.3 and 2.4.

With $\mathbf{b}$ and $\mathbf{x}$ fixed, the basic problem is

$$\min_{\mathbf{d}\in\mathbb{R}^{MD}} l(\mathbf{d}, \mathbf{x}; \mathbf{b}) + \iota_{\mathcal{C}}(\mathbf{d}) , \tag{2.11}$$

where $\iota_{\mathcal{C}}(\cdot)$ is the indicator function[1] of set $\mathcal{C}$. To solve this problem we can apply projected gradient descent (GD) [Ber99]

$$\mathbf{d}^{(t)} = \mathrm{Proj}_{\mathcal{C}}\Big(\mathbf{d}^{(t-1)} - \eta^{(t)}\mathbf{X}^T\big(\mathbf{X}\mathbf{d}^{(t-1)} - \mathbf{b}\big)\Big) , \tag{2.12}$$

where (t) is the iteration index and $\mathbf{X}^T\big(\mathbf{X}\mathbf{d} - \mathbf{b}\big)$ is the gradient of l with respect to $\mathbf{d}$. Since $\mathbf{X}$ is a linear operator from $\mathbb{R}^{MD}$ to $\mathbb{R}^N$, the cost of directly computing (2.12) is $\mathcal{O}(NMD)$. However, we can exploit the sparsity or the structure of operator $\mathbf{X}$ to yield a more efficient computation that greatly reduces the time complexity.

2.2.3.1 Computing with sparsity property

The first option is to utilize the sparsity of $\mathbf{X}$. Specifically, $\mathbf{X}$ is saved as a triple array (i, j, v), which records the indices (i, j) and values v of the non-zero elements of $\mathbf{X}$, so that only the

[1] The indicator function is defined as: $\iota_{\mathcal{C}}(\mathbf{d}) = \begin{cases} 0, & \text{if } \mathbf{d} \in \mathcal{C} \\ +\infty, & \text{otherwise} \end{cases}$.

nonzero entries in $\mathbf{X}$ contribute to the computational time. This triple array is commonly referred as a *coordinate list* and is a standard way of representing a sparse matrix.

Let us compute the non-zero entries of operator $\mathbf{X}$. The operator form $\mathbf{X}_m$ of the N-dimensional vectors $\mathbf{x}_m = ((x_m)_1, \cdots, (x_m)_N)^T$ can be written as

$$\mathbf{X}_m = \begin{bmatrix} (x_m)_1 & (x_m)_N & (x_m)_{N-1} & \dots & (x_m)_{N-D+2} \\ (x_m)_2 & (x_m)_1 & (x_m)_N & \dots & (x_m)_{N-D+3} \\ (x_m)_3 & (x_m)_2 & (x_m)_1 & \dots & (x_m)_{N-D+4} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (x_m)_N & (x_m)_{N-1} & (x_m)_{N-2} & \dots & (x_m)_{N-D+1} \end{bmatrix},$$

where each column is a circular shift of $\mathbf{x}_m$ and D is the dimension of each dictionary kernel. Thus, the density of $\mathbf{x}_m$ and $\mathbf{X}_m$ are the same. Assuming the density of vector $\mathbf{x}$ is ρ, the number of nonzero entries of operator $\mathbf{X}$ is $NMD\rho$, giving a single step complexity of $\mathcal{O}(NMD\rho)$ for computing (2.12).

2.2.3.2 Computing in the frequency domain

Another option is to utilize the structure of $\mathbf{X}$. It is well known that convolving two signals of the same size corresponds to the pointwise multiplication of their frequency representations. Our method below takes advantage of this property. First, we zero-pad each $\mathbf{d}_m$ from $\mathbb{R}^D$ to $\mathbb{R}^N$ to match the size of $\mathbf{b}$. Then the basic problem can be written as

$$\min_{\mathbf{d} \in \mathbb{R}^{MN}} l(\mathbf{d}, \mathbf{x}; \mathbf{b}) + \iota_{\mathcal{C}_{\mathrm{PN}}}(\mathbf{d}) , \tag{2.13}$$

where the set $\mathcal{C}_{\mathrm{PN}}$ is defined as

$$\mathcal{C}_{\mathrm{PN}} \triangleq \{\mathbf{d}_m \in \mathbb{R}^N : (\mathbf{I} - \mathbf{P})\mathbf{d}_m = 0, \|\mathbf{d}_m\|^2 \leq 1\} . \tag{2.14}$$

Operator $\mathbf{P}$ preserves the desired support of $\mathbf{d}_m$ and masks the remaining part to zeros. Projected GD (2.12) has an equivalent form:

$$\mathbf{d}^{(t)} = \mathrm{Proj}_{\mathcal{C}_{\mathrm{PN}}}\Big(\mathbf{d}^{(t-1)} - \eta^{(t)} \frac{\partial l}{\partial \mathbf{d}}(\mathbf{d}^{(t-1)}, \mathbf{x}; \mathbf{b})\Big) . \tag{2.15}$$

Then, using the Plancherel formula, we can write the loss function[2] l as

$$l(\mathbf{d},\mathbf{x};\mathbf{b}) = \underbrace{\Big\|\sum_m \mathbf{d}_m * \mathbf{x}_m - \mathbf{b}\Big\|_2^2}_{\|\mathbf{X}\mathbf{d}-\mathbf{b}\|^2} = \underbrace{\Big\|\sum_m \hat{\mathbf{d}}_m \odot \hat{\mathbf{x}}_m - \hat{\mathbf{b}}\Big\|_2^2}_{\|\hat{\mathbf{X}}\hat{\mathbf{d}}-\hat{\mathbf{b}}\|^2}, \tag{2.16}$$

where $\hat{\cdot}$ denotes the corresponding quantity in the frequency domain and $\odot$ means pointwise multiplication. Therefore, we have $\hat{\mathbf{d}} \in \mathbb{C}^{MN}$, and $\hat{\mathbf{X}} = \big(\hat{\mathbf{X}}_1\ \hat{\mathbf{X}}_2 \cdots \hat{\mathbf{X}}_M\big)$ is a linear operator. Define the loss function in the frequency domain

$$\hat{l}(\hat{\mathbf{d}},\hat{\mathbf{x}};\hat{\mathbf{b}}) = (1/2)\big\|\hat{\mathbf{X}}\hat{\mathbf{d}} - \hat{\mathbf{b}}\big\|^2 , \tag{2.17}$$

which is a real valued function defined in the complex domain. The Cauchy-Riemann condition [Ahl79] implies that (2.17) is not differentiable unless it is constant. However, the *conjugate cogradient*[3] [SBL12]

$$\frac{\partial \hat{l}}{\partial \hat{\mathbf{d}}}(\hat{\mathbf{d}},\hat{\mathbf{x}};\hat{\mathbf{b}}) \triangleq \hat{\mathbf{X}}^H(\hat{\mathbf{X}}\hat{\mathbf{d}} - \hat{\mathbf{b}}) . \tag{2.18}$$

exists and can be used for minimizing (2.17) by gradient descent.

Since each item $\hat{\mathbf{X}}_m$ in $\hat{\mathbf{X}}$ is diagonal, the gradient is easy to compute, with a complexity of $\mathcal{O}(NM)$, instead of $\mathcal{O}(NMD)$. Based on (2.18), we have the following modified gradient descent:

$$\mathbf{d}^{(t)} = \mathrm{Proj}_{\mathcal{C}_{\mathrm{PN}}}\left(\mathrm{IFFT}\Big(\hat{\mathbf{d}}^{(t-1)} - \eta^{(t)}\frac{\partial \hat{l}}{\partial \hat{\mathbf{d}}}\big(\hat{\mathbf{d}}^{(t-1)},\hat{\mathbf{x}};\hat{\mathbf{b}}\big)\Big)\right) . \tag{2.19}$$

To compute (2.19), we transform $\mathbf{d}^{(t)}$ into its frequency domain counterpart $\hat{\mathbf{d}}^{(t)}$, perform gradient descent in the frequency domain, return to the spatial domain, and project the result onto the set $\mathcal{C}_{\mathrm{PN}}$.

In our modified method (2.19), the iterate $\mathbf{d}^{(t)}$ is transformed between the frequency and spatial domains because the gradient is cheaper to compute in the frequency domain, but projection is cheaper to compute in the spatial domain.

Equivalence of (2.15) and (2.19). We can prove

$$\hat{\mathbf{X}}^H(\hat{\mathbf{X}}\hat{\mathbf{d}} - \hat{\mathbf{b}}) = \mathrm{FFT}\big(\mathbf{X}^T(\mathbf{X}\mathbf{d} - \mathbf{b})\big) , \quad \forall \mathbf{x},\mathbf{d},\mathbf{b} , \tag{2.20}$$

[2]We ignore the term $\lambda\|\mathbf{x}\|_1$ in l here because $\mathbf{x}$ is fixed in this problem.

[3]The conjugate cogradient of function $f(x) : \mathbb{C}^n \to \mathbb{R}$ is defined as: $\frac{\partial f}{\partial \Re(x)} + i\frac{\partial f}{\partial \Im(x)}$, where $\Re(x)$, $I\Im(x)$ are the real part and imaginary part of x. The derivation of (2.17) is given in Appendix 2.A.

which means that the conjugate cogradient of $\hat{l}$ is equivalent to the gradient of l. Thus, modified GD (2.19) coincides with standard GD (2.15) using conjugate cogradient.

A proof of (2.20) given in Appendix 2.B. A similar result is also given in [RSA15] under the name "conjugate symmetry".

2.3 First-order method: Algorithm 1

Recall the Projected SGD step (2.9)

$$\mathbf{d}^{(t)} = \mathrm{Proj}_{\mathcal{C}}\Big(\mathbf{d}^{(t-1)} - \eta^{(t)}\nabla f(\mathbf{d}^{(t-1)}; \mathbf{b}^{(t)})\Big) \ ,$$

where parameter $\eta^{(t)}$ is the *step size*[4]. Given the definition of f in (2.7), $\nabla f(\mathbf{d}^{(t-1)}; \mathbf{b}^{(t)})$ is the partial derivative with respect to $\mathbf{d}$ at the optimal $\mathbf{x}$ [MBP10, Dan66], i.e. $\nabla f(\mathbf{d}; \mathbf{b}) = \frac{\partial l}{\partial \mathbf{d}}(\mathbf{d}, \mathbf{x}^*(\mathbf{d}, \mathbf{b}); \mathbf{b})$, where $\mathbf{x}^*$ is defined by (2.7).

Thus, to compute the gradient $\nabla f(\mathbf{d}^{(t-1)}; \mathbf{b}^{(t)})$, we should first compute the coefficient maps $\mathbf{x}^{(t)}$ of the t^{th} training signal $\mathbf{b}^{(t)}$ with dictionary $\mathbf{d}^{(t-1)}$, which is given by (2.10a). Then we can compute the gradient as

$$\nabla f\big(\mathbf{d}^{(t-1)}; \mathbf{b}^{(t)}\big) = \frac{\partial l}{\partial \mathbf{d}}\big(\mathbf{d}^{(t-1)}, \mathbf{x}^{(t)}; \mathbf{b}^{(t)}\big) = \big(\mathbf{X}^{(t)}\big)^T \Big(\mathbf{X}^{(t)}\mathbf{d}^{(t-1)} - \mathbf{b}^{(t)}\Big) \ .$$

Based on the discussion in Section 2.2.3, we can perform gradient descent either in the spatial-domain or the frequency-domain. In the frequency domain, the conjugate cogradient of $\nabla \hat{f}$ is:

$$\nabla \hat{f}(\hat{\mathbf{d}}^{(t-1)}; \hat{\mathbf{b}}^{(t)}) = \frac{\partial \hat{l}}{\partial \hat{\mathbf{d}}}\big(\hat{\mathbf{d}}^{(t-1)}, \hat{\mathbf{x}}^{(t)}; \hat{\mathbf{b}}^{(t)}\big) = \big(\hat{\mathbf{X}}^{(t)}\big)^H \Big(\hat{\mathbf{X}}^{(t)}\hat{\mathbf{d}}^{(t-1)} - \hat{\mathbf{b}}^{(t)}\Big) \ .$$

The full algorithm is summarized in Algorithm 1.

Complexity analysis of Algorithm 1. We list the single-step complexity and memory usage of different options in Table 2.1. Both the frequency-domain update and sparse matrix technique reduce single-step complexities. The comparison between these two computational techniques depends on the sparsity of $\mathbf{X}^{(t)}$ and the dictionary kernel size D. In Section 2.5.1, we will numerically compare these methods.

[4]Some authors refer to it as the *learning rate*.

Algorithm 1: Online Convolutional Dictionary Learning (Modified SGD)

```
Initialize: Initialize $\mathbf{d}^{(0)}$ with a random dictionary.
1 for $t = 1, \cdots, T$ do
2     Sample a signal $\mathbf{b}^{(t)}$.
3     Solve convolutional sparse coding problem (2.10a) to obtain $\mathbf{x}^{(t)}$.
4     if Option I then
5         Update dictionary in the spatial-domain with sparse matrix $\mathbf{X}^{(t)}$:
              $\mathbf{d}^{(t)} = \mathrm{Proj}_{\mathcal{C}}\Big(\mathbf{d}^{(t-1)} - \eta^{(t)}\big(\mathbf{X}^{(t)}\big)^T\big(\mathbf{X}^{(t)}\mathbf{d}^{(t-1)} - \mathbf{b}^{(t)}\big)\Big)$
6     else if Option II then
7         Update dictionary in the frequency-domain:
              $\hat{\mathbf{x}}^{(t)} = \mathrm{FFT}(\mathbf{x}^{(t)})$
              $\mathbf{d}^{(t)} = \mathrm{Proj}_{\mathcal{C}_{\mathrm{PN}}}\bigg(\mathrm{IFFT}\Big(\hat{\mathbf{d}}^{(t-1)} - \eta^{(t)}\big(\hat{\mathbf{X}}^{(t)}\big)^H\big(\hat{\mathbf{X}}^{(t)}\hat{\mathbf{d}}^{(t-1)} - \hat{\mathbf{b}}^{(t)}\big)\Big)\bigg)$
8     end
9 end
Output: $\mathbf{d}^{(T)}$
```

Convergence of Algorithm 1. Algorithm 1, by (2.20), is equivalent to the standard projected SGD. Thus, by properly choosing step sizes $\eta^{(t)}$, Algorithm 1 converges to a stationary point [GL13]. A diminishing step size rule $\eta^{(t)} = a/(b+t)$ is used in other dictionary learning works [AE08, MBP09]. The convergence performance with different step sizes are numerically tested in Section 2.5.1.

2.4 Second-order method: Algorithm 2

In this section, we first introduce some details of directly applying second order stochastic approximation method (2.10) to CDL problems, then we discuss some issues and our resolutions.

Aggregating the true loss function $f(\mathbf{d}; \mathbf{b}^{(t)})$ on the t^{th} sample $\mathbf{b}^{(t)}$, the objective function

Table 2.1: Single step complexity and memory usage of Algorithm 1. N: signal dimension; M: number of dictionary kernels; D: size of each kernel; ρ: average density of the coefficient maps.

Scheme	Single step complexity	Memory usage
Spatial (dense matrix)	$T_{\text{CBPDN}} + \mathcal{O}(NMD)$	$\mathcal{O}(NMD)$
Spatial (sparse matrix)	$T_{\text{CBPDN}} + \mathcal{O}(NMD\rho)$	$\mathcal{O}(NMD\rho)$
Frequency update	$T_{\text{CBPDN}} + \mathcal{O}(NM\log(N)) + \mathcal{O}(NM)$	$\mathcal{O}(MN)$

on the first t training samples is

$$F^{(t)}(\mathbf{d}) = \frac{1}{t}\Big(\sum_{\tau=1}^{t} f\big(\mathbf{d};\mathbf{b}^{(\tau)}\big)\Big) \approx F(\mathbf{d}) = \mathbb{E}_{\mathbf{b}}[f(\mathbf{d};\mathbf{b})] \,. \tag{2.21}$$

The central limit theorem tells us that $F^{(t)} \to F$ as $t \to \infty$. However, as discussed in Section 2.2.2, $F^{(t)}$ is not computationally tractable. To update $\mathbf{d}$ efficiently, we introduce the *surrogate function* $\mathcal{F}^{(t)}$ of $F^{(t)}$. Given $\mathbf{b}^{(t)}$, $\mathbf{x}^{(t)}$ is computed by CBPDN (2.7) using the latest dictionary $\mathbf{d}^{(t-1)}$, then a surrogate of $f(\mathbf{d};\mathbf{b}^{(t)})$ is given as

$$\mathbf{x}^{(t)} = \operatorname*{arg\,min}_{\mathbf{x}} \ell(\mathbf{d}^{(t-1)},\mathbf{x};\mathbf{b}^{(t)}), \qquad f^{(t)}(\mathbf{d}) \triangleq l\Big(\mathbf{d},\mathbf{x}^{(t)};\mathbf{b}^{(t)}\Big) \,, \tag{2.22}$$

The surrogate function of $F^{(t)}$ is defined as

$$\mathcal{F}^{(t)}(\mathbf{d}) = \frac{1}{t}\Big(f^{(1)}(\mathbf{d}) + \cdots + f^{(t)}(\mathbf{d})\Big) \,. \tag{2.23}$$

Then, at the t^{th} step, the dictionary is updated as

$$\mathbf{d}^{(t)} = \operatorname*{arg\,min}_{\mathbf{d}\in\mathbb{R}^{MD}} \mathcal{F}^{(t)}(\mathbf{d}) + \iota_{\mathcal{C}}(\mathbf{d}) \,. \tag{2.24}$$

Solving subproblem (2.24). To solve (2.24), we apply Fast Iterative Shrinkage-Thresholding (FISTA) [BT09], which needs to compute a gradient at each step. The gradient for the surrogate function can be computed as

$$\nabla\mathcal{F}^{(t)}(\mathbf{d}) = \frac{1}{t}\Big(\sum_{\tau=1}^{t} (\mathbf{X}^{(\tau)})^T\mathbf{X}^{(\tau)}\Big)\mathbf{d} - \frac{1}{t}\Big(\sum_{\tau=1}^{t} (\mathbf{X}^{(\tau)})^T\mathbf{b}^{(\tau)}\Big) \,.$$

We cannot follow this formula directly since the cost increases linearly in t. Instead we perform the recursive updates

$$\mathbf{H}^{(t)} = \mathbf{H}^{(t-1)} + (\mathbf{X}^{(t)})^T\mathbf{X}^{(t)} \ , \quad \mathbf{c}^{(t)} = \mathbf{c}^{(t-1)} + (\mathbf{X}^{(t)})^T\mathbf{b}^{(t)} \ , \tag{2.25}$$

where $(\mathbf{X}^{(t)})^T\mathbf{X}^{(t)}$ is the Hessian matrix of $f^{(t)}$. These updates, which have a constant cost per step, yield $\nabla\mathcal{F}^{(t)}(\mathbf{d}) = (\mathbf{H}^{(t)}\mathbf{d} - \mathbf{c}^{(t)})/t$. The matrix $\mathbf{H}^{(t)}/t$, the Hessian matrix of the surrogate function $\mathcal{F}^{(t)}$, accumulates the Hessian matrices of all the past loss functions. This is why we call this method the *second-order stochastic approximation* method.

There are some practical issues that prevents us from using the above algorithm directly in our problem.

- Inaccurate loss function: The surrogate function $\mathcal{F}^{(t)}$ involves old loss functions $f^{(1)}, f^{(2)}, \cdots$, which contain old information $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots$. For example, $\mathbf{x}^{(1)}$ is computed using $\mathbf{d}^{(0)}$ (cf. (2.22)).
- Large single step complexity and memory usage: handling a whole image $\mathbf{b}^{(t)}$ at each time is still a large-scale problem.
- FISTA is slow at solving subproblem (2.24): FISTA takes many steps to reach a sufficient accuracy.

To address these points, four modifications are given in this section[5].

2.4.1 Improvement I: forgetting factor

At time t, the dictionary is the result of an accumulation of past coefficient maps $\mathbf{x}_m^{(\tau)}$, $\tau < t$, which were computed with the then-available dictionaries. A way to balance accumulated past contributions and the information provided by the new training samples is to compute a weighted combination of these contributions [SE10, MBP10, SPL11, SG14]. This combination

[5]Improvements I and II have been addressed in our previous work [LGW17]. In the present article, we include their theoretical analysis and introduce the new enhancement of the stopping criterion (Improvement III).

gives more weight to more recent updates since those are the result of a more extensively trained dictionary. Specifically, we consider the following weighted (or modified) surrogate function:

$$\mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d}) = \frac{1}{\Lambda^{(t)}} \sum_{\tau=1}^{t} (\tau/t)^p f^{(\tau)}(\mathbf{d}) \ , \quad \Lambda^{(t)} = \sum_{\tau=1}^{t} (\tau/t)^p \ . \tag{2.26}$$

This function can be written in recursive form as

$$\Lambda^{(t)} = \alpha^{(t)} \Lambda^{(t-1)} + 1 \ , \tag{2.27}$$

$$\Lambda^{(t)} \mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d}) = \alpha^{(t)} \Lambda^{(t-1)} \mathcal{F}^{(t-1)}_{\mathrm{mod}}(\mathbf{d}) + f^{(t)}(\mathbf{d}) \ . \tag{2.28}$$

Here $\alpha^{(t)} \in (0,1)$ is a *forgetting factor*, which has its own time evolution:

$$\alpha^{(t)} = (1 - 1/t)^p \tag{2.29}$$

regulated by the *forgetting exponent* $p > 0$. As t increases, the factor $\alpha^{(t)}$ increases ($\alpha^{(t)} \to 1$ as $t \to \infty$), reflecting the increasing accuracy of the past information as the training progresses. The dictionary update (2.24) is modified correspondingly to

$$\mathbf{d}^{(t)} = \operatorname*{arg\,min}_{\mathbf{d} \in \mathbb{R}^{MD}} \mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d}) + \iota_{\mathcal{C}}(\mathbf{d}) \ . \tag{2.30}$$

This technique has been used in some previous dictionary learning works, as we mentioned before, but was not theoretically analyzed. In this paper, we prove in Propositions 1 and 2 that $F^{(t)}_{\mathrm{mod}} \to F$ as $t \to \infty$, where $F^{(t)}_{\mathrm{mod}}$ is a weighted approximation of F:

$$F^{(t)}_{\mathrm{mod}}(\mathbf{d}) = \frac{1}{\Lambda^{(t)}} \Big(\sum_{\tau=1}^{t} (\tau/t)^p f(\mathbf{d}; \mathbf{b}^{(\tau)}) \Big) \ . \tag{2.31}$$

Moreover, in Theorem 1, $\mathcal{F}^{(t)}_{\mathrm{mod}}$, the surrogate of $F^{(t)}_{\mathrm{mod}}$, is also proved to be convergent on the current dictionary, i.e. $\mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d}^{(t)}) - F^{(t)}_{\mathrm{mod}}(\mathbf{d}^{(t)}) \to 0$.

Effect of the forgetting exponent p. A small p tends to lead to a stable algorithm since all the training signals are given nearly equal weights and $F^{(t)}_{\mathrm{mod}}$ is a stochastic approximation of F with small variance. Propositions 1 and 2 give theoretical explanations of this phenomenon. However, a small p leads to an inaccurate surrogate loss function $\mathcal{F}^{(t)}_{\mathrm{mod}}$ since it gives large weights to old information. In the extreme case, as $p \to 0$, the modified surrogate function (2.26) reduces to the standard one (2.23). Section 2.5.2.1 reports the related numerical results.

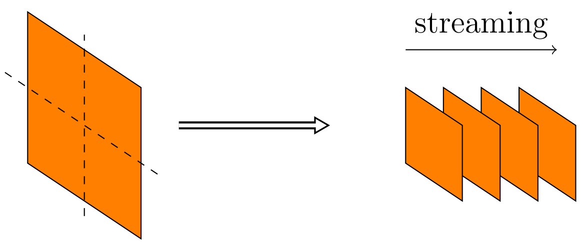


Figure 2.1: An example of image splitting: $N = 256 \times 256 \rightarrow \tilde{N} = 128 \times 128$.

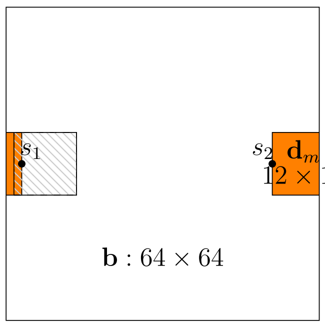
2.4.2 Improvement II: image-splitting

Both the single-step complexity and memory usage are related to the signal dimension N. For a typical imaging problem, $N = 256 \times 256$ or greater, which is large. To reduce the complexities, we use small regions [6] instead of the whole signal. Specifically, as illustrated in Fig. 2.1, we split a signal $\mathbf{b}^{(t)} \in N$ into small regions $\mathbf{b}^{(t)}_{\text{split},1}, \mathbf{b}^{(t)}_{\text{split},2}, \ldots \in \tilde{N}$, with $\tilde{N} < N$, and treat them as if they were distinct signals. In this way, the training signal sequence becomes

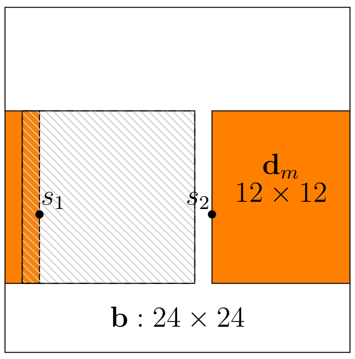
$$\{\mathbf{b}_{\text{split}}\} \triangleq \{\mathbf{b}^{(1)}_{\text{split},1}, \cdots, \mathbf{b}^{(1)}_{\text{split},n}, \mathbf{b}^{(2)}_{\text{split},1}, \cdots, \mathbf{b}^{(2)}_{\text{split},n}, \cdots\} \; .$$

Boundary issues. The use of *circular boundary conditions* for signals that are not periodic has the potential to introduce boundary artifacts in the representation, and therefore also in the learned dictionary [ZKT10]. When the size of the training images is much larger than the kernels, there is some evidence that the effect on the learned dictionary is negligible [BEL13], but it is reasonable to expect that these effects will become more pronounced for smaller training images, such as the regions we obtain when using a small splitting size $\tilde{N}$. The possibility of severe artifacts when the image size approaches the kernel size is illustrated in Fig. 2.2. In Sec. 2.5.2.2, we study this effect and show that using a splitting size that is twice the kernel size in each dimension is sufficient to avoid artifacts, as expected from the

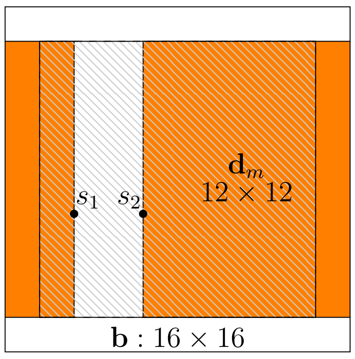
[6] In our previous work [LGW17], we sample some small regions from the whole signals in the limited memory algorithm, which performs worse than the algorithm training with the whole signals. We claimed that the performance sacrifice is caused by the circular boundary condition. In fact, this is caused by the sampling. In that paper, we sample small regions with random center position and fixed size. If we sample small regions in this way, some parts of the image are not sampled, but some are sampled several times. Consequently, in the present paper, we propose the "image-splitting" technique in Algorithm 2, which avoids this issue. It only shows worse performance when the splitting size is smaller than a threshold, which is actually caused by the boundary condition.



(a) When the signal size 64×64 is much larger than the kernel size 12×12, pixels s_1, s_2 in the same filter are far from each other. Thus, they do not interact with each other.



(b) When the signal size 24×24 is twice the kernel size 12×12, s_1, s_2 still do not interact. It is the smallest signal size to avoid boundary artifacts.



(c) When the signal size 16×16 is less than twice the kernel size 12×12, s_1, s_2 interact with one another. This leads to artifacts in practice.

Figure 2.2: An illustration of the boundary artifacts.

argument illustrated in Fig. 2.2.

2.4.3 Improvement III: stopping FISTA early

Another issue in surrogate function method is the stopping condition of FISTA. A small fixed tolerance will result in too many inner-loop iterations for the initial steps. Another strategy, as used in SPAMS [MBP09, JMB10] is a fixed number of inner-loop iterations, but it does not have any theoretical convergence guarantee.

In this article, we propose a "diminishing tolerance" scheme in which subproblem (2.30) is solved *inexactly*, but the online learning algorithm is still theoretically guaranteed to converge. The stopping accuracy is increasing as t increases. Specifically, the stopping tolerance is decreased as t increases. Moreover, with a warm start (using $\mathbf{d}^{(t-1)}$ as the initial solution for the t^{th} step), the number of inner-loop iterations stays moderate as t increases, which is validated by the results in Fig. 2.8.

Stopping metric. We use the Fixed Point Residual (FPR) [DY16]

$$\mathbf{R}^{(t)}(\mathbf{g}) \triangleq \left\| \mathbf{g} - \mathrm{Proj}_{\mathcal{C}}\big(\mathbf{g} - \eta \nabla \mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{g})\big) \right\| . \tag{2.32}$$

for two reasons. One is its simplicity; if FISTA is used to solve (2.30), this metric can be computed directly as $\mathbf{R}^{(t)}(\mathbf{g}^j_{\mathrm{aux}}) = \|\mathbf{g}^{j+1} - \mathbf{g}^j_{\mathrm{aux}}\|$. The other is that a small FPR implies a small distance to the exact solution of the subproblem, as shown in Proposition 3 below.

Stopping condition. In this paper, we consider the following stopping condition:

$$\mathbf{R}^{(t)}(\mathbf{g}^j_{\mathrm{aux}}) \leq \tau^{(t)} \triangleq \tau_0/(1+\alpha t) , \tag{2.33}$$

where the tolerance $\tau^{(t)}$ is large during the first several steps and reduces to zeros at the rate of $\mathcal{O}(1/t)$ as t increases. In the t^{th} step, once (2.33) is satisfied, we stop the $\mathbf{D}$-update (FISTA) and continue to the next step. The effect of this stopping condition is theoretically analyzed in Propositions 3 and 4, and numerically demonstrated in Sec. 2.5.2.3 below.

2.4.4 Improvement IV: updating gradients

Based on the discussion in Section 2.2.3, we have two options to solve subproblem (2.30). One is to solve in the spatial domain utilizing sparsity. The gradient of $\mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d})$ is

$$\nabla \mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d}) = \frac{1}{\Lambda^{(t)}} \sum_{\tau=1}^{t} (\tau/t)^p \Big((\mathbf{X}^{(t)})^T \mathbf{X}^{(t)} \mathbf{d} - (\mathbf{X}^{(\tau)})^T \mathbf{b}^{(\tau)} \Big) = \frac{1}{\Lambda^{(t)}} \Big(\mathbf{H}^{(t)}_{\mathrm{mod}} \mathbf{d} - \mathbf{c}^{(t)}_{\mathrm{mod}} \Big) ,$$

where $\mathbf{H}^{(t)}_{\mathrm{mod}}$ and $\mathbf{c}^{(t)}_{\mathrm{mod}}$ are calculated in a recursive form in the line 5 of Algorithm 2. The other option is to update in the frequency domain. The conjugate cogradient of $\hat{\mathcal{F}}^{(t)}_{\mathrm{mod}}(\hat{\mathbf{d}})$ is

$$\nabla \hat{\mathcal{F}}^{(t)}_{\mathrm{mod}}(\hat{\mathbf{d}}) = \frac{1}{\Lambda^{(t)}} \sum_{\tau=1}^{t} (\tau/t)^p \Big((\hat{\mathbf{X}}^{(t)})^T \hat{\mathbf{X}}^{(t)} \hat{\mathbf{d}} - (\hat{\mathbf{X}}^{(\tau)})^T \hat{\mathbf{b}}^{(\tau)} \Big) = \frac{1}{\Lambda^{(t)}} \Big(\hat{\mathbf{H}}^{(t)}_{\mathrm{mod}} \hat{\mathbf{d}} - \hat{\mathbf{c}}^{(t)}_{\mathrm{mod}} \Big) ,$$

where $\hat{\mathbf{H}}^{(t)}_{\mathrm{mod}}$ and $\hat{\mathbf{c}}^{(t)}_{\mathrm{mod}}$ are calculated in a recursive form in the line 7 of Algorithm 2. With the gradients, we can apply FISTA or frequency-domain FISTA on the problem (2.30), as in Algorithm 2.

Complexity analysis of Algorithm 2. If we solve (2.30) directly, the operator $\mathbf{X}^{(t)}$ is a linear operator from $\mathbb{R}^{DM}$ to $\mathbb{R}^N$. Thus, the complexity of computing the Hessian matrix of $f^{(t)}$, $(\mathbf{X}^{(t)})^T\mathbf{X}^{(t)}$, is $\mathcal{O}(D^2M^2N)$ and the memory cost is $\mathcal{O}(D^2M^2)$. Otherwise, if we solve (2.30) utilizing the sparsity of $\mathbf{X}$, the computational cost of computing $(\mathbf{X}^{(t)})^T\mathbf{X}^{(t)}$ can be reduced to $\mathcal{O}(D^2M^2N\rho)$, where ρ is the density of sparse matrix $\mathbf{X}^{(t)}$, but the memory cost is still $\mathcal{O}(D^2M^2)$ because $(\mathbf{X}^{(t)})^T\mathbf{X}^{(t)}$ is not sparse although $\mathbf{X}^{(t)}$ is. In comparison, if we solve (2.30) in the frequency domain, the frequency-domain operator $\hat{\mathbf{X}}^{(t)} = (\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2, \cdots, \hat{\mathbf{X}}_M)$ is a linear operator from $\mathbb{C}^{MN}$ to $\mathbb{C}^N$, which seems to lead to a larger complexity to compute the Hessian: $\mathcal{O}(M^2N^3)$ flops and $\mathcal{O}(M^2N^2)$ memory cost. However, since each component $\hat{\mathbf{X}}_m$ is diagonal, the frequency-domain product $(\hat{\mathbf{X}}^{(t)})^H\hat{\mathbf{X}}^{(t)}$ has only $\mathcal{O}(M^2N)$ non-zero values. Both the number of flops and memory cost are $\mathcal{O}(M^2N)$. The complexities are listed in Table 2.2.

2.4.5 Convergence of Algorithm 2

First, we start with some assumptions[7]:

Assumption 1. *All the signals are drawn from a distribution with a compact support.*

Assumption 2. *Each sparse coding step (2.7) has a unique solution.*

Assumption 3. *The surrogate functions are strongly convex.*

Assumption 1 can easily be guaranteed by normalizing each training signal. Assumption 2 is a common assumption in dictionary learning and other linear regression papers [MBP09, EHJ04]. Practically, it must be guaranteed by choosing a sufficiently large penalty parameter λ in (2.7), because a larger penalty parameter leads to a sparser $\mathbf{x}$. See Appendix 2.D for details. Assumption 3 is a common assumption in RLS (see Definition (3.1) in [JJB82]) and dictionary learning (see Assumption B in [MBP10]).

Proposition 1 (Weighted central limit theorem)**.** *Suppose $Z_i \overset{i.i.d}{\sim} P_Z(z)$, with a compact support, expectation μ, and variance σ^2. Define the weighted approximation of Z: $\hat{Z}^n_{\text{mod}} \triangleq \frac{1}{\sum_{i=1}^n (i/n)^p} \sum_{i=1}^n (i/n)^p Z_i$. Then, we have*

$$\sqrt{n}(\hat{Z}^n_{\text{mod}} - \mu) \xrightarrow{d} N\Big(0, \frac{p+1}{\sqrt{2p+1}}\sigma\Big) \,. \tag{2.34}$$

[7]The specific formulas for Assumptions 2 and 3 are shown in Appendix 2.D.

$$\mathbb{E}\left[\sqrt{n}\left|\hat{Z}^n_{\text{mod}} - \mu\right|\right] = \mathcal{O}(1) \; . \tag{2.35}$$

This proposition is an extension of the central limit theorem (CLT). As $p \to 0$, it reduces to the standard CLT. The proof is given in Appendix 2.E.3.

Proposition 2 (Convergence of functions)**.** *With Assumptions 1-3, we have*

$$\mathbb{E}\left[\sqrt{t}\left\|F - F^{(t)}\right\|_\infty\right] \leq M \; , \tag{2.36}$$

$$\mathbb{E}\left[\sqrt{t}\left\|F - F^{(t)}_{\text{mod}}\right\|_\infty\right] \leq \frac{p+1}{\sqrt{2p+1}} M \; , \tag{2.37}$$

where $M > 0$ is some constant unrelated with t, and $\|f\|_\infty = \sup_{\mathbf{d}\in\mathcal{C}} \|f(\mathbf{d})\|$.

This proposition is an extension of Donsker's theorem (see Lemma 7 in [MBP10] and Chapter 19 in [Vaa00]). The proof is given in Appendix 2.E.4.

Moreover, it shows that weighted approximation $F^{(t)}_{\text{mod}}$ and standard approximation $F^{(t)}$ have the same asymptotic convergence rate $\mathcal{O}(1/\sqrt{t})$. However, the error bound factor $(p+1)/\sqrt{2p+1}$ is a monotone increasing function in $p \geq 0$. Thus, a larger p leads to a larger variance and slower convergence of $F^{(t)}_{\text{mod}}$. This explains why we cannot choose p to be too large.

Proposition 3 (Convergence of FPR implies convergence of iterates)**.** *Let $(\mathbf{d}^*)^{(t)}$ be the exact minimizer of the t^{th} subproblem:*

$$(\mathbf{d}^*)^{(t)} = \arg\min_{\mathbf{d}} \mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}) + \iota_{\mathcal{C}}(\mathbf{d}) \; . \tag{2.38}$$

Let $\mathbf{d}^{(t)}$ be the solution obtained by the frequency-domain FISTA (Algorithm 3) with our proposed stopping condition (2.33). Then, we have

$$\left\|\mathbf{d}^{(t)} - (\mathbf{d}^*)^{(t)}\right\| \leq \mathcal{O}\left(t^{-1}\right) \; . \tag{2.39}$$

The proof is given in Appendix 2.E.1.

Proposition 4 (The convergence rate of Algorithm 2)**.** *Let $\mathbf{d}^{(t)}$ be the sequence generated by Algorithm 2. Then, we have*

$$\left\|\mathbf{d}^{(t+1)} - \mathbf{d}^{(t)}\right\| = \mathcal{O}\left(t^{-1}\right) . \tag{2.40}$$

Compared with Lemma 1 in [MBP10], which shows the convergence rate of the surrogate function method with exact $\mathbf{D}$-update, our Proposition 4 shows that the inexact $\mathbf{D}$-update (2.33) shares the same rate. Since our inexact version stops FISTA earlier, it is faster. The proof of this proposition is given in Appendix 2.E.2.

Theorem 1 (Almost sure convergence of Algorithm 2)**.** *Let $\mathcal{F}_{\text{mod}}^{(t)}$ be the surrogate function sequence, $\mathbf{d}^{(t)}$ the iterate sequence, both generated by Algorithm 2. Then we have, with probability 1:*

1. *$\mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})$ converges.*
2. *$\mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)}) - F(\mathbf{d}^{(t)}) \to 0$.*
3. *$F(\mathbf{d}^{(t)})$ converges.*
4. *$\text{dist}(\mathbf{d}^{(t)}, V) \to 0$, where V is the set of stationary points of the CDL problem (2.8).*

The proof is given in Appendix 2.E.5.

2.5 Numerical results

All the experiments are computed using MATLAB R2016a running on a workstation with 2 Intel Xeon(R) X5650 CPUs clocked at 2.67GHz. Implementations of these algorithms are available in the Matlab version of the SPORCO software library [Woh16f], and will be included in a future release of the Python version of this library. The dictionary size is $12 \times 12 \times 64$, and the signal size is 256×256. Dictionaries are evaluated by comparing the functional values obtained by computing CBPDN (2.7) on the test set. A smaller functional value indicates a better dictionary. Similar methods to evaluate the dictionary are also used in other dictionary learning works [MBP10, TYG12]. The regularization parameter is chosen as $\lambda = 0.1$.

The training set consists of 40 images selected from the MIRFLICKR-1M dataset[8] [HTL10], and the test set consists of 20 different images from the same source. All of the images used

[8]The actual image data contained in this dataset is of very low resolution since the dataset is primarily

were originally of size 512×512. To accelerate the experiments, we crop the borders of both the training images and testing images and preserve the central part to yield 256×256. The training and testing images are pre-processed by dividing by 255 to rescale the pixel values to the range $[0, 1]$ and highpass filtering[9].

In this work we solve the convolutional sparse coding step using an ADMM algorithm [Woh14] with an adaptive penalty parameter scheme [Woh17a]. The stopping condition is that both primal and dual normalized residuals [Woh17a] be less than 10^{-3}, and the relaxation parameter is set to 1.8 [Woh16d].

2.5.1 Validation of Algorithm 1

First we test the effect of step size $\eta^{(t)}$ in Algorithm 1. We can choose either a fixed step size or a diminishing step size:

$$\eta^{(t)} = \eta_0 \quad \text{or} \quad \eta^{(t)} = a/(t+b).$$

The results of experiments to determine the best choice of η are reported in Fig. 2.3. We test the convergence performance of fixed step size scheme with values: $\eta_0 \in \{1,\ 0.3,\ 0.1,\ 0.03,\ 0.01\}$. We also test the convergence performance of the diminishing step size scheme with values: $a \in \{5, 10, 20\}; b \in \{5, 10, 20\}$ and report the best $(a = 10, b = 5)$ in Fig. 2.3. When a large fixed step size is used, the functional value decreases fast initially but becomes unstable later on. A smaller step size causes the opposite. A diminishing step size balances accuracy and convergence rate.

Second, we test the computational techniques (computing with sparsity / computing in the frequency domain), as Table 2.3 shows. To get the table, we set $\lambda = 0.1$, and the average density of $\mathbf{X}$ is 0.0037. Both techniques reduce the complexity of updating $\mathbf{d}^{(t)}$. Option I has

targeted at image classification tasks. The original images from which those used here were derived were obtained by downloading the original images from Flickr that were used to derive the MIRFLICKR-1M images.

[9]The pre-processing is applied due to the inability of the standard CSC model to effectively represent low-frequency/large-scale image components [Woh16c, Sec. 3]. In this case the highpass component is computed as the difference between the input signal and a lowpass component computed by Tikhonov regularization with a gradient term [Woh17b, pg. 3], with regularization parameter 5.0.

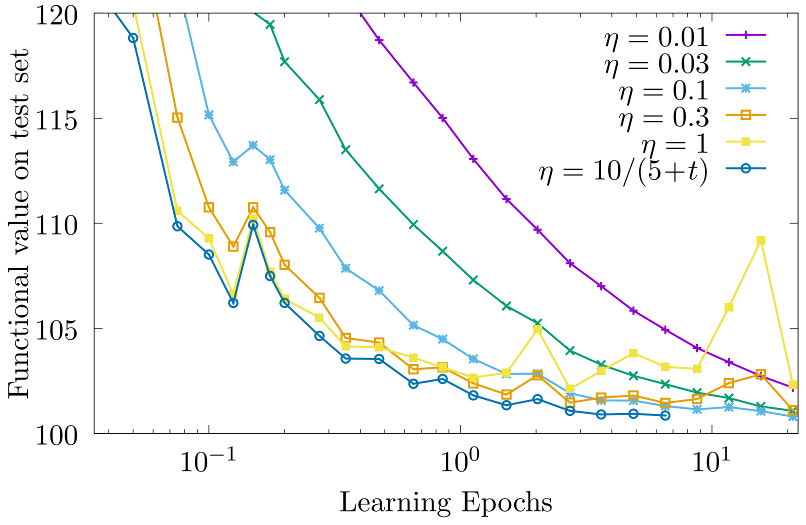


Figure 2.3: Tuning the step size of Algorithm 1.

better memory cost while Option II has better calculation time. Fig. 2.4 shows the objective values versus training time. Frequency-domain update (Option II) performs the best.

2.5.2 Validation of Algorithm 2

For Algorithm 2, we test the four techniques separately: the forgetting exponent p, image splitting with size $\tilde{N}$, and stopping tolerance of FISTA $\tau^{(t)}$, and computational techniques (sparsity or frequency-domain update).

2.5.2.1 Validation of Improvement I: forgetting exponent p

In this section, we fix $\tilde{N} = 256 \times 256$ (no splitting) and $\tau^{(t)} = 10^{-4}$, which is small enough to give an accurate solution. Fig. 2.5 shows that, when $p = 0$, the curve is monotonic and with small oscillation, but it converges to a higher functional value. When p is larger, the algorithm converges to lower functional values. When p is too large, for instance, $p \in \{40, 80\}$, the curve oscillates severely, which indicates large variance. These results are consistent with Propositions 1 and 2. In the remaining sections we fix $p = 10$ since it is shown to be a good choice.

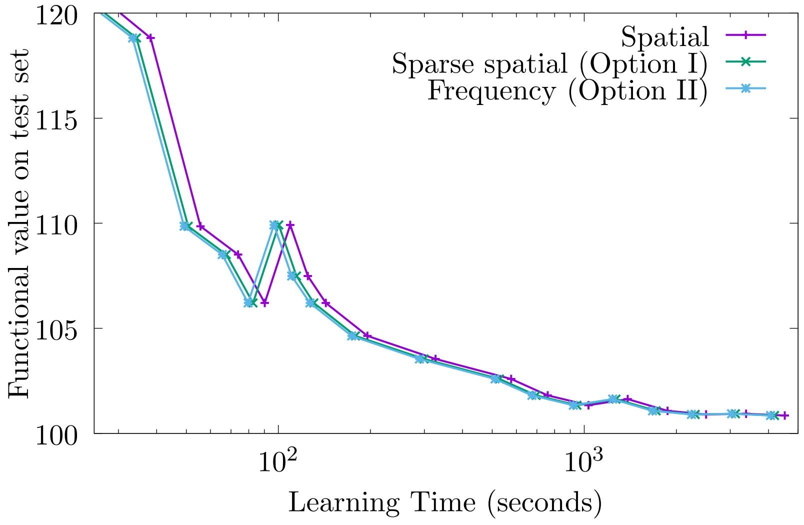


Figure 2.4: Different options of Algorithm 1.

2.5.2.2 Validation of Improvement II: image splitting with size $\tilde{N}$ and boundary artifacts

In this section, we again fix $\tau^{(t)} = 10^{-4}$. Convergence comparisons are shown in Fig. 2.6, and the dictionaries obtained with different $\tilde{N}$ are displayed Fig. 2.7. In our experiments, we only consider square signals ($\tilde{N} = 12\times12, 16\times16, 32\times32, 64\times64, 256\times256$) and square dictionary kernels ($D = 12 \times 12$). When $\tilde{N} \geq 2^2D$, say $\tilde{N} = 32 \times 32$ or $\tilde{N} = 64 \times 64$, the algorithm converges to a good functional value, which is the same as that without image-splitting. However, when $\tilde{N}$ is smaller than the threshold 2^2D, say $\tilde{N} = 16 \times 16$ or 12×12, the algorithm converges to a higher functional value, which implies worse dictionaries. Thus, we can conclude that *the splitting size should be at least twice the dictionary kernel size in each dimension. Otherwise, it will lead to boundary artifacts.* This phenomenon is consistent with the discussion in Section 2.4.2. The artifacts are specifically displayed in Fig. 2.7. When $\tilde{N}$ is smaller than the threshold, say 12×12, the features learned are incomplete.

This section only studies the effect, due to boundary artifacts, of image-splitting on objective functional values. As Table 2.2 shows, it also helps reducing computing time and memory cost, which is numerically validated in Section 2.5.2.4.

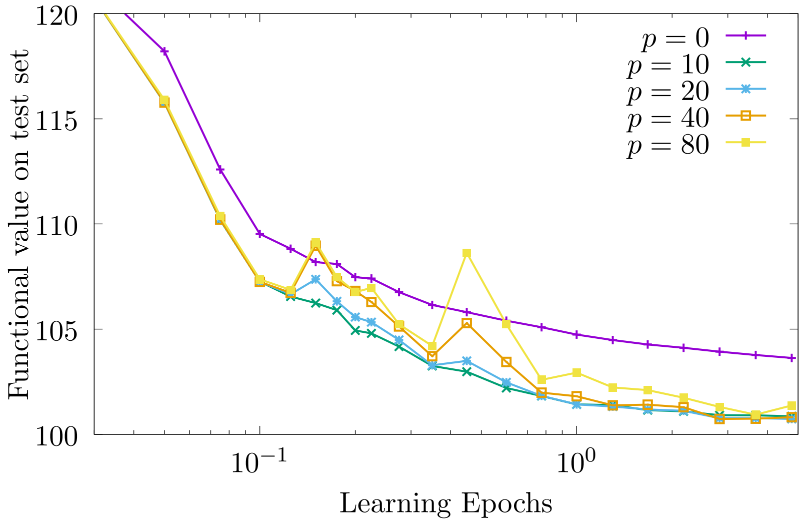


Figure 2.5: Effect of forgetting exponent p in Algorithm 2.

2.5.2.3 Validation of Improvement III: stopping tolerance of FISTA $\tau^{(t)}$

In this section, we fix $p = 10, \tilde{N} = 256 \times 256$ (no splitting). Fig. 2.8 shows the effect of using different $\tau^{(t)}$. Using a small stopping tolerance $\tau^{(t)} = 10^{-4}$ leads to a good functional value 101.1 but large number of FISTA iterations, while a large tolerance 10^{-2} leads to a large functional value 104.4 and small number of FISTA iterations. Consider our proposed diminishing tolerance rule (2.33) $\tau^{(t)} = 0.01/t$. When the algorithm starts, $t = 1$, we have $\tau^{(1)} = 10^{-2}$. At the end of the algorithm, $t = 100$, $\tau^{(100)} = 10^{-4}$. Based on the results in Fig. 2.8, our diminishing tolerance avoids large number of FISTA loops, especially at the initial steps, while losing little accuracy, as the final objective, 101.3 is close to 101.1.

2.5.2.4 Validation of Improvement IV: computational techniques

In this section, we fix $p = 10, \tau^{(t)} = 0.01/t$ and $\lambda = 0.1$, and compare the calculation time and memory usage of spatial-domain update and frequency-domain update. Table 2.4 illustrates that image-splitting helps reduce the single-step complexity and memory usage for both Option I (spatial-domain update) and Option II (frequency-domain update). For option II, the advantage of smaller splitting size $\tilde{N}$ is more significant than that of option I. When $\tilde{N} = 256 \times 256$, option I is much better than option II; but when $\tilde{N} = 64 \times 64$, the single step

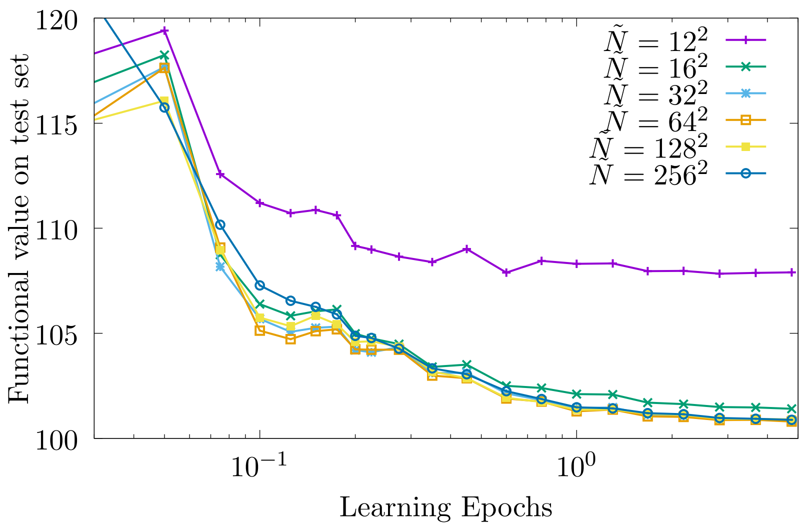


Figure 2.6: Effect of the Technique II (image-splitting with size $\tilde{N}$) in Algorithm 2.

time of option II is comparable with that of option I. The reason for this is that, for option I, reducing $\tilde{N}$ only helps reduce the single-step time cost of CBPDN, updating Hessian matrix $\mathbf{H}^{(t)}$ and the loops of FISTA, but does not help reduce the time cost of single-step time cost in FISTA. However, for option II, image-splitting not only reduces those three complexities, but also reduces the single-step complexity of FISTA. Furthermore, option II uses much less memory than option I when $\tilde{N} = 64 \times 64$.

Fig. 2.9(a) and Fig. 2.9(b) compare the objective functional value versus time. Fig. 2.9(a) indicates that reducing $\tilde{N}$ does *not* help a lot for Option I. Table 2.4 shows that smaller $\tilde{N}$ reduces the single step complexity, but it also reduces the gain in each step because a smaller splitting size leads to less information used for training. This is a trade-off. By Fig. 2.9(a), $\tilde{N} = 128 \times 128$ is a good choice.

Option II, in contrast, benefits more from smaller $\tilde{N}$, as can be seen from Fig. 2.9(b) and Table 2.4. Although splitting a training image reduces the gain in each step, the benefit overwhelms the loss. Thus, for Option II, the smaller the splitting size the better, as long as $\tilde{N}$ is larger than the threshold for boundary artifacts.

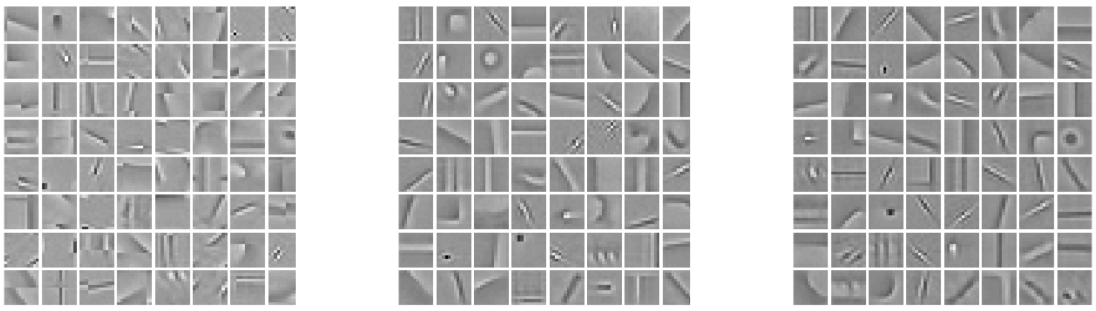

(a) Dictionaries learned by $\tilde{N} = 12 \times 12$: some incomplete features. (b) Dictionaries learned by $\tilde{N} = 64 \times 64$. (c) Dictionaries learned by $\tilde{N} = 256 \times 256$ (no splitting).

Figure 2.7: Visualization of boundary artifacts.

2.5.3 Main result I: convergence speed

In this section, we study the convergence speeds of all the methods on the clean data set, without a masking operator. We compare our methods with two leading batch learning algorithms: the method of Papyan et al. [PRS17], which uses K-SVD and updates the dictionary in the spatial domain, and an algorithm [GW17] that uses the ADMM consensus dictionary update [vv16], which is computed in the frequency domain. For batch learning algorithms, we test on subsets of 10, 20, and 40 images selected from the training set. For online learning algorithms, since they are scalable in the size of the training set, we just test our methods on the whole training set of 40 images. All the parameters are tuned as follows. For batch learning algorithm (Papyan et al.), we use the software they released, and for batch learning algorithm (ADMM consensus update), we use the "adaptive penalty parameter" scheme in [Woh17a]. For modified SGD (Algorithm 1), we use the step size of $10/(5+t)$. For Surrogate-Splitting (Algorithm 2), we use $p = 10, \tau^{(t)} = 0.01/t, \tilde{N} = 128 \times 128$ for spatial-domain update, $\tilde{N} = 64 \times 64$ for frequency-domain update, as we tuned in the previous sections. For our algorithm proposed in [LGW17], we use $p = 10, \tau^{(t)} = 10^{-3}, \tilde{N} = 64 \times 64$.

The performance comparison of batch and online methods is presented in Fig. 2.10. The advantage of online learning is significant (note that the time axis is logarithmically scaled).

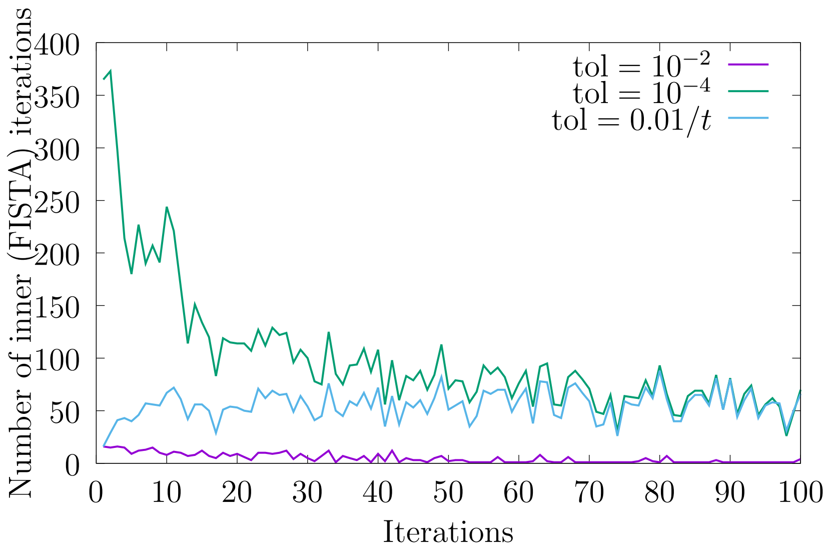


Figure 2.8: Effect of Technique III (stopping FISTA early) in Algorithm 2.

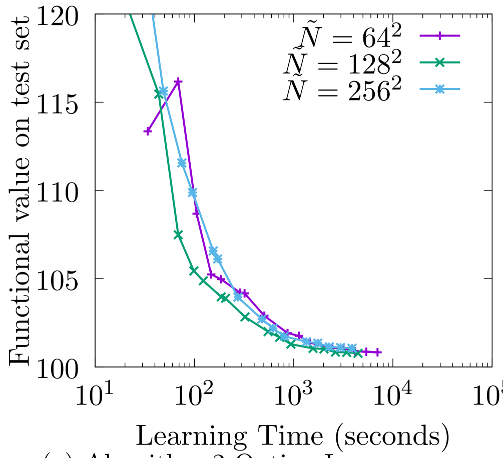
To obtain the same functional value 101 on the test set, batch learning takes 15 hours, our previous method [LGW17] takes around 1.5 hours, Algorithm 2 with option II takes around 1 hour, and Algorithm 1 and Algorithm 2 with option I takes less than 1 hour. We can conclude that, both modified SGD (Algorithm 1) and Surrogate-Splitting (Algorithm 2) converge faster than the batch learning algorithms and our previous online algorithm.

2.5.4 Main result II: memory usage

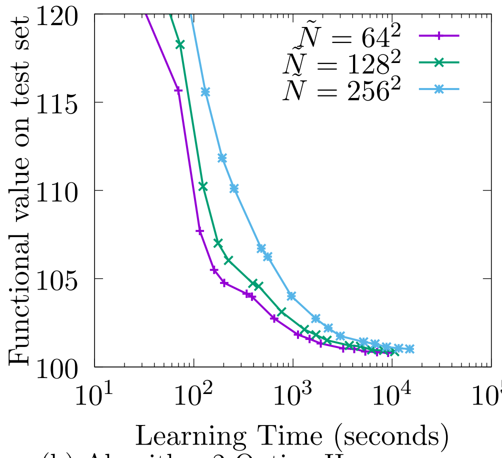
As Table 2.5 shows, both Algorithm 1 and 2 save a large amount of memory.

2.6 Conclusions

We have proposed two efficient online convolutional dictionary learning methods. Both of them have a theoretical convergence guarantee and show good performance on both time and memory usage. Compared to recent online CDL works [DKB17, WYK18], our second-order method improves the framework by several practical techniques. Our first-order method, to the best of our knowledge, is the first attempt to use first order methods in online CDL. It shows better performance in time and memory usage, and requires fewer parameters to tune.



(a) Algorithm 2 Option I.



(b) Algorithm 2 Option II.

Figure 2.9: Effect of splitting region size $\tilde{N}$ on different options.

Although only single-channel images are considered in this article, our online methods can easily be extended to the multi-channel case [Woh16b].

Appendix 2.A Derivation of (2.18)

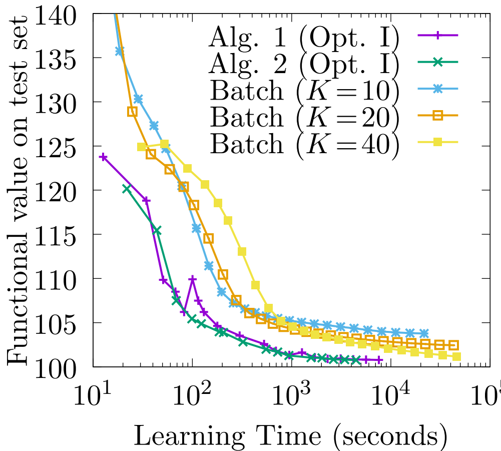
Consider a real-valued function defined on the complex domain $f : \mathbb{C}^n \to \mathbb{R}$, which can be viewed as a function defined on the $2n$ dimensional real domain: $f(\mathbf{x}) = f\big(\Re(\mathbf{x}) + i\Im(\mathbf{x})\big)$, where $\Re(\mathbf{x}), \Im(\mathbf{x}) \in \mathbb{R}^n$ are the real part and imaginary part, respectively. By [SBL12], "conjugate cogradient" is defined as

$$\nabla f(\mathbf{x}) \triangleq \frac{\partial f}{\partial \Re(\mathbf{x})} + i \frac{\partial f}{\partial \Im(\mathbf{x})} \; . \tag{2.41}$$

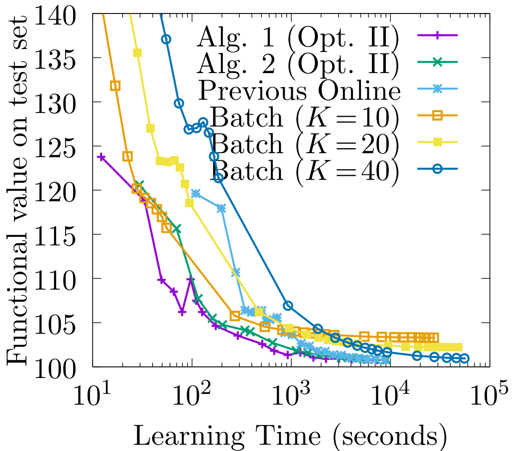
Based on (2.41), we give a derivation of (2.18).

Recall the definition $\hat{l}(\hat{\mathbf{d}}, \hat{\mathbf{x}}; \hat{\mathbf{b}}) = 1/2\big\|\hat{\mathbf{X}}\hat{\mathbf{d}} - \hat{\mathbf{b}}\big\|^2$. Substituting $\hat{\mathbf{X}} = \Re(\hat{\mathbf{X}}) + i\Im(\hat{\mathbf{X}})$, $\hat{\mathbf{d}} = \Re(\hat{\mathbf{d}}) + i\Im(\hat{\mathbf{d}})$, and $\hat{\mathbf{b}} = \Re(\hat{\mathbf{b}}) + i\Im(\hat{\mathbf{b}})$ into $\hat{l}$, we have

$$\begin{aligned}
&\hat{l}(\hat{\mathbf{d}}, \hat{\mathbf{x}}; \hat{\mathbf{b}}) \\
=&\frac{1}{2}\Big\|\Re(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Re(\hat{\mathbf{b}}) + i\big(\Im(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) + \Re(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{b}})\big)\Big\|^2 \\
=&\frac{1}{2}\Big\|\Re(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Re(\hat{\mathbf{b}})\Big\|^2 + \frac{1}{2}\Big\|\Im(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) + \Re(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{b}})\Big\|^2 \; .
\end{aligned}$$



(a) Methods with spatial-domain dictionary update scheme.



(b) Methods with frequency-domain dictionary update scheme.

Figure 2.10: Main Result I: convergence speed comparison.

The partial derivatives on $\Re(\hat{\mathbf{d}})$ and $\Im(\hat{\mathbf{d}})$ are, respectively,

$$\begin{aligned}\frac{\partial\hat{l}}{\partial\Re(\hat{\mathbf{d}})} =&\Re(\hat{\mathbf{X}})^T\big(\Re(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Re(\hat{\mathbf{b}})\big)\\ &+ \Im(\hat{\mathbf{X}})^T\big(\Im(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) + \Re(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{b}})\big)\\ \frac{\partial\hat{l}}{\partial\Im(\hat{\mathbf{d}})} =&\Im(\hat{\mathbf{X}})^T\big(- \Re(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) + \Im(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) + \Re(\hat{\mathbf{b}})\big)\\ &+ \Re(\hat{\mathbf{X}})^T\big(\Im(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) + \Re(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{b}})\big)\ .\end{aligned}$$

Therefore,

$$\begin{aligned}&\hat{\mathbf{X}}^H(\hat{\mathbf{X}}\hat{\mathbf{d}} - \hat{\mathbf{b}})\\ =&(\Re(\hat{\mathbf{X}}) - i\Im(\hat{\mathbf{X}}))^T\Big((\Re(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Re(\hat{\mathbf{b}}))\\ &+ i\big(\Im(\hat{\mathbf{X}})\Re(\hat{\mathbf{d}}) + \Re(\hat{\mathbf{X}})\Im(\hat{\mathbf{d}}) - \Im(\hat{\mathbf{b}})\big)\Big)\\ =&\frac{\partial\hat{l}}{\partial\Re(\hat{\mathbf{d}})} + i\frac{\partial\hat{l}}{\partial\Im(\hat{\mathbf{d}})}\ .\end{aligned}$$

By the definition of conjugate cogradient (2.41), the right side of the above equation is the conjugate cogradient of $\hat{l}$, i.e.

$$\nabla\hat{l}(\hat{\mathbf{d}}, \hat{\mathbf{x}}; \hat{\mathbf{b}}) = \hat{\mathbf{X}}^H(\hat{\mathbf{X}}\hat{\mathbf{d}} - \hat{\mathbf{b}})\ .$$

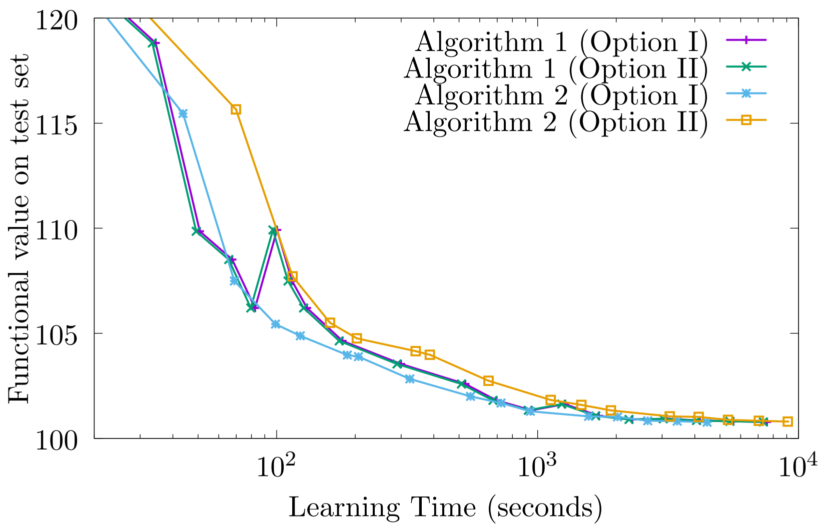


Figure 2.11: Main Result I: convergence speed comparison.

Appendix 2.B Proof of (2.20)

Proof. Let $\mathcal{F}$ be the Fourier operator from $\mathbb{C}^N$ to $\mathbb{C}^N$, so that $\mathcal{F}^{-1} = \mathcal{F}^H$ is the inverse Fourier operator. $\mathbf{x}$ and $\mathbf{X}$ are the vector form and operator form of the coefficient map, respectively. $\hat{\mathbf{x}}$ and $\hat{\mathbf{X}}$ are the corresponding vector and operator in the frequency domain. By definition, we have that $\hat{\mathbf{x}} = \mathcal{F}\mathbf{x}$. We claim that

$$\hat{\mathbf{X}} = \mathcal{F}\mathbf{X}\mathcal{F}^H \ . \tag{2.42}$$

To prove this, notice that

$$\hat{\mathbf{X}}\hat{\mathbf{d}} = \mathcal{F}\big(\mathbf{x} * \mathbf{d}\big) = \mathcal{F}\big(\mathbf{X}\mathbf{d}\big) = \mathcal{F}\mathbf{X}\mathcal{F}^H\mathcal{F}\mathbf{d} = \mathcal{F}\mathbf{X}\mathcal{F}^H\hat{\mathbf{d}} \ , \quad \forall \mathbf{d} \in \mathbb{R}^N \ .$$

Thus we have $\hat{\mathbf{X}} = \mathcal{F}\mathbf{X}\mathcal{F}^H$. With this equation, we have

$$\begin{aligned}\hat{\mathbf{X}}^H(\hat{\mathbf{X}}\hat{\mathbf{d}} - \hat{\mathbf{b}}) &= (\mathcal{F}\mathbf{X}\mathcal{F}^H)^H(\mathcal{F}\mathbf{X}\mathcal{F}^H\mathcal{F}\mathbf{d} - \mathcal{F}\mathbf{b}) = (\mathcal{F}\mathbf{X}^T\mathcal{F}^H)(\mathcal{F}\mathbf{X}\mathbf{d} - \mathcal{F}\mathbf{b}) \\ &= \mathcal{F}\big(\mathbf{X}^T(\mathbf{X}\mathbf{d} - \mathbf{b})\big) \ ,\end{aligned}$$

which is exactly (2.20). □

Appendix 2.C Frequency-domain FISTA

To solve (2.30), we propose frequency-domain FISTA, Algorithm 3. It calculates the gradient in the frequency domain and do projection and extrapolation in the spatial domain. Mathematically speaking, (2.20) illustrates that frequency-domain FISTA is actually equivalent with standard FISTA. However, calculating convolutional operator in the frequency domain reduces computing time. Thus, our algorithm is faster.

Appendix 2.D Details of the assumptions

2.D.1 Description of Assumption 2

To represent Assumption 2 in a concise way, we use the notation

$$\mathbf{D}\mathbf{x} = \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{x}_m \approx \mathbf{b} \ ,$$

where $\mathbf{x} \in \mathbb{R}^{MN}$, $\mathbf{b} \in \mathbb{R}^N$, $\mathbf{D} : \mathbb{R}^{MN} \to \mathbb{R}^N$ is the convolutional dictionary. Then CBPDN problem (2.3) could be written as

$$\min_{\mathbf{x} \in \mathbb{R}^{MN}} (1/2)\|\mathbf{D}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_1 \ . \tag{2.45}$$

The coefficient map $\mathbf{x}$ is usually sparse, and Λ is the set of indices of non-zero elements in $\mathbf{x}$. Then, we have $\mathbf{D}\mathbf{x} = \mathbf{D}_\Lambda \mathbf{x}_\Lambda$. By the results in [Fuc05], problem (2.45) has the unique solution if $\mathbf{D}_\Lambda^T \mathbf{D}_\Lambda$ is invertible[10], and its unique solution satisfies

$$\mathbf{x}_\Lambda^* = (\mathbf{D}_\Lambda^T \mathbf{D}_\Lambda)^{-1} (\mathbf{D}_\Lambda^T \mathbf{b} - \lambda \text{sign}(\mathbf{x}_\Lambda^*)) \ . \tag{2.46}$$

Specifically, Assumption 2 is: *for all signals* $\mathbf{b}$ *and dictionaries* $\mathbf{d}$*, the smallest singular value of* $\mathbf{D}_\Lambda^T \mathbf{D}_\Lambda$ *is lower bounded by a positive number, i.e.*

$$\sigma_{\min}(\mathbf{D}_\Lambda^T \mathbf{D}_\Lambda) \geq \kappa \ . \tag{2.47}$$

[10]Although [Fuc05] only studies standard sparse coding, the uniqueness condition can be applied to the convolutional case because the only condition in their proof is "for a convex function $f(x)$ on $\mathbb{R}^n$, x a minimum if and only if $0 \in \partial f(x)$". The only assumption is the convexity of the function, with no assumptions on the signals and dictionaries. Thus, large signals and convolutional dictionaries as in our case are consistent with the condition in [Fuc05].

Except for condition (2.47), other types of uniqueness conditions of CSC are studied in recent works [PRE17, PSE17, SPR17].

2.D.2 Description of Assumption 3

Specifically, Assumption 3 is, *the surrogate functions* $\mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d})$ *are uniformly strongly convex, i.e.*

$$\langle \nabla \mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d}) - \nabla \mathcal{F}^{(t)}_{\mathrm{mod}}(\tilde{\mathbf{d}}), \mathbf{d} - \tilde{\mathbf{d}} \rangle \geq \mu \big\| \mathbf{d} - \tilde{\mathbf{d}} \big\|^2 \ , \tag{2.48}$$

for all $t, \mathbf{d}, \tilde{\mathbf{d}}$*, for some* $\mu > 0$*.*

Appendix 2.E Proofs of propositions and the theorem

Before proving propositions, we introduce a useful lemma.

Lemma 1 (Uniform smoothness of surrogate functions)**.** *Under Assumptions 1 and 2, we have* $f^{(t)}$ *(2.22) and* $\mathcal{F}^{(t)}_{\mathrm{mod}}$(2.31) *are uniformly L-smooth, i.e.*

$$\begin{aligned} \big\| \nabla f^{(t)}(\mathbf{d}) - \nabla f^{(t)}(\tilde{\mathbf{d}}) \big\| &\leq L_f \big\| \mathbf{d} - \tilde{\mathbf{d}} \big\| \\ \big\| \nabla \mathcal{F}^{(t)}_{\mathrm{mod}}(\mathbf{d}) - \nabla \mathcal{F}^{(t)}_{\mathrm{mod}}(\tilde{\mathbf{d}}) \big\| &\leq L_{\mathcal{F}} \big\| \mathbf{d} - \tilde{\mathbf{d}} \big\| \ , \end{aligned} \tag{2.49}$$

for all $t, \mathbf{d}, \tilde{\mathbf{d}}$*, for some constants* $L_f > 0, L_{\mathcal{F}} > 0$*.*

Proof. First, we consider a single surrogate function:

$$\big\| \nabla f^{(t)}(\mathbf{d}) - \nabla f^{(t)}(\tilde{\mathbf{d}}) \big\| = \big\| (\mathbf{X}^{(t)})^T (\mathbf{X}^{(t)}) (\mathbf{d} - \tilde{\mathbf{d}}) \big\| \ .$$

By $\mathbf{d} \in \mathcal{C}$ (the compact support of $\mathbf{d}$), Assumption 1 (the compact support of $\mathbf{b}$), and equation (2.46) (regularity of convolutional sparse coding), we have $\mathbf{x}^{(t)}$ is uniformly bounded. Therefore, $\mathbf{X}^{(t)}$, the operator form of $\mathbf{x}^{(t)}$, is also uniformly bounded:

$$\big\| \mathbf{X}^{(t)} \big\| \leq M, \tag{2.50}$$

for all t, for some $M > 0$, which is independent of t.

By (2.26), we have

$$\begin{aligned}\left\|\nabla\mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d})-\nabla\mathcal{F}^{(t)}_{\text{mod}}(\tilde{\mathbf{d}})\right\| =&\left\|\frac{1}{\Lambda^{(t)}}\sum_{\tau=1}^{t}(\tau/t)^p(\mathbf{X}^{(\tau)})^T(\mathbf{X}^{(\tau)})(\mathbf{d}-\tilde{\mathbf{d}})\right\| \\ \le&\frac{1}{\Lambda^{(t)}}\sum_{\tau=1}^{t}(\tau/t)^p\left\|(\mathbf{X}^{(\tau)})^T(\mathbf{X}^{(\tau)})(\mathbf{d}-\tilde{\mathbf{d}})\right\| ,\end{aligned}$$

which, together with (2.50), implies (2.49). □

2.E.1 Proof of Proposition 3

Given the strong-convexity (2.48) and smoothness (2.49) of the surrogate function, we start to prove Proposition 3.

Proof. To prove (2.39), we consider a more general case. Let $\mathbf{g}^*$ be the minimizer of the following subproblem:

$$\mathbf{g}^* = \arg\min_{\mathbf{d}} \mathcal{F}(\mathbf{d}) + \iota_{\mathcal{C}}(\mathbf{d}) ,$$

where $\mathcal{F}$ is μ-strongly convex and L-smooth. Moreover, $\mathbf{g}^j$ and $\mathbf{g}^j_{\text{aux}}$ are the iterates generated in Algorithm 3, and j is the loop index. Then, we want to show that

$$\|\mathbf{g}^{j+1}-\mathbf{g}^*\| \le C\mathbf{R}^{(t)}(\mathbf{g}^j_{\text{aux}}) , \quad \forall j \ge 0 . \tag{2.51}$$

By (2.20), it is enough to prove the above for the spatial-domain FISTA. By strong convexity and smoothness of $\mathcal{F}$, we obtain

$$\begin{aligned}&\|\mathbf{g}^{j+1}-\mathbf{g}^*\|^2 \\ =&\left\|\text{Proj}(\mathbf{g}^j_{\text{aux}}-\eta\nabla\mathcal{F}(\mathbf{g}^j_{\text{aux}}))-\text{Proj}(\mathbf{g}^*-\eta\nabla\mathcal{F}(\mathbf{g}^*))\right\|^2 \\ \le&\left\|\mathbf{g}^j_{\text{aux}}-\eta\nabla\mathcal{F}(\mathbf{g}^j_{\text{aux}})-\mathbf{g}^*-\eta\nabla\mathcal{F}(\mathbf{g}^*)\right\|^2 \\ =&\left\|\mathbf{g}^j_{\text{aux}}-\mathbf{g}^*-\eta\big(\nabla\mathcal{F}(\mathbf{g}^j_{\text{aux}})-\nabla\mathcal{F}(\mathbf{g}^*)\big)\right\|^2 \\ =&\|\mathbf{g}^j_{\text{aux}}-\mathbf{g}^*\|^2-2\eta\Big\langle\mathbf{g}^j_{\text{aux}}-\mathbf{g}^*,\nabla\mathcal{F}(\mathbf{g}^j_{\text{aux}})-\nabla\mathcal{F}(\mathbf{g}^*)\Big\rangle+\eta^2\left\|\nabla\mathcal{F}(\mathbf{g}^j_{\text{aux}})-\nabla\mathcal{F}(\mathbf{g}^*)\right\|^2 \\ \le&(1-2\mu\eta+\eta^2L^2)\|\mathbf{g}^j_{\text{aux}}-\mathbf{g}^*\|^2 .\end{aligned}$$

Combining the above inequality and the definition of FPR (2.32), we have

$$\begin{aligned}
\mathbf{R}(\mathbf{g}^j_{\text{aux}}) =& \Big\| \mathbf{g}^j_{\text{aux}} - \text{Proj}\big(\mathbf{g}^j_{\text{aux}} - \eta \nabla \mathcal{F}(\mathbf{g}^j_{\text{aux}})\big) \Big\| \\
=& \|\mathbf{g}^j_{\text{aux}} - \mathbf{g}^{(j+1)}\| \\
=& \|\mathbf{g}^j_{\text{aux}} - \mathbf{g}^* - (\mathbf{g}^{(j+1)} - \mathbf{g}^*)\| \\
\geq& \|\mathbf{g}^j_{\text{aux}} - \mathbf{g}^*\| - \|\mathbf{g}^{(j+1)} - \mathbf{g}^*\| \\
\geq& \Big(1 - \sqrt{1 - 2\mu\eta + \eta^2 L^2}\Big) \|\mathbf{g}^j_{\text{aux}} - \mathbf{g}^*\| \\
\geq& \frac{1 - \sqrt{1 - 2\mu\eta + \eta^2 L^2}}{\sqrt{1 - 2\mu\eta + \eta^2 L^2}} \|\mathbf{g}^{j+1} - \mathbf{g}^*\| \,.
\end{aligned}$$

Let the step size be small enough $\eta \leq \min(\mu/L^2, 1/\mu)$, we have $0 \leq 1 - 2\mu\eta + \eta^2 L^2 \leq 1$, which implies (2.51). Combining (2.51) and (2.33), we get (2.39). □

2.E.2 Proof of Proposition 4

Proof. Recall $(\mathbf{d}^*)^{(t)}$ (2.38) is the "exact solution" of the t^{th} iterate, and $\mathbf{d}^{(t)}$ is the "inexact solution" of the t^{th} iterate (i.e. the approximated solution obtained by stopping condition (2.33)). Then, by the strong convexity of $\mathcal{F}^{(t)}_{\text{mod}}$, we have

$$\begin{aligned}
& \mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t+1)}) - \mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t)}) \\
=& \mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t+1)}) - \mathcal{F}^{(t)}_{\text{mod}}((\mathbf{d}^*)^{(t)}) - \Big(\mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t)}) - \mathcal{F}^{(t)}_{\text{mod}}((\mathbf{d}^*)^{(t)})\Big) \\
\geq& \mu \|\mathbf{d}^{(t+1)} - (\mathbf{d}^*)^{(t)}\|^2 - L\|\mathbf{d}^{(t)} - (\mathbf{d}^*)^{(t)}\|^2 \\
\geq& \mu\Big(\|\mathbf{d}^{(t+1)} - \mathbf{d}^{(t)}\| - \|\mathbf{d}^{(t)} - (\mathbf{d}^*)^{(t)}\|\Big)^2 - L\|\mathbf{d}^{(t)} - (\mathbf{d}^*)^{(t)}\|^2 \,.
\end{aligned}$$

Let $r^{(t)} = \|\mathbf{d}^{(t+1)} - \mathbf{d}^{(t)}\|$. If $r^{(t)} \leq C/t$, Proposition 4 is directly proved. Otherwise, Proposition 3 (2.39) implies $r^{(t)} - \|\mathbf{d}^{(t)} - (\mathbf{d}^*)^{(t)}\| \geq r^{(t)} - C/t \geq 0$ and

$$\mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t+1)}) - \mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t)}) \geq \mu\Big(r^{(t)} - \frac{C}{t}\Big)^2 - \frac{LC^2}{t^2} \,. \tag{2.52}$$

On the other hand,

$$\begin{aligned}
\mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t+1)}) - \mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t)}) =& \underbrace{\mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t+1)}) - \mathcal{F}^{(t+1)}_{\text{mod}}(\mathbf{d}^{(t+1)})}_{\mathbf{T}_1} \\
+ \underbrace{\mathcal{F}^{(t+1)}_{\text{mod}}(\mathbf{d}^{(t+1)}) - \mathcal{F}^{(t+1)}_{\text{mod}}(\mathbf{d}^{(t)})}_{\mathbf{T}_2} &+ \underbrace{\mathcal{F}^{(t+1)}_{\text{mod}}(\mathbf{d}^{(t)}) - \mathcal{F}^{(t)}_{\text{mod}}(\mathbf{d}^{(t)})}_{\mathbf{T}_3} \,,
\end{aligned}$$

Now we will give the upper bounds of $\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3$. Given the smoothness of $\mathcal{F}_{\text{mod}}^{(t)}$(2.49) and $(\mathbf{d}^*)^{(t+1)}$ being the minimizer of $\mathcal{F}_{\text{mod}}^{(t)}$, we have an upper bound of $\mathbf{T}_2$:

$$\begin{aligned}\mathbf{T}_2 =&\mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t+1)}) - \mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t)})\\ =&\mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t+1)}) - \mathcal{F}_{\text{mod}}^{(t+1)}((\mathbf{d}^*)^{(t+1)}) - \Big(\mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t)}) - \mathcal{F}_{\text{mod}}^{(t+1)}((\mathbf{d}^*)^{(t+1)})\Big)\\ \le& L\|\mathbf{d}^{(t+1)} - (\mathbf{d}^*)^{(t+1)}\|^2 - 0 \le \frac{LC^2}{t^2}\ .\end{aligned} \tag{2.53}$$

Based on (2.26), the gradient of $\mathcal{F}_{\text{mod}}^{(t)} - \mathcal{F}_{\text{mod}}^{(t+1)}$ is bounded by

$$\begin{aligned}&\|\nabla\mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}) - \nabla\mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d})\|\\ =&\|\nabla\mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}) - \frac{\alpha^{(t+1)}\Lambda^{(t)}}{\Lambda^{(t+1)}}\nabla\mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}) - \frac{1}{\Lambda^{(t+1)}}\nabla f^{(t+1)}(\mathbf{d})\|\\ \le&\frac{1}{\Lambda^{(t+1)}}\|\nabla\mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d})\| + \frac{1}{\Lambda^{(t+1)}}\|\nabla f^{(t+1)}(\mathbf{d})\| \le C_0/(\Lambda^{(t+1)}) \le C_1/t\ ,\end{aligned}$$

for some constant $C_1 > 0$. The second inequality follows from $\mathbf{d} \in \mathcal{C}$ (the compact support of $\mathbf{d}$), Assumption 1 (the compact support of $\mathbf{b}$), and equation (2.50) (boundedness of $\mathbf{X}$). The last inequality is derived by the follows:

$$\frac{1}{\Lambda^{(t+1)}} = \frac{(t+1)^p}{\sum_{\tau=1}^{(t+1)}\tau^p} \le \frac{(t+1)^p}{\int_0^{(t+1)}\tau^p \mathrm{d}\tau} = \frac{p}{t+1}\ .$$

Then, $\mathcal{F}_{\text{mod}}^{(t)} - \mathcal{F}_{\text{mod}}^{(t+1)}$ is a Lipschitz continuous function with $L = C_1/t$, which implies

$$\mathbf{T}_1 + \mathbf{T}_3 \le \frac{C_1}{t}r^{(t)}\ .$$

Therefore,

$$\mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t+1)}) - \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)}) \le \frac{C_1}{t}r^{(t)} + \frac{LC^2}{t^2}\ . \tag{2.54}$$

Combining (2.52) and (2.54), we have

$$\mu\Big(r^{(t)} - \frac{C}{t}\Big)^2 - \frac{LC^2}{t^2} \le \frac{C_1}{t}r^{(t)} + \frac{LC^2}{t^2}\ ,$$

which implies

$$(r^{(t)})^2 - \frac{2C + C_1}{t}r^{(t)} \le \frac{2LC^2}{\mu t^2}\ .$$

This can be written more neatly as

$$(r^{(t)})^2 - 2\frac{C_2}{t}r^{(t)} \le \frac{C_3}{t^2}\ , \quad \text{for some } C_2 > 0, C_3 > 0\ .$$

Finally, $r^{(t)}$ is bounded by $r^{(t)} \le (C_2 + \sqrt{C_2^2 + C_3})/t$. (2.40) is proved. □

2.E.3 Proof of Proposition 1

Proof. Define a sequence of random variables $Y_i = i^p Z_i$. Their expectations and variances are $\mu_i = i^p\mu$ and $\sigma_i^2 = i^{2p}\sigma^2$, respectively. Now we apply the Lyapunov central limit theorem on the stochastic sequence $\{Y_i\}$. First, we check the Lyapunov condition [Bil08]. Let

$$s_n^2 = \sum_{i=1}^n \sigma_i^2 = \sum_{i=1}^n i^{2p}\sigma^2 = \Theta(n^{2p+1}) \ ,$$

then we have

$$\frac{1}{s_n^{2+\delta}}\sum_{i=1}^n \mathbb{E}\Big[|Y_i-\mu_i|^{2+\delta}\Big] \le \frac{1}{s_n^{2+\delta}}\sum_{i=1}^n (i^p\sigma)^{2+\delta} = \mathcal{O}\left(\frac{n^{2p+1+\delta p}}{n^{2p+1+\delta p+\delta/2}}\right) = \mathcal{O}(n^{-\delta/2}) \ . \tag{2.55}$$

The Lyapunov condition is satisfied, so, by the Lyapunov central limit theorem, we have $\frac{1}{s_n}\sum_{i=1}^n (Y_i-\mu_i) \xrightarrow{\mathrm{d}} N(0,1)$. Furthermore, the definition of $\hat{Z}^n_{\mathrm{mod}}$ indicates

$$\begin{aligned}\frac{1}{s_n}\sum_{i=1}^n (Y_i-\mu_i) = \frac{1}{s_n}\sum_{i=1}^n i^p(Z_i-\mu) &= \frac{\sum_{i=1}^n i^p}{\sqrt{\sum_{i=1}^n i^{2p}}\sigma}\left(\frac{1}{\sum_{i=1}^n i^p}\sum_{i=1}^n i^p(Z_i-\mu)\right)\\ &= \frac{\sum_{i=1}^n i^p}{\sqrt{\sum_{i=1}^n i^{2p}}\sigma}(\hat{Z}^n_{\mathrm{mod}}-\mu) \ .\end{aligned}$$

Given the following inequalities:

$$\sum_{i=1}^n i^p < \int_1^{n+1} s^p \mathrm{d}s < \frac{1}{p+1}(n+1)^{p+1} \ , \quad \sum_{i=1}^n i^p > \int_0^n s^p\mathrm{d}s = \frac{1}{p+1}(n)^{p+1} \ ,$$

we have

$$\begin{aligned}\frac{1}{s_n}\sum_{i=1}^n (Y_i-\mu_i) &\le \Big(1+\frac{1}{n}\Big)^{p+1}\frac{1}{\sigma}\frac{\sqrt{2p+1}}{p+1}\sqrt{n}(\hat{Z}^n_{\mathrm{mod}}-\mu) \ ,\\ \frac{1}{s_n}\sum_{i=1}^n (Y_i-\mu_i) &\ge \Big(1+\frac{1}{n}\Big)^{-(p+1)}\frac{1}{\sigma}\frac{\sqrt{2p+1}}{p+1}\sqrt{n}(\hat{Z}^n_{\mathrm{mod}}-\mu) \ .\end{aligned}$$

Then (2.34) is obtained by $\frac{1}{s_n}\sum_{i=1}^n (Y_i-\mu_i) \xrightarrow{\mathrm{d}} N(0,1)$ and $(1+1/n)\to 1$.

The formula $\mathrm{Var}(\mathbf{X}) = \mathbb{E}\mathbf{X}^2 - (\mathbb{E}\mathbf{X})^2 \ge 0$ implies

$$\left(\mathbb{E}\Big[\sqrt{n}\big|\hat{Z}^n_{\mathrm{mod}}-\mu\big|\Big]\right)^2 \le \mathbb{E}\Big[n\big|\hat{Z}^n_{\mathrm{mod}}-\mu\big|^2\Big] \ .$$

By the independence of different Z_i, we have

$$\mathbb{E}\Big[n\big|\hat{Z}^n_{\mathrm{mod}}-\mu\big|^2\Big] = \frac{n}{(\sum_{i=1}^n i^p)^2}\sum_{i=1}^n \mathbb{E}\Big[i^{2p}\big|Z_i-\mu\big|^2\Big] \le \frac{(p+1)^2}{2p+1}B^2 \ ,$$

where B is the upper bound of Z_i as Z_i is compact supported. (2.35) is proved. □

2.E.4 Proof of Proposition 2

Proof. First, we fix $\mathbf{d} \in \mathcal{C}$. Let $i \to \tau, n \to t, Z_i \to f(\mathbf{d}; \mathbf{b}^{(\tau)})$, then, by Proposition 1, we have

$$\mathbb{E}\Big[\sqrt{t}\big|F(\mathbf{d}) - F_{\text{mod}}^{(t)}(\mathbf{d})\big|\Big] \leq \frac{p+1}{\sqrt{2p+1}}B \ , \quad \forall t \in \{1, 2, \cdots\}$$

for some $B > 0$, for fixed $\mathbf{d}$. Since F and $F_{\text{mod}}^{(t)}$ are continuously differentiable and have uniformly bounded derivatives (2.50), we have $\mathbb{E}\Big[\sqrt{t}\big|F(\mathbf{d}) - F_{\text{mod}}^{(t)}(\mathbf{d})\big|\Big]$ is uniformly continuous w.r.t $\mathbf{d}$ on a compact set $\mathcal{C}$. Thus, the boundedness of $\mathbb{E}\Big[\sqrt{t}\big|F(\mathbf{d}) - F_{\text{mod}}^{(t)}(\mathbf{d})\big|\Big]$ on each $\mathbf{d}$ implies the boundedness for all $\mathbf{d} \in \mathcal{C}$. Inequality (2.37) is proved. Taking $p \to 0$, we have (2.36). $\square$

2.E.5 Proof of Theorem 1

Proof. Let $u^{(t)} = \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})$. Inspired by the proof of Proposition 3 in [MBP10], we will show that $u^{(t)}$ is a "quasi-martingale" [Fis65].

$$\begin{aligned}
& u^{(t+1)} - u^{(t)} \\
=& \mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t+1)}) - \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)}) \\
=& \underbrace{\mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t+1)}) - \mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t)})}_{\mathbf{T}_2} + \underbrace{\mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t)}) - \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})}_{\mathbf{T}_4} \ .
\end{aligned}$$

The bound of $\mathbf{T}_2$ is given by (2.53). Furthermore, definition (2.22) tells us $f^{(t+1)}(\mathbf{d}^{(t)}) = f(\mathbf{d}^{(t)}; \mathbf{b}^{(t+1)})$, which implies

$$\begin{aligned}
\mathbf{T}_4 =& \mathcal{F}_{\text{mod}}^{(t+1)}(\mathbf{d}^{(t)}) - \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)}) \\
=& \left(\frac{1}{\Lambda^{(t+1)}} f(\mathbf{d}^{(t)}; \mathbf{b}^{(t+1)}) + \frac{\alpha^{(t+1)}\Lambda^{(t)}}{\Lambda^{(t+1)}} \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})\right) - \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)}) \\
=& \frac{f(\mathbf{d}^{(t)}; \mathbf{b}^{(t+1)}) - F_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})}{\Lambda^{(t+1)}} + \frac{F_{\text{mod}}^{(t)}(\mathbf{d}^{(t)}) - \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})}{\Lambda^{(t+1)}} \ .
\end{aligned}$$

By the definitions of f (2.7) and F (2.31), we have $F_{\text{mod}}^{(t)}(\mathbf{d}^{(t)}) \leq \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})$. Define $\mathcal{G}^t$ as all the previous information: $\mathcal{G}^t \triangleq \{\mathbf{x}^{(\tau)}, \mathbf{b}^{(\tau)}, \mathbf{d}^{(\tau)}\}_{\tau=1}^t$. Thus, taking conditional expectation, we obtain

$$\mathbb{E}[\mathbf{T}_4|\mathcal{G}^t] \leq \frac{1}{\Lambda^{(t+1)}}\Big(\mathbb{E}[f(\mathbf{d}^{(t)}; \mathbf{b}^{(t+1)})|\mathcal{G}^t] - F_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})\Big) = \frac{1}{\Lambda^{(t+1)}}\Big(F(\mathbf{d}^{(t)}) - F_{\text{mod}}^{(t)}(\mathbf{d}^{(t)})\Big) \ .$$

Therefore, the positive part of $\mathbb{E}[\mathbf{T}_4|\mathcal{G}^t]$ is bounded by

$$\mathbb{E}[\mathbf{T}_4|\mathcal{G}^t]^+ \leq \frac{1}{\Lambda^{(t+1)}}\|F - F^{(t)}_{\text{mod}}\|_\infty = \mathcal{O}\left(\frac{1}{t^{3/2}}\right) ,$$

where the second inequality follows from (2.37). Given the bound of $\mathbf{T}_2$ (2.53) and $\mathbf{T}_4$, we have

$$\sum_{t=1}^{\infty} \mathbb{E}\big[\mathbb{E}[u^{(t+1)} - u^{(t)}|\mathcal{G}^t]^+\big] \leq \sum_{t=1}^{\infty}\left(\mathcal{O}\left(\frac{1}{t^{3/2}}\right) + \mathcal{O}\left(\frac{1}{t^2}\right)\right) < +\infty ,$$

which implies that $u^{(t+1)}$ generated by Algorithm 2 is a quasi-martingale. Thus, by results in [Bot99, Sec. 4.4] or [MBP10, Theorem 6], we have $u^{(t)}$ converges almost surely.

For the proofs of 2, 3 and 4, using the results in Proposition 4, 2 in this paper, following the same proof line of Proposition 3 and 4 in [MBP10], we can obtain the results in 2, 3 and 4. $\square$

```
Algorithm 2: Online Convolutional Dictionary Learning (Surrogate-Splitting)
  Initialize: Initialize $\mathbf{d}^{(0)}$, let $\mathbf{H}_{\text{mod}}^{(0)} \leftarrow 0, \mathbf{c}_{\text{mod}}^{(0)} \leftarrow 0$ or $\hat{\mathbf{H}}_{\text{mod}}^{(0)} \leftarrow 0, \hat{\mathbf{c}}_{\text{mod}}^{(0)} \leftarrow 0$.
1 for $t = 1, \cdots, T$ do
2     Sample a signal $\mathbf{b}^{(t)}$ from $\{\mathbf{s}_{\text{split}}\}$.
3     Solve convolutional sparse coding problem (2.10a) to obtain $\mathbf{x}^{(t)}$.
4     if Option I then
5         Update $\mathbf{H}_{\text{mod}}^{(t)}, \mathbf{c}_{\text{mod}}^{(t)}$ in the spatial-domain with sparse matrix $\mathbf{X}^{(t)}$:
              $\mathbf{H}_{\text{mod}}^{(t)} = \alpha^{(t)} \mathbf{H}_{\text{mod}}^{(t-1)} + (\mathbf{X}^{(t)})^T \mathbf{X}^{(t)},\ \mathbf{c}_{\text{mod}}^{(t)} = \alpha^{(t)} \mathbf{c}_{\text{mod}}^{(t-1)} + (\mathbf{X}^{(t)})^T \mathbf{b}^{(t)}$
6         Solve the following subproblem with FISTA (stopping condition (2.33)):
              $\mathbf{d}^{(t)} = \operatorname*{arg\,min}_{\mathbf{d} \in \mathbb{R}^{MD}} \mathcal{F}_{\text{mod}}^{(t)}(\mathbf{d}) + \iota_{\mathcal{C}}(\mathbf{d})\ .$
7     else if Option II then
8         Update $\hat{\mathbf{H}}_{\text{mod}}^{(t)}, \hat{\mathbf{c}}_{\text{mod}}^{(t)}$ in the frequency-domain:
              $\hat{\mathbf{H}}_{\text{mod}}^{(t)} = \alpha^{(t)} \hat{\mathbf{H}}_{\text{mod}}^{(t-1)} + (\hat{\mathbf{X}}^{(t)})^H \hat{\mathbf{X}}^{(t)},\ \hat{\mathbf{c}}_{\text{mod}}^{(t)} = \alpha^{(t)} \hat{\mathbf{c}}_{\text{mod}}^{(t-1)} + (\hat{\mathbf{X}}^{(t)})^H \hat{\mathbf{b}}^{(t)}$
9         Solve the following subproblem with frequency-domain FISTA (stopping
            condition (2.33), see Appendix 2.C):
              $\mathbf{d}^{(t)} = \operatorname*{arg\,min}_{\mathbf{d} \in \mathbb{R}^{MN}} \hat{\mathcal{F}}_{\text{mod}}^{(t)}(\hat{\mathbf{d}}) + \iota_{\mathcal{C}_{\text{PN}}}(\mathbf{d})\ .$
10    end
11 end
  Output: $\mathbf{d}^{(T)}$
```

Table 2.2: Single step complexity and memory usage of Algorithm 2. N: signal dimension; M: number of dictionary kernels; D: size of each kernel; ρ: average density of the coefficient maps; J: average loops of FISTA in each step.

Scheme	Single step complexity	Memory usage
Spatial (dense)	$T_{\text{CBPDN}} + \mathcal{O}(D^2M^2N) + \mathcal{O}(JD^2M^2)$	$\mathcal{O}(D^2M^2) + \mathcal{O}(DMN)$
Spatial (sparse)	$T_{\text{CBPDN}} + \mathcal{O}(D^2M^2N\rho) + \mathcal{O}(JD^2M^2)$	$\mathcal{O}(D^2M^2) + \mathcal{O}(DMN\rho)$
Frequency update	$T_{\text{CBPDN}} + \mathcal{O}(JM^2N) + \mathcal{O}(JMN\log(N))$	$\mathcal{O}(M^2N)$

Table 2.3: Comparison between different options of Algorithm 1.

Schemes	Average single-step complexity (seconds)				Memory
	CBPDN	FFT/IFFT	Update $\mathbf{d}^{(t)}$	Total	Usage (MB)
Spatial (dense matrix)	14.8	0	1.978	16.8	2346.44
Spatial (sparse matrix)	14.8	0	0.241	15.1	111.38
Frequency domain	14.8	0.047	0.025	14.9	154.84

Table 2.4: Comparison of two options in Algorithm 2 with different splitting size $\tilde{N}$.

$\tilde{N}$	Average single-step complexity (seconds)				Memory Usage (MB)
	CBPDN	Update $\mathbf{H}^{(t)}$	FISTA (Loops $\times$ Single step)	Total	
Update in the spatial domain with dense matrix					
256×256	14.8	25.1	57×0.017	40.9	3058.56
128×128	3.42	6.80	37×0.017	10.8	1258.37
64×64	1.05	2.25	24×0.017	3.71	808.32
(Option I) Update in the spatial domain with sparse matrix					
256×256	14.8	4.47	57×0.017	20.3	486.91
128×128	3.42	1.77	37×0.017	5.82	366.51
64×64	1.05	0.84	24×0.017	2.30	342.90
(Option II) Update in the frequency domain (including extra time caused by FFT)					
256×256	14.8	0.89	57×1.068	76.6	2458.84
128×128	3.42	0.22	37×0.244	12.7	622.28
64×64	1.05	0.06	24×0.072	2.84	158.11

Table 2.5: Main Result II: memory usage comparison.

Scheme	Memory (MB)
Batch learning (consensus update, batch $K = 10$)	1959.58
Batch learning (consensus update, batch $K = 20$)	3887.08
Batch learning (consensus update, batch $K = 40$)	7742.08
Batch learning (Papyan et al. [PRS17], batch $K = 10$)	1802.29
Batch learning (Papyan et al. [PRS17], batch $K = 20$)	3390.24
Batch learning (Papyan et al. [PRS17], batch $K = 40$)	6566.15
Our algorithm "Online-Samp" in [LGW17]	158.11
Algorithm 1 Option I (sgd-spatial)	111.38
Algorithm 1 Option II (sgd-frequency)	154.84
Algorithm 2 Option I (surro-spatial)	342.90
Algorithm 2 Option II (surro-frequency)	158.11

Algorithm 3: Frequency-domain FISTA for solving subproblem (2.30)

```
Input: Hessian matrix $\hat{\mathbf{H}}^{(t)}_{\text{mod}}$ and vector $\hat{\mathbf{c}}^{(t)}_{\text{mod}}$.
Dictionary of last iterate: $\mathbf{d}^{(t-1)}$.
Initialize: Let $\mathbf{g}^0 = \mathbf{d}^{(t-1)}$ (warm start), $\mathbf{g}^0_{\text{aux}} = \mathbf{g}^0$, $\gamma^0 = 1$.
1 for $j = 0, 1, 2, \ldots$ until condition (2.33) is satisfied do
2     Compute DFT: $\hat{\mathbf{g}}^j_{\text{aux}} = \text{FFT}(\mathbf{g}^j_{\text{aux}})$.
3     Compute conjugate cogradient: $\nabla\hat{\mathcal{F}}^{(t)}_{\text{mod}}(\hat{\mathbf{g}}^j_{\text{aux}}) = \frac{1}{\Lambda^{(t)}}\left(\hat{\mathbf{H}}^{(t)}_{\text{mod}}\hat{\mathbf{g}}^j_{\text{aux}} - \hat{\mathbf{c}}^{(t)}_{\text{mod}}\right)$ .
4     Compute the next iterate:
          $$\mathbf{g}^{j+1} = \text{Proj}_{\mathcal{C}_{\text{PN}}}\Big(\text{IFFT}\big(\hat{\mathbf{g}}^j_{\text{aux}} - \eta\nabla\hat{\mathcal{F}}^{(t)}_{\text{mod}}(\hat{\mathbf{g}}^j_{\text{aux}})\big)\Big) \,. \quad (2.43)$$
      Let $\gamma^{j+1} = \big(1 + \sqrt{1 + 4(\gamma^j)^2}\big)/2$, then compute the auxiliary variable:
          $$\mathbf{g}^{j+1}_{\text{aux}} = \mathbf{g}^{j+1} + \frac{\gamma^j - 1}{\gamma^{j+1}}(\mathbf{g}^{j+1} - \mathbf{g}^j) \,. \quad (2.44)$$
5 end
Output: $\mathbf{d}^{(t)} \leftarrow \mathbf{g}^J$, where $J$ is the last iterate.
```

CHAPTER 3

Learning Solvers

This chapter deals with the sparse coding problem (1.2) with a *fixed and known* dictionary $\mathbf{D}$. We develop fast sparse coding solvers by learning to optimize (L2O). Section 3.1 describes *learned ISTA (LISTA)*, a successful example of L2O for sparse coding; Section 3.2 introduces some theories on the convergence of LISTA and simplifies the model based on the theory; In Section 3.3, we propose "support selection", a technique based on prior knowledge that further improves the performance of LISTA; In Section 3.4, we study the structure of the parameters and further simplify the model; Section 3.5 extends the algorithms and theories to the convolutional sparse coding; Section 3.6 provides the numerical results; Section 3.7 concludes this chapter.

3.1 Model 0: Learned ISTA

3.1.1 Background

We consider sparse vector recovery, or sparse coding introduced in Chapter 2:

$$\mathbf{b} = \sum_{m=1}^{M} \mathbf{d}_m x_m^* + \varepsilon = \mathbf{D}\mathbf{x}^* + \varepsilon, \tag{3.1}$$

where $\mathbf{b} \in \mathbb{R}^N$ is the observation, $\mathbf{x}^* = [x_1^*, \cdots, x_M^*]^T \in \mathbb{R}^M$ is the unknown vector we want to recover, $\mathbf{D} = [\mathbf{d}_1, \cdots, \mathbf{d}_M] \in \mathbb{R}^{N\times M}$ is the dictionary which is *known* in this chapter, and $\varepsilon \in \mathbb{R}^N$ is additive Gaussian white noise. For simplicity, each column of $\mathbf{D}$, named as a dictionary kernel, is normalized, that is, $\|\mathbf{d}_m\|_2 = \|\mathbf{D}_{:,m}\|_2 = 1,\ m = 1, 2, \cdots, M$. Typically, we have $N \ll M$, so Equation (3.1) is an under-determined system.

However, when $\mathbf{x}^*$ is sufficiently sparse, it can be recovered faithfully. A popular approach is to solve the LASSO problem below (where λ is a scalar):

$$\underset{\mathbf{x}}{\text{minimize}} \frac{1}{2}\|\mathbf{b}-\mathbf{D}\mathbf{x}\|_2^2+\lambda\|\mathbf{x}\|_1 \tag{3.2}$$

using iterative algorithms such as the iterative shrinkage thresholding algorithm (ISTA):

$$\mathbf{x}^{(k+1)}=\eta_{\lambda/L}\Big(\mathbf{x}^{(k)}+\frac{1}{L}\mathbf{D}^T(\mathbf{b}-\mathbf{D}\mathbf{x}^{(k)})\Big),\quad k=0,1,2,\dots \tag{3.3}$$

where η_θ is the soft-thresholding function[1] and L is usually taken as the largest eigenvalue of $\mathbf{D}^T\mathbf{D}$.

3.1.2 Learned ISTA

In [GL10], inspired by ISTA, the authors proposed a learning-based model named Learned ISTA (LISTA). They view ISTA as a recurrent neural network (RNN) that is illustrated in Figure 3.1(a), where $\mathbf{W}_1^{(k)}\equiv\frac{1}{L}\mathbf{D}^T$, $\mathbf{W}_2^{(k)}\equiv\mathbf{I}-\frac{1}{L}\mathbf{D}^T\mathbf{D}$, $\theta^{(k)}\equiv\frac{1}{L}\lambda$. LISTA, illustrated in Figure 3.1(b), unrolls the RNN and truncates it into K iterations:

$$\mathbf{x}^{(k+1)}=\eta_{\theta^{(k)}}(\mathbf{W}_1^{(k)}\mathbf{b}+\mathbf{W}_2^{(k)}\mathbf{x}^{(k)}),\quad k=0,1,\cdots,K-1. \tag{3.4}$$

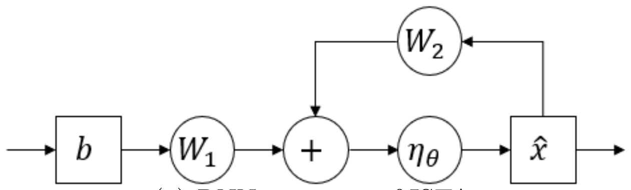
leading to a K-layer feed-forward neural network. Learning the parameters $\{\mathbf{W}_1^{(k)},\mathbf{W}_2^{(k)},\theta^{(k)}\}_k$ in the network can be viewed as learning a new "solver" parameterized by ISTA.

Given each pair of sparse vector and its noisy measurements $(\mathbf{x}^*,\mathbf{b})$, applying (3.4) from some initial point $\mathbf{x}^{(0)}$ and using $\mathbf{b}$ as the input yields $\mathbf{x}^{(k)}$. Our goal is to choose the parameters Θ such that $\mathbf{x}^{(k)}$ is close to $\mathbf{x}^*$ for all sparse $\mathbf{x}^*$ following some distribution $\mathcal{P}$. Therefore, given the distribution $\mathcal{P}$, all parameters in $\Theta=\{\mathbf{W}_1^{(k)},\mathbf{W}_2^{(k)},\theta^{(k)}\}_{k=0}^{K-1}$ are subject to learning:

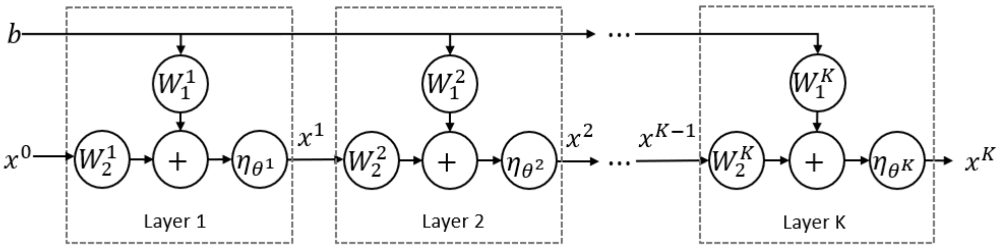
$$\underset{\Theta}{\text{minimize}}\,\mathbb{E}_{\mathbf{x}^*,\mathbf{b}\sim\mathcal{P}}\Big\|\mathbf{x}^{(K)}\Big(\Theta,\mathbf{b},\mathbf{x}^{(0)}\Big)-\mathbf{x}^*\Big\|_2^2. \tag{3.5}$$

This problem is approximately solved over a training dataset $\{(\mathbf{x}_i^*,\mathbf{b}_i)\}_{i=1}^N$ sampled from $\mathcal{P}$. Since $\mathbf{x}^{(K)}$ is actually a neural network, (3.5) is solvable with recent machine learning platforms (TensorFlow [AAB15], PyTorch [PGM19], etc.).

[1]Soft- thresholding function is defined in a component-wise way: $\eta_\theta(\mathbf{x})=\text{sign}(\mathbf{x})\max(0,|\mathbf{x}|-\theta)$



(a) RNN structure of ISTA.



(b) Unfolded learned ISTA Network.

Figure 3.1: Diagrams of ISTA and LISTA.

Many empirical results, e.g., [GL10, SBS15, WLH16], show that a trained K-layer LISTA (with K usually set to $10 \sim 20$) or its variants can generalize more than well to unseen samples $(\mathbf{x}', \mathbf{b}')$ from the same distribution and recover $\mathbf{x}'$ from $\mathbf{b}'$ to the same accuracy within one or two order-of-magnitude fewer iterations than the original ISTA. Additionally, the accuracies of the outputs $\{\mathbf{x}^{(k)}\}$ of the layers $k = 1, .., K$ gradually improve.

Despite the empirical success in constructing fast trainable regressors for approximating iterative solvers [GL10, SBS15, WCZ16, WLH16, WLC16, WYC16, SLX16, BSR17, ZG18a, AO18, ZDD18, ITW18], the theoretical understanding of such approximations remains limited. In this chapter, we target on the following problems:

- Is there a theoretical guarantee to ensure that the learned solver LISTA (3.4) converges faster and/or produces a better solution than ISTA (3.3)? If the answer is affirmative, can we quantize the amount of acceleration?
- When the parameters $\{\mathbf{W}_1^{(k)}, \mathbf{W}_2^{(k)}, \theta^{(k)}\}_{k=0}^{K-1}$ are ideal? Are there any explanations on the learned parameters?
- Rather than training (3.4) as a conventional "black-box", can we benefit from exploiting the structure of its parameters to simplify the model and improve the recovery results?

3.1.3 Related Works

Some related works on analyzing and understanding LISTA are presented here. [MB17] re-factorized the Gram matrix of dictionary, by trying to nearly diagonalize the Gram matrix with a basis, subject to a small ℓ_1 perturbation. They thus re-parameterized LISTA a new factorized architecture that achieved similar acceleration gain to LISTA, hence ending up with an "indirect" proof. They concluded that LISTA can converge faster than ISTA, but still sublinearly. [GEB18] interpreted LISTA as a projected gradient descent descent (PGD) where the projection step was inaccurate, which enables a trade-off between approximation error and convergence speed.

Several other works examined the theoretical properties of some sibling architectures to LISTA. [XWG16] studied the model proposed by [WLH16], which unfolded/truncated the iterative hard thresholding (IHT) algorithm instead of ISTA, for approximating the solution to ℓ_0-minimization. They showed that the learnable fast regressor can be obtained by using a transformed dictionary with improved restricted isometry property (RIP). However, their discussions are not applicable to LISTA directly, although IHT is linearly convergent [BD09] under rather strong assumptions. Their discussions were also limited to linear sparse coding and resulting fully-connected networks only. [BSR17, MMB17] studied a similar learning-based model inspired from another LASSO solver, called approximated message passing (AMP). [BSR17] showed the MMSE-optimality of an AMP-inspired model, but not accompanied with any convergence rate result. Also, the popular assumption in analyzing AMP algorithms (called "state evolution") does not hold when analyzing ISTA.

3.2 Model 1: LISTA-CP

In this section, we study the necessary condition that LISTA converges to $\mathbf{x}^*$. Based on that condition, LISTA can be simplified. Then we establish the convergence guarantee for LISTA and the simplified LISTA. Before that, we consider some mild assumptions.

Assumption 4 (Basic assumptions)**.** *The signal* $\mathbf{x}^*$ *and the observation noise* ε *are sampled* *from the following set:*

$$(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, \sigma) \triangleq \Big\{(\mathbf{x}^*, \varepsilon)\Big| |x_i^*| \leq B, \forall i,\ \|\mathbf{x}^*\|_0 \leq s, \|\varepsilon\|_1 \leq \sigma\Big\}. \tag{3.6}$$

In other words, $\mathbf{x}^*$ *is bounded and* s*-sparse*[2] $(s \geq 2)$*, and* ε *is bounded.*

3.2.1 Partial Weight Coupling

Now we focus on the convergence of LISTA (3.4) and suggest a minimal set of parameters that are really necessary to learn.

Theorem 2 (Necessary Condition)**.** *Given* $\{\mathbf{W}_1^{(k)}, \mathbf{W}_2^{(k)}, \theta^{(k)}\}_{k=0}^{\infty}$ *and* $\mathbf{x}^{(0)} = 0$*, let* $\mathbf{b}$ *be observed by (3.1) and* $\{\mathbf{x}^{(k)}\}_{k=1}^{\infty}$ *be generated layer-wise by LISTA (3.4). If the following holds uniformly for any* $(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, 0)$ *(no observation noise):*

$$\mathbf{x}^{(k)}\Big(\{\mathbf{W}_1^{(\tau)}, \mathbf{W}_2^{(\tau)}, \theta^{(\tau)}\}_{\tau=0}^{k-1}, \mathbf{b}, \mathbf{x}^{(0)}\Big) \to \mathbf{x}^*, \quad \text{as } k \to \infty$$

and $\{\mathbf{W}_2^{(k)}\}_{k=1}^{\infty}$ *are bounded*

$$\|\mathbf{W}_2^{(k)}\|_2 \leq B_W, \quad \forall k = 0, 1, 2, \cdots,$$

then $\{\mathbf{W}_1^{(k)}, \mathbf{W}_2^{(k)}, \theta^{(k)}\}_{k=0}^{\infty}$ *must satisfy*

$$\mathbf{W}_2^{(k)} - \big(\mathbf{I} - \mathbf{W}_1^{(k)}\mathbf{D}\big) \to 0, \quad \text{as } k \to \infty \tag{3.7}$$

$$\theta^{(k)} \to 0, \quad \text{as } k \to \infty. \tag{3.8}$$

Proofs of the results throughout this chapter can be found in the appendix. The conclusion (3.7) demonstrates that the weights $\{\mathbf{W}_1^{(k)}, \mathbf{W}_2^{(k)}\}_{k=0}^{\infty}$ in LISTA asymptotically satisfies the following partial weight coupling structure:

$$\mathbf{W}_2^{(k)} = \mathbf{I} - \mathbf{W}_1^{(k)}\mathbf{D}. \tag{3.9}$$

We adopt the above partial weight coupling for all layers, letting $\mathbf{W}^{(k)} = (\mathbf{W}_1^{(k)})^T \in \mathbb{R}^{N\times M}$, thus simplifying LISTA (3.4) to *LISTA-CP* (LISTA with weights CouPled):

$$\mathbf{x}^{(k+1)} = \eta_{\theta^{(k)}}\Big(\mathbf{x}^{(k)} + (\mathbf{W}^{(k)})^\top(\mathbf{b} - \mathbf{D}\mathbf{x}^{(k)})\Big), \quad k = 0, 1, \cdots, K-1, \tag{3.10}$$

[2] *A signal is* s*-sparse if it has no more than* s *non-zero entries.*

where $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_{k=0}^{(K-1)}$ remain as free parameters to train.

The coupled structure (3.9) for soft-thresholding based algorithms was empirically studied in [BSR17]. The similar structure was also theoretically studied in Proposition 1 of [XWG16] for IHT algorithms using the fixed-point theory, but they let all layers share the same weights, i.e. $\mathbf{W}_2^{(k)} = \mathbf{W}_2, \mathbf{W}_1^{(k)} = \mathbf{W}_1, \forall k$.

3.2.2 Convergence

In this section, we formally establish the linear convergence of LISTA. The output of the k^{th} layer $\mathbf{x}^{(k)}$ depends on the parameters $\{\mathbf{W}^{(\tau)}, \theta^{(\tau)}\}_{\tau=0}^{k-1}$, the observed measurement $\mathbf{b}$ and the initial point $\mathbf{x}^{(0)}$. Strictly speaking, $\mathbf{x}^{(k)}$ should be written as $\mathbf{x}^{(k)}\Big(\{\mathbf{W}^{(\tau)}, \theta^{(\tau)}\}_{\tau=0}^{k-1}, \mathbf{b}, \mathbf{x}^{(0)}\Big)$. By the observation model $\mathbf{b} = \mathbf{D}\mathbf{x}^* + \varepsilon$, since $\mathbf{D}$ is given and $\mathbf{x}^{(0)}$ can be taken as 0, $\mathbf{x}^{(k)}$ therefore depends on $\{(\mathbf{W}^{(\tau)}, \theta^{(\tau)})\}_{\tau=0}^{(k)}$, $\mathbf{x}^*$ and ε. So, we can write $\mathbf{x}^{(k)}\Big(\{\mathbf{W}^{(\tau)}, \theta^{(\tau)}\}_{\tau=0}^{k-1}, \mathbf{x}^*, \varepsilon\Big)$. For simplicity, we instead just write $\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon)$.

Theorem 3 (Convergence of LISTA-CP). *Given $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_{k=0}^{\infty}$ and $x^{(0)} = 0$, let $\{\mathbf{x}^{(k)}\}_{k=1}^{\infty}$ be generated by (3.10). If Assumption 4 holds and s is sufficiently small, then there exists a sequence of parameters $\{\mathbf{W}^{(k)}, \theta^{(k)}\}$ such that, for all $(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, \sigma)$, we have the error bound:*

$$\|\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon) - \mathbf{x}^*\|_2 \leq sB\exp(-ck) + C\sigma, \quad \forall k = 1, 2, \cdots, \tag{3.11}$$

where $c > 0, C > 0$ are constants that depend only on $\mathbf{D}$ and s. Recall s (sparsity of the signals) and σ (noise-level) are defined in (3.6).

If $\sigma = 0$ (noiseless case), (3.11) reduces to

$$\|\mathbf{x}^{(k)}(\mathbf{x}^*, 0) - \mathbf{x}^*\|_2 \leq sB\exp(-ck). \tag{3.12}$$

The recovery error converges to 0 at a linear rate as the number of layers goes to infinity. Combined with Theorem 2, we see that the partial weight coupling structure (3.10) is both necessary and sufficient to guarantee convergence in the noiseless case.

Discussion: The bound (3.12) also explains why LISTA (or its variants) can converge faster than ISTA and fast ISTA (FISTA) [BT09]. With a proper λ (see (3.2)), ISTA converges at an $O(1/k)$ rate and FISTA converges at an $O(1/k^2)$ rate [BT09]. With a large enough λ, ISTA achieves a linear rate [BL08, ZHL17]. With $\bar{\mathbf{x}}(\lambda)$ being the solution of LASSO (noiseless case), these results can be summarized as: before the iterates $\mathbf{x}^{(k)}$ settle on a support[3],

$$\mathbf{x}^{(k)} \to \bar{\mathbf{x}}(\lambda) \text{ sublinearly}, \quad \|\bar{\mathbf{x}}(\lambda) - \mathbf{x}^*\| = O(\lambda), \quad \lambda > 0$$

$$\mathbf{x}^{(k)} \to \bar{\mathbf{x}}(\lambda) \text{ linearly}, \quad \|\bar{\mathbf{x}}(\lambda) - \mathbf{x}^*\| = O(\lambda), \quad \lambda \text{ large enough.}$$

Based on the choice of λ in LASSO, the above observation reflects an inherent trade-off between convergence rate and approximation accuracy in solving the problem (3.1), see a similar conclusion in [GEB18]: a larger λ leads to faster convergence but a less accurate solution, and vice versa.

However, if λ is not constant throughout all iterations/layers, but instead chosen adaptively for each step, more promising trade-off can arise[4]. LISTA and LISTA-CP, with the thresholds $\{\theta^{(k)}\}_{k=0}^{(K-1)}$ free to train, actually adopt this idea because $\{\theta^{(k)}\}_{k=0}^{(K-1)}$ corresponds to a path of LASSO parameters $\{\lambda^{(k)}\}_{k=0}^{(K-1)}$. With extra free trainable parameters, $\{\mathbf{W}^{(k)}\}_{k=0}^{(K-1)}$ (LISTA-CP) or $\{\mathbf{W}_1^{(k)}, \mathbf{W}_2^{(k)}\}_{k=0}^{(K-1)}$ (LISTA), learning based solvers are able to converge to an accurate solution at a fast convergence rate. Theorem 3 demonstrates the existence of such sequence $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_k$ in LISTA-CP (3.10). The experiment results in Fig. 3.4 show that such $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_k$ can be obtained by training.

3.3 Model 2: LISTA-CPSS

3.3.1 Support Selection

We introduce a special thresholding scheme to LISTA, called *support selection* (SS), which is inspired by “kicking” [OMD10] in linearized Bregman iteration. This technique shows advantages on recoverability and convergence. Its impact on improving LISTA convergence rate and reducing recovery errors will be analyzed in Section 3.3.2. With support selection,

[3]After $\mathbf{x}^{(k)}$ settles on a support, i.e. as k large enough such that support$(\mathbf{x}^{(k)})$ is fixed, even with small λ, ISTA reduces to a linear iteration, which has a linear convergence rate [TBZ16].

[4]This point was studied in [HYZ08, XZ13] with classical compressive sensing settings, while our learning settings can learn a good path of parameters without a complicated thresholding rule or any manual tuning.

at each LISTA layer *before* applying soft thresholding, we will select a certain percentage of entries with largest magnitudes, and trust them as "true support" and won't pass them through thresholding. Those entries that do not go through thresholding will be directly fed into next layer, together with other thresholded entires.

Assume we select $p^{(k)}\%$ of entries as the trusted support at layer k. LISTA with support selection (LISTA-SS) can be generally formulated as

$$\mathbf{x}^{(k+1)} = \eta_{\mathrm{ss}}{}_{\theta^{(k)}}^{p^{(k)}}\Big(\mathbf{W}_1^{(k)}\mathbf{b} + \mathbf{W}_2^{(k)}\mathbf{x}^{(k)}\Big), \quad k = 0, 1, \cdots, K-1, \tag{3.13}$$

where η_{ss} is the thresholding operator with support selection, formally defined as:

$$(\eta_{\mathrm{ss}}{}_{\theta^{(k)}}^{p^{(k)}}(\mathbf{v}))_i = \begin{cases} v_i & : v_i > \theta^{(k)}, \qquad i \in S^{p^{(k)}}(\mathbf{v}), \\ v_i - \theta^{(k)} & : v_i > \theta^{(k)}, \qquad i \notin S^{p^{(k)}}(\mathbf{v}), \\ 0 & : -\theta^{(k)} \leq v_i \leq \theta^{(k)} \\ v_i + \theta^{(k)} & : v_i < -\theta^{(k)}, \qquad i \notin S^{p^{(k)}}(\mathbf{v}), \\ v_i & : v_i < -\theta^{(k)}, \qquad i \in S^{p^{(k)}}(\mathbf{v}), \end{cases}$$

where $S^{p^{(k)}}(\mathbf{v})$ includes the elements with the largest $p^{(k)}\%$ magnitudes in vector $\mathbf{v}$:

$$S^{p^{(k)}}(\mathbf{v}) = \Big\{ i_1, i_2, \cdots, i_{p^{(k)}} \Big| |v_{i_1}| \geq |v_{i_2}| \geq \cdots |v_{i_{p^{(k)}}}| \cdots \geq |v_{i_n}| \Big\}. \tag{3.14}$$

To clarify, in (3.13), $p^{(k)}$ is a hyperparameter to be manually tuned, and $\theta^{(k)}$ is a parameter to train. We use an empirical formula to select $p^{(k)}$ for layer k: $p^{(k)} = \min(p \cdot k, p_{\max})$, where p is a positive constant and $p_{\max}$ is an upper bound of the percentage of the support cardinality. Here p and $p_{\max}$ are both hyperparameters to be manually tuned.

If we adopt the partial weight coupling in (3.9), then (3.13) is modified as

$$\mathbf{x}^{(k+1)} = \eta_{\mathrm{ss}}{}_{\theta^{(k)}}^{p^{(k)}}\Big(\mathbf{x}^{(k)} + (\mathbf{W}^{(k)})^T(\mathbf{b} - \mathbf{D}\mathbf{x}^{(k)})\Big), \quad k = 0, 1, \cdots, K-1. \tag{3.15}$$

Algorithm abbreviations For simplicity, hereinafter we will use the abbreviation "CP" for the partial weight coupling in (3.9), and "SS" for the support selection technique. *LISTA-CP* denotes the LISTA model with weights coupling (3.10). *LISTA-SS* denotes the LISTA model with support selection (3.13). Similarly, *LISTA-CPSS* stands for a model using both techniques (3.15), which has the best performance. Unless otherwise specified, *LISTA* refers to the baseline LISTA (3.4).

3.3.2 Convergence

In this subsection, we study the convergence of LISTA-CPSS and compare it with LISTA-CP. To measure the advantage of support selection, we consider a mildly stronger assumption than Assumption 4.

Assumption 5. *Signal* $\mathbf{x}^*$ *and observation noise* ε *are sampled from the following set:*

$$(\mathbf{x}^*, \varepsilon) \in \bar{\mathcal{X}}(B, \underline{B}, s, \sigma) \triangleq \Big\{(\mathbf{x}^*, \varepsilon)\Big| |x_i^*| \le B, \forall i,\ \|\mathbf{x}^*\|_1 \ge \underline{B}, \|\mathbf{x}^*\|_0 \le s, \|\varepsilon\|_1 \le \sigma\Big\}. \quad (3.16)$$

The only difference between Assumptions 4 and 5 is that $\|\mathbf{x}^*\|_1 \ge \underline{B}$ is required in Assumption 5.

Theorem 4 (Convergence of LISTA-CPSS). *Given* $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_{k=0}^{\infty}$ *and* $\mathbf{x}^{(0)} = 0$, *let* $\{\mathbf{x}^{(k)}\}_{k=1}^{\infty}$ *be generated by (3.15). With Assumption 4 and the same parameters as in Theorem 3, the approximation error can be bounded for all* $(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, \sigma)$:

$$\|\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon) - \mathbf{x}^*\|_2 \le sB \exp\Big(-\sum_{t=0}^{k-1} c_{\mathrm{ss}}^{(t)}\Big) + C_{\mathrm{ss}}\sigma, \quad \forall k = 1, 2, \cdots, \quad (3.17)$$

where $c_{\mathrm{ss}}^{(k)} \ge c$ *for all* k *and* $C_{\mathrm{ss}} \le C$.

If Assumption 5 holds, s *is small enough, and* $\underline{B} \ge 2C\sigma$ *(SNR is not too small), then there exists another sequence of parameters* $\{\tilde{\mathbf{W}}^{(k)}, \tilde{\theta}^{(k)}\}$ *that yields the following improved error bound: for all* $(\mathbf{x}^*, \varepsilon) \in \bar{\mathcal{X}}(B, \underline{B}, s, \sigma)$,

$$\|\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon) - \mathbf{x}^*\|_2 \le sB \exp\Big(-\sum_{t=0}^{k-1} \tilde{c}_{\mathrm{ss}}^{(t)}\Big) + \tilde{C}_{\mathrm{ss}}\sigma, \quad \forall k = 1, 2, \cdots, \quad (3.18)$$

where $\tilde{c}_{\mathrm{ss}}^{(k)} \ge c$ *for all* k, $\tilde{c}_{\mathrm{ss}}^{(k)} > c$ *for large enough* k, *and* $\tilde{C}_{\mathrm{ss}} < C$.

The bound in (3.17) ensures that, with the same assumptions and parameters, LISTA-CPSS is *at least no worse* than LISTA-CP. The bound in (3.18) shows that, under stronger assumptions, LISTA-CPSS can be *strictly better* than LISTA-CP in both folds: $\tilde{c}_{\mathrm{ss}}^{(k)} > c$ is the better convergence rate of LISTA-CPSS; $\tilde{C}_{\mathrm{ss}} < C$ means that the LISTA-CPSS can achieve smaller approximation error than the minimum error that LISTA can achieve.

3.4 Model 3: Analytic LISTA

In LISTA-CP (3.10) and LISTA-CPSS (3.15), a sequence of matrices $\{\mathbf{W}^{(k)}\}_k$ and a sequence of scalars $\{\theta^{(k)}\}_k$ have to be trained to obtain good performance. In this section, We will further study properties of "good" parameters in LISTA-CP[5], and then discuss how to analytically compute the sequence of matrices $\{\mathbf{W}^{(k)}\}_k$ rather than relying solely on black-box training. In this way, the model could be further significantly simplified, with little performance loss.

3.4.1 Structure of the parameters

Theorem 3 only shows the existence of parameters $\{\mathbf{W}^{(k)},\theta^{(k)}\}_{k=1}^{(k)}$ that make LISTA converges linearly. In this section, we go deeper on the structure of such parameters.

The *mutual coherence* of the dictionary $\mathbf{D}$ is a significant concept in compressive sensing [DE03, Ela07, LLL18]. A dictionary with small coherence possesses better sparse recovery performance. Motivated by this point, we introduce the following definition.

Definition 1. *Given $\mathbf{D}\in\mathbb{R}^{N\times M}$ with each of its column normalized, we define the generalized mutual coherence:*

$$\tilde{\mu}(\mathbf{D}) = \inf_{\substack{\mathbf{W}\in\mathbb{R}^{N\times M}\\ (\mathbf{W}_{:,i})^T\mathbf{D}_{:,i}=1, 1\le i\le M}} \Big\{ \max_{\substack{i\neq j\\ 1\le i,j\le M}} (\mathbf{W}_{:,i})^T\mathbf{D}_{:,j}\Big\}. \tag{3.19}$$

Additionally, We define $\mathcal{W}(\mathbf{D}) = \big\{\mathbf{W}\in\mathbb{R}^{N\times M} : \mathbf{W}$ attains the infimum given (3.19)$\big\}$. A weight matrix $\mathbf{W}$ is "good" if $\mathbf{W}\in\mathcal{W}(\mathbf{D})$.

In the above definition, problem (3.19) is feasible and attainable, i.e., $\mathcal{W}(\mathbf{D})\neq\varnothing$, which was proven in Lemma 2.

Theorem 5 (Structure of the parameters)**.** *Take any $(\mathbf{x}^*,\varepsilon)\in\mathcal{X}(B,s,0)$, any $\mathbf{W}\in\mathcal{W}(\mathbf{D})$, and any sequence $\gamma^{(k)}\in(0,\frac{2}{2\tilde{\mu}s-\tilde{\mu}+1})$. Using them, construct the parameters $\{\mathbf{W}^{(k)},\theta^{(k)}\}$:*

$$\mathbf{W}^{(k)} = \gamma^{(k)}\mathbf{W},\quad \theta^{(k)} = \gamma^{(k)}\tilde{\mu}(\mathbf{D}) \sup_{\mathbf{x}^*\in\mathcal{X}(B,s)} \big\{\|\mathbf{x}^{(k)}(\mathbf{x}^*,\varepsilon)-\mathbf{x}^*\|_1\big\}, \tag{3.20}$$

[5]For simplicity of the proofs, we analyze LISTA-CP rather than LISTA-CPSS, and All the analysis in this section can be generalized to models with support selection.

while the sequence $\{\mathbf{x}^{(k)}(\mathbf{x}^*,\varepsilon)\}_{k=1}^{\infty}$ *is generated by (3.10) using the above parameters and* $\mathbf{x}^{(0)} = \mathbf{0}$ *(Note that each* $\mathbf{x}^{(k)}(\mathbf{x}^*,\varepsilon)$ *depends only on* $\theta^{(k-1)}, \theta^{(k-2)}, \ldots$ *and defines* $\theta^{(k)}$*). Let Assumption 4 hold with any* $B > 0$ *and* $s < (1 + 1/\tilde{\mu})/2$*. Then, for* $k = 1, 2, \ldots$*, it holds that*

$$\begin{aligned} &\operatorname{support}\big(\mathbf{x}^{(k)}(\mathbf{x}^*,\varepsilon)\big) \subset \mathbb{S}, \\ &\|\mathbf{x}^{(k)}(\mathbf{x}^*,\varepsilon) - \mathbf{x}^*\|_2 \le sB \exp\Big(-\sum_{\tau=0}^{k-1} c^{(\tau)}\Big), \end{aligned} \tag{3.21}$$

where $\mathbb{S}$ *is the support of* $\mathbf{x}^*$ *and* $c^{(k)} = -\log\big((2\tilde{\mu}s - \tilde{\mu})\gamma^{(k)} + |1-\gamma^{(k)}|\big)$ *is a positive constant.*

In Theorem 5, Eqn. (3.20) defines the properties of "good" parameters:

- The weights $\mathbf{W}^{(k)}$ can be separated as the product of a scalar $\gamma^{(k)}$ and a matrix $\mathbf{W}$ independent of layer index k, where $\mathbf{W}$ has small coherence with $\mathbf{D}$.
- $\gamma^{(k)}$ is bounded in an interval.
- $\theta^{(k)}/\gamma^{(k)}$ is proportional to the ℓ_1 error of the output of the k^{th} layer.

The factor $c^{(k)}$ takes the maximum at $\gamma^{(k)} = 1$. If $\gamma^{(k)} \equiv 1$, the recovery error converges to zero in the same rate with (3.12). Although $\gamma^{(k)} \equiv 1$ gives the optimal theoretical upper bound if there are infinitely many layers $k = 0, 1, 2, \cdots$, it is not the optimal choice for finite k. Practically, there are finitely many layers and $\gamma^{(k)}$ obtained by learning is bounded in an interval.

3.4.2 Optimality of the parameters

In this subsection, we introduce a lower bound of the recovery error of LISTA, which illustrates that the parameters analytically given by (3.20) in Theorem 5 are optimal in the convergence order (linear).

Assumption 6. *The signal* $\mathbf{x}^*$ *is a random variable following the distribution* P_X*. Let* $\mathbb{S} = $ *support*$(\mathbf{x}^*)$*.* P_X *satisfies:* $2 \le |\,\mathbb{S}\,| \le s$*;* $\mathbb{S}$ *uniformly distributes on the whole index set; non-zero part* $\mathbf{x}^*_{\mathbb{S}}$ *satisfies the uniform distribution with bound* B*:* $|x_i^*| \le B, \forall i \in \mathbb{S}$*. Moreover, the observation noise* $\varepsilon = \mathbf{0}$*.*

Theorem 5 tells that an ideal weight $\mathbf{W} \in \mathcal{W}(\mathbf{D})$ satisfies $\mathbf{I} - \mathbf{W}^T\mathbf{D} \approx \mathbf{0}$. But this cannot be met exactly in the overcomplete $\mathbf{D}$ case, i.e., $N < M$. Definition 2 defines the set of matrices $\mathbf{W}$ such that $\mathbf{W}^T\mathbf{D}$ is bounded away from the identity $\mathbf{I}$.

Definition 2. *Given* $\mathbf{D} \in \mathbb{R}^{N\times M}, s \geq 2, \bar{\sigma}_{\min} > 0$*, we define a set that* $\mathbf{W}^{(k)}$ *are chosen from:*

$$\bar{\mathcal{W}}(\mathbf{D}, s, \bar{\sigma}_{\min}) = \Big\{\mathbf{W} \in \mathbb{R}^{N\times M} \Big| \sigma_{\min}\Big(\mathbf{I} - (\mathbf{W}_{:,\mathbb{S}})^T\mathbf{D}_{:,\mathbb{S}}\Big) \geq \bar{\sigma}_{\min}, \forall \mathbb{S} \text{ with } 2 \leq |\,\mathbb{S}\,| \leq s\Big\}. \tag{3.22}$$

Based on Definition 2, we define a set that $\Theta = \{\mathbf{W}^{(k)}, \theta^{(k)}\}_{k=0}^{\infty}$ are chosen from:

Definition 3. *Let* $\{\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon)\}_{k=1}^{\infty}$ *be generated by (3.10) with* $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_{k=0}^{\infty}$ *and* $\mathbf{x}^{(0)} = \mathbf{0}$*. Then we define* $\mathbf{T}$ *as the set of parameters that guarantee there is no false positive in* $\mathbf{x}^{(k)}$*:*

$$\begin{aligned}\mathbf{T} = \Big\{\{\mathbf{W}^{(k)} \in \bar{\mathcal{W}}(\mathbf{D}, s, \bar{\sigma}_{min}), \theta^{(k)}\}_{k=0}^{\infty}\Big| \\ \text{support}(\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon)) \subset \mathbb{S},\ \forall(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, 0),\ \forall k\Big\}\end{aligned} \tag{3.23}$$

The conclusion (3.21) demonstrates that $\mathbf{T}$ is nonempty because "support$(\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon)) \subset \mathbb{S}$" is satisfied as long as $\theta^{(k-1)}$ large enough. Actually, $\mathbf{T}$ contains almost all "good" parameters because considerable false positives lead to large recovery errors. With $\mathbf{T}$ defined, we have:

Theorem 6 (Optimality of the parameters)**.** *Let the sequence* $\{\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon)\}_{k=1}^{\infty}$ *be generated by (3.10) with* $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_{k=0}^{\infty}$ *and* $\mathbf{x}^{(0)} = \mathbf{0}$*. Under Assumption 6, for all parameters* $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_{k=0}^{\infty} \in \mathbf{T}$ *and any sufficient small* $\epsilon > 0$*, we have*

$$\|\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon) - \mathbf{x}^*\|_2 \geq \epsilon\|\mathbf{x}^*\|_2 \exp(-\bar{c}k), \tag{3.24}$$

with probability at least $(1 - \epsilon s^{3/2} - \epsilon^2)$*, where* $\bar{c} = s\log(3) - \log(\bar{\sigma}_{min})$*.*

This theorem illustrates that, with high probability, the convergence rate of LISTA cannot be faster than a linear rate. Thus, the parameters given in (3.20), that leads to the linear convergence if $\gamma^{(k)}$ is bounded within an interval near 1, are optimal with respect to the order of convergence of LISTA.

3.4.3 Analytic LISTA: calculating weights without training

Following Theorem 5 and Theorem 6, we set $\mathbf{W}^{(k)} = \gamma^{(k)}\mathbf{W}$, where $\gamma^{(k)}$ is a scalar, and propose *Tied LISTA (TiLISTA)*:

$$\mathbf{x}^{(k+1)} = \eta_{\theta^{(k)}}\Big(\mathbf{x}^{(k)} - \gamma^{(k)}\mathbf{W}^T(\mathbf{D}\mathbf{x}^{(k)} - \mathbf{b})\Big), \tag{3.25}$$

where $\Theta = \big\{\{\gamma^{(k)}\}_k, \{\theta^{(k)}\}_k, \mathbf{W}\big\}$ are parameters to train. The matrix $\mathbf{W}$ is tied over all the layers. Further, we notice that the selection of $\mathbf{W}$ from $\mathcal{W}(\mathbf{D})$ depends on $\mathbf{D}$ only. Hence we propose the **analytic LISTA (ALISTA)** that decomposes tied-LISTA into two stages:

$$\mathbf{x}^{(k+1)} = \eta_{\theta^{(k)}}\Big(\mathbf{x}^{(k)} - \gamma^{(k)}\tilde{\mathbf{W}}^T(\mathbf{D}\mathbf{x}^{(k)} - \mathbf{b})\Big), \tag{3.26}$$

where $\tilde{\mathbf{W}}$ is pre-computed by solving the following problem (**Stage 1**)[6]:

$$\tilde{\mathbf{W}} \in \mathop{\arg\min}_{\mathbf{W}\in\mathbb{R}^{N\times M}} \big\|\mathbf{W}^T\mathbf{D}\big\|_F^2, \quad \text{s.t. } (\mathbf{W}_{:,m})^T\mathbf{D}_{:,m} = 1,\ \forall m = 1, 2, \cdots, M, \tag{3.27}$$

Then with $\tilde{\mathbf{W}}$ fixed, $\{\gamma^{(k)}, \theta^{(k)}\}_k$ in (3.26) are learned from end to end (**Stage 2**). (3.27) reformulates (3.19) to minimizing the Frobenius norm of $\mathbf{W}^T\mathbf{D}$ (a quadratic objective), over linear constraints. This is a standard convex quadratic program, which is easier to solve than to solve (3.19) directly.

Table 3.1: Summary: variants of LISTA and the number of parameters to learn.

Vanilla LISTA (3.4)	LISTA-CPSS (3.15)	TiLISTA (3.25)	ALISTA (3.26)
$O(KM^2 + K + MN)$	$O(KNM + K)$	$O(NM + K)$	$O(K)$

3.5 Convolutional Analytic LISTA

As we introduced in Chapter 2, *convolutional sparse coding* (CSC) is an extension of the sparse coding (3.1) that gains increasingly attention in the machine learning area. [SG18] showed that the CSC could be similarly approximated and accelerated by a LISTA-type feed-forward

[6]Some details and a complexity analysis of Stage 1 are discussed in Appendix 3.H.1

network. [TDB18] designed a structure of sparse auto-encoder inspired by multi-layer CSC. [PRE16, SPR17] also revealed CSC as a potentially useful tool for understanding general convolutional neural networks (CNNs). In this section, we will study how to apply ALISTA in the convolutional case.

3.5.1 Convolutional sparse coding

We extend the analytic LISTA to the convolutional case in this section, starting from discussing the convolutional sparse coding (CSC) where the general linear transform is replaced by convolutions in order to learn spatially invariant features:

$$\mathbf{b} = \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{x}_m^* + \varepsilon. \tag{3.28}$$

Each $\mathbf{d}_m$ is a dictionary kernel (or filter) and $\{\mathbf{d}_m\}_{m=1}^M$ is the dictionary of filters, M denotes the number of filters. $\{\mathbf{x}_m^*\}_{m=1}^M$ is the set of coefficient maps that are assumed to have sparse structure, and $*$ is the convolution operator.

Now we consider 2D convolution and take[7] $\mathbf{b} \in \mathbb{R}^{N^2}, \mathbf{d}_m \in \mathbb{R}^{D^2}, \mathbf{x}_m \in \mathbb{R}^{(N+D-1)^2}$. Equation (3.28) is pointwisely defined as[8]:

$$\mathbf{b}(i,j) = \sum_{k=0}^{D-1}\sum_{l=0}^{D-1}\sum_{m=1}^{M} \mathbf{d}_m(k,l)\mathbf{x}_m(i+k,j+l) + \varepsilon(i,j), \quad 0 \le i,j \le N-1. \tag{3.29}$$

We concatenate $\mathbf{d}_m$s and $\mathbf{x}_m$s: $\mathbf{d} = [\mathbf{d}_1, \cdots, \mathbf{d}_M]^T$, $\mathbf{x} = [\mathbf{x}_1, \cdots, \mathbf{x}_M]^T$, and rewrite (3.29) as:

$$\mathbf{b} = \sum_{m=1}^{M} \mathbf{D}_{\text{conv},m}^N(\mathbf{d}_m)\mathbf{x}_m + \varepsilon = \mathbf{D}_{\text{conv}}^N(\mathbf{d})\mathbf{x} + \varepsilon, \tag{3.30}$$

where the matrix $\mathbf{D}_{\text{conv}}^N(\mathbf{d}) = [\mathbf{D}_{\text{conv},1}^N(\mathbf{d}_1), \cdots, \mathbf{D}_{\text{conv},M}^N(\mathbf{d}_M)] \in \mathbb{R}^{N^2 \times (N+D-1)^2 M}$, depending on the signal size N and the dictionary $\mathbf{d}$, is defined in detail in (3.68) in Appendix 3.F.2.

[7]Here, $\mathbf{b}, \mathbf{d}_m, \mathbf{x}_m$ are vectors. The notion $\mathbf{b}(i,j)$ means the $(iN+j)^{\text{th}}$ entry of $\mathbf{b}$. Additionally, $\mathbf{d}_m, \mathbf{x}_m$ are defined in the same way for all $m = 1, \cdots, M$.

[8]Strictly speaking, (3.29) is the cross-correlation rather than convolution. However in TensorFlow, that operation is named as convolution, and we follow that convention to be consistent with the learning community.

3.5.2 Convolutional ALISTA

From (3.28), the convolutional LISTA becomes a natural extension of the fully-connected LISTA-CP (3.10):

$$\mathbf{x}_m^{(k+1)} = \eta_{\theta^{(k)}}\Big(\mathbf{x}_m^{(k)} - \big(\mathbf{w}_m^{(k)}\big)' * \Big(\sum_{\bar{m}=1}^{M} \mathbf{d}_{\bar{m}} * \mathbf{x}_{\bar{m}}^{(k)} - \mathbf{b}\Big)\Big), \quad m = 1, 2, \cdots, M, \tag{3.31}$$

where $\{\mathbf{w}_m^{(k)}\}_{m=1}^M$ share the same sizes with $\{\mathbf{d}_m\}_{m=1}^M$ and $(\cdot)'$ means a 180 rotation of the filter [CPR13]. We concatenate the filters together: $\mathbf{w}^{(k)} = [\mathbf{w}_1^{(k)}, \cdots, \mathbf{w}_M^{(k)}]^T \in \mathbb{R}^{D^2 M}$. Parameters to train are $\Theta = \{\mathbf{w}^{(k)}, \theta^{(k)}\}_k$.

Let $\mathbf{W}_{\text{conv}}^N(\mathbf{w}^{(k)})$ be the matrix induced by dictionary $\mathbf{w}^{(k)}$ with the same dimensionality as $\mathbf{D}_{\text{conv}}^N(\mathbf{d})$. Since convolution can be written as a matrix form (3.30), (3.31) is equivalent to

$$\mathbf{x}^{(k+1)} = \eta_{\theta^{(k)}}\Big(\mathbf{x}^{(k)} - (\mathbf{W}_{\text{conv}}^N(\mathbf{w}^{(k)}))^T(\mathbf{D}_{\text{conv}}^N(\mathbf{d})\mathbf{x}^{(k)} - \mathbf{b})\Big). \tag{3.32}$$

Then by just substituting $\mathbf{D}, \mathbf{W}^{(k)}$ with $\mathbf{D}_{\text{conv}}^N(\mathbf{d}), \mathbf{W}_{\text{conv}}^N(\mathbf{w}^{(k)})$ respectively, Theorem 5 and Theorem 6 can be applied to the convolutional LISTA.

Proposition 5. *Let* $\mathbf{D} = \mathbf{D}_{\text{conv}}^N(\mathbf{d})$ *and* $\mathbf{W}^{(k)} = \mathbf{W}_{\text{conv}}^N(\mathbf{w}^{(k)})$*. With Assumption 4 and other settings the same with those in Theorem 5, (3.21) holds. With Assumption 6 and other settings the same with those in Theorem 6, (3.24) holds.*

Similar to the fully connected case (3.26), based on the results in Proposition 5, we should set $\mathbf{w}_m^{(k)} = \gamma_m^{(k)}\tilde{\mathbf{w}}_m,\ m = 1, 2, \cdots, M$, where $\tilde{\mathbf{w}} = [\tilde{\mathbf{w}}_1, \cdots, \tilde{\mathbf{w}}_M]^T$ is chosen from

$$\tilde{\mathbf{w}} \in \mathcal{W}_{\text{conv}}^N = \mathop{\arg\min}_{\substack{\mathbf{w}\in\mathbb{R}^{D^2 M} \\ \mathbf{w}_m\cdot\mathbf{d}_m=1,\ 1\le m\le M}} \Big\| \big(\mathbf{W}_{\text{conv}}^N(\mathbf{w})\big)^T \mathbf{D}_{\text{conv}}^N(\mathbf{d}) \Big\|_F^2. \tag{3.33}$$

However, (3.33) is not as efficient to solve as (3.27). To see that, matrices $\mathbf{D}_{\text{conv}}^N(\mathbf{d})$ and $\mathbf{W}_{\text{conv}}^N(\mathbf{w})$ are both of size $N^2 \times (N + D - 1)^2 M$, the coherence matrix $\big(\mathbf{W}_{\text{conv}}^N(\mathbf{w})\big)^T \mathbf{D}_{\text{conv}}^N(\mathbf{d})$ is thus of size $(N + D - 1)^2 M \times (N + D - 1)^2 M$. In the typical application setting of CSC, $\mathbf{b}$ is usually an image rather than a small patch. For example, if the image size is 100×100, dictionary size is $7 \times 7 \times 64$, $N = 100, D = 7, M = 64$, then $(N+D-1)^2 M \times (N+D-1)^2 M \approx 5 \times 10^{11}$.

3.5.3 Calculating convolutional weights analytically and efficiently

To overcome the computational challenge of solving (3.33), we exploit the following *circular convolution* as an efficient approximation:

$$\mathbf{b}(i,j) = \sum_{k=0}^{D-1}\sum_{l=0}^{D-1}\sum_{m=1}^{M} \mathbf{d}_m(k,l)\mathbf{x}_m\big((i+k)_{\mathrm{mod}N},(j+l)_{\mathrm{mod}N}\big)+\varepsilon(i,j), \quad 0\le i,j\le N-1, \tag{3.34}$$

where $\mathbf{b}\in\mathbb{R}^{N^2}, \mathbf{d}_m\in\mathbb{R}^{D^2}, \mathbf{x}_m\in\mathbb{R}^{N^2}$. Similar to (3.29), we rewrite (3.34) in a compact way:

$$\mathbf{b} = \sum_{m=1}^{M} \mathbf{D}_{\mathrm{cir},m}^{N}(\mathbf{d}_m)\mathbf{x}_m + \varepsilon = \mathbf{D}_{\mathrm{cir}}^{N}(\mathbf{d})\mathbf{x}+\varepsilon,$$

where $\mathbf{D}_{\mathrm{cir}}^{N}(\mathbf{d}):\mathbb{R}^{N^2M}\to\mathbb{R}^{N^2}$ is a matrix depending on the signal size N and the dictionary $\mathbf{d}$. Then the coherence minimization with the circular convolution is given by

$$\mathcal{W}_{\mathrm{cir}}^{N} = \underset{\substack{\mathbf{w}\in\mathbb{R}^{D^2M}\\ \mathbf{w}_m\cdot\mathbf{d}_m=1,\ 1\le m\le M}}{\arg\min} \left\|\big(\mathbf{W}_{\mathrm{cir}}^{N}(\mathbf{w})\big)^T\mathbf{D}_{\mathrm{cir}}^{N}(\mathbf{d})\right\|_F^2. \tag{3.35}$$

The following theorem motivates us to use the solution to (3.35) to approximate that of (3.33).

Theorem 7. *The solution sets of (3.33) and (3.35) satisfy the following properties:*

1. $\mathcal{W}_{\mathrm{cir}}^{N} = \mathcal{W}_{\mathrm{cir}}^{2D-1}, \forall N\ge 2D-1.$
2. *If at least one of the matrices* $\{\mathbf{D}_{\mathrm{cir},1}^{2D-1},\cdots,\mathbf{D}_{\mathrm{cir},M}^{2D-1}\}$ *is non-singular,* $\mathcal{W}_{\mathrm{cir}}^{2D-1}$ *involves only a unique element. Furthermore,*

$$\lim_{N\to\infty}\mathcal{W}_{\mathrm{conv}}^{N} = \mathcal{W}_{\mathrm{cir}}^{2D-1}. \tag{3.36}$$

The solution set $\mathcal{W}_{\mathrm{cir}}^{N}$ is not related with the image size N as long as $N\ge 2D-1$, thus one can deal with a much smaller-size problem (let $N=2D-1$). Further, (3.36) indicates that as N gets (much) larger than D, the boundary condition becomes less important. Thus, one can use $\mathcal{W}_{\mathrm{cir}}^{2D-1}$ to approximate $\mathcal{W}_{\mathrm{conv}}^{N}$. In Appendix 3.H.2, we introduce the algorithm details of solving (3.35).

Based on Proposition 5 and Theorem 7, we obtain the **convolutional ALISTA**:

$$\mathbf{x}_m^{(k+1)} = \eta_{\theta^{(k)}}\Big(\mathbf{x}_m^{(k)} - \gamma_m^{(k)}\big(\tilde{\mathbf{w}}_m\big)' * \Big(\sum_{\bar{m}=1}^{M} \mathbf{d}_{\bar{m}} * \ \mathbf{x}_{\bar{m}}^{(k)} - \mathbf{b}\Big)\Big), \quad m = 1, 2, \cdots, M, \tag{3.37}$$

where $\tilde{\mathbf{w}} = [\tilde{\mathbf{w}}_1, \cdots, \tilde{\mathbf{w}}_M]^T \in \mathcal{W}_{\text{cir}}^{2D-1}$ and $\Theta = \{\{\gamma_m^{(k)}\}_{m,k}, \{\theta^{(k)}\}_k\}$ are the parameters to train. (3.37) is a simplified form, compared to the empirically unfolded CSC model recently proposed in [SG18]

3.6 Numerical Results

3.6.1 Training Strategy

In this section we have a detailed discussion on the stage-wise training strategy in empirical experiments. Denote $\Theta = \{(\mathbf{W}_1^{(k)}, \mathbf{W}_2^{(k)}, \theta^{(k)})\}_{k=0}^{(K-1)}$ as all the weights in the network. Note that $(\mathbf{W}_1^{(k)}, \mathbf{W}_2^{(k)})$ can be coupled as in (3.7). Denote $\Theta^{(\tau)} = \{(\mathbf{W}_1^{(k)}, \mathbf{W}_2^{(k)}, \theta^{(k)})\}_{k=0}^{\tau}$ all the weights in the τ-th and all the previous layers. Define an initial learning rate α_0 and two decayed learning rates α_1, α_2. In real training, we have $\alpha_1 = 0.2\alpha_0, \alpha_2 = 0.02\alpha_0$. Our training strategy is described as below:

- Train the network layer by layer. Training in each layer consists of 3 stages.
- In layer τ, $\Theta^{(\tau-1)}$ is pre-trained.
  - Train $(\mathbf{W}_1^{(\tau)}, \mathbf{W}_2^{(\tau)}, \theta^{(\tau)})$ the initial learning rate α_0.
  - Train $\Theta^{(\tau)} = \Theta^{\tau-1} \cup (\mathbf{W}_1^{(\tau)}, \mathbf{W}_2^{(\tau)}, \theta^{(\tau)})$ with the learning rates α_1 and α_2.
- Proceed training to the next layer.

The layer-wise training is widely adopted in previous LISTA-type networks. We add the learning rate decaying that is able to stabilize the training process. It will make the previous layers change very slowly when the training proceeds to deeper layers because learning rates of first several layers will exponentially decay and quickly go to near zero when the training process progresses to deeper layers, which can prevent them varying too far from pre-trained positions. It works well especially when the unfolding goes deep to $K > 10$. All models trained and reported in experiments section are trained using the above strategy.

Remark While adopting the above stage-wise training strategy, we first finish a complete training pass, calculate the intermediate results and final outputs, and then draw curves and evaluate the performance based on these results, instead of logging how the best performance changes when the training process goes deeper. This manner possibly accounts for the reason why some curves plotted in Section 3.6.2 display some unexpected fluctuations.

3.6.2 Simulation Experiments

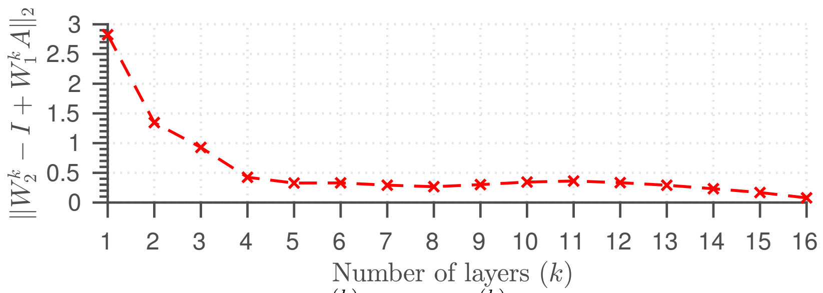
Experiments Setting. We choose $N = 250, M = 500$. We sample the entries of $\mathbf{D}$ i.i.d. from the standard Gaussian distribution, $\mathbf{D}_{ij} \sim N(0, 1/N)$ and then normalize its columns to have the unit ℓ_2 norm. We fix a matrix $\mathbf{D}$ in each setting where different networks are compared. To generate sparse vectors $\mathbf{x}^*$, we decide each of its entry to be non-zero following the Bernoulli distribution with $p_b = 0.1$. The values of the non-zero entries are sampled from the standard Gaussian distribution. A test set of 1000 samples generated in the above manner is fixed for all tests in our simulations.

All the networks have $K = 16$ layers. In LISTA models with support selection, we add p% of entries into support and maximally select $p_{\max}$% in each layer. We manually tune the value of p and $p_{\max}$ for the best final performance. With $p_b = 0.1$ and $K = 16$, we choose $p = 1.2$ for all models in simulation experiments and $p_{\max} = 12$ for LISTA-SS but $p_{\max} = 13$ for LISTA-CPSS. The recovery performance is evaluated by NMSE (in dB):

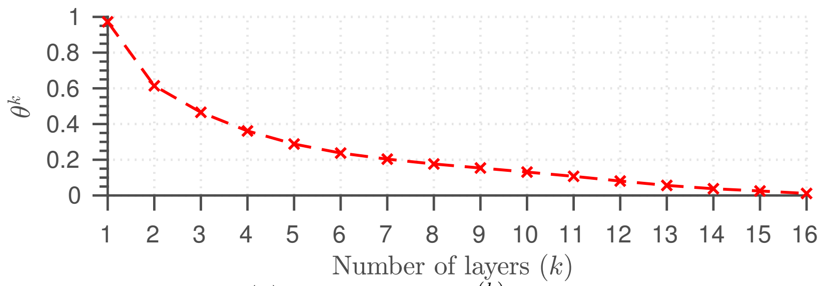
$$\mathrm{NMSE}(\hat{\mathbf{x}}, \mathbf{x}^*) = 10 \log_{10} \left(\frac{\mathbb{E}\|\hat{\mathbf{x}} - \mathbf{x}^*\|^2}{\mathbb{E}\|\mathbf{x}^*\|^2} \right),$$

where $\mathbf{x}^*$ is the ground truth and $\hat{\mathbf{x}}$ is the estimate obtained by the recovery algorithms (ISTA, FISTA, LISTA, etc.).

Validation of Theorem 2. In Fig 3.2, we report two values, $\|\mathbf{W}_2^{(k)} - (\mathbf{I} - \mathbf{W}_1^{(k)}\mathbf{D})\|_2$ and $\theta^{(k)}$, obtained by the baseline LISTA model (3.4) trained under the noiseless setting. The plot clearly demonstrates that $\mathbf{W}_2^{(k)} \to \mathbf{I} - \mathbf{W}_1^{(k)}\mathbf{D}$, and $\theta^{(k)} \to 0$, as $k \to \infty$. Theorem 2 is directly validated.



(a) Weight $\mathbf{W}_2^{(k)} \to \mathbf{I} - \mathbf{W}_1^{(k)}\mathbf{D}$ as $k \to \infty$.



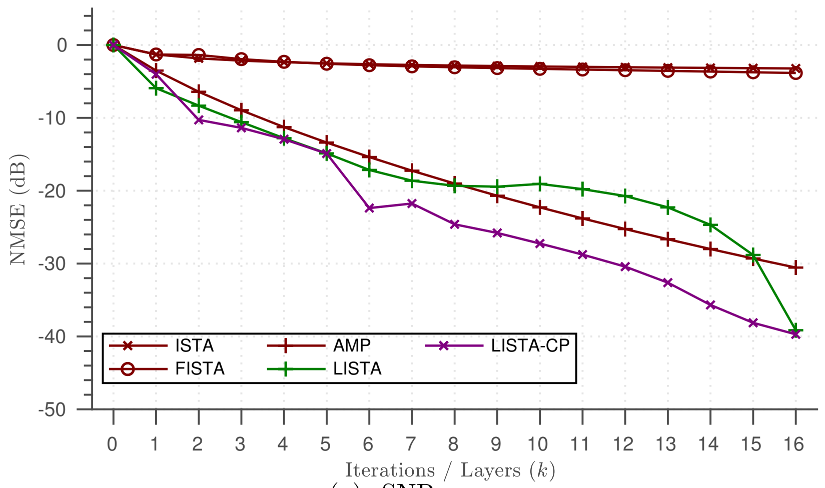
(b) The threshold $\theta^{(k)} \to 0$.

Figure 3.2: Validation of Theorem 2.

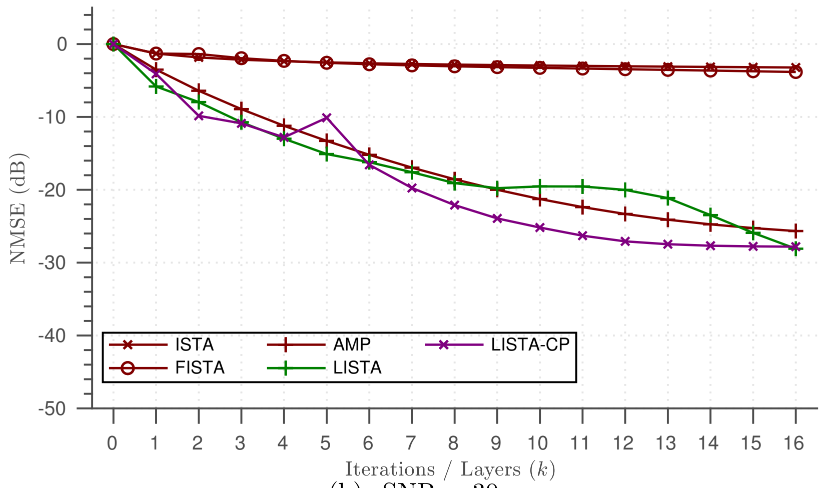
Validation of Theorem 3. We report the test-set NMSE of LISTA-CP (3.10) in Fig. 3.3. Although (3.10) fixes the structure between $\mathbf{W}_1^{(k)}$ and $\mathbf{W}_2^{(k)}$, the final performance remains the same with the baseline LISTA (3.4), and outperforms AMP, in both noiseless and noisy cases. Moreover, the output of interior layers in LISTA-CP are even better than the baseline LISTA. In the noiseless case, NMSE converges exponentially to 0; in the noisy case, NMSE converges to a stationary level related with the noise-level. This supports Theorem 3: there indeed exist a sequence of parameters $\{\mathbf{W}^{(k)}, \theta^{(k)}\}_{k=0}^{(K-1)}$ leading to linear convergence for LISTA-CP, and they can be obtained by data-driven learning.

Validation of Discussion after Theorem 3. In Fig 3.4, We compare LISTA-CP and ISTA with different λs (see the LASSO problem (3.2)) as well as an adaptive threshold rule similar to one in [HYZ08], which is described in Algorithm 4.

As we have discussed after Theorem 3, LASSO has an inherent tradeoff based on the choice of λ. A smaller λ leads to a more accurate solution but slower convergence. The



(a) SNR $= \infty$



(b) SNR $= 30$

Figure 3.3: Validation of Theorem 3.

adaptive thresholding rule fixes this issue: it uses large $\lambda^{(k)}$ for small k, and gradually reduces it as k increases to improve the accuracy [HYZ08]. Except for adaptive thresholds $\{\theta^{(k)}\}_k$ ($\theta^{(k)}$ corresponds to $\lambda^{(k)}$ in LASSO), LISTA-CP has adaptive weights $\{\mathbf{W}^{(k)}\}_k$, which further greatly accelerate the convergence. Note that we only ran ISTA and FISTA for 16 iterations, just enough and fair to compare them with the learned models. The number of iterations is so small that the difference between ISTA and FISTA is not quite observable.

Validation of Theorem 4. We compare the recovery NMSEs of LISTA-CP (3.10) and LISTA-CPSS (3.15) in Fig. 3.5. The result of the noiseless case (Fig. 3.5(a)) shows that

Algorithm 4: A thresholding rule for LASSO (Similar to that in [HYZ08])

```
  Input          : Maximum iteration $K$, initial $\lambda^{(0)} = 0.2, \epsilon^{(0)} = 0.05$.
  Initialization : Let $\mathbf{x}^{(0)} = 0, \lambda^{(1)} = \lambda^{(0)}, \epsilon^{(1)} = \epsilon^{(0)}$.
1 for $k = 1, 2, \cdots, K$ do
2     Conduct ISTA: $\mathbf{x}^{(k)} = \eta_{\lambda^{(k)}/L}\Big(\mathbf{x}^{(k-1)} - \frac{1}{L}\mathbf{D}^T(\mathbf{D}\mathbf{x}^{(k-1)} - \mathbf{b})\Big)$.
3     if $\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| < \epsilon^{(k)}$ then
4         Let $\lambda^{(k+1)} \leftarrow 0.5\lambda^{(k)}$, $\epsilon^{(k+1)} \leftarrow 0.5\epsilon^{(k)}$.
5     else
6         Let $\lambda^{(k+1)} \leftarrow \lambda^{(k)}$, $\epsilon^{(k+1)} \leftarrow \epsilon^{(k)}$.
7     end
8 end
  Output: $\mathbf{x}^{(K)}$
```

the recovery error of LISTA-SS converges to 0 at a faster rate than that of LISTA-CP. The difference is significant with the number of layers $k \geq 10$, which supports our theoretical result: "$\tilde{c}_{\text{ss}}^{(k)} > c$ as k large enough" in Theorem 4. The result of the noisy case (Fig. 3.5(b)) shows that LISTA-CPSS has better recovery error than LISTA-CP. This point supports $\tilde{C}_{\text{ss}} < C$ in Theorem 4. Notably, LISTA-CPSS also outperforms LAMP [BSR17], when $k > 10$ in the noiseless case, and even earlier as SNR becomes lower.

Validation of Theorems 5 and 6 In Figure 3.6 (a) noise-less case, all four learned models apparently converge much faster than two iterative solvers (ISTA/FISTA curves almost overlap in this y-scale, at the small number of iterations). Among the four networks, classical-LISTA is inferior to the other three by an obvious margin. LISTA-CPSS, TiLISTA and ALISTA perform comparably: ALISTA is observed to eventually achieve the lowest NMSE. Figure 3.6(a) also supports Theorem 6, that all networks have at most linear convergence, regardless of how freely their parameters can be end-to-end learned.

Figure 3.6 (b) - (d) further show that even in the presence of noise, ALISTA can empirically perform comparably with LISTA-CPSS and TiLISTA, and stay clearly better than LISTA

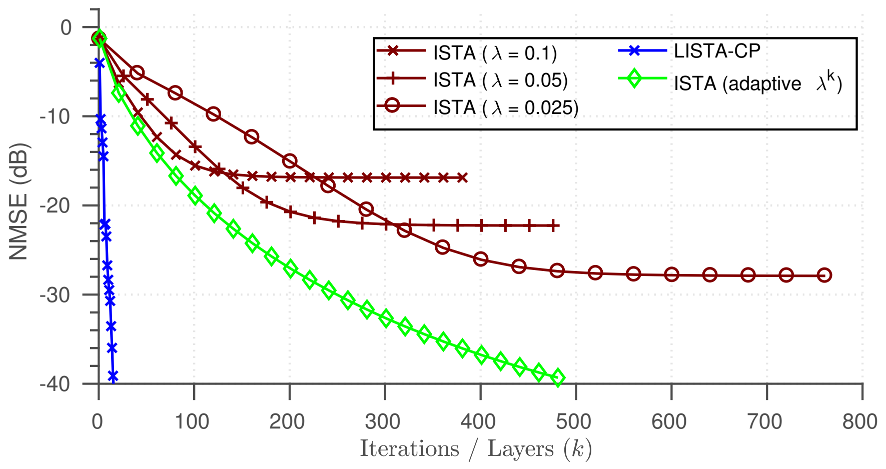


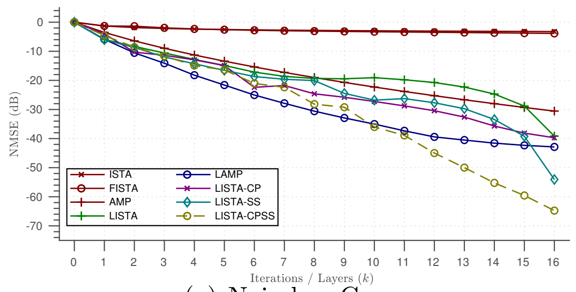
Figure 3.4: Validating Discussion after Theorem 3 (SNR $= \infty$).

and ISTA/FISTA. Always note that ALISTA the smallest amount of parameters to learn from the end-to-end training (Stage 2). The above results endorse that:

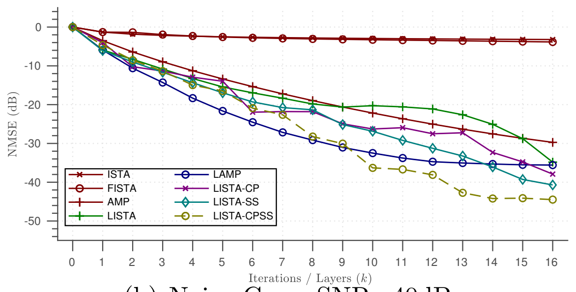
- The optimal LISTA layer-wise weights could be structured as $\mathbf{W}^{(k)} = \gamma^{(k)}\mathbf{W}$.
- $\mathbf{W}$ could be analytically solved rather than learned from data, without incurring performance loss.

We also observe the significant reduction of training time for ALISTA: while LISTA-CPSS of the same depth took ~1.5 hours to train, ALISTA was trained within only 6 minutes (0.1 hours) to achieve comparable performance, on the same hardware (one 1080 Ti on server).

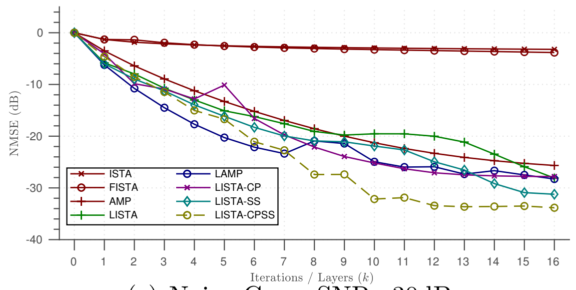
We further supply Figures 3.7 and 3.8 to justify Theorem 5 from different perspectives. Figure 3.7 plots the learned parameters $\{\gamma^{(k)}, \theta^{(k)}\}$ in ALISTA (Stage 2), showing that they satisfy the properties proposed in Theorem 5: $\gamma^{(k)}$ bounded; $\theta^{(k)}$ and $\gamma^{(k)}$ is proportional to $\sup_{\mathbf{x}^*} \|\mathbf{x}^{(k)}(\mathbf{x}^*) - \mathbf{x}^*\|_1$ ("$\sup_{\mathbf{x}^*}$" is taken over the test set). Figure 3.8 reports the average magnitude of the false positives and the true positives in $\mathbf{x}^{(k)}(\mathbf{x}^*)$ of ALISTA: the "true positives" curve draws the values of $\mathbb{E}\{\|\mathbf{x}^{(k)}_{\mathbb{S}}(\mathbf{x}^*)\|_2^2/\|\mathbf{x}^{(k)}(\mathbf{x}^*)\|_2^2\}$ w.r.t. k (the expectation is taken over the test set), while "false positives" for $\mathbb{E}\{\|\mathbf{x}^{(k)}_{\mathbb{S}^c}(\mathbf{x}^*)\|_2^2/\|\mathbf{x}^{(k)}(\mathbf{x}^*)\|_2^2\}$. False positives take up small proportion over the positives, which supports the Theorem 5 conclusion that $\text{support}(\mathbf{x}^{(k)}(\mathbf{x}^*)) \subset \mathbb{S}$.



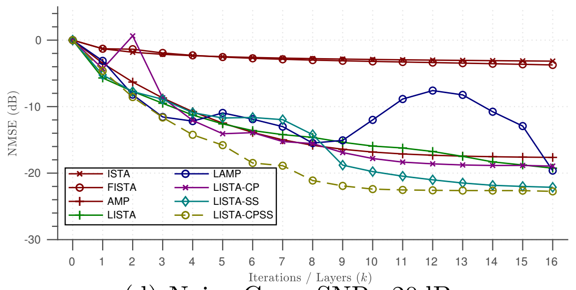
(a) Noiseless Case



(b) Noisy Case: SNR=40dB



(c) Noisy Case: SNR=30dB



(d) Noisy Case: SNR=20dB

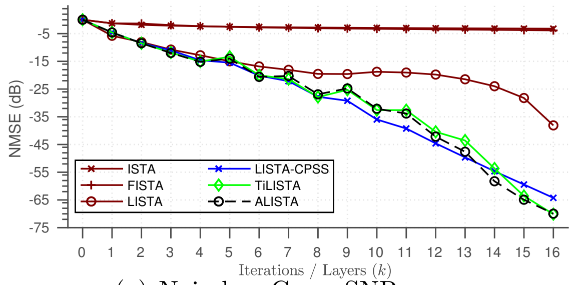
Figure 3.5: Validation of Theorem 4.

The number and proportion of false alarms are a more straightforward performance metric. However, they are sensitive to the threshold. We found that, although using a smaller threshold leads to more false alarms, the final recovery quality is better and those false alarms have small magnitudes and are easy to remove by thresholding during post-processing. That's why we chose to show their magnitudes, implying that we get easy-to-remove false alarms.

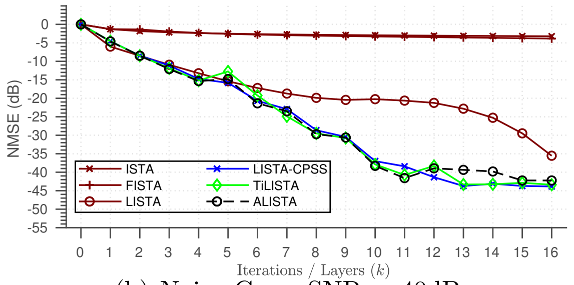
3.6.3 Convolutional Analytic LISTA

Validation of Theorem 7 For convolutional cases, we use real image data to verify Theorem 7. We train a convolutional dictionary $\mathbf{d}$ with $D = 7, M = 64$ on the BSD500 training set (400 images), using the Algorithm 1 in [LGW18]. We then use it for problems (3.33) and (3.35) and solve them with different Ns.

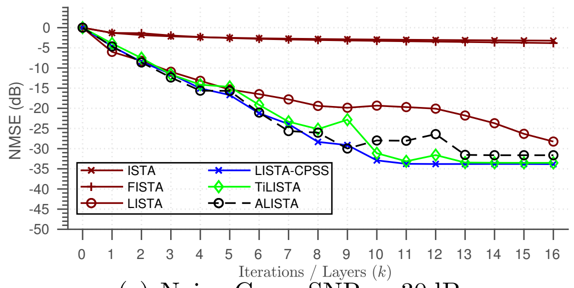
In Table 3.2, we take $\mathbf{w}_{\text{cir}}^N \in \mathcal{W}_{\text{cir}}^N$, $\mathbf{w}^* \in \mathcal{W}_{\text{cir}}^{50}$ (consider 50 as large enough) For this example, $\mathcal{W}_{\text{cir}}^N$ has only one element. Table 3.2 shows that $\mathbf{w}_{\text{cir}}^N = \mathbf{w}^*$ for $N \geq 13$, i.e., the solution of the problem (3.35) is independent of N if $N \geq 2D - 1$, justifying the first conclusion in Theorem 7. In Table 3.3, we take $\mathbf{w}_{\text{conv}}^N \in \mathcal{W}_{\text{conv}}^N$ and $\mathbf{w}^* \in \mathbf{w}_{\text{cir}}^{13}$, where $\mathcal{W}_{\text{conv}}^N$



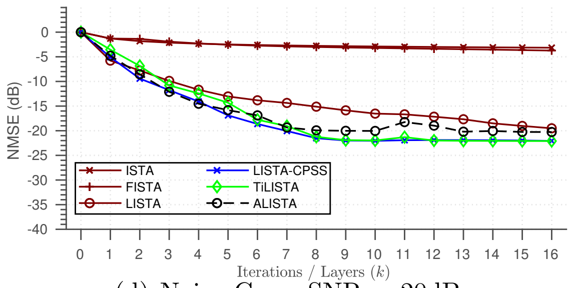
(a) Noiseless Case: SNR = ∞



(b) Noisy Case: SNR = 40dB



(c) Noisy Case: SNR = 30dB



(d) Noisy Case: SNR = 20dB

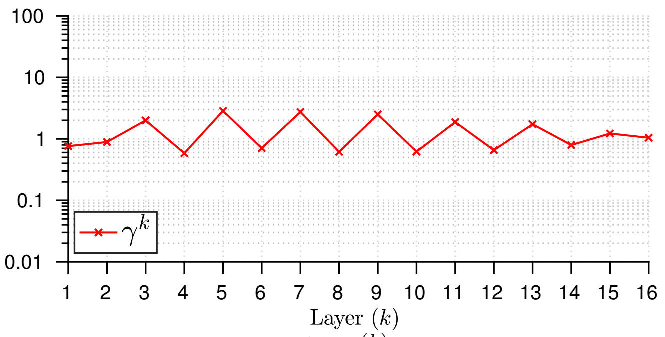
Figure 3.6: Justification of Theorems 5 and 6: comparision among LISTA variants.

also has only one element. Table 3.3 shows $\mathbf{w}^N_{\text{conv}} \to \mathbf{w}^*$, i.e., the solution of the problem (3.33) converges to that of (3.35) as N increases, validating the second conclusion of Theorem 7. Visualized $\mathbf{w}^* \in \mathbf{w}^{13}_{\text{cir}}$ is displayed in Appendix 3.6.3.

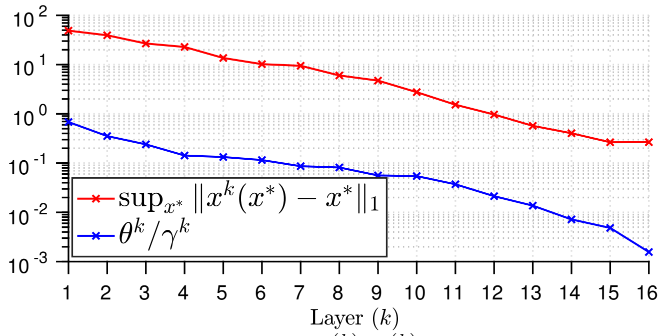
Table 3.2: Validation of Conclusion 1 in Theorem 7

$\lVert\mathbf{w}^N_{\text{cir}} - \mathbf{w}^*\rVert^2/\lVert\mathbf{w}^*\rVert^2$ (We take $\mathbf{w}^N_{\text{cir}} \in \mathcal{W}^N_{\text{cir}}$ and $\mathbf{w}^* \in \mathcal{W}^{50}_{\text{cir}}$)					
$N = 10$	$N = 11$	$N = 12$	$N = 13$	$N = 15$	$N = 20$
2.0×10^{-2}	9.3×10^{-3}	3.9×10^{-3}	1.4×10^{-12}	8.8×10^{-13}	5.9×10^{-13}

Visualization of the analytic convolutional weights Besides validating Theorem 7, we also present a real image denoising experiment to verify the effectiveness of Conv ALISTA. Fig. 3.9 visualizes the dictionary $\mathbf{d}$ ($7 \times 7 \times 64$) and the weights $\tilde{\mathbf{w}} \in \mathcal{W}^{13}_{\text{cir}}$, used in the convolutional A-LISTA simulation of Section 3.6.3. It is obtained by Algorithm 6 in Appendix 3.H.2. Kernel $\tilde{\mathbf{w}}$ keeps the high-frequency texture in $\mathbf{d}$. The support of $\tilde{\mathbf{w}}$ is small, most of the pixels in $\tilde{\mathbf{w}}$ are zeros. Then the coherence between shifted $\mathbf{d}$ and $\tilde{\mathbf{w}}$ is nearly 0.



(a) $\gamma^{(k)}$



(b) $\theta^{(k)}/\gamma^{(k)}$

Figure 3.7: Justification of Theorem 5 (noiseless case)

Table 3.3: Validation of Conclusion 2 in Theorem 7

$\|\mathbf{w}^N_{\text{conv}} - \mathbf{w}^*\|^2/\|\mathbf{w}^*\|^2$ (We take $\mathbf{w}^N_{\text{conv}} \in \mathcal{W}^N_{\text{conv}}$ and $\mathbf{w}^* \in \mathbf{w}^{13}_{\text{cir}}$)				
$N = 3$	$N = 5$	$N = 10$	$N = 15$	$N = 20$
0.1892	0.0850	0.0284	0.0161	0.0113

3.7 Conclusions

Based on the recent theoretical advances of LISTA, we have made further steps to reduce the training complexity and improve the robustness of LISTA. Specifically, we no longer train any matrix for LISTA but directly use the solution to an analytic minimization problem to solve for its layer-wise weights. Therefore, only two scalar sequences (stepsizes and thresholds) still need to be trained. Excluding the matrix from training is backed by our theoretical upper and lower bounds. The resulting method, Analytic LISTA or ALISTA, is not only faster to

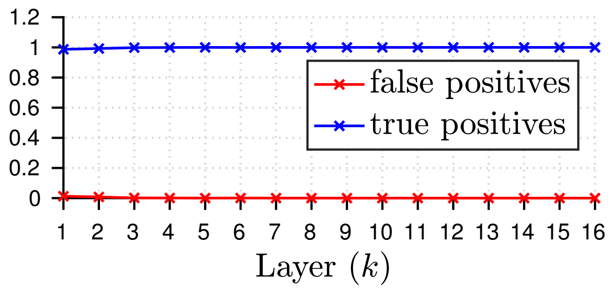
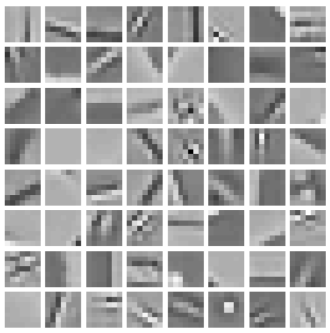
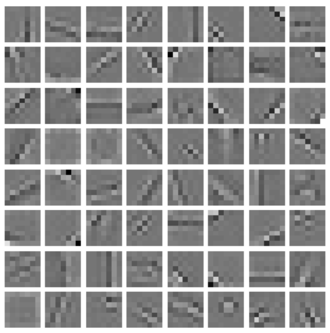


Figure 3.8: Justification of Theorem 5 (noiseless case)



(a) The dictionary $\mathbf{d}$.



(b) The $\tilde{\mathbf{w}}$ obtained by solving (3.35).

Figure 3.9: A visualization of convolutional kernels $\mathbf{d}$ and $\tilde{\mathbf{w}}$.

train but performs as well as the state-of-the-art.

Appendix 3.A Proof of Theorem 2

Proof. By LISTA model (3.4), the output of the k-th layer $\mathbf{x}^{(k)}$ depends on parameters, observed signal $\mathbf{b}$ and initial point $\mathbf{x}^{(0)}$: $\mathbf{x}^{(k)}\Big(\{\mathbf{W}_1^{(\tau)}, \mathbf{W}_2^{(\tau)}, \theta^{(\tau)}\}_{\tau=0}^{k-1}, \mathbf{b}, \mathbf{x}^{(0)}\Big)$. Since we assume $(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, 0)$, the noise $\varepsilon = 0$. Moreover, $\mathbf{D}$ is fixed and $\mathbf{x}^{(0)}$ is taken as 0. Thus, $\mathbf{x}^{(k)}$ therefore depends on parameters and $\mathbf{x}^*$: $\mathbf{x}^{(k)}\Big(\{\mathbf{W}_1^{(\tau)}, \mathbf{W}_2^{(\tau)}, \theta^{(\tau)}\}_{\tau=0}^{k-1}, \mathbf{x}^*\Big)$ In this proof, for simplicity, we use $\mathbf{x}^{(k)}$ denote $\mathbf{x}^{(k)}\Big(\{\mathbf{W}_1^{(\tau)}, \mathbf{W}_2^{(\tau)}, \theta^{(\tau)}\}_{\tau=0}^{k-1}, \mathbf{x}^*\Big)$.

Step 1 Firstly, we prove $\theta^{(k)} \to 0$ as $k \to \infty$.

We define a subset of $\mathcal{X}(B,s,0)$ given $0<\tilde{B}\le B$:

$$\tilde{\mathcal{X}}(B,\tilde{B},s,0)\triangleq\Big\{(\mathbf{x}^*,\varepsilon)\Big|\tilde{B}\le|x_i^*|\le B,\forall i,\ \|\mathbf{x}^*\|_0\le s,\varepsilon=0\Big\}\subset\mathcal{X}(B,s,0).$$

Since $\mathbf{x}^{(k)}\to\mathbf{x}^*$ uniformly for all $(\mathbf{x}^*,0)\in\mathcal{X}(B,s,0)$, so does for all $(\mathbf{x}^*,0)\in\tilde{\mathcal{X}}(B,B/10,s,0)$. Then there exists a uniform $K_1>0$ for all $(\mathbf{x}^*,0)\in\tilde{\mathcal{X}}(B,B/10,s,0)$, such that $|x_i^{(k)}-x_i^*|<B/10$ for all $i=1,2,\cdots,n$ and $k\ge K_1$, which implies

$$\operatorname{sign}(\mathbf{x}^{(k)})=\operatorname{sign}(\mathbf{x}^*),\quad\forall k\ge K_1.\tag{3.38}$$

The relationship between $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k+1)}$ is

$$\mathbf{x}^{(k+1)}=\eta_{\theta^{(k)}}\Big(\mathbf{W}_2^{(k)}\mathbf{x}^{(k)}+\mathbf{W}_1^{(k)}\mathbf{b}\Big).$$

Let $\mathbb{S}=\operatorname{support}(\mathbf{x}^*)$. Then, (3.38) implies that, for any $k\ge K_1$ and $(\mathbf{x}^*,0)\in\tilde{\mathcal{X}}(B,B/10,s,0)$, we have

$$\mathbf{x}_{\mathbb{S}}^{(k+1)}=\eta_{\theta^{(k)}}\Big(\mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S})\mathbf{x}_{\mathbb{S}}^{(k)}+\mathbf{W}_1^{(k)}(\mathbb{S},:)\mathbf{b}\Big).$$

The fact (3.38) means $x_i^{(k+1)}\ne0,\forall i\in\mathbb{S}$. By the definition of $\eta_\theta(\mathbf{x})$:

$$\eta_\theta(\mathbf{x})=\operatorname{sign}(\mathbf{x})\max(0,|\mathbf{x}|-\theta),$$

as long as $\eta_\theta(\mathbf{x})_i\ne0$, we have $\eta_\theta(\mathbf{x})_i=x_i-\theta\operatorname{sign}(x_i)$. Thus,

$$\mathbf{x}_{\mathbb{S}}^{(k+1)}=\mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S})\mathbf{x}_{\mathbb{S}}^{(k)}+\mathbf{W}_1^{(k)}(\mathbb{S},:)\mathbf{b}-\theta^{(k)}\operatorname{sign}(\mathbf{x}_{\mathbb{S}}^*).$$

Furthermore, the uniform convergence of $\mathbf{x}^{(k)}$ tells us, for any $\epsilon>0$ and

$$(\mathbf{x}^*,0)\in\tilde{\mathcal{X}}(B,B/10,s,0),$$

there exists a large enough constant $K_2>0$ and $\xi_1,\xi_2\in\mathbb{R}^{|\mathbb{S}|}$ such that $\mathbf{x}_{\mathbb{S}}^{(k)}=\mathbf{x}_{\mathbb{S}}^*+\xi_1,\mathbf{x}_{\mathbb{S}}^{(k+1)}=\mathbf{x}_{\mathbb{S}}^*+\xi_2$ and $\|\xi_1\|_2\le\epsilon$, $\|\xi_2\|_2\le\epsilon$. Then

$$\mathbf{x}_{\mathbb{S}}^*+\xi_2=\mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S})(\mathbf{x}_{\mathbb{S}}^*+\xi_1)+\mathbf{W}_1^{(k)}(\mathbb{S},:)\mathbf{b}-\theta^{(k)}\operatorname{sign}(\mathbf{x}_{\mathbb{S}}^*).$$

Since the noise is supposed to be zero $\varepsilon=0$, $\mathbf{b}=\mathbf{D}\mathbf{x}^*$. Substituting $\mathbf{b}$ with $\mathbf{D}\mathbf{x}^*$ in the above equality, we obtain

$$\mathbf{x}_{\mathbb{S}}^*=\mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S})\mathbf{x}_{\mathbb{S}}^*+\mathbf{W}_1^{(k)}(\mathbb{S},:)\mathbf{D}(:,\mathbb{S})\mathbf{x}_{\mathbb{S}}^*-\theta^{(k)}\operatorname{sign}(\mathbf{x}_{\mathbb{S}}^*)+\xi,$$

where $\|\xi\|_2 = \|\mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S})\xi_1 - \xi_2\|_2 \le (1+B_W)\epsilon$, B_W is defined in Theorem 2. Equivalently,

$$\Big(I - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S})\Big)\mathbf{x}^*_{\mathbb{S}} = \theta^{(k)}\ \mathrm{sign}(\mathbf{x}^*_{\mathbb{S}}) - \xi. \tag{3.39}$$

For any $(\mathbf{x}^*, 0) \in \tilde{\mathcal{X}}(B/2, B/10, s, 0)$, $(2\mathbf{x}^*, 0) \in \tilde{\mathcal{X}}(B, B/10, s, 0)$ holds. Thus, the above argument holds for all $2\mathbf{x}^*$ if $(\mathbf{x}^*, 0) \in \tilde{\mathcal{X}}(B/2, B/10, s, 0)$. Substituting $\mathbf{x}^*$ with $2\mathbf{x}^*$ in (3.39), we get

$$\Big(I - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S})\Big)2\mathbf{x}^*_{\mathbb{S}} = \theta^{(k)}\ \mathrm{sign}(2\mathbf{x}^*_{\mathbb{S}}) - \xi' = \theta^{(k)}\ \mathrm{sign}(\mathbf{x}^*_{\mathbb{S}}) - \xi', \tag{3.40}$$

where $\|\xi'\|_2 \le (1+B_W)\epsilon$. Taking the difference between (3.39) and (3.40), we have

$$\Big(I - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S})\Big)\mathbf{x}^*_{\mathbb{S}} = -\xi' + \xi. \tag{3.41}$$

Equations (3.39) and (3.41) imply

$$\theta^{(k)}\ \mathrm{sign}(\mathbf{x}^*_{\mathbb{S}}) - \xi = -\xi' + \xi.$$

Then $\theta^{(k)}$ can be bounded with

$$\theta^{(k)} \le \frac{3(1+B_W)}{\sqrt{|\,\mathbb{S}\,|}}\epsilon, \quad \forall k \ge \max(K_1, K_2). \tag{3.42}$$

The above conclusion holds for all $|\,\mathbb{S}\,| \ge 1$. Moreover, as a threshold in η_θ, $\theta^{(k)} \ge 0$. Thus, $0 \le \theta^{(k)} \le 3(1+B_W)\epsilon$ for any $\epsilon > 0$ as long as k large enough. In another word, $\theta^{(k)} \to 0$ as $k \to \infty$.

Step 2 We prove that $\mathbf{I} - \mathbf{W}_2^{(k)} - \mathbf{W}_1^{(k)}\mathbf{D} \to 0$ as $k \to \infty$.

LISTA model (3.4) and $\mathbf{b} = \mathbf{D}\mathbf{x}^*$ gives

$$\begin{aligned}
\mathbf{x}_{\mathbb{S}}^{(k+1)} =& \eta_{\theta^{(k)}}\Big(\mathbf{W}_2^{(k)}(\mathbb{S},:)\mathbf{x}^{(k)} + \mathbf{W}_1^{(k)}(\mathbb{S},:)\mathbf{b}\Big)\\
=& \eta_{\theta^{(k)}}\Big(\mathbf{W}_2^{(k)}(\mathbb{S},:)\mathbf{x}^{(k)} + \mathbf{W}_1^{(k)}(\mathbb{S},:)\mathbf{D}(:,\mathbb{S})\mathbf{x}^*_{\mathbb{S}}\Big)\\
\in& \mathbf{W}_2^{(k)}(\mathbb{S},:)\mathbf{x}^{(k)} + \mathbf{W}_1^{(k)}(\mathbb{S},:)\mathbf{D}(:,\mathbb{S})\mathbf{x}^*_{\mathbb{S}} - \theta^{(k)}\partial\ell_1(\mathbf{x}_{\mathbb{S}}^{(k+1)}),
\end{aligned}$$

where $\partial\ell_1(x)$ is the sub-gradient of $\|\mathbf{x}\|_1$. It is a set defined component-wisely:

$$\partial\ell_1(x)_i = \begin{cases} \{\mathrm{sign}(x_i)\} & \text{if } x_i \ne 0, \\ [-1, 1] & \text{if } x_i = 0. \end{cases} \tag{3.43}$$

The uniform convergence of $\mathbf{x}^{(k)}$ implies, for any $\epsilon > 0$ and $(\mathbf{x}^*, 0) \in \mathcal{X}(B, s, 0)$, there exists a large enough constant $K_3 > 0$ and $\xi_1, \xi_2 \in \mathbb{R}^M$ such that $\mathbf{x}^{(k)} = \mathbf{x}^* + \xi_3, \mathbf{x}^{(k+1)} = \mathbf{x}^* + \xi_4$ and $\|\xi_3\|_2 \leq \epsilon, \|\xi_4\|_2 \leq \epsilon$. Thus,

$$x^*_{\mathbb{S}} + (\xi_4)_{\mathbb{S}} \in \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S})\mathbf{x}^*_{\mathbb{S}} + \mathbf{W}_2^{(k)}(\mathbb{S},:)\xi_3 + \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S})\mathbf{x}^*_{\mathbb{S}} - \theta^{(k)}\partial\ell_1(\mathbf{x}_{\mathbb{S}}^{(k+1)})$$

$$\Big(\mathbf{I} - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S})\Big)x^*_{\mathbb{S}} \in \mathbf{W}_2^{(k)}(\mathbb{S},:)\xi_3 - (\xi_4)_{\mathbb{S}} - \theta^{(k)}\partial\ell_1(\mathbf{x}_{\mathbb{S}}^{(k+1)})$$

By the definition (3.43) of $\partial\ell_1$, every element in $\partial\ell_1(x), \forall x \in \mathbb{R}$ has a magnitude less than or equal to 1. Thus, for any $\xi \in \ell_1(\mathbf{x}_{\mathbb{S}}^{(k+1)})$, we have $\|\xi\|_2 \leq \sqrt{|\mathbb{S}|}$, which implies

$$\Big\|\Big(\mathbf{I} - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S})\Big)x^*_{\mathbb{S}}\Big\|_2 \leq \|\mathbf{W}_2^{(k)}\|_2\epsilon + \epsilon + \theta^{(k)}\sqrt{|\mathbb{S}|}.$$

Combined with (3.42), we obtain the following inequality for all $k \geq \max(K_1, K_2, K_3)$:

$$\Big\|\Big(\mathbf{I} - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S})\Big)x^*_{\mathbb{S}}\Big\|_2 \leq \|\mathbf{W}_2^{(k)}\|_2\epsilon + \epsilon + 3(1 + B_W)\epsilon = 4(1 + B_W)\epsilon.$$

The above inequality holds for all $(\mathbf{x}^*, 0) \in \mathcal{X}(B, s, 0)$, which implies,

$$\begin{aligned}
&\sigma_{\max}\Big(\mathbf{I} - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S})\Big) \\
=& \sup_{\substack{\text{support}(\mathbf{x}^*)=S \\ \|x_i^*\|_2=B}} \left\{\frac{\|(\mathbf{I} - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S}))\mathbf{x}^*_{\mathbb{S}}\|_2}{B}\right\} \\
\leq& \sup_{(\mathbf{x}^*,0)\in\mathcal{X}(B,s,0)} \left\{\frac{\|(\mathbf{I} - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S}))\mathbf{x}^*_{\mathbb{S}}\|_2}{B}\right\} \\
\leq& \frac{4(1+B_W)}{B}\epsilon.
\end{aligned}$$

for all $k \geq \max(K_1, K_2, K_3)$. Since $s \geq 2$, $\mathbf{I} - \mathbf{W}_2^{(k)}(\mathbb{S},\mathbb{S}) - \mathbf{W}_1^{(k)}\mathbf{D}(\mathbb{S},\mathbb{S}) \to 0$ uniformly for all S with $2 \leq |\mathbb{S}| \leq s$. Then, $\mathbf{I} - \mathbf{W}_2^{(k)} - \mathbf{W}_1^{(k)}\mathbf{D} \to 0$ as $k \to \infty$. $\square$

Appendix 3.B Proof of Theorem 3

Before proving Theorem 3, we introduce a lemma that tells us the generalized mutual coherence is attached at some $\tilde{\mathbf{W}} \in \mathbb{R}^{N\times M}$.

Lemma 2. *There exists a matrix $\widetilde{\mathbf{W}} \in \mathbb{R}^{N\times M}$ that attaches the infimum given in (3.19).*

Proof. Optimization problem given in (3.19) is a linear programming because it minimizing a piece-wise linear function with linear constraints. Since each column of $\mathbf{D}$ is normalized, there is at least one matrix in the feasible set:

$$\mathbf{D} \in \{\mathbf{W} \in \mathbb{R}^{N\times M} : (\mathbf{W}_{:,i})^T\mathbf{D}_{:,i} = 1, 1 \leq i \leq M\}.$$

In another word, optimization problem (3.19) is feasible. Moreover, by the definition of infimum bound (3.19), we have

$$0 \leq \tilde{\mu}(\mathbf{D}) \leq \max_{\substack{i\neq j\\ 1\leq i,j\leq M}} |(\mathbf{D}_{:,i})^\top \mathbf{D}_{:,j}| = \mu(\mathbf{D}).$$

Thus, $\tilde{\mu}$ is bounded. According to Corollary 2.3 in [BT97], a feasible and bounded linear programming problem has an optimal solution. Thus $\mathcal{W}(\mathbf{D})$ is nonempty. □

With definition (3.19), we propose a choice of parameters:

$$\mathbf{W}^{(k)} \in \mathcal{W}(\mathbf{D}), \quad \theta^{(k)} = \sup_{(\mathbf{x}^*,\varepsilon)\in\mathcal{X}(B,s,\sigma)} \{\tilde{\mu}\|\mathbf{x}^{(k)}(\mathbf{x}^*,\varepsilon) - \mathbf{x}^*\|_1\} + C_W\sigma, \tag{3.44}$$

which are uniform for all $(\mathbf{x}^*,\varepsilon) \in \mathcal{X}(B,s,\sigma)$. In the following proof line, we prove that (3.44) leads to the conclusion (3.11) in Theorem 3.

Proof of Theorem 3

Proof. In this proof, we use the notation $\mathbf{x}^{(k)}$ to replace $\mathbf{x}^{(k)}(\mathbf{x}^*,\varepsilon)$ for simplicity.

Step 1: no false positives. Firstly, we take $(\mathbf{x}^*,\varepsilon) \in \mathcal{X}(B,s,\sigma)$. Let $\mathbb{S} = \text{support}(\mathbf{x}^*)$. We want to prove by induction that, as long as (3.44) holds, $x_i^{(k)} = 0, \forall i \notin \mathbb{S}, \forall k$ (no false positives). When $k = 0$, it is satisfied since $\mathbf{x}^{(0)} = 0$. Fixing k, and assuming $x_i^{(k)} = 0, \forall i \notin \mathbb{S}$, we have

$$\begin{aligned} x_i^{(k+1)} =& \eta_{\theta^{(k)}}\Big(x_i^{(k)} - \sum_{j\in S}(\mathbf{W}_{:,i}^{(k)})^T(\mathbf{D}\mathbf{x}^{(k)} - \mathbf{b})\Big) \\ =& \eta_{\theta^{(k)}}\Big(- \sum_{j\in S}(\mathbf{W}_{:,i}^{(k)})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*) + (\mathbf{W}_i^{(k)})^T\varepsilon\Big), \quad \forall i \notin S. \end{aligned}$$

Since $\theta^{(k)} = \tilde{\mu} \sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1\} + C_W\sigma$ and $\mathbf{W}^{(k)} \in \mathcal{W}(\mathbf{D})$,

$$\theta^{(k)} \geq \tilde{\mu}\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1 + C_W\|\varepsilon\|_1 \geq \Big| - \sum_{j\in S}(\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*) + (\mathbf{W}^{(k)}_{:,i})^T\varepsilon\Big|, \forall i \notin S,$$

which implies $x_i^{(k+1)} = 0, \forall i \notin \mathbb{S}$ by the definition of $\eta_{\theta^{(k)}}$. By induction, we have

$$x_i^{(k)} = 0, \forall i \notin \mathbb{S}, \quad \forall k. \tag{3.45}$$

In another word, threshold rule in (3.44) ensures no false positives[9] for all $\mathbf{x}^{(k)}, k = 1, 2, \cdots$

Step 2: error bound for one $(\mathbf{x}^*, \varepsilon)$. Next, let's consider the components on $\mathbb{S}$. For all $i \in \mathbb{S}$,

$$\begin{aligned} x_i^{(k+1)} &= \eta_{\theta^{(k)}}\Big(x_i^{(k)} - (\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,\mathbb{S}}(\mathbf{x}^{(k)}_{\mathbb{S}} - \mathbf{x}^*_{\mathbb{S}}) + (\mathbf{W}^{(k)}_{:,i})^T\varepsilon\Big) \\ &\in x_i^{(k)} - (\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,\mathbb{S}}(\mathbf{x}^{(k)}_{\mathbb{S}} - \mathbf{x}^*_{\mathbb{S}}) + (\mathbf{W}^{(k)}_{:,i})^T\varepsilon - \theta^{(k)}\partial\ell_1(x_i^{(k+1)}), \end{aligned}$$

where $\partial\ell_1(x)$ is defined in (3.43). Since $(\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,i} = 1$, we have

$$\begin{aligned} x_i^{(k)} - (\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,\mathbb{S}}(\mathbf{x}^{(k)}_{\mathbb{S}} - \mathbf{x}^*_{\mathbb{S}}) =& x_i^{(k)} - \sum_{j\in S, j\neq i}(\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*) - (x_i^{(k)} - x_i^*) \\ =& x_i^* - \sum_{j\in\mathbb{S}, j\neq i}(\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*). \end{aligned}$$

Then,

$$x_i^{(k+1)} - x_i^* \in - \sum_{j\in\mathbb{S}, j\neq i}(\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*) + (\mathbf{W}^{(k)}_{:,i})^T\varepsilon - \theta^{(k)}\partial\ell_1(x_i^{(k+1)}), \quad \forall i \in \mathbb{S}.$$

By the definition (3.43) of $\partial\ell_1$, every element in $\partial\ell_1(x), \forall x \in \mathbb{R}$ has a magnitude less than or equal to 1. Thus, for all $i \in \mathbb{S}$,

$$\begin{aligned} |x_i^{(k+1)} - x_i^*| \leq& \sum_{j\in S, j\neq i}\Big|(\mathbf{W}^{(k)}_{:,i})^T\mathbf{D}_{:,j}\Big||x_j^{(k)} - x_j^*| + \theta^{(k)} + |(\mathbf{W}^{(k)}_{:,i})^T\varepsilon| \\ \leq& \tilde{\mu}\sum_{j\in S, j\neq i}|x_j^{(k)} - x_j^*| + \theta^{(k)} + C_W\|\varepsilon\|_1 \end{aligned}$$

[9] In practice, if we obtain $\theta^{(k)}$ by training, but not (3.44), the learned $\theta^{(k)}$ may not guarantee no false positives for all layers. However, the magnitudes on the false positives are actually small compared to those on true positives. Our proof sketch are qualitatively describing the learning-based ISTA.

Equation (3.45) implies $\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1 = \|\mathbf{x}^{(k)}_{\mathbb{S}} - \mathbf{x}^*_{\mathbb{S}}\|_1$ for all k. Then

$$
\begin{aligned}
\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1 = \sum_{i\in S} |x_i^{(k+1)} - x_i^*| \leq & \sum_{i\in S} \Big(\tilde{\mu} \sum_{j\in S, j\neq i} |x_j^{(k)} - x_j^*| + \theta^{(k)} + C_W\sigma \Big) \\
= & \tilde{\mu}(|\,\mathbb{S}\,| - 1) \sum_{i\in S} |x_i^{(k)} - x_i^*| + \theta^{(k)}|\,\mathbb{S}\,| + |\,\mathbb{S}\,| C_W\sigma \\
\leq & \tilde{\mu}(|\,\mathbb{S}\,| - 1)\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1 + \theta^{(k)}|\,\mathbb{S}\,| + |\,\mathbb{S}\,| C_W\sigma
\end{aligned}
$$

Step 3: error bound for the whole data set. Finally, we take supremum over $(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, x, \sigma)$, by $|\,\mathbb{S}\,| \leq s$,

$$
\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq \tilde{\mu}(s-1) \sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1\} + s\theta^{(k)} + sC_W\sigma.
$$

By $\theta^{(k)} = \sup_{\mathbf{x}^*,\varepsilon}\{\tilde{\mu}\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1\} + C_W\sigma$, we have

$$
\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq (2\tilde{\mu}s - \tilde{\mu}) \sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1\} + 2sC_W\sigma.
$$

By induction, with $c = -\log(2\tilde{\mu}s - \tilde{\mu}), C = \frac{2sC_W}{1+\tilde{\mu}-2\tilde{\mu}s}$, we obtain

$$
\begin{aligned}
\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq & (2\tilde{\mu}s - \tilde{\mu})^{k+1} \sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(0)} - \mathbf{x}^*\|_1\} + 2sC_W\sigma\Big(\sum_{\tau=0}^{k+1}(2\tilde{\mu}s - \tilde{\mu})^{(\tau)}\Big) \\
\leq & (2\tilde{\mu}s - \tilde{\mu})^k sB + C\sigma = sB\exp(-ck) + C\sigma.
\end{aligned}
$$

Since $\|x\|_2 \leq \|x\|_1$ for any $x \in \mathbb{R}^M$, we can get the upper bound for ℓ_2 norm:

$$
\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_2\} \leq \sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq sB\exp(-ck) + C\sigma.
$$

As long as $s < (1 + 1/\tilde{\mu})/2$, $c = -\log(2\tilde{\mu}s - \tilde{\mu}) > 0$, then the error bound (3.11) holds uniformly for all $(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, \sigma)$. $\square$

Appendix 3.C Proof of Theorem 4

Proof. In this proof, we use the notation $\mathbf{x}^{(k)}$ to replace $\mathbf{x}^{(k)}(\mathbf{x}^*, \varepsilon)$ for simplicity.

Step 1: proving (3.17). Firstly, we assume Assumption 1 holds. Take $(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, \sigma)$. Let $\mathbb{S} = \text{support}(\mathbf{x}^*)$. By the definition of selecting-support operator $\eta_{\text{ss}_{\theta^{(k)}}^{p^k}}$, using the same argument with the proof of Theorem 3, we have LISTA-CPSS also satisfies $x_i^{(k)} = 0, \forall i \notin \mathbb{S}, \forall k$ (no false positive) with the same parameters as (3.44).

For all $i \in \mathbb{S}$, by the definition of $\eta_{\mathrm{ss}_{\theta^{(k)}}^{p^k}}$, there exists $\xi^k \in \mathbb{R}^M$ such that

$$\begin{aligned} x_i^{(k+1)} =& \eta_{\mathrm{ss}_{\theta^{(k)}}^{p^k}}\Big(x_i^{(k)} - (\mathbf{W}_{:,i}^{(k)})^T \mathbf{D}_{:,\mathbb{S}}(\mathbf{x}_{\mathbb{S}}^{(k)} - \mathbf{x}_{\mathbb{S}}^*) + (\mathbf{W}_{:,i}^{(k)})^T \varepsilon\Big) \\ =& x_i^{(k)} - (\mathbf{W}_{:,i}^{(k)})^T \mathbf{D}_{:,\mathbb{S}}(\mathbf{x}_{\mathbb{S}}^{(k)} - \mathbf{x}_{\mathbb{S}}^*) + (\mathbf{W}_{:,i}^{(k)})^T \varepsilon - \theta^{(k)} \xi_i^k, \end{aligned}$$

where

$$\xi_i^k \begin{cases} = 0 & \text{if } i \notin \mathbb{S} \\ \in [-1, 1] & \text{if } i \in \mathbb{S}, x_i^{(k+1)} = 0 \\ = \mathrm{sign}(x_i^{(k+1)}) & \text{if } i \in \mathbb{S}, x_i^{(k+1)} \neq 0,\ i \notin \mathbb{S}^{p^k}(\mathbf{x}^{(k+1)}), \\ = 0 & \text{if } i \in \mathbb{S}, x_i^{(k+1)} \neq 0,\ i \in \mathbb{S}^{p^k}(\mathbf{x}^{(k+1)}). \end{cases}$$

The set $\mathbb{S}^{p^k}$ is defined in (3.14). Let

$$\mathbb{S}^k(\mathbf{x}^*, \varepsilon) = \{i | i \in \mathbb{S}, x_i^{(k+1)} \neq 0, i \in \mathbb{S}^{p^k}(\mathbf{x}^{(k+1)})\},$$

where $\mathbb{S}^k$ depends on $\mathbf{x}^*$ and ε because $\mathbf{x}^{(k+1)}$ depends on $\mathbf{x}^*$ and ε. Then, using the same argument with that of LISTA-CP (Theorem 3), we have

$$\|\mathbf{x}_{\mathbb{S}}^{(k+1)} - \mathbf{x}_{\mathbb{S}}^*\|_1 \leq \tilde{\mu}(|\,\mathbb{S}\,| - 1)\|\mathbf{x}_{\mathbb{S}}^{(k)} - \mathbf{x}_{\mathbb{S}}^*\|_1 + \theta^{(k)}\big(|\,\mathbb{S}\,| - |\,\mathbb{S}^k(\mathbf{x}^*, \varepsilon)|\big) + |\,\mathbb{S}\,| C_W \|\varepsilon\|_1.$$

Since $x_i^{(k)} = 0, \forall i \notin \mathbb{S}$, $\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_2 = \|\mathbf{x}_{\mathbb{S}}^{(k)} - \mathbf{x}_{\mathbb{S}}^*\|_2$ for all k. Taking supremum over $(\mathbf{x}^*, \varepsilon) \in \mathcal{X}(B, s, \sigma)$, we have

$$\sup_{\mathbf{x}^*,\varepsilon} \|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1 \leq (\tilde{\mu}s - 1) \sup_{\mathbf{x}^*,\varepsilon} \|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1 + \theta^{(k)}(s - \inf_{\mathbf{x}^*,\varepsilon} |\,\mathbb{S}^k(\mathbf{x}^*, \varepsilon)|) + sC_W\sigma.$$

By $\theta^{(k)} = \sup_{\mathbf{x}^*,\varepsilon}\{\tilde{\mu}\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1\} + C_W\sigma$, we have

$$\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq \Big(2\tilde{\mu}s - \tilde{\mu} - \tilde{\mu} \inf_{\mathbf{x}^*,\varepsilon} |\,\mathbb{S}^k(\mathbf{x}^*, \varepsilon)|\Big) \sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1\} + 2sC_W\sigma.$$

Let

$$\begin{aligned} c_{\mathrm{ss}}^k =& -\log\Big(2\tilde{\mu}s - \tilde{\mu} - \tilde{\mu} \inf_{\mathbf{x}^*,\varepsilon} |\,\mathbb{S}^k(\mathbf{x}^*, \varepsilon)|\Big) \\ C_{\mathrm{ss}} =& 2sC_W \sum_{k=0}^{\infty} \prod_{t=0}^{k} \exp(-c_{\mathrm{ss}}^t)). \end{aligned}$$

Then,

$$\begin{aligned}
&\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k)}-\mathbf{x}^*\|_1\}\\
\leq&\Big(\prod_{t=0}^{k-1}\exp(-c_{\mathrm{ss}}^t)\Big)\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(0)}-\mathbf{x}^*\|_1\}+2sC_W\Big(\prod_{t=0}^{0}\exp(-c_{\mathrm{ss}}^t))+\cdots+\prod_{t=0}^{k-1}\exp(-c_{\mathrm{ss}}^t))\Big)\sigma\\
\leq& sB\Big(\prod_{t=0}^{k-1}\exp(-c_{\mathrm{ss}}^t)\Big)+C_{\mathrm{ss}}\sigma\leq B\exp\Big(-\sum_{t=0}^{k-1}c_{\mathrm{ss}}^t\Big)+C_{\mathrm{ss}}\sigma.
\end{aligned}$$

With $\|\mathbf{x}\|_2\leq\|\mathbf{x}\|_1$, we have

$$\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k)}-\mathbf{x}^*\|_2\}\leq\sup_{\mathbf{x}^*,\varepsilon}\{\|\mathbf{x}^{(k)}-\mathbf{x}^*\|_1\}\leq sB\Big(\prod_{t=0}^{k-1}\exp(-c_{\mathrm{ss}}^t)\Big)+C_{\mathrm{ss}}\sigma.$$

Since $|\,\mathbb{S}^k\,|$ means the number of elements in $\mathbb{S}^k$, $|\,\mathbb{S}^k\,|\geq 0$. Thus, $c_{\mathrm{ss}}^k\geq c$ for all k. Consequently,

$$C_{\mathrm{ss}}\leq 2sC_W\Big(\sum_{k=0}^{\infty}\exp(-ck))\Big)=2sC_W\Big(\sum_{k=0}^{\infty}(2\tilde{\mu}s-\tilde{\mu})^k\Big)=\frac{2sC_W}{1+\tilde{\mu}-2\tilde{\mu}s}=C.$$

Step 2: proving (3.18). Secondly, we assume Assumption 2 holds. Take $(\mathbf{x}^*,\varepsilon)\in\bar{\mathcal{X}}(B,\underline{B},s,\sigma)$. The parameters are taken as

$$\mathbf{W}^{(k)}\in\mathcal{W}(\mathbf{D}),\quad\theta^{(k)}=\sup_{(\mathbf{x}^*,\varepsilon)\in\bar{\mathcal{X}}(B,\underline{B},s,\sigma)}\{\tilde{\mu}\|\mathbf{x}^{(k)}(\mathbf{x}^*,\varepsilon)-\mathbf{x}^*\|_1\}+C_W\sigma.$$

With the same argument as before, we get

$$\sup_{(\mathbf{x}^*,\varepsilon)\in\bar{\mathcal{X}}(B,\underline{B},s,\sigma)}\{\|\mathbf{x}^{(k)}-\mathbf{x}^*\|_2\}\leq sB\exp\Big(-\sum_{t=0}^{k-1}\tilde{c}_{\mathrm{ss}}^t\Big)+\tilde{C}_{\mathrm{ss}}\sigma,$$

where

$$\begin{aligned}
\tilde{c}_{\mathrm{ss}}^k=&-\log\Big(2\tilde{\mu}s-\tilde{\mu}-\tilde{\mu}\inf_{(\mathbf{x}^*,\varepsilon)\in\bar{\mathcal{X}}(B,\underline{B},s,\sigma)}|\,\mathbb{S}^k(\mathbf{x}^*,\varepsilon)|\Big)\geq c\\
\tilde{C}_{\mathrm{ss}}=&2sC_W\Big(\sum_{k=0}^{\infty}\prod_{t=0}^{k}\exp(-\tilde{c}_{\mathrm{ss}}^t))\Big)\leq C.
\end{aligned}$$

Now we consider $\mathbb{S}^k$ in a more precise way. The definition of $\mathbb{S}^k$ implies

$$|\,\mathbb{S}^k(\mathbf{x}^*,\varepsilon)|=\min\big(p^k,\#\text{ of non-zero elements of }\mathbf{x}^{(k+1)}\big).\tag{3.46}$$

By Assumption 5, it holds that $\|\mathbf{x}^*\|_1 \geq \underline{B} \geq 2C\sigma$. Consequently, if $k > 1/c(\log(sB/C\sigma))$, then

$$sB\exp(-ck) + C\sigma < 2C\sigma \leq \|\mathbf{x}^*\|_1,$$

which implies

$$\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1 \leq sB(\prod_{t=0}^{k} \exp(-\tilde{c}_{\text{ss}}^t)) + \tilde{C}_{\text{ss}}\sigma \leq sB\exp(-ck) + C\sigma < \|\mathbf{x}^*\|_1.$$

Then # of non-zero elements of $\mathbf{x}^{(k+1)} \geq 1$. (Otherwise, $\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1 = \|0 - \mathbf{x}^*\|_1$, which contradicts.) Moreover, $p^k = \min(pk, s)$ for some constant $p > 0$. Thus, as long as $k \geq 1/p$, we have $p^k \geq 1$. By (3.46), we obtain

$$|\,\mathbb{S}^k(\mathbf{x}^*, \varepsilon)| > 0, \quad \forall k > \max\Big(\frac{1}{p}, \frac{1}{c}\log\Big(\frac{sB}{C\sigma}\Big)\Big),\ \forall(\mathbf{x}^*, \varepsilon) \in \bar{\mathcal{X}}(B, \underline{B}, s, \sigma).$$

Then, we have $\tilde{c}_{\text{ss}}^k > c$ for large enough k, consequently, $\tilde{C}_{\text{ss}} < C$. □

Appendix 3.D Proof of Theorem 5

In this proof, we use the notion $\mathbf{x}^{(k)}$ to replace $\mathbf{x}^{(k)}(\mathbf{x}^*)$ for simplicity. We fix $\mathbf{D}$ in the proof, $\tilde{\mu}(\mathbf{D})$ can be simply written as $\tilde{\mu}$.

Before proving Theorem 5, we present and prove a lemma.

Lemma 3. *With all the settings the same with those in Theorem 5, we have*

$$\text{support}(\mathbf{x}^{(k)}) \subset \mathbb{S}, \quad \forall k. \tag{3.47}$$

In another word, there are no false positives in $\mathbf{x}^{(k)}$: $x_i^{(k)} = 0, \forall i \notin \mathbb{S}, \forall k$.

Proof. Take arbitrary $\mathbf{x}^* \in \mathcal{X}(B, s)$. We prove Lemma 3 by induction. As $k = 0$, (3.47) is satisfied since $\mathbf{x}^{(0)} = \mathbf{0}$. Fixing k, and assuming support$(\mathbf{x}^{(k)}) \subset \mathbb{S}$, we have

$$\begin{aligned} x_i^{(k+1)} =& \eta_{\theta^{(k)}}\Big(x_i^{(k)} - \gamma^{(k)}(\mathbf{W}_{:,i})^T(\mathbf{D}\mathbf{x}^{(k)} - \mathbf{b})\Big) \\ =& \eta_{\theta^{(k)}}\Big(-\gamma^{(k)}\sum_{j\in\mathbb{S}}(\mathbf{W}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*)\Big), \quad \forall i \notin \mathbb{S}\,. \end{aligned}$$

By (3.20), the thresholds are taken as $\theta^{(k)} = \tilde{\mu}\gamma^{(k)} \sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(k)}-\mathbf{x}^*\|_1\}$. Also, since $\mathbf{W} \in \mathcal{W}(\mathbf{D})$, we have $|(\mathbf{W}_{:,i})^T\mathbf{D}_{:,j}| \le \tilde{\mu}$ for all $j \neq i$. Thus, for all $i \notin \mathbb{S}$,

$$\begin{aligned}\theta^{(k)} \ge& \tilde{\mu}\gamma^{(k)}\big\|\mathbf{x}^{(k)} - \mathbf{x}^*\big\|_1 = \sum_{j\in\text{support}(\mathbf{x}^{(k)})} \tilde{\mu}\gamma^{(k)}\big|x_j^{(k)} - x_j^*\big| = \sum_{j\in\mathbb{S}} \tilde{\mu}\gamma^{(k)}\big|x_j^{(k)} - x_j^*\big| \\ \ge& \Big| -\gamma^{(k)}\sum_{j\in\mathbb{S}}(\mathbf{W}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*)\Big|,\end{aligned}$$

which implies $x_i^{(k+1)} = 0, \forall i \notin \mathbb{S}$ by the definition of $\eta_{\theta^{(k)}}$, i.e.,

$$\text{support}(\mathbf{x}^{(k+1)}) \subset \mathbb{S}$$

By induction, (3.47) is proved. □

With Lemma 3, we are able to prove Theorem 5 now.

Proof of Theorem 5. Take arbitrary $\mathbf{x}^* \in \mathcal{X}(B, s)$. For all $i \in \mathbb{S}$, by (3.47), we obtain

$$\begin{aligned}x_i^{(k+1)} &= \eta_{\theta^{(k)}}\Big(x_i^{(k)} - \gamma^{(k)}(\mathbf{W}_{:,i})^T\mathbf{D}_{:,\mathbb{S}}(\mathbf{x}_{\mathbb{S}}^{(k)} - \mathbf{x}_{\mathbb{S}}^*)\Big) \\ &\in x_i^{(k)} - \gamma^{(k)}(\mathbf{W}_{:,i})^T\mathbf{D}_{:,\mathbb{S}}(\mathbf{x}_{\mathbb{S}}^{(k)} - \mathbf{x}_{\mathbb{S}}^*) - \theta^{(k)}\partial\ell_1(x_i^{(k+1)}),\end{aligned}$$

where $\partial\ell_1(x)$ is the sub-gradient of $|x|, x \in \mathbb{R}$:

$$\partial\ell_1(x) = \begin{cases}\{\text{sign}(x)\} & \text{if } x \neq 0, \\ [-1, 1] & \text{if } x = 0.\end{cases}$$

The choice of $\mathbf{W} \in \mathcal{W}(\mathbf{D})$ gives $(\mathbf{W}_{:,i})^T\mathbf{D}_{:,i} = 1$. Thus,

$$\begin{aligned}& x_i^{(k)} - \gamma^{(k)}(\mathbf{W}_{:,i})^T\mathbf{D}_{:,\mathbb{S}}(\mathbf{x}_{\mathbb{S}}^{(k)} - \mathbf{x}_{\mathbb{S}}^*) \\ =& x_i^{(k)} - \gamma^{(k)}\sum_{j\in\mathbb{S}, j\neq i}(\mathbf{W}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*) - \gamma^{(k)}(x_i^{(k)} - x_i^*) \\ =& x_i^* - \gamma^{(k)}\sum_{j\in\mathbb{S}, j\neq i}(\mathbf{W}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*) + (1 - \gamma^{(k)})(x_i^{(k)} - x_i^*).\end{aligned}$$

Then the following inclusion formula holds for all $i \in \mathbb{S}$,

$$x_i^{(k+1)} - x_i^* \in -\gamma^{(k)}\sum_{j\in\mathbb{S}, j\neq i}(\mathbf{W}_{:,i})^T\mathbf{D}_{:,j}(x_j^{(k)} - x_j^*) - \theta^{(k)}\partial\ell_1(x_i^{(k+1)}) + (1 - \gamma^{(k)})(x_i^{(k)} - x_i^*).$$

By the definition of $\partial \ell_1$, every element in $\partial \ell_1(x), \forall x \in \mathbb{R}$ has a magnitude less than or equal to 1. Thus, for all $i \in \mathbb{S}$,

$$\begin{aligned}
|x_i^{(k+1)} - x_i^*| \leq & \sum_{j \in \mathbb{S}, j \neq i} \gamma^{(k)} \Big|(\mathbf{W}_{:,i})^T \mathbf{D}_{:,j}\Big| |x_j^{(k)} - x_j^*| + \theta^{(k)} + |1 - \gamma^{(k)}||x_i^{(k)} - x_i^*| \\
\leq & \tilde{\mu}\gamma^{(k)} \sum_{j \in \mathbb{S}, j \neq i} |x_j^{(k)} - x_j^*| + \theta^{(k)} + |1 - \gamma^{(k)}||x_i^{(k)} - x_i^*|.
\end{aligned}$$

Equation (3.47) implies $\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1 = \|\mathbf{x}_{\mathbb{S}}^{(k)} - \mathbf{x}_{\mathbb{S}}^*\|_1$ for all k. Then

$$\begin{aligned}
\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1 = & \sum_{i \in \mathbb{S}} |x_i^{(k+1)} - x_i^*| \\
\leq & \sum_{i \in \mathbb{S}} \Big(\tilde{\mu}\gamma^{(k)} \sum_{j \in \mathbb{S}, j \neq i} |x_j^{(k)} - x_j^*| + \theta^{(k)} + |1 - \gamma^{(k)}||x_i^{(k)} - x_i^*| \Big) \\
= & \tilde{\mu}\gamma^{(k)}(|\mathbb{S}| - 1) \sum_{i \in \mathbb{S}} |x_i^{(k)} - x_i^*| + \theta^{(k)}|\mathbb{S}| + |1 - \gamma^{(k)}| \|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1 \\
= & \tilde{\mu}\gamma^{(k)}(|\mathbb{S}| - 1)\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1 + \theta^{(k)}|\mathbb{S}| + |1 - \gamma^{(k)}| \|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1.
\end{aligned}$$

Taking supremum of the above inequality over $\mathbf{x}^* \in \mathcal{X}(B, s)$, by $|\mathbb{S}| \leq s$,

$$\sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq \Big(\tilde{\mu}\gamma^{(k)}(s - 1) + |1 - \gamma^{(k)}| \Big) \sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1\} + \theta^{(k)} s.$$

By the value of $\theta^{(k)}$ given in (3.20), we have

$$\sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq \Big(\gamma^{(k)}(2\tilde{\mu}s - \tilde{\mu}) + |1 - \gamma^{(k)}| \Big) \sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(k)} - \mathbf{x}^*\|_1\}.$$

Let $c^{(\tau)} = -\log\big((2\tilde{\mu}s - \tilde{\mu})\gamma^{(\tau)} + |1 - \gamma^{(\tau)}|\big)$. Then, by induction,

$$\sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq \exp\Big(-\sum_{\tau=0}^{(k)} c^{(\tau)} \Big) \sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(0)} - \mathbf{x}^*\|_1\} \leq \exp\Big(-\sum_{\tau=0}^{(k)} c^{(\tau)} \Big) sB.$$

Since $\|\mathbf{x}\|_2 \leq \|\mathbf{x}\|_1$ for any $\mathbf{x} \in \mathbb{R}^n$, we can get the upper bound for ℓ_2 norm:

$$\sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_2\} \leq \sup_{\mathbf{x}^*}\{\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\|_1\} \leq sB \exp\Big(-\sum_{\tau=0}^{(k)} c^{(\tau)} \Big).$$

The assumption $s < (1 + 1/\tilde{\mu})/2$ gives $2\tilde{\mu}s - \tilde{\mu} < 1$. If $0 < \gamma^{(k)} \leq 1$, we have $c^{(k)} > 0$. If $1 < \gamma^{(k)} < 2/(1 + 2\tilde{\mu}s - \tilde{\mu})$, we have

$$(2\tilde{\mu}s - \tilde{\mu})\gamma^{(k)} + |1 - \gamma^{(k)}| = (2\tilde{\mu}s - \tilde{\mu})\gamma^{(k)} + \gamma^{(k)} - 1 < 1,$$

which implies $c^{(k)} > 0$. Theorem 5 is proved. $\square$

Appendix 3.E Proof of Theorem 6

Proof of Theorem 6. We fix $\mathbf{D}$ and sample a $\mathbf{x}^* \sim P_X$.

If we can prove

$$P\Big((3.24) \text{ does not hold}\Big|\text{support}(\mathbf{x}^*) = \mathbb{S}\Big) \leq \epsilon|\mathbb{S}| + \epsilon^{|\mathbb{S}|}, \tag{3.48}$$

then the lower bound (3.24) in Theorem 6 is proved by

$$\begin{aligned} P\Big((3.24) \text{ holds}\Big) =& \sum_{\mathbb{S}, 2\leq|\mathbb{S}|\leq s} P\Big((3.24) \text{ holds}\Big|\text{support}(\mathbf{x}^*) = \mathbb{S}\Big)P\Big(\text{support}(\mathbf{x}^*) = \mathbb{S}\Big) \\ \geq& (1 - \epsilon s^{3/2} - \epsilon^2) \sum_{2\leq|\mathbb{S}|\leq s} P\Big(\text{support}(\mathbf{x}^*) = \mathbb{S}\Big) \\ =& 1 - \epsilon s^{3/2} - \epsilon^2. \end{aligned}$$

Now we fix k and prove inequality (3.48) by three steps:

Step 1: If (3.24) does not hold, then what condition $\mathbf{x}^*$ should satisfy?

Fixing k, we define a set $\mathcal{X}^{(k)}(\epsilon)$, which involves all the $\mathbf{x}^*$ that does not satisfy (3.24):

$$\mathcal{X}^{(k)}(\epsilon) = \{(3.24) \text{ does not hold}\} = \Big\{\mathbf{x}^*\Big|\|\mathbf{x}^{(k)}(\mathbf{x}^*) - \mathbf{x}^*\|_2 < \epsilon\|\mathbf{x}^*\|_2\Big(\frac{\bar{\sigma}_{\min}}{3^s}\Big)^{(k)}\Big\}.$$

Let $\mathbb{S} = \text{support}(\mathbf{x}^*)$. For $\mathbf{x}^* \in \mathcal{X}^{(k)}(\epsilon)$, we consider two cases:

1. $|x_i^*| > \epsilon\|\mathbf{x}^*\|_2(\bar{\sigma}_{\min}/3^s)^{(k)}, \ \forall i \in \mathbb{S}.$

2. $|x_i^*| \leq \epsilon\|\mathbf{x}^*\|_2(\bar{\sigma}_{\min}/3^s)^{(k)}, \ \text{for some } i \in \mathbb{S}.$

If case 1 holds, we obtain that the support of $\mathbf{x}^{(k)}$ is exactly the same with that of $\mathbf{x}^*$:

$$\text{support}(\mathbf{x}^{(k)}(\mathbf{x}^*)) = \mathbb{S}\,.$$

Then the relationship between $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k-1)}$ can be reduced to an affine transform:

$$\begin{aligned} \mathbf{x}_{\mathbb{S}}^{(k)} =& \eta_{\theta^{(k)}}\Big(\mathbf{x}_{\mathbb{S}}^{(k-1)} - (\mathbf{W}_{:,\mathbb{S}}^{(k-1)})^T(\mathbf{D}\mathbf{x}^{(k-1)} - \mathbf{b})\Big) \\ =& \mathbf{x}_{\mathbb{S}}^{(k-1)} - (\mathbf{W}_{:,\mathbb{S}}^{(k-1)})^T\mathbf{D}_{:,\mathbb{S}}(\mathbf{x}_{\mathbb{S}}^{(k-1)} - \mathbf{x}_{\mathbb{S}}^*) - \theta^{(k-1)}\text{sign}(\mathbf{x}_{\mathbb{S}}^{(k)}). \end{aligned} \tag{3.49}$$

Subtracting $\mathbf{x}^*$ from the two sides of (3.49), we obtain

$$\left\|\big(\mathbf{I}-(\mathbf{W}_{:,\mathbb{S}}^{(k-1)})^T\mathbf{D}_{:,\mathbb{S}}\big)(\mathbf{x}_{\mathbb{S}}^{(k-1)}-\mathbf{x}_{\mathbb{S}}^*)-\theta^{(k-1)}\mathrm{sign}(\mathbf{x}_{\mathbb{S}}^{(k)})\right\|_2=\|\mathbf{x}_{\mathbb{S}}^{(k)}-\mathbf{x}_{\mathbb{S}}^*\|_2=\|\mathbf{x}^{(k)}-\mathbf{x}^*\|_2,$$

where the last equality is due to Definition 3. Thus, for all $\mathbf{x}^*\in\mathcal{X}^{(k)}(\epsilon)$, if case 1 holds, we have

$$\left\|\big(\mathbf{I}-(\mathbf{W}_{:,\mathbb{S}}^{(k-1)})^T\mathbf{D}_{:,\mathbb{S}}\big)(\mathbf{x}_{\mathbb{S}}^{(k-1)}-\mathbf{x}_{\mathbb{S}}^*)-\theta^{(k-1)}\mathrm{sign}(\mathbf{x}_{\mathbb{S}}^{(k)})\right\|_2\le\epsilon\|\mathbf{x}^*\|_2(\bar{\sigma}_{\min}/3^s)^{(k)}. \tag{3.50}$$

Multiplying both sides of (3.50) by $(\mathbf{I}-(\mathbf{W}_{:,\mathbb{S}}^{(k-1)})^T\mathbf{D}_{:,\mathbb{S}})^{-1}$, we have

$$\begin{aligned}&\|\mathbf{x}_{\mathbb{S}}^{(k-1)}-\mathbf{x}_{\mathbb{S}}^*-\theta^{(k-1)}(\mathbf{I}-(\mathbf{W}_{:,\mathbb{S}}^{(k-1)})^T\mathbf{D}_{:,\mathbb{S}})^{-1}\mathrm{sign}(\mathbf{x}_{\mathbb{S}}^{(k)})\|_2\\ \le&\|(\mathbf{I}-(\mathbf{W}_{:,\mathbb{S}}^{(k-1)})^T\mathbf{D}_{:,\mathbb{S}})^{-1}\|_2\cdot\epsilon\|\mathbf{x}^*\|_2(\bar{\sigma}_{\min}/3^s)^{(k)}\le\epsilon\|\mathbf{x}^*\|_2(\bar{\sigma}_{\min})^{k-1}3^{-ks},\end{aligned}$$

where the last inequality is due to (3.22). Let $\tilde{\mathbf{x}}^{(k-1)}$ denote the bias of $\mathbf{x}^{(k-1)}$:

$$\tilde{\mathbf{x}}^{(k-1)}\triangleq\theta^{(k-1)}(\mathbf{I}-(\mathbf{W}_{:,\mathbb{S}}^{(k-1)})^T\mathbf{D}_{:,\mathbb{S}})^{-1}\mathrm{sign}(\mathbf{x}_{\mathbb{S}}^{(k)}),$$

then we get a condition that $\mathbf{x}^*$ satisfies if case 1 holds:

$$\mathcal{X}^{(k-1)}(\epsilon)=\left\{\mathbf{x}^*\middle|\big\|\mathbf{x}_{\mathbb{S}}^{(k-1)}(\mathbf{x}^*)-\mathbf{x}_{\mathbb{S}}^*-\tilde{\mathbf{x}}^{(k-1)}(\mathbf{x}^*)\big\|_2\le\epsilon\|\mathbf{x}^*\|_2(\bar{\sigma}_{\min})^{k-1}3^{-ks}\right\}.$$

If case 2 holds, $\mathbf{x}^*$ belongs to the following set:

$$\tilde{\mathcal{X}}^{(k)}(\epsilon)=\left\{\mathbf{x}^*\middle||x_i^*|\le\epsilon\|\mathbf{x}^*\|_2\big(\bar{\sigma}_{\min}/3^s\big)^{(k)},\ \text{for some } i\in\mathbb{S}\right\}.$$

Then for any $\mathbf{x}^*\in\mathcal{X}^{(k)}(\epsilon)$, either $\mathbf{x}^*\in\mathcal{X}^{(k-1)}(\epsilon)$ or $\mathbf{x}^*\in\tilde{\mathcal{X}}^{(k)}(\epsilon)$ holds. In another word,

$$\mathcal{X}^{(k)}(\epsilon)\subset\tilde{\mathcal{X}}^{(k)}(\epsilon)\cup\mathcal{X}^{(k-1)}(\epsilon).$$

Step 2: By imitating the construction of $\mathcal{X}^{(k)}(\epsilon)$, we construct

$$\mathcal{X}^{(k-2)}(\epsilon),\mathcal{X}^{(k-3)}(\epsilon),\cdots.$$

Similar to Step 1, we divide $\mathcal{X}^{(k-1)}(\epsilon)$ into two sets: $\tilde{\mathcal{X}}^{(k-1)}(\epsilon)$ and $\mathcal{X}^{(k-2)}(\epsilon)$, then we divide $\mathcal{X}^{(k-2)}(\epsilon)$ into $\tilde{\mathcal{X}}^{(k-2)}(\epsilon)$ and $\mathcal{X}^{(k-3)}(\epsilon)$. Repeating the process, until dividing $\mathcal{X}^{(1)}(\epsilon)$ into $\tilde{\mathcal{X}}^{(1)}(\epsilon)$ and $\mathcal{X}^{(0)}(\epsilon)$.

By induction, we have

$$\mathcal{X}^{(k)}(\epsilon) \subset \tilde{\mathcal{X}}^{(k)}(\epsilon) \cup \tilde{\mathcal{X}}^{(k-1)}(\epsilon) \cup \tilde{\mathcal{X}}^{(k-2)}(\epsilon) \cup \cdots \cup \tilde{\mathcal{X}}^{(1)}(\epsilon) \cup \mathcal{X}^{(0)}(\epsilon), \tag{3.51}$$

where the sets are defined as follows for all $j = 0, 1, 2, \cdots, k$:

$$\tilde{\mathcal{X}}^{(k-j)}(\epsilon) = \Big\{ \mathbf{x}^* \Big| |x_i^* + \tilde{x}_i^{(k-j)}(\mathbf{x}^*)| < \epsilon \|\mathbf{x}^*\|_2 \big(\bar{\sigma}_{\min}\big)^{k-j} 3^{-ks}, \text{ for some } i \in \mathbb{S}. \Big\}, \tag{3.52}$$

$$\mathcal{X}^{(k-j)}(\epsilon) = \Big\{ \mathbf{x}^* \Big| \|\mathbf{x}_{\mathbb{S}}^{(k-j)}(\mathbf{x}^*) - \mathbf{x}_{\mathbb{S}}^* - \tilde{\mathbf{x}}^{(k-j)}(\mathbf{x}^*)\|_2 \leq \epsilon \|\mathbf{x}^*\|_2 \big(\bar{\sigma}_{\min}\big)^{k-j} 3^{-ks} \Big\} \tag{3.53}$$

and the bias is defined as following for all $j = 0, 1, 2, \cdots, k$:

$$\tilde{\mathbf{x}}^{(k-j)}(\mathbf{x}^*) = \sum_{t=1}^{j} \bigg(\mathbf{I} - \Big(\mathbf{W}_{:,\mathbb{S}}^{(k-j+t-1)} \Big)^T \mathbf{D}_{:,\mathbb{S}} \bigg)^{-t} \theta^{(k-j+t-1)} \mathrm{sign}\big(\mathbf{x}_{\mathbb{S}}^{(k-j+t)}(\mathbf{x}^*) \big). \tag{3.54}$$

Step 3: Estimating the probabilities of all the sets in (3.51).

By (3.51), we have

$$\begin{aligned} & P\Big(\mathbf{x}^* \in \mathcal{X}^{(k)}(\epsilon) \Big| \mathrm{support}(\mathbf{x}^*) = \mathbb{S} \Big) \\ \leq & \sum_{j=1}^{k-1} P\Big(\mathbf{x}^* \in \tilde{\mathcal{X}}^{(k-j)}(\epsilon) \Big| \mathrm{support}(\mathbf{x}^*) = \mathbb{S} \Big) + P\Big(\mathbf{x}^* \in \mathcal{X}^{(0)}(\epsilon) \Big| \mathrm{support}(\mathbf{x}^*) = \mathbb{S} \Big). \end{aligned}$$

Now we have to prove that each of the above terms is small, then $P(\mathbf{x}^* \in \mathcal{X}^{(k)}(\epsilon) | \mathrm{support}(\mathbf{x}^*) = \mathbb{S})$ is small and (3.48) will be proved.

Define a set of n-dimensional sign numbers

$$\mathrm{Si}(n) = \Big\{ (s_1, s_2, \cdots, s_n) \Big| s_i \in \{0, -1, 1\}, \forall i = 1, \cdots, n \Big\}.$$

Since $\mathrm{sign}\big(\mathbf{x}_{\mathbb{S}}^{(k-j+t)}\big) \in \mathrm{Si}(|\mathbb{S}|)$ for all $t = 1, 2, \cdots, j$, $\{\mathrm{sign}(\mathbf{x}_{\mathbb{S}}^{(k-j+t)})\}_{t=1}^{j}$ has finitely possible values. Let $\mathrm{sign}(\mathbf{x}_{\mathbb{S}}^{(k-j+t)}) = \mathbf{s}^{(t)}$ for $t = 1, 2, \cdots, j$. Then $\tilde{x}_i^{(k-j)}(\mathbf{x}^*)$ is independent of $\mathbf{x}^*$ and can be written as $\tilde{x}_i^{(k-j)}(\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \cdots, \mathbf{s}^{(j)})$. Thus, we have

$$\begin{aligned}
&P(\mathbf{x}^* \in \tilde{\mathcal{X}}^{(k-j)}(\epsilon)|\text{support}(\mathbf{x}^*) = \mathbb{S}) \\
=&\sum_{i\in\mathbb{S}} \sum_{\mathbf{s}^{(1)}\in\text{Si}(|\mathbb{S}|)} \sum_{\mathbf{s}^{(2)}\in\text{Si}(|\mathbb{S}|)} \cdots \sum_{\mathbf{s}^{(j)}\in\text{Si}(|\mathbb{S}|)} \\
&P\Big(|x_i^* + \tilde{x}_i^{(k-j)}(\mathbf{x}^*)| < \epsilon\|\mathbf{x}^*\|_2(\bar{\sigma}_{\min})^{k-j}3^{-ks}, \text{sign}(\mathbf{x}_{\mathbb{S}}^{(k)}) = \mathbf{s}^{(1)}, \cdots, \\
&\text{sign}(\mathbf{x}_{\mathbb{S}}^{(k-j+1)}) = \mathbf{s}^{(j)}\Big|\text{support}(\mathbf{x}^*) = \mathbb{S}\Big) \\
\leq&\sum_{i\in\mathbb{S}} \sum_{\mathbf{s}^{(1)}\in\text{Si}(|\mathbb{S}|)} \sum_{\mathbf{s}^{(2)}\in\text{Si}(|\mathbb{S}|)} \cdots \sum_{\mathbf{s}^{(j)}\in\text{Si}(|\mathbb{S}|)} \\
&P\Big(|x_i^* + \tilde{x}_i^{(k-j)}(\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \cdots, \mathbf{s}^{(j)})| < \epsilon\sqrt{|\mathbb{S}|}B(\bar{\sigma}_{\min})^{k-j}3^{-ks}\Big|\text{support}(\mathbf{x}^*) = \mathbb{S}\Big) \\
\leq&\sum_{i\in\mathbb{S}} \sum_{\mathbf{s}^{(1)}\in\text{Si}(|\mathbb{S}|)} \sum_{\mathbf{s}^{(2)}\in\text{Si}(|\mathbb{S}|)} \cdots \sum_{\mathbf{s}^{(j)}\in\text{Si}(|\mathbb{S}|)} \frac{\epsilon\sqrt{|\mathbb{S}|}B(\bar{\sigma}_{\min})^{k-j}3^{-ks}}{B} \\
=&|\mathbb{S}|3^{j|\mathbb{S}|}(\epsilon\sqrt{|\mathbb{S}|}\Big((\bar{\sigma}_{\min})^{k-j}3^{-ks}\Big) \leq \epsilon|\mathbb{S}|^{3/2}(\bar{\sigma}_{\min})^{k-j}3^{(j-k)|\mathbb{S}|}
\end{aligned}$$

where the second inequality comes from the uniform distribution of $\mathbf{x}_{\mathbb{S}}^*$ (Assumption 6), the last inequality comes from $|\mathbb{S}| \leq s$.

The last term, due to the uniform distribution of $\mathbf{x}_{\mathbb{S}}^*$ and $\mathbf{x}^{(0)} = \mathbf{0}$, can be bounded by

$$\begin{aligned}
&P(\mathbf{x}^* \in \mathcal{X}^{(0)}(\epsilon)|\text{support}(\mathbf{x}^*) = \mathbb{S}) \\
=&P\Big(\|\mathbf{x}^* + \tilde{\mathbf{x}}^{(0)}(\mathbf{x}^*)\|_2 \leq \epsilon\|\mathbf{x}^*\|_2 3^{-ks}\Big|\text{support}(\mathbf{x}^*) = \mathbb{S}\Big) \\
=&\sum_{\mathbf{s}^{(1)}\in\text{Si}(|\mathbb{S}|)} \sum_{\mathbf{s}^{(2)}\in\text{Si}(|\mathbb{S}|)} \cdots \sum_{\mathbf{s}^{(k)}\in\text{Si}(|\mathbb{S}|)} \\
&P\Big(\|\mathbf{x}^* + \tilde{\mathbf{x}}^{(0)}(\mathbf{x}^*)\|_2 \leq \epsilon\|\mathbf{x}^*\|_2 3^{-ks}, \text{sign}(\mathbf{x}_{\mathbb{S}}^{(1)}) = \mathbf{s}^{(1)}, \cdots, \text{sign}(\mathbf{x}_{\mathbb{S}}^{(k)}) = \mathbf{s}^{(k)}\Big|\text{support}(\mathbf{x}^*) = \mathbb{S}\Big) \\
\leq&3^{k|\mathbb{S}|}\Big((\epsilon 3^{-ks})^{|\mathbb{S}|}\Big) \leq \epsilon^{|\mathbb{S}|}.
\end{aligned}$$

Then we obtain

$$\begin{aligned}
&P(\mathbf{x}^* \in \mathcal{X}^{(k)}(\epsilon)|\text{support}(\mathbf{x}^*) = \mathbb{S}) \\
\leq&\sum_{j=0}^{k-1} \epsilon|\mathbb{S}|^{3/2}(\bar{\sigma}_{\min})^{k-j}3^{(j-k)|\mathbb{S}|} + \epsilon^{|\mathbb{S}|} = \sum_{j=1}^{(k)} \epsilon|\mathbb{S}|^{3/2}(\bar{\sigma}_{\min})^{j}3^{-j|\mathbb{S}|} + \epsilon^{|\mathbb{S}|} \\
=&\epsilon|\mathbb{S}|^{3/2}\frac{\bar{\sigma}_{\min}3^{-|\mathbb{S}|}}{1-\bar{\sigma}_{\min}3^{-|\mathbb{S}|}}\Big(1 - (\bar{\sigma}_{\min}3^{-|\mathbb{S}|})^{(k)}\Big) + \epsilon^{|\mathbb{S}|} \leq \epsilon|\mathbb{S}|^{3/2} + \epsilon^{|\mathbb{S}|}.
\end{aligned}$$

Then (3.48) is proved. □

Appendix 3.F Proof of Theorem 7

There are two conclusions in Theorem 7. We prove the two conclusions in the following two subsections respectively.

3.F.1 Proof of Conclusion 1.

Before proving Conclusion 1, we analyze the operator $\mathbf{D}_{\text{cir}}^{N}$ in detail.

The circular convolution (3.34) is equivalent with:

$$\mathbf{b}(i,j) = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\sum_{m=1}^{M} \mathbf{D}_{\text{cir}}^{N}(i,j;k,l,m)\mathbf{x}_m(k,l), \quad 0 \le i,j \le N-1,$$

where the circulant matrix is element-wise defined as:

$$\mathbf{D}_{\text{cir}}^{N}(i,j;k,l,m) = \begin{cases} \mathbf{d}_m\big((k-i)_{\text{mod}N},(l-j)_{\text{mod}N}\big), & 0 \le (k-i)_{\text{mod}N},(l-j)_{\text{mod}N} \le D-1 \\ 0, & \text{others} \end{cases} \tag{3.55}$$

Similarly, the corresponding circulant matrix $\mathbf{W}_{\text{cir}}^{N}(i,j;k,l,m)$ of dictionary $\mathbf{w}$ is:

$$\mathbf{W}_{\text{cir}}^{N}(i,j;k,l,m) = \begin{cases} \mathbf{w}_m\big((k-i)_{\text{mod}N},(l-j)_{\text{mod}N}\big), & 0 \le (k-i)_{\text{mod}N},(l-j)_{\text{mod}N} \le D-1 \\ 0, & \text{others} \end{cases} \tag{3.56}$$

As we defined in Section 3.5, $\mathbf{b}$ is a vector. With $\mathbf{x} = [\mathbf{x}_1, \cdots, \mathbf{x}_M]^T$, $\mathbf{x}$ is a vector. Then the operator $\mathbf{D}_{\text{cir}}^{N}$ is a matrix, where (i,j) is its row index and (k,l,m) is its column index.

Define a function measuring the difference between i and k:

$$I(i,k) \triangleq (k-i)_{\text{mod}N}, \quad 0 \le i,k \le N-1.$$

The coherence between $\mathbf{D}_{\text{cir}}^{N}(i,j;k,l,m)$ and $\mathbf{W}_{\text{cir}}^{N}(i,j;k,l,m)$: $\mathbf{B}_{\text{coh}} = (\mathbf{D}_{\text{cir}}^{N})^T\mathbf{W}_{\text{cir}}^{N}$ is element-wise defined by:

$$\begin{aligned}\mathbf{B}_{\text{coh}}(k_1,l_1,m_1;k_2,l_2,m_2) &= \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} \mathbf{D}_{\text{cir}}^{N}(i,j;k_1,l_1,m_1)\mathbf{W}_{\text{cir}}^{N}(i,j;k_2,l_2,m_2) \\ &= \sum_{i\in\mathcal{I}(k_1,k_2)}\ \sum_{j\in\mathcal{J}(l_1,l_2)} \mathbf{d}_{m_1}\big(I(i,k_1),I(j,l_1)\big)\mathbf{w}_{m_2}\big(I(i,k_2),I(j,l_2)\big).\end{aligned}$$

where

$$\mathcal{I}(k_1,k_2) = \{i | 0 \le i \le N-1,\ 0 \le I(i,k_1) \le D-1,\ 0 \le I(i,k_2) \le D-1\},$$
$$\mathcal{J}(l_1,l_2) = \{j | 0 \le j \le N-1,\ 0 \le I(j,l_1) \le D-1,\ 0 \le I(j,l_2) \le D-1\}.$$

Lemma 4. *Given $N \ge 2D-1$, it holds that:*

(a) $\mathcal{I}(k_1,k_2) \ne \varnothing$ if and only if "$0 \le (k_1-k_2)_{\text{mod}N} \le D-1$" or "$0 < (k_2-k_1)_{\text{mod}N} \le D-1$" holds.

(b) $\mathcal{J}(l_1,l_2) \ne \varnothing$ if and only if "$0 \le (l_1-l_2)_{\text{mod}N} \le D-1$" or "$0 < (l_2-l_1)_{\text{mod}N} \le D-1$" holds.

Proof. Now we prove Conclusion (a). Firstly, we prove "if." If $0 \le (k_1-k_2)_{\text{mod}N} \le D-1$ and $N \ge 2D-1$, we have

$$\mathcal{I}(k_1,k_2) = \big\{(k_1-\delta)_{\text{mod}N} \big| \delta \in \mathbb{Z},\ (k_1-k_2)_{\text{mod}N} \le \delta \le D-1\big\} \ne \varnothing. \tag{3.57}$$

If $0 < (k_2-k_1)_{\text{mod}N} \le D-1$ and $N \ge 2D-1$, we have

$$\mathcal{I}(k_1,k_2) = \big\{(k_2-\delta)_{\text{mod}N} \big| \delta \in \mathbb{Z},\ (k_2-k_1)_{\text{mod}N} \le \delta \le D-1\big\} \ne \varnothing. \tag{3.58}$$

Secondly, we prove "only if." If $\mathcal{I}(k_1,k_2) \ne \varnothing$, we can select an $i \in \mathcal{I}(k_1,k_2)$. Let $r_1 = (k_1-i)_{\text{mod}N}$ and $r_2 = (k_2-i)_{\text{mod}N}$. By the definition of $\mathcal{I}(k_1,k_2)$, we have $0 \le r_1, r_2 \le D-1$. Two cases should be considered here. Case 1: $r_1 \ge r_2$. Since $0 \le r_1 - r_2 \le D-1 \le N-1$, it holds that $r_1 - r_2 = (r_1-r_2)_{\text{mod}N}$. Thus,

$$\begin{aligned} r_1 - r_2 = (r_1-r_2)_{\text{mod}N} =& \big((k_1-i)_{\text{mod}N} - (k_2-i)_{\text{mod}N}\big)_{\text{mod}N} \\ =& \big((k_1-i)-(k_2-i)\big)_{\text{mod}N} \\ =& (k_1-k_2)_{\text{mod}N}. \end{aligned}$$

The equality "$0 \le r_1 - r_2 \le D-1$" leads to the conclusion "$0 \le (k_1-k_2)_{\text{mod}N} \le D-1$". In case 2 where $r_1 < r_2$, we can obtain $0 < (k_2-k_1)_{\text{mod}N} \le D-1$ with the similar arguments.

Conclusion (b) can be proved by the same argument with the proof of (a). Lemma 4 is proved. □

Now we fix k_1, l_1 and consider what values of k_2, l_2 give $\mathcal{I}(k_1, k_2) \neq \varnothing$ and $\mathcal{J}(l_1, l_2) \neq \varnothing$. Define four index sets given $0 \leq k_1, l_1 \leq N - 1$:

$$\begin{aligned} \mathcal{K}(k_1) =& \{k | 0 \leq (k_1 - k)_{\text{mod}N} \leq D - 1\} \\ \bar{\mathcal{K}}(k_1) =& \{k | 0 < (k - k_1)_{\text{mod}N} \leq D - 1\} \end{aligned}$$

$$\begin{aligned} \mathcal{L}(l_1) =& \{l | 0 \leq (l_1 - l)_{\text{mod}N} \leq D - 1\} \\ \bar{\mathcal{L}}(l_1) =& \{l | 0 < (l - l_1)_{\text{mod}N} \leq D - 1\} \end{aligned}$$

Lemma 5. *If $N \geq 2D - 1$, we have:*

(a) The cardinality of $\mathcal{K}(k_1), \bar{\mathcal{K}}(k_1)$: $|\mathcal{K}(k_1)| = D, |\bar{\mathcal{K}}(k_1)| = D - 1$.

(b) $\mathcal{K}(k_1) \cap \bar{\mathcal{K}}(k_1) = \varnothing$.

(c) The cardinality of $\mathcal{L}(l_1), \bar{\mathcal{L}}(l_1)$: $|\mathcal{L}(l_1)| = D, |\bar{\mathcal{L}}(l_1)| = D - 1$.

(d) $\mathcal{L}(l_1) \cap \bar{\mathcal{L}}(l_1) = \varnothing$.

Proof. Now we prove Conclusion (a). The set $\mathcal{K}(k_1)$ can be equivalently written as

$$\mathcal{K}(k_1) = \{(k_1 - r_k)_{\text{mod}N} | r_k = 0, 1, \cdots, D - 1\} \tag{3.59}$$

Let $k(r_k) = (k_1 - r_k)_{\text{mod}N}$. We want to show that $k(r_k^1) \neq k(r_k^2)$ as long as $r_k^1 \neq r_k^2$. Without loss of generality, we assume $0 \leq r_k^1 < r_k^2 \leq D - 1$. By the definition of modulo operation, There exist two integers q, q' such that

$$k(r_k^1) = qN + k_1 - r_k^1, \quad k(r_k^2) = q'N + k_1 - r_k^2.$$

Suppose $k(r_k^1) = k(r_k^2)$. Taking the difference between the above two equations, we obtain $r_k^2 - r_k^1 = (q' - q)N$, i.e, N divides $r_k^2 - r_k^1$. However, $0 \leq r_k^1 < r_k^2 \leq D - 1$ implies $1 \leq r_k^2 - r_k^1 \leq D - 1 \leq N - 1$, which contradicts with "N dividing $r_k^2 - r_k^1$." Thus, it holds that $k(r_k^1) \neq k(r_k^2)$. Then we have $|\mathcal{K}(k_1)| = D$.

In the same way, we have

$$\bar{\mathcal{K}}(k_1) = \{(k_1 + r_k)_{\text{mod}N} | r_k = 1, 2, \cdots, D - 1\} \tag{3.60}$$

and $|\bar{\mathcal{K}}(k_1)| = D-1$. Conclusion (a) is proved.

Now we prove Conclusion (b). Suppose $\mathcal{K}(k_1) \cap \bar{\mathcal{K}}(k_1) \neq \varnothing$. Pick a $k_2 \in \mathcal{K}(k_1) \cap \bar{\mathcal{K}}(k_1)$. Let $r_3 = (k_1 - k_2)_{\text{mod}N}$ and $r_4 = (k_2 - k_1)_{\text{mod}N}$. Then we have $0 \leq r_3 \leq D-1$ and $0 < r_4 \leq D-1$. By the definition of modulo operation, There exist two integers q, q' such that

$$k_1 - k_2 = qN + r_3, \quad k_2 - k_1 = q'N + r_4$$

which imply

$$r_3 + r_4 + (q+q')N = 0.$$

However, $0 < r_3 + r_4 \leq 2D-2$ contradicts with "$q \in \mathbb{Z}, q' \in \mathbb{Z}, N \in \mathbb{Z}, N \geq 2D-1$." Conclusion (b) is proved.

Conclusions (c) and (d) are actually the same with Conclusions (a) and (b) respectively. Thus, it holds that

$$\mathcal{L}(l_1) = \{(l_1 - r_l)_{\text{mod}N} | r_l = 0, 1, \cdots, D-1\} \tag{3.61}$$

$$\bar{\mathcal{L}}(l_1) = \{(l_1 + r_l)_{\text{mod}N} | r_l = 1, 2, \cdots, D-1\} \tag{3.62}$$

and $|\mathcal{L}(l_1)| = D, |\bar{\mathcal{L}}(l_1)| = D-1$. Lemma 5 is proved. □

With the preparations, we can prove Conclusion 1 of Theorem 7 now.

Proof of Theorem 7, Conclusion 1. Firstly we fix $k_1 \in \{0, 1, \cdots, N-1\}$ and consider $k_2 \in \mathcal{K}(k_1)$. Let $r_k = (k_1 - k_2)_{\text{mod}N}$. Then equation (3.57) implies that, for any $i \in \mathcal{I}(k_1, k_2)$, there exists a δ ($r_k \leq \delta \leq D-1$) such that

$$\begin{aligned} I(i, k_1) =& \big(k_1 - (k_1 - \delta)_{\text{mod}N}\big)_{\text{mod}N} = (\delta)_{\text{mod}N} = \delta, \\ I(i, k_2) =& \big(k_2 - (k_1 - \delta)_{\text{mod}N}\big)_{\text{mod}N} = (\delta - r_k)_{\text{mod}N} = \delta - r_k. \end{aligned} \tag{3.63}$$

Now we consider another case for k_2: $k_2 \in \bar{\mathcal{K}}(k_1)$, $r_k = (k_2 - k_1)_{\text{mod}N}$. Equation (3.58) implies that, for any $i \in \mathcal{I}(k_1, k_2)$, there exists a δ ($r_k \leq \delta \leq D-1$) such that

$$\begin{aligned} I(i, k_1) =& \big(k_1 - (k_2 - \delta)_{\text{mod}N}\big)_{\text{mod}N} = (\delta - r_k)_{\text{mod}N} = \delta - r_k, \\ I(i, k_2) =& \big(k_2 - (k_2 - \delta)_{\text{mod}N}\big)_{\text{mod}N} = (\delta)_{\text{mod}N} = \delta. \end{aligned} \tag{3.64}$$

Similarly, for any $l_1 \in \{0, 1, \cdots, N-1\}$ and $l_2 \in \mathcal{L}(l_1)$, we denote $r_l = (l_1 - l_2)_{\mathrm{mod} N}$. For any $j \in \mathcal{J}(l_1, l_2)$, there exists a δ ($r_l \le \delta \le D-1$) such that

$$\begin{aligned} I(j, l_1) =& \big(l_1 - (l_1 - \delta)_{\mathrm{mod} N}\big)_{\mathrm{mod} N} = (\delta)_{\mathrm{mod} N} = \delta, \\ I(j, l_2) =& \big(l_2 - (l_1 - \delta)_{\mathrm{mod} N}\big)_{\mathrm{mod} N} = (\delta - r_l)_{\mathrm{mod} N} = \delta - r_l. \end{aligned} \tag{3.65}$$

Another case for l_2: $l_2 \in \bar{\mathcal{L}}(l_1)$, $r_l = (l_2 - l_1)_{\mathrm{mod} N}$. For any $j \in \mathcal{J}(l_1, l_2)$, there exists a δ ($r_l \le \delta \le D-1$) such that

$$\begin{aligned} I(j, l_1) =& \big(l_1 - (l_2 - \delta)_{\mathrm{mod} N}\big)_{\mathrm{mod} N} = (\delta - r_l)_{\mathrm{mod} N} = \delta - r_l, \\ I(j, l_2) =& \big(l_2 - (l_2 - \delta)_{\mathrm{mod} N}\big)_{\mathrm{mod} N} = (\delta)_{\mathrm{mod} N} = \delta. \end{aligned} \tag{3.66}$$

Now let us consider the following function. By results in Lemmas 4 and 5, we have

$$\begin{aligned} f(k_1, l_1, m_1, m_2) =& \sum_{k_2=0}^{N-1} \sum_{l_2=0}^{N-1} \Big(\mathbf{B}_{\mathrm{coh}}(k_1, l_1, m_1; k_2, l_2, m_2)\Big)^2 \\ =& f_1 + f_2 + f_3 + f_4, \end{aligned}$$

where

$$\begin{aligned} f_1 &= \sum_{k_2 \in \mathcal{K}(k_1)} \sum_{l_2 \in \mathcal{L}(l_1)} \Big(\mathbf{B}_{\mathrm{coh}}(k_1, l_1, m_1; k_2, l_2, m_2)\Big)^2 \\ f_2 &= \sum_{k_2 \in \bar{\mathcal{K}}(k_1)} \sum_{l_2 \in \mathcal{L}(l_1)} \Big(\mathbf{B}_{\mathrm{coh}}(k_1, l_1, m_1; k_2, l_2, m_2)\Big)^2 \\ f_3 &= \sum_{k_2 \in \mathcal{K}(k_1)} \sum_{l_2 \in \bar{\mathcal{L}}(l_1)} \Big(\mathbf{B}_{\mathrm{coh}}(k_1, l_1, m_1; k_2, l_2, m_2)\Big)^2 \\ f_4 &= \sum_{k_2 \in \bar{\mathcal{K}}(k_1)} \sum_{l_2 \in \bar{\mathcal{L}}(l_1)} \Big(\mathbf{B}_{\mathrm{coh}}(k_1, l_1, m_1; k_2, l_2, m_2)\Big)^2. \end{aligned}$$

Combining equations (3.59), (3.61), (3.63) and (3.65), we obtain

$$f_1 = \sum_{r_k=0}^{D-1} \sum_{r_l=0}^{D-1} \sum_{\delta_k=r_k}^{D-1} \sum_{\delta_l=r_l}^{D-1} \Big(\mathbf{d}_{m_1}(\delta_k, \delta_l) \mathbf{w}_{m_2}(\delta_k - r_k, \delta_l - r_l)\Big)^2.$$

Combining (3.60), (3.61), (3.64) and (3.65), we obtain

$$f_2 = \sum_{r_k=1}^{D-1} \sum_{r_l=0}^{D-1} \sum_{\delta_k=r_k}^{D-1} \sum_{\delta_l=r_l}^{D-1} \Big(\mathbf{d}_{m_1}(\delta_k - r_k, \delta_l) \mathbf{w}_{m_2}(\delta_k, \delta_l - r_l)\Big)^2.$$

Combining (3.59), (3.62), (3.63) and (3.66), we obtain

$$f_3 = \sum_{r_k=0}^{D-1} \sum_{r_l=1}^{D-1} \sum_{\delta_k=r_k}^{D-1} \sum_{\delta_l=r_l}^{D-1} \Big(\mathbf{d}_{m_1}(\delta_k, \delta_l - r_l) \mathbf{w}_{m_2}(\delta_k - r_k, \delta_l)\Big)^2.$$

Combining (3.60), (3.62), (3.64) and (3.66), we obtain

$$f_4 = \sum_{r_k=1}^{D-1}\sum_{r_l=1}^{D-1}\sum_{\delta_k=r_k}^{D-1}\sum_{\delta_l=r_l}^{D-1}\Big(\mathbf{d}_{m_1}(\delta_k - r_k, \delta_l - r_l)\mathbf{w}_{m_2}(\delta_k, \delta_l)\Big)^2.$$

By the above explicit formulas of $f_i, 1 \le i \le 4$, we have f_1, f_2, f_3, f_4 are all independent of k_1, l_1 and N. They are only related with m_1, m_2 for fixed $\mathbf{d}$ and $\mathbf{m}$. Thus, we are able to denote $f(k_1, l_1, m_1, m_2)$ as $f(m_1, m_2)$ for simplicity. Consequently,

$$\begin{aligned}\frac{1}{N^2}\|(\mathbf{D}_{\text{cir}}^N)^T\mathbf{W}_{\text{cir}}^N\|_F^2 =& \frac{1}{N^2}\sum_{k_1=0}^{N-1}\sum_{l_1=0}^{N-1}\sum_{k_2=0}^{N-1}\sum_{l_2=0}^{N-1}\sum_{m_1=1}^{M}\sum_{m_2=1}^{M}\Big(\mathbf{B}_{\text{coh}}(k_1, l_1, m_1; k_2, l_2, m_2)\Big)^2\\ =&\frac{1}{N^2}\sum_{k_1=0}^{N-1}\sum_{l_1=0}^{N-1}\sum_{m_1=1}^{M}\sum_{m_2=1}^{M} f(k_1, l_1, m_1, m_2)\\ =&\frac{1}{N^2}\sum_{k_1=0}^{N-1}\sum_{l_1=0}^{N-1}\sum_{m_1=1}^{M}\sum_{m_2=1}^{M} f(m_1, m_2)\\ =&\frac{1}{N^2}\cdot N^2\cdot\sum_{m_1=1}^{M}\sum_{m_2=1}^{M} f(m_1, m_2) = \sum_{m_1=1}^{M}\sum_{m_2=1}^{M} f(m_1, m_2)\end{aligned}$$

Thus, $\frac{1}{N^2}\|(\mathbf{D}_{\text{cir}}^N)^T\mathbf{W}_{\text{cir}}^N\|_F^2$ is dependent of N:

$$\frac{1}{N^2}\|(\mathbf{D}_{\text{cir}}^N)^T\mathbf{W}_{\text{cir}}^N\|_F^2 = \frac{1}{(2D-1)^2}\|(\mathbf{D}_{\text{cir}}^{2D-1})^T\mathbf{W}_{\text{cir}}^{2D-1}\|_F^2, \quad \forall N \ge 2D-1, \tag{3.67}$$

which implies $\mathcal{W}_{\text{cir}}^N = \mathcal{W}_{\text{cir}}^{2D-1}, \forall N \ge 2D-1$. □

3.F.2 Proof of Conclusion 2.

Before proving Conclusion 2, let us analyze the relationship between $\mathbf{D}_{\text{conv}}^N$ and $\mathbf{D}_{\text{cir}}^{N+D-1}$.

Similar to $\mathbf{D}_{\text{cir}}$, we use (i, j) as the row index and (k, l, m) as the column index of $\mathbf{D}_{\text{conv}}$. For $0 \le i, j \le N-1, 1 \le m \le M$,

$$\mathbf{D}_{\text{cir}}^{N+D-1}(i, j; k, l, m) = \mathbf{D}_{\text{conv}}^N(i, j; k, l, m) = \begin{cases}\mathbf{d}_m(k-i, l-j), & 0 \le k-i, l-j \le D-1\\ 0, & k, l \text{ taken as others}\end{cases} \tag{3.68}$$

Matrix $\mathbf{D}_{\text{cir}}^{N+D-1}$ is of dimension $(N+D-1)^2 \times (N+D-1)^2 M$, where $0 \le i, j \le N+D-2$; matrix $\mathbf{D}_{\text{conv}}^N$ is of dimension $(N)^2 \times (N+D-1)^2 M$, where $0 \le i, j \le N-1$. Thus, $\mathbf{D}_{\text{conv}}^N$ is a block in $\mathbf{D}_{\text{cir}}^{N+D-1}$, i.e.,

$$\mathbf{D}_{\text{cir}}^{N+D-1} = \begin{bmatrix} \mathbf{D}_{\text{conv}}^{N} \\ \Delta_{\mathbf{D}}^{N} \end{bmatrix}.$$

The matrix $\Delta_{\mathbf{D}}^{N}$ is of dimension $((N+D-1)^2 - N^2) \times (N+D-1)^2 M$:

$$\Delta_{\mathbf{D}}^{N} = \left[\mathbf{D}_{\text{cir}}^{N+D-1}(i,j;:,:,:)\right], \quad (i,j) \in \mathcal{I}_{\Delta}$$

where

$$\begin{aligned}
\mathcal{I}_{\Delta} =& \mathcal{I}_1 \cup \mathcal{I}_2 \cup \mathcal{I}_3 \\
\mathcal{I}_1 =& \{(i,j) | N \le i \le N+D-2,\ 0 \le j \le N-1\} \\
\mathcal{I}_2 =& \{(i,j) | 0 \le i \le N-1,\ N \le j \le N+D-2\} \\
\mathcal{I}_3 =& \{(i,j) | N \le i \le N+D-2,\ N \le j \le N+D-2\}.
\end{aligned}$$

Similarly,

$$\mathbf{W}_{\text{cir}}^{N+D-1} = \begin{bmatrix} \mathbf{W}_{\text{conv}}^{N} \\ \Delta_{\mathbf{W}}^{N} \end{bmatrix}, \quad \Delta_{\mathbf{W}}^{N} = \left[\mathbf{W}_{\text{cir}}^{N+D-1}(i,j;:,:,:)\right], \quad (i,j) \in \mathcal{I}_{\Delta}.$$

Then,

$$(\mathbf{D}_{\text{cir}}^{N+D-1})^T \mathbf{W}_{\text{cir}}^{N+D-1} = (\mathbf{D}_{\text{conv}}^{N})^T \mathbf{W}_{\text{conv}}^{N} + (\Delta_{\mathbf{D}}^{N})^T \Delta_{\mathbf{W}}^{N}. \tag{3.69}$$

Lemma 6. *For any $(i,j) \in \mathcal{I}_{\Delta}$, one has*

$$\|\mathbf{D}_{\text{cir}}^{N+D-1}(i,j;:,:,:)\|_2^2 = \|\mathbf{d}\|_2^2, \tag{3.70}$$

$$\|\mathbf{W}_{\text{cir}}^{N+D-1}(i,j;:,:,:)\|_2^2 = \|\mathbf{w}\|_2^2. \tag{3.71}$$

Proof. Equation (3.55) implies that, for $(i,j) \in \mathcal{I}_1$, $1 \le m \le M$,

$$\mathbf{D}_{\text{cir}}^{N+D-1}(i,j;k,l,m) = \begin{cases} \mathbf{d}_m(k-i, l-j), & i \le k \le N+D-2, \text{ and} \\ & j \le l \le j+D-1 \\ \mathbf{d}_m(k-i+N+D-1, l-j), & 0 \le k \le i-N, j \le l \le j+D-1 \\ 0, & k,l \text{ taken as others} \end{cases}$$

Thus, for any $(i,j) \in \mathcal{I}_1$,

$$\|\mathbf{D}_{\text{cir}}^{N+D-1}(i,j;:,:,:)\|_2^2 = \sum_{k=0}^{N+D-2} \sum_{l=0}^{N+D-2} \sum_{m=1}^{M} \left|\mathbf{D}_{\text{cir}}^{N+D-1}(i,j;k,l,m)\right|^2 = \|\mathbf{d}\|_2^2$$

Similarly,

$$\|\mathbf{D}_{\text{cir}}^{N+D-1}(i,j;:,:,:)\|_2^2 = \|\mathbf{d}\|_2^2, \quad (i,j) \in \mathcal{I}_2 \cup \mathcal{I}_3.$$

Equation (3.70) is proved. With the same argument, equation (3.71) is also proved. □

Lemma 7. *If $N \geq 2D-1$, we have*

$$\|(\Delta_{\mathbf{D}}^N)^T \Delta_{\mathbf{W}}^N\|_F^2 \leq \big(2N(D-1)+(D-1)^2\big)(2D-1)^2\|\mathbf{d}\|_2^2\|\mathbf{w}\|_2^2. \tag{3.72}$$

Proof. For simplicity, we denote two row vectors:

$$\mathbf{d}_{i,j} \triangleq \mathbf{D}_{\text{cir}}^{N+D-1}(i,j;:,:,:) \in \mathbb{R}^{1\times(N+D-1)^2M}$$
$$\mathbf{w}_{i,j} \triangleq \mathbf{W}_{\text{cir}}^{N+D-1}(i,j;:,:,:) \in \mathbb{R}^{1\times(N+D-1)^2M}$$

Then,

$$\|(\Delta_{\mathbf{D}}^N)^T \Delta_{\mathbf{W}}^N\|_F^2 = \left\| \sum_{(i,j)\in\mathcal{I}_\Delta} \mathbf{d}_{i,j}^T\mathbf{w}_{i,j}\right\|_F^2 = \sum_{(i_1,j_1)\in\mathcal{I}_\Delta}\sum_{(i_2,j_2)\in\mathcal{I}_\Delta} \left\langle \mathbf{d}_{i_1,j_1}^T\mathbf{w}_{i_1,j_1}, \mathbf{d}_{i_2,j_2}^T\mathbf{w}_{i_2,j_2}\right\rangle_F,$$

where

$$\left\langle \mathbf{d}_{i_1,j_1}^T\mathbf{w}_{i_1,j_1}, \mathbf{d}_{i_2,j_2}^T\mathbf{w}_{i_2,j_2}\right\rangle_F = \text{trace}\Big(\mathbf{w}_{i_1,j_1}^T\mathbf{d}_{i_1,j_1}\mathbf{d}_{i_2,j_2}^T\mathbf{w}_{i_2,j_2}\Big) = (\mathbf{d}_{i_1,j_1}\mathbf{d}_{i_2,j_2}^T)\cdot(\mathbf{w}_{i_1,j_1}\mathbf{w}_{i_2,j_2}^T).$$

Since

$$\mathbf{d}_{i_1,j_1}\mathbf{d}_{i_2,j_2}^T = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\sum_{m=1}^{M} = \mathbf{D}_{\text{cir}}^{N+D-1}(i_1,j_1;k,l,m)\mathbf{D}_{\text{cir}}^{N+D-1}(i_2,j_2;k,l,m),$$

with the same argument in Lemma 4, we have: $\mathbf{d}_{i_1,j_1}\mathbf{d}_{i_2,j_2}^T \neq 0$ implies

$$i_2 \in \mathcal{I}'_\Delta \triangleq \{i | 0 \leq (i_1 - i)_{\text{mod}(N+D-1)} \leq D-1 \text{ or } 0 \leq (i-i_1)_{\text{mod}(N+D-1)} \leq D-1\}$$

$$j_2 \in \mathcal{J}'_\Delta \triangleq \{j | 0 \leq (j_1 - j)_{\text{mod}(N+D-1)} \leq D-1 \text{ or } 0 \leq (j-j_1)_{\text{mod}(N+D-1)} \leq D-1\}$$

Then

$$\begin{aligned}\|(\Delta_{\mathbf{D}}^N)^T \Delta_{\mathbf{W}}^N\|_F^2 &= \sum_{(i_1,j_1)\in\mathcal{I}_\Delta}\sum_{i_2\in\mathcal{I}'_\Delta}\sum_{j_2\in\mathcal{J}'_\Delta}(\mathbf{d}_{i_1,j_1}\mathbf{d}_{i_2,j_2}^T)\cdot(\mathbf{w}_{i_1,j_1}\mathbf{w}_{i_2,j_2}^T)\\ &\leq \sum_{(i_1,j_1)\in\mathcal{I}_\Delta}\sum_{i_2\in\mathcal{I}'_\Delta}\sum_{j_2\in\mathcal{J}'_\Delta}\|\mathbf{d}\|_2^2\|\mathbf{w}\|_2^2\\ &=|\mathcal{I}_\Delta|\cdot|\mathcal{I}'_\Delta|\cdot|\mathcal{J}'_\Delta|\cdot\|\mathbf{d}\|_2^2\|\mathbf{w}\|_2^2\\ &=\big(2N(D-1)+(D-1)^2\big)(2D-1)^2\|\mathbf{d}\|_2^2\|\mathbf{w}\|_2^2,\end{aligned}$$

where the inequality in the second line follows from (3.70) and (3.71). Inequality (3.72) is proved. □

With these preparations, we can prove Theorem 7, Conclusion 2 now.

Proof of Theorem 7, Conclusion 2. Define set

$$\mathcal{W}_{\text{normal}} = \Big\{ \mathbf{w} \in \mathbb{R}^{D^2 M} \Big| \mathbf{w}_m \cdot \mathbf{d}_m = 1,\ \forall m = 1, \cdots, M \Big\}. \tag{3.73}$$

Since $\mathbf{d} \in \mathcal{W}_{\text{normal}}$, the set is nonempty:

$$\mathcal{W}_{\text{normal}} \neq \varnothing. \tag{3.74}$$

Define functions $F_{\text{conv}}^N : \mathbb{R}^{D^2 M} \to \mathbb{R}$, $F_{\text{cir}}^N : \mathbb{R}^{D^2 M} \to \mathbb{R}$.

$$\begin{aligned} F_{\text{conv}}^N(\mathbf{w}) &= \frac{1}{N+D-1} \Big\| \big(\mathbf{D}_{\text{conv}}^N(\mathbf{d})\big)^T \mathbf{W}_{\text{conv}}^N(\mathbf{w}) \Big\|_F + \iota_{\mathcal{W}_{\text{normal}}}(\mathbf{w}) \\ F_{\text{cir}}^N(\mathbf{w}) &= \frac{1}{N} \Big\| (\mathbf{D}_{\text{cir}}^N(\mathbf{d}))^T \mathbf{W}_{\text{cir}}^N(\mathbf{w}) \Big\|_F + \iota_{\mathcal{W}_{\text{normal}}}(\mathbf{w}) \end{aligned}$$

By the definitions of $\mathcal{W}_{\text{conv}}^N, \mathcal{W}_{\text{cir}}^N$, we have

$$\mathcal{W}_{\text{conv}}^N = \arg\min_{\mathbf{w}} F_{\text{conv}}^N(\mathbf{w}), \quad \mathcal{W}_{\text{cir}}^N = \arg\min_{\mathbf{w}} F_{\text{cir}}^N(\mathbf{w})$$

Step 1: Proving $F_{\text{conv}}^N(\mathbf{w})$ uniformly converges to $F_{\text{cir}}^{2D-1}(\mathbf{w})$ on $X \cap \mathcal{W}_{\text{normal}}$ for any compact set $X \subset \mathbb{R}^{D^2 M}$.

We arbitrarily choose such a compact set X. Based on (3.67), (3.69) and (3.72), one has, for all $\mathbf{w} \in X \cap \mathcal{W}_{\text{normal}}$,

$$\begin{aligned} |F_{\text{conv}}^N(\mathbf{w}) - F_{\text{cir}}^{2D-1}(\mathbf{w})| =& |F_{\text{conv}}^N(\mathbf{w}) - F_{\text{cir}}^{N+D-1}(\mathbf{w})| \\ =& \frac{1}{N+D-1} \Big| \Big\| (\mathbf{D}_{\text{cir}}^{N+D-1})^T \mathbf{W}_{\text{cir}}^{N+D-1} \Big\|_F - \Big\| (\mathbf{D}_{\text{conv}}^N)^T \mathbf{W}_{\text{conv}}^N \Big\|_F \Big| \\ \leq& \frac{1}{N+D-1} \| (\Delta_{\mathbf{D}}^N)^T \Delta_{\mathbf{W}}^N \|_F \\ \leq& \frac{\sqrt{\big(2N(D-1) + (D-1)^2\big)(2D-1)}}{N+D-1} \|\mathbf{d}\|_2 \|\mathbf{w}\|_2 \\ \leq& \frac{(2D-1)\sqrt{2(D-1)}}{\sqrt{N+D-1}} \|\mathbf{d}\|_2 \|\mathbf{w}\|_2. \end{aligned}$$

Thus, there exists a constant $B > 0$, which is independent of N, such that

$$|F_{\text{conv}}^N(\mathbf{w}) - F_{\text{cir}}^{2D-1}(\mathbf{w})| \leq \frac{B}{\sqrt{N}} \sup_{\mathbf{w} \in X \cap \mathcal{W}_{\text{normal}}} \|\mathbf{w}\|, \quad \forall\ \mathbf{w} \in X \cap \mathcal{W}_{\text{normal}}. \tag{3.75}$$

Step 2: Proving $F^N_{\text{conv}}(\mathbf{w})$ epigraphically converges[10] to $F^{2D-1}_{\text{cir}}(\mathbf{w})$.

We want to show, at each point $\mathbf{w}$ it holds that

$$\liminf_{N\to\infty} F^N_{\text{conv}}(\mathbf{w}^N) \geq F^{2D-1}_{\text{cir}}(\mathbf{w}) \quad \text{for every sequence } \mathbf{w}^N \to \mathbf{w} \tag{3.76}$$

$$\limsup_{N\to\infty} F^N_{\text{conv}}(\mathbf{w}^N) \leq F^{2D-1}_{\text{cir}}(\mathbf{w}) \quad \text{for some sequence } \mathbf{w}^N \to \mathbf{w} \tag{3.77}$$

Firstly, we prove (3.76). We arbitrarily pick a sequence $\{\mathbf{w}^N\}_{N=0}^{\infty}$ such that $\mathbf{w}^N \to \mathbf{w}$.

If $\mathbf{w} \notin \mathcal{W}_{\text{normal}}$, $F^{2D-1}_{\text{cir}}(\mathbf{w}) = +\infty$. Since $\mathcal{W}_{\text{normal}}$ is a closed set, there exists a N^+ such that $\mathbf{w}^N \notin \mathcal{W}_{\text{normal}}$ for all $N \geq N^+$. Thus, one has $F^N_{\text{conv}}(\mathbf{w}^N) = +\infty$ for all $N \geq N^+$, i.e.,

$$\liminf_{N\to\infty} F^N_{\text{conv}}(\mathbf{w}^N) = F^{2D-1}_{\text{cir}}(\mathbf{w}) = +\infty.$$

If $\mathbf{w} \in \mathcal{W}_{\text{normal}}$, two cases should be considered. The first case is that any subsequences of $\{\mathbf{w}^N\}_{N=0}^{\infty}$ are not kept within $\mathcal{W}_{\text{normal}}$, i.e., there exists a N^+ such that $\mathbf{w}^N \notin \mathcal{W}_{\text{normal}}$ for all $N \geq N^+$. Then we have

$$\liminf_{N\to\infty} F^N_{\text{conv}}(\mathbf{w}^N) = +\infty > F^{2D-1}_{\text{cir}}(\mathbf{w}).$$

The second case is that there exists a subsequence $\{\mathbf{w}^{N_k}\}_{k=0}^{\infty} \subset \{\mathbf{w}^N\}_{N=0}^{\infty}$ such that

$$\mathbf{w}^{N_k} \in \mathcal{W}_{\text{normal}}, \quad \forall k = 0, 1, 2, \cdots .$$

Since $\mathbf{w}^N$ converges to $\mathbf{w}$, any subsequences should be Cauchy. Given any Cauchy sequence $\{\mathbf{w}^{N_k}\}_{k=0}^{\infty}$ in finite dimensional Euclidean space, there exists a compact set X such that

$$\mathbf{w}^{N_k} \in X, \quad \forall k = 0, 1, 2, \cdots$$

Let $B' = \sup_{\mathbf{w}\in X\cap\mathcal{W}_{\text{normal}}} \|\mathbf{w}\|$. By (3.75), we obtain

$$\begin{aligned}|F^{N_k}_{\text{conv}}(\mathbf{w}^{N_k}) - F^{2D-1}_{\text{cir}}(\mathbf{w})| \leq& |F^{N_k}_{\text{conv}}(\mathbf{w}^{N_k}) - F^{2D-1}_{\text{cir}}(\mathbf{w}^{N_k})| + |F^{2D-1}_{\text{cir}}(\mathbf{w}^{N_k}) - F^{2D-1}_{\text{cir}}(\mathbf{w})| \\ \leq& \frac{BB'}{\sqrt{N_k}} + |F^{2D-1}_{\text{cir}}(\mathbf{w}^{N_k}) - F^{2D-1}_{\text{cir}}(\mathbf{w})|.\end{aligned}$$

[10]Epigraphic convergence is a standard tool to prove the convergence of a sequence of minimization problems. The definition of epigraphic convergence refers to Definition 7.1 and Proposition 7.2 in [RW09].

For any $\epsilon > 0$, by the continuity of F_{cir}^{2D-1}, we are able to find a $K > 0$ such that $|F_{\text{cir}}^{2D-1}(\mathbf{w}^{N_k}) - F_{\text{cir}}^{2D-1}(\mathbf{w})| < \epsilon$ for all $k \geq K$. Pick a K' such that $N_{K'} \geq (BB'/\epsilon)^2$. Then, for all $k \geq \max(K, K')$, we have $|F_{\text{conv}}^{N_k}(\mathbf{w}^{N_k}) - F_{\text{cir}}^{2D-1}(\mathbf{w})| < 2\epsilon$, i.e.,

$$\lim_{k\to\infty} F_{\text{conv}}^{N_k}(\mathbf{w}^{N_k}) = F_{\text{cir}}^{2D-1}(\mathbf{w}).$$

The above conclusion holds for all subsequences $\{\mathbf{w}^{N_k}\}_{k=0}^{\infty} \subset \mathcal{W}_{\text{normal}}$. $F_{\text{cir}}^{2D-1}(\mathbf{w})$ is an accumulation point of $\{F_{\text{conv}}^{N}(\mathbf{w}^{N})\}_{N=0}^{\infty}$. All the other accumulation points of $\{F_{\text{conv}}^{N}(\mathbf{w}^{N})\}_{N=0}^{\infty}$ must be $+\infty$ because $F_{\text{conv}}^{N}(\mathbf{w}) = F_{\text{cir}}^{2D-1}(\mathbf{w}) = +\infty$ for all $\mathbf{w} \notin \mathcal{W}_{\text{normal}}$. Thus,

$$\liminf_{N\to\infty} F_{\text{conv}}^{N}(\mathbf{w}^{N}) = F_{\text{cir}}^{2D-1}(\mathbf{w}) < +\infty.$$

Secondly, we prove (3.77). We set $\mathbf{w}^N = \mathbf{w}$ for all $N = 0, 1, 2, \cdots$. Then (3.77) is a direct result of (3.75).

Step 3: proving (3.36). Define

$$G(\mathbf{w}) = \left\|(\mathbf{D}_{\text{cir}}^{2D-1})^T \mathbf{W}_{\text{cir}}^{2D-1}\right\|_F^2.$$

We want to show that $G(\mathbf{w})$ is strongly convex.

Let $\tilde{\mathbf{w}}_i \in \mathbb{R}^{(2D-1)^2}$ be the ith column of $\mathbf{W}_{\text{cir}}^{2D-1}$, i.e.,

$$\mathbf{W}_{\text{cir}}^{2D-1} = \left[\tilde{\mathbf{w}}_1, \tilde{\mathbf{w}}_2, \cdots, \tilde{\mathbf{w}}_{(2D-1)^2 M}\right]$$

Then

$$G(\mathbf{w}) = \sum_{i=1}^{(2D-1)^2 M} (\tilde{\mathbf{w}}_i)^T \left(\mathbf{D}_{\text{cir}}^{2D-1} (\mathbf{D}_{\text{cir}}^{2D-1})^T\right) \tilde{\mathbf{w}}_i.$$

Let $\tilde{\mathbf{w}} \in \mathbb{R}^{(2D-1)^4 M}$ vectorize $\mathbf{W}_{\text{cir}}^{2D-1}$, i.e.,

$$\tilde{\mathbf{w}} = \left[(\tilde{\mathbf{w}}_1)^T, (\tilde{\mathbf{w}}_2)^T, \cdots, (\tilde{\mathbf{w}}_{(2D-1)^2 M})^T\right]^T.$$

Then $G(\mathbf{w})$ can be written as a quadratic form of $\tilde{\mathbf{w}}$:

$$G(\mathbf{w}) = \tilde{\mathbf{w}}^T Q \tilde{\mathbf{w}},$$

where

$$Q = \underbrace{\begin{bmatrix} \left(\mathbf{D}_{\text{cir}}^{2D-1}(\mathbf{D}_{\text{cir}}^{2D-1})^T\right) & & \\ & \cdots & \\ & & \left(\mathbf{D}_{\text{cir}}^{2D-1}(\mathbf{D}_{\text{cir}}^{2D-1})^T\right) \end{bmatrix}}_{\text{totally } (2D-1)^2 M \text{ diagonal blocks}}.$$

As long as at least one of the matrices $\{\mathbf{D}_{\text{cir},0}^{2D-1}, \cdots, \mathbf{D}_{\text{cir},M-1}^{2D-1}\}$ is non-singular, $\mathbf{D}_{\text{cir}}^{2D-1}$ is full row rank, which implies that $\mathbf{D}_{\text{cir}}^{2D-1}(\mathbf{D}_{\text{cir}}^{2D-1})^T$ is non-singular. Then Q is positive definite.

The transform between $\mathbf{w}$ and $\tilde{\mathbf{w}}$ is linear. We denote the transform as T, i.e.,

$$\tilde{\mathbf{w}} = T\mathbf{w}.$$

It's trivial that $\|\tilde{\mathbf{w}}\|_2^2 = 0$ implies $\|\mathbf{W}_{\text{cir}}^{2D-1}\|_F^2 = 0$. By the definition of $\mathbf{W}_{\text{cir}}^{2D-1}$, $\|\mathbf{W}_{\text{cir}}^{2D-1}\|_F^2 = 0$ implies $\|\mathbf{w}\|_2^2 = 0$. Thus, linear operator T is full column rank. Thus, T^TQT is positive definite, and

$$G(\mathbf{w}) = \mathbf{w}^T(T^TQT)\mathbf{w}$$

is strongly convex. Then $F_{\text{cir}}^{2D-1}(\mathbf{w}) = \sqrt{G(\mathbf{w})} + \iota_{\mathcal{W}_{\text{normal}}}(\mathbf{w})$ has only one minimizer, i.e., $\mathcal{W}_{\text{cir}}^{2D-1}$ involves only a unique element.

Now we check the conditions of Propositions 7.32(c) and 7.33 in [RW09] to apply them.

1. $F_{\text{conv}}^N \xrightarrow{\text{e}} F_{\text{cir}}^{2D-1}$. This is proved in Step 2.

2. F_{cir}^{2D-1} is level bounded. Since $G(\mathbf{w})$ is strongly convex, $F_{\text{cir}}^{2D-1}(\mathbf{w}) = \sqrt{G(\mathbf{w})} + \iota_{\mathcal{W}_{\text{normal}}}(\mathbf{w})$ must be level bounded.

3. $F_{\text{cir}}^{2D-1} \not\equiv +\infty$. Since $\mathcal{W}_{\text{normal}}$ is nonempty (3.74), dom $F_{\text{cir}}^{2D-1} \neq \varnothing$, F_{cir}^{2D-1} is not constantly $+\infty$.

4. All the level set of F_{conv}^N are connected. This can be derived from the convexity of F_{conv}^N.

5. F_{cir}^{2D-1} and F_{conv}^N are all lower semi-continuous and proper. This condition follows from the fact that the functions F_{cir}^{2D-1} and F_{conv}^N are all continuous functions defined on a nonempty closed convex domain $\mathcal{W}_{\text{normal}}$.

Applying Proposition 7.32(c), we have $\{F^N_{\text{conv}}\}$ is eventually level bounded. If we arbitrarily pick a $\mathbf{w}^N \in \mathcal{W}^N_{\text{conv}}$ and let $\mathbf{w}_{\text{cir}}$ be the unique point in $\mathcal{W}^{2D-1}_{\text{cir}}$. Applying Proposition 7.33, we have $\mathbf{w}^N \to \mathbf{w}_{\text{cir}}$. By Definition 4.1 in [RW09], we obtain the convergence of the sequence of sets $\{\mathcal{W}^N_{\text{conv}}\}$: $\lim_{N\to\infty} \mathcal{W}^N_{\text{conv}} = \mathcal{W}^{2D-1}_{\text{cir}}$. □

Appendix 3.G Discussion of Definition 2 (3.22)

In this section, we want to numerically show that, given typical $\mathbf{D}$ and s, there is a $\bar{\sigma}_{\min} > 0$ such that a random generated matrix $\mathbf{W} \in \bar{\mathcal{W}}(\mathbf{D}, s, \bar{\sigma}_{\min})$. However, given $\mathbf{D}$ and $\mathbf{W}$, it's intractable to completely check (3.22):

$$\sigma_{\min}\Big(\mathbf{I} - (\mathbf{W}_{:,\mathbb{S}})^T \mathbf{D}_{:,\mathbb{S}}\Big) \geq \bar{\sigma}_{\min}, \forall \mathbb{S} \text{ with } 2 \leq |\,\mathbb{S}\,| \leq s.$$

The reason is that there are extremely large amount of possible $\mathbb{S}$ s. For example, we take $M = 250, N = 500, s = 50$. There are totally

$$\binom{500}{50} + \binom{500}{49} + \cdots + \binom{500}{2}$$

possible $\mathbb{S}$s satisfying $2 \leq |\,\mathbb{S}\,| \leq s$. It's impossible to check (3.22) on all possible $\mathbb{S}$s.

Instead of checking all possible $\mathbb{S}$s, we sample 5000 $\mathbb{S}$s from the whole set:

$$\mathcal{S}' \subset \mathcal{S} = \{\mathbb{S} : \mathbb{S} \subset \{1, 2, \cdots, 500\} | 2 \leq |S| \leq s\},$$

where $\mathcal{S}'$ is the set of all the samples. Then we estimate $\bar{\sigma}_{\min}$ with the following quantity:

$$\bar{\sigma}'(\mathbf{D}, \mathbf{W}) = \min_{\mathbb{S}\in\mathcal{S}'} \Big\{\sigma_{\min}\Big(\mathbf{I} - (\mathbf{W}_{:,\mathbb{S}})^T \mathbf{D}_{:,\mathbb{S}}\Big)\Big\}$$

Furthermore, we use the same $\mathbf{D}$ as that in Section 3.6 and generate 1000 $\mathbf{W}$s with each entry i.i.d sampled from the normal distribution. Then we normalize each column of the generated $\mathbf{W}$s. This technique is commonly used in sparse coding. Finally, we report the distribution of $\bar{\sigma}'(\mathbf{D}, \mathbf{W})$ with the fixed $\mathbf{D}$ and the 1000 sampled $\mathbf{W}$s in Figure 3.10.

Figure 3.10 demonstrates that, with the fixed $\mathbf{D}$, most of the random generated $\mathbf{W}$s have a $\bar{\sigma}'(\mathbf{D}, \mathbf{W})$ within the interval $[0.25, 0.35]$. Thus, the numerical results support our claim:

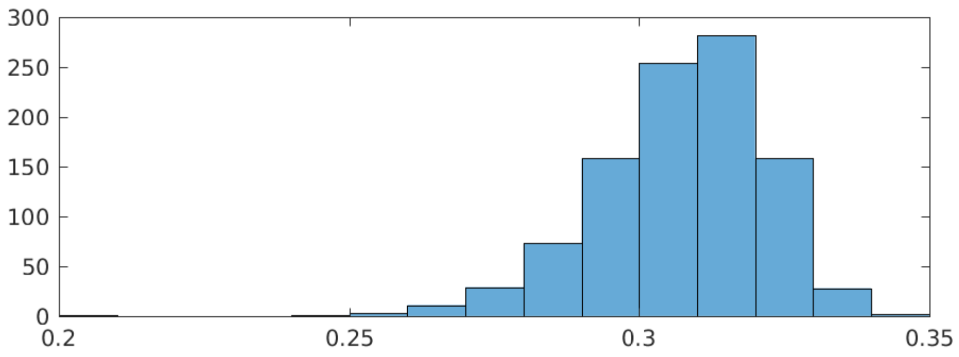


Figure 3.10: Discussion of Definition 2: distribution of $\bar{\sigma}'(\mathbf{D},\mathbf{W})$ on random generated $\mathbf{W}$s.

with high probability, a random generated $\mathbf{W}$ satisfies

$$\min_{\mathbb{S}\in\mathcal{S}}\left\{\sigma_{\min}\Big(\mathbf{I}-(\mathbf{W}_{:,\mathbb{S}})^T\mathbf{D}_{:,\mathbb{S}}\Big)\right\}\geq\bar{\sigma}_{\min}>0,$$

that is, $\mathbf{W}\in\bar{\mathcal{W}}(\mathbf{D},s,\bar{\sigma}_{\min})$.

Appendix 3.H An efficient algorithm to calculate analytic weights

3.H.1 An efficient algorithm to solve (3.27)

In this section, we introduce an algorithm to solve (3.27) (we copy (3.27) below to facilitate reading):

$$\min_{\mathbf{W}\in\mathbb{R}^{N\times M}}\left\|\mathbf{W}^T\mathbf{D}\right\|_F^2,\quad \text{s.t. } (\mathbf{W}_{:,m})^T\mathbf{D}_{:,m}=1,\ \forall m=1,2,\cdots,M,$$

By the definition of the Frobenius norm, it holds that

$$\|\mathbf{W}^T\mathbf{D}\|_F^2=\|(\mathbf{W}^T\mathbf{D})^T\|_F^2=\|\mathbf{D}^T\mathbf{W}\|_F^2. \tag{3.78}$$

Thus, the above problem is equivalent with

$$\min_{\mathbf{W}\in\mathbb{R}^{N\times M}}\left\|\mathbf{D}^T\mathbf{W}\right\|_F^2,\quad \text{s.t. } (\mathbf{D}_{:,m})^T\mathbf{W}_{:,m}=1,\ \forall m=1,2,\cdots,M.$$

We apply projected gradient descent (PGD) to solve the above problem. The gradient of $\|\mathbf{D}^T\mathbf{W}\|_F^2$ is $\nabla\|\mathbf{D}^T\mathbf{W}\|_F^2=\mathbf{D}\mathbf{D}^T\mathbf{W}$. Denote the set by

$$\mathcal{W}=\{\mathbf{W}\in\mathbb{R}^{N\times M}|(\mathbf{D}_{:,m})^T\mathbf{W}_{:,m}=1,\ \forall m=1,2,\cdots,M.\}$$

Then the projection onto $\mathcal{W}$ can be calculated by

$$\mathrm{Proj}_{\mathcal{W}}(\mathbf{W}) = \mathbf{W} + \Delta\mathbf{W},\ \Delta\mathbf{W} = \Big[(1 - (\mathbf{D}_{:,1})^T\mathbf{W}_{:,1})\mathbf{W}_{:,1},\ \cdots,\ (1 - (\mathbf{D}_{:,M})^T\mathbf{W}_{:,M})\mathbf{W}_{:,M}\Big]$$

With these formulas, we are able to write down the PGD, which is listed in Algorithm 5.

Algorithm 5: Projected gradient descent for solving (3.27)

```
  Input: Dictionary $\mathbf{D} \in \mathbb{R}^{N\times M}$.
  Initialize: Let $\mathbf{W}^{(0)} = \mathbf{D}$.
1 for $j = 0, 1, 2, \ldots$ until convergence do
2     Update $\mathbf{W}$ by $\mathbf{W}^{j+1} = \mathrm{Proj}_{\mathcal{W}}\Big(\mathbf{W}^j - \eta\mathbf{D}(\mathbf{D})^T\mathbf{W}^j\Big)$.
3 end
  Output: $\mathbf{W}^J$, where $J$ is the last iterate.
```

In each step, calculating the gradient has the complexity of $O(N^2M)$ because $\mathbf{D}\mathbf{D}^T$ can be pre-computed. Calculating the projection takes $O(NM)$ time consumptions. Due to the objective function to minimize in (3.27) is restricted strongly convex, Algorithm 5 is linear convergent [ZC15]. To get an ϵ-accurate solution, PGD takes $O(\log(1/\epsilon))$ steps. Thus, the complexity of Algorithm 5 is $O(\log(1/\epsilon)N^2M)$. We should note that the bounds given in Table 3.1 are the number of parameters to train, not the training complexity. The training complexity can be estimated by "Number of iterations $\times$ complexity of back-propagation", i.e., $O(IBKNM)$,where I is the number of iterations for training, B is the batch size , and K is the number of layers. Actually, Algorithm 5 (Stage 1) only takes a few seconds on an example of $\mathbf{D}$: 250×500, while the training process (Stage 2) of, for example, ALISTA, takes around 0.1 hours.

3.H.2 An efficient algorithm to solve (3.35)

In this section, we introduce an algorithm to solve (3.35) (we copy (3.35) below to facilitate reading):

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^{D^2M} \\ \mathbf{w}_m\cdot\mathbf{d}_m=1,\ 1\le m\le M}} \Big\|\big(\mathbf{W}^N_{\mathrm{cir}}(\mathbf{w})\big)^T\mathbf{D}^N_{\mathrm{cir}}(\mathbf{d})\Big\|_F^2.$$

Similarly, by (3.78), the above problem is equivalent with

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^{D^2M}\\ \mathbf{d}_m\cdot\mathbf{w}_m=1,\ 1\le m\le M}} \left\|\left(\mathbf{D}^N_{\text{cir}}(\mathbf{d})\right)^T \mathbf{W}^N_{\text{cir}}(\mathbf{w})\right\|_F^2. \tag{3.79}$$

Since the circular convolution is very efficient to calculate in the frequency domain, we consider solving (3.79) utilizing the fast Fourier transform (FFT).

Firstly, we introduce the operators $\mathbf{D}^N_{\text{cir}}(\mathbf{d}), \mathbf{W}^N_{\text{cir}}(\mathbf{w})$ in the frequency domain. To simplify the notation, we denote the operators as $\mathbf{D}^N_{\text{cir}}$ and $\mathbf{W}^N_{\text{cir}}$ respectively. Let $\mathcal{F}$ be the FFT operator. Thus, $\mathbf{b} = \mathbf{D}^N_{\text{cir}}\mathbf{x}$ is equivalent with

$$\mathcal{F}\mathbf{b} = \mathcal{F}\mathbf{D}^N_{\text{cir}}\mathcal{F}^H\mathcal{F}\mathbf{x}.$$

Let $\hat{\mathbf{b}} = \mathcal{F}\mathbf{b}, \hat{\mathbf{x}} = \mathcal{F}\mathbf{x}$ be the frequency domain signals, let $\hat{\mathbf{D}}^N_{\text{cir}} = \mathcal{F}\mathbf{D}^N_{\text{cir}}\mathcal{F}^H$ be the frequency domain operator. The above equation is:

$$\hat{\mathbf{b}} = \hat{\mathbf{D}}^N_{\text{cir}}\hat{\mathbf{x}}.$$

The frequency domain operator $\hat{\mathbf{D}}^N_{\text{cir}}$ is much cheaper to calculate than the operator $\mathbf{D}^N_{\text{cir}}$ in the spacial domain because it is block diagonal [Woh16e]. Specifically, we zero pad $\mathbf{d}$ to $N \times N$ and do FFT: $\hat{\mathbf{d}}_m = \text{FFT}\big(\text{zeropad}(\mathbf{d}_m, N-D)\big)$, then the above operator can be explicitly written as:

$$\hat{\mathbf{b}} = \sum_{m=1}^{M} \overline{\hat{\mathbf{d}}_m} \odot \hat{\mathbf{x}}_m,$$

where $\bar{\cdot}$ means complex conjugate. This is due to $\mathbf{D}^N_{\text{cir}}$ is actually cross-correlation, not convolution (see (3.29)). Cross-correlation is equal to the transpose of convolution. Thus, there should be complex conjugate in the frequency domain.

Further, since

$$\begin{aligned}\left\|\left(\hat{\mathbf{D}}^N_{\text{cir}}\right)^H\hat{\mathbf{W}}^N_{\text{cir}}\right\|_F^2 = \left\|\left(\mathcal{F}\mathbf{D}^N_{\text{cir}}\mathcal{F}^H\right)^H\mathcal{F}\mathbf{W}^N_{\text{cir}}\mathcal{F}^H\right\|_F^2 &= \left\|\mathcal{F}\left(\mathbf{D}^N_{\text{cir}}\right)^T\mathbf{W}^N_{\text{cir}}\mathcal{F}^H\right\|_F^2\\ &= \left\|\left(\mathbf{D}^N_{\text{cir}}\right)^T\mathbf{W}^N_{\text{cir}}\right\|_F^2,\end{aligned}$$

problem (3.79) is equivalent with

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^{D^2M}\\ \mathbf{d}_m\cdot\mathbf{w}_m=1,\ 1\le m\le M}} \left\|\left(\hat{\mathbf{D}}^N_{\text{cir}}\right)^H\hat{\mathbf{W}}^N_{\text{cir}}\right\|_F^2,$$

which can be efficiently solved by the frequency domain ISTA in [LGW17]. The details are outlined in Algorithm 6.

```
Algorithm 6: Frequency-domain ISTA for solving (3.35)
   Input: Dictionary $\mathbf{d} = [\mathbf{d}_1, \cdots, \mathbf{d}_M]^T,\ \mathbf{d}_m \in \mathbb{R}^{D^2},\ m = 1, 2, \cdots, M$.
   Initialize: Let $\mathbf{w}^{(0)} = \mathbf{d}$.
 1 for $j = 0, 1, 2, \ldots$ until convergence do
 2     Zeropad and FFT:
           $$\hat{\mathbf{w}}_m^j = \mathrm{FFT}\Big(\mathrm{zeropad}\big(\mathbf{w}_m^j, N - D\big)\Big), \quad m = 1, \cdots, M.$$
 3     Compute frequency domain gradient:
           $$(\nabla f)_m = \Big(\sum_{m=1}^{M} \hat{\mathbf{d}}_m \odot \bar{\hat{\mathbf{d}}}_m\Big) \odot \hat{\mathbf{w}}_m^j, \quad m = 1, \cdots, M,$$
       where $\bar{\cdot}$ represents the conjugate of a complex number.
 4     Compute the next iterate:
           $$\mathbf{w}_m^{j+1} = \mathrm{Proj}_{\mathcal{W}_{\mathrm{normal}}}\Big(\mathrm{IFFT}\big(\hat{\mathbf{w}}_m^j - \eta(\nabla f)_m\big)\Big), \quad m = 1, \cdots, M,$$
       where the set $\mathcal{W}_{\mathrm{normal}}$ is defined in (3.73).
 5 end
   Output: $\mathbf{w}^J$, where $J$ is the last iterate.
```

CHAPTER 4

Learning Regularizers

Many modern image processing algorithms recover or denoise an image through the optimization problem (introduced in Chapter 1)

$$\underset{\mathbf{x}\in\mathbb{R}^M}{\text{minimize}} \quad f(\mathbf{x}) + \gamma g(\mathbf{x}), \tag{4.1}$$

where $\mathbf{x} \in \mathbb{R}^M$ represents the image, $f(\mathbf{x})$ measures data fidelity, $g(\mathbf{x})$ regularizes the signal or image to be less noisy or less complex and $\gamma \geq 0$ is a parameter controls the balance between f and g.

The regularizer $g(\mathbf{x})$ plays a significant role in such problems. In the literature, plenty of *manually designed* regularizers have been developed, such as Total Variation [ROF92], Tikhonov regularization [Tik63, GHO99], Markov random field model-based priors [KS80], patch based prior [KB06, ZWZ18], etc. Recent developed *learning based* regularizers [SCH17, AMJ18, QSC18, KHS19] show better empirical results although the interpretability is still limited.

In this chapter, we study learning-based regularizers g in the framework of *Plug-and-Play* (introduced in Chapter 1) and target on the following problems:

- Is plug-and-play convergent? If the answer is positive, what assumptions should we make on the regularizers?
- Do commonly-used regularizers satisfy the assumptions required by convergence?
- How to guarantee the assumptions met by a learning-based regularizer?

The remainder of the chapter is organized as follows: Section 4.1 introduces the plug-and-play methods used in this chapter and the concept of fixed point; Section 4.2 describes related works; In Section 4.3, we establish linear convergence theories of plug-and-play methods and propose an assumption required by convergence; Section 4.4 provides a learning method to enforce the learned regularizers satisfy the assumption; empirical results in Section 4.5 support our theories and more applications can be found in Section 4.6; Section 4.7 concludes this chapter.

4.1 PNP-FBS/ADMM and their fixed points

We now present the two PnP methods we investigate in this work. We quickly note that although PNP-FBS and PNP-ADMM are distinct methods, they share the same fixed points by Remark 3.1 of [MMH17] and Proposition 3 of [SWK19].

We call the method

$$\mathbf{x}^{(k+1)} = H_\sigma(I - \alpha\nabla f)(\mathbf{x}^{(k)}) \tag{PNP-FBS}$$

for any $\alpha > 0$, plug-and-play forward-backward splitting (PNP-FBS) or plug-and-play proximal gradient method. H_σ is a learning-based denoiser (regularizer). We interpret PNP-FBS as a fixed-point iteration, and we say $\mathbf{x}^\star$ is a fixed point of PNP-FBS if

$$\mathbf{x}^\star = H_\sigma(I - \alpha\nabla f)(\mathbf{x}^\star).$$

Fixed points of PNP-FBS have a simple, albeit non-rigorous, interpretation. An image denoising algorithm must trade off the two goals of making the image agree with measurements and making the image less noisy. PNP-FBS applies $I - \alpha\nabla f$ and H_σ, each promoting such objectives, repeatedly in an alternating fashion. If PNP-FBS converges to a fixed point, we can expect the limit to represent a compromise. We call the method

$$\begin{aligned} \mathbf{x}^{(k+1)} &= H_\sigma(\mathbf{y}^{(k)} - \mathbf{u}^{(k)}) \\ \mathbf{y}^{(k+1)} &= \mathrm{Prox}_{\alpha f}(\mathbf{x}^{(k+1)} + \mathbf{u}^{(k)}) \\ \mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \mathbf{x}^{(k+1)} - \mathbf{y}^{(k+1)} \end{aligned} \tag{PNP-ADMM}$$

for any $\alpha > 0$, plug-and-play alternating directions method of multipliers (PNP-ADMM). We interpret PNP-ADMM as a fixed-point iteration, and we say $(\mathbf{x}^\star, \mathbf{u}^\star)$ is a fixed point of

PNP-ADMM if

$$\begin{aligned}\mathbf{x}^\star &= H_\sigma(\mathbf{x}^\star - \mathbf{u}^\star)\\ \mathbf{x}^\star &= \mathrm{Prox}_{\alpha f}(\mathbf{x}^\star + \mathbf{u}^\star).\end{aligned}$$

If we let $\mathbf{y}^{(k)} = \mathbf{x}^\star$ and $\mathbf{u}^{(k)} = \mathbf{u}^\star$ in (PNP-ADMM), then we get $\mathbf{x}^{(k+1)} = \mathbf{y}^{(k+1)} = \mathbf{x}^\star$ and $\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} = \mathbf{u}^\star$. We call the method

$$\begin{aligned}\mathbf{x}^{(k+1/2)} &= \mathrm{Prox}_{\alpha f}(\mathbf{z}^{(k)})\\ \mathbf{x}^{(k+1)} &= H_\sigma(2\mathbf{x}^{(k+1/2)} - \mathbf{z}^{(k)})\\ \mathbf{z}^{(k+1)} &= \mathbf{z}^{(k)} + \mathbf{x}^{(k+1)} - \mathbf{x}^{(k+1/2)}\end{aligned} \tag{PNP-DRS}$$

plug-and-play Douglas–Rachford splitting (PNP-DRS). We interpret PNP-DRS as a fixed-point iteration, and we say $\mathbf{z}^\star$ is a fixed point of PNP-DRS if

$$\begin{aligned}\mathbf{x}^\star &= \mathrm{Prox}_{\alpha f}(\mathbf{z}^\star)\\ \mathbf{x}^\star &= H_\sigma(2\mathbf{x}^\star - \mathbf{z}^\star).\end{aligned}$$

PNP-ADMM and PNP-DRS are equivalent. Although this is not surprising as the equivalence between convex ADMM and DRS is well known, we show the steps establishing equivalence in the appendix.

We introduce PNP-DRS as an analytical tool for analyzing PNP-ADMM. It is straightforward to verify that PNP-DRS can be written as $\mathbf{z}^{(k+1)} = T(\mathbf{z}^{(k)})$, where

$$T = \frac{1}{2}I + \frac{1}{2}(2H_\sigma - I)(2\mathrm{Prox}_{\alpha f} - I).$$

We use this form to analyze the convergence of PNP-DRS and translate the result to PNP-ADMM.

4.2 Related works

Plug-and-play: Practice. The first PnP method was the Plug-and-play ADMM proposed in [VBW13]. Since then, other schemes such as the primal-dual method [HST14, MMH17, Ono17], ADMM with increasing penalty parameter [BRE16, CWE17], generalized approximate message passing [MMB16], Newton iteration [BCS18], Fast Iterative Shrinkage-Thresholding Algorithm [KMW17, SXL18], (stochastic) forward-backward splitting [SWK19, SWK18, SXL18], and alternating minimization [DWY18] have been combined with the PnP technique.

PnP method reported empirical success on a large variety of imaging applications: bright field electron tomography [SVW16], camera image processing [HST14], compression-artifact reduction [DBE16], compressive imaging [TBF16], deblurring [TBF16, RGE16, WC17], electron microscopy [SVB17], Gaussian denoising [BCS18, DWY18], nonlinear inverse scattering [KMW17], Poisson denoising [RGE16], single-photon imaging [CWE17], super-resolution [BRE16, SVW16, CWE17], diffraction tomography [SWK19], Fourier ptychographic microscopy [SXL18], low-dose CT imaging [VBW13, HYW18, YST18, LRH19], hyperspectral sharpening [TBF17, TBF19], inpainting [Cha19, TG19], and superresolution [DWY18].

A wide range of denoisers have been used for the PnP framework. BM3D has been used the most [HST14, DBE16, RGE16, SVW16, CWE17, KMW17, Ono17, WC17], but other denoisers such as sparse representation [BRE16], non-local means [VBW13, HST14, SVW16, SVB17, Cha19], Gaussian mixture model [TBF16, TBF17, SF18, TBF19], Patch-based Wiener filtering [VBW13], nuclear norm minimization [KMW17], deep learning-based denoisers [MMH17, HYW18, YST18, TG19] and deep projection model based on generative adversarial networks [CLP17] have also been considered.

Plug-and-play: Theory. Compared to the empirical success, much less progress was made on the theoretical aspects of PnP optimization. [CWE17] analyzed convergence with a bounded denoiser assumption, establishing convergence using an increasing penalty parameter. [BCS18] provided an interpretation of fixed points via “consensus equilibrium”. [SVW16, SWK19, TBF17, Cha19, TBF19] proved convergence of PNP-ADMM and PNP-FBS with the assumption that the denoiser is (averaged) nonexpansive by viewing the methods to be fixed-point iterations. The nonexpansiveness assumption is not met with most denoisers as is, but [Cha19] proposed modifications to the non-local means and Gaussian mixture model denoisers, which make them into linear filters, to enforce nonexpansiveness. [DWY18] presented a proof that relies on the existence of a certain Lyapunov function that is monotonic under H_σ, which holds only for simple H_σ. [TG19] analyzed a variant of PnP, but did not establish local convergence since their key assumption is only expected to be satisfied "in early iterations".

Other PnP-type methods. There are other lines of works that incorporate modern denoisers into model-based optimization methods. The plug-in idea with half quadratic splitting, as opposed to ADMM, was discussed [ZW11] and this approach was carried out with deep learning-based denoisers in [ZZG17]. [DKE12, EK15] use the notion of Nash equilibrium to propose a scheme similar to PnP. [DKE10] proposed an augmented Lagrangian method similar to PnP. [REM17, RS19] presented Regularization by Denoising (RED), which uses the (nonconvex) regularizer $\mathbf{x}^T(\mathbf{x} - H_\sigma(\mathbf{x}))$ given a denoiser H_σ, and use denoiser evaluations in its iterations. [FPR18] applies the plug-in approach to vector approximate message passing. [SLX16, FWW19] replaced both the proximal operator enforcing data fidelity and the denoiser with two neural networks and performed end-to-end training. Broadly, there are more works that incorporate model-based optimization with deep learning [CLW18, LCW19].

Image denoising using deep learning. Deep learning-based denoising methods have become state-of-the-art. [ZZC17] proposed an effective denoising network called DnCNN, which adopted batch normalization [IS15] and ReLU [KSH12] into the residual learning [HZR16]. Other represenative deep denoising models include the deep convolutional encoder-decoder with symmetric skip connection [MSY16], N^3Net [PR18], and MWCNN [LZZ18]. The recent FFDNet [ZZZ18] handles spatially varying Gaussian noise.

Regularizing Lipschitz continuity. Lipschitz continuity and its variants have started to receive attention as a means for regularizing deep classifiers [BFT17, BCW18, OC18] and GANs [MKK18, BDS19]. Regularizing Lipschitz continuity stabilizes training, improves the final performance, and enhances robustness to adversarial attacks [WZC18, QW19].

Specifically, [MKK18] proposed to normalize all weights to be of unit spectral norms to thereby constrain the Lipschitz constant of the overall network to be no more than one.

4.3 Convergence via contraction

We now present conditions that ensure the PnP methods are contractive and thereby convergent.

If we assume $2H_\sigma - I$ is nonexpansive, standard tools of monotone operator theory tell us that PnP-ADMM converges. However, this assumption is too strong. Chan et al. presented a counter example demonstrating that $2H_\sigma - I$ is not nonexpansive for the NLM denoiser [CWE17].

Rather, we assume $H_\sigma : \mathbb{R}^M \to \mathbb{R}^M$ satisfies

$$\|(H_\sigma - I)(\mathbf{x}) - (H_\sigma - I)(\mathbf{y})\|^2 \leq \varepsilon^2 \|\mathbf{x} - \mathbf{y}\|^2 \tag{A}$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$ for some $\varepsilon \geq 0$. Since σ controls the strength of the denoising, we can expect H_σ to be close to identity for small σ. If so , Assumption (A) is reasonable.

Under this assumption, we show that the PNP-FBS and PNP-DRS iterations are **contractive** in the sense that we can express the iterations as $\mathbf{x}^{(k+1)} = T(\mathbf{x}^{(k)})$, where $T : \mathbb{R}^M \to \mathbb{R}^M$ satisfies

$$\|T(\mathbf{x}) - T(\mathbf{y})\| \leq \delta \|\mathbf{x} - \mathbf{y}\|$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$ for some $\delta < 1$. We call δ the contraction factor. If $\mathbf{x}^\star$ satisfies $T(\mathbf{x}^\star) = \mathbf{x}^\star$, i.e., $\mathbf{x}^\star$ is a fixed point, then $\mathbf{x}^{(k)} \to \mathbf{x}^\star$ geometrically by the classical Banach contraction principle.

Theorem 8 (Convergence of PNP-FBS)**.** *Assume H_σ satisfies assumption (A) for some $\varepsilon \geq 0$. Assume f is μ-strongly convex, f is differentiable, and ∇f is L-Lipschitz. Then*

$$T = H_\sigma(I - \alpha \nabla f)$$

satisfies

$$\|T(\mathbf{x}) - T(\mathbf{y})\| \leq \max\{|1 - \alpha\mu|, |1 - \alpha L|\}(1 + \varepsilon)\|\mathbf{x} - \mathbf{y}\|$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$*. The coefficient is less than* 1 *if*

$$\frac{1}{\mu(1+1/\varepsilon)} < \alpha < \frac{2}{L} - \frac{1}{L(1+1/\varepsilon)}.$$

Such an α *exists if* $\varepsilon < 2\mu/(L-\mu)$*.*

Theorem 9 (Convergence of PNP-DRS)**.** *Assume* H_σ *satisfies assumption (A) for some* $\varepsilon \geq 0$*. Assume* f *is* μ*-strongly convex and differentiable. Then*

$$T = \frac{1}{2}I + \frac{1}{2}(2H_\sigma - I)(2\mathrm{Prox}_{\alpha f} - I)$$

satisfies

$$\|T(\mathbf{x}) - T(\mathbf{y})\| \leq \frac{1+\varepsilon+\varepsilon\alpha\mu+2\varepsilon^2\alpha\mu}{1+\alpha\mu+2\varepsilon\alpha\mu}\|\mathbf{x}-\mathbf{y}\|$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$*. The coefficient is less than* 1 *if*

$$\frac{\varepsilon}{(1+\varepsilon-2\varepsilon^2)\mu} < \alpha, \quad \varepsilon < 1.$$

The proofs of Theorems 8 and 9 are provided in the appendix and can also be found in [RLW19].

Corollary 1 (Convergence of PNP-ADMM)**.** *Assume* H_σ *satisfies assumption (A) for some* $\varepsilon \in [0,1)$*. Assume* f *is* μ*-strongly convex. Then PNP-ADMM converges for*

$$\frac{\varepsilon}{(1+\varepsilon-2\varepsilon^2)\mu} < \alpha.$$

Proof. This follows from Theorem 9 and the equivalence of PNP-DRS and PNP-ADMM. □

For PNP-FBS, we assume f is μ-strongly convex and ∇f is L-Lipschitz. For PNP-DRS and PNP-ADMM, we assume f is μ-strongly convex. These are standard assumptions that are satisfied in application such as image denoising/deblurring and single photon imaging. Strong convexity, however, does exclude a few applications such as compressed sensing, sparse interpolation, and super-resolution.

PNP-FBS and PNP-ADMM are distinct methods for finding the same set of fixed points. Sometimes, PNP-FBS is easier to implement since it only requires the computation of ∇f rather than $\mathrm{Prox}_{\alpha f}$. On the other hand, PNP-ADMM has better convergence properties as demonstrated theoretically by Theorems 8 and 9 and empirically by our experiments.

The proof of Theorem 9 relies on the notion of "negatively averaged" operators of [Gis17]. It is straightforward to modify Theorems 8 and 9 to establish local convergence when Assumption (A) holds locally. Theorem 9 can be generalized to the case when f is strongly convex but non-differentiable using the notion of subgradients.

Recently, [FPR18] proved convergence of "plug-and-play" vector approximate message passing, a method similar to ADMM, assuming Lipschitz continuity of the denoiser. Although the method, the proof technique, and the notion of convergence are different from ours, the similarities are noteworthy.

4.4 Real spectral normalization: enforcing Assumption (A)

We now present real spectral normalization, a technique for training denoisers to satisfy Assumption (A) and connect the practical implementations to the theory of Section 4.3.

4.4.1 Deep learning denoisers: SimpleCNN and DnCNN

We use a deep denoising model called **DnCNN** [ZZC17], which learns the residual mapping with a 17-layer CNN and reports state-of-the-art results on natural image denoising. Given a noisy observation $\mathbf{y} = \mathbf{x} + \mathbf{e}$, where $\mathbf{x}$ is the clean image and $\mathbf{e}$ is noise, the residual mapping R outputs the noise, i.e., $R(\mathbf{y}) = \mathbf{e}$ so that $\mathbf{y} - R(\mathbf{y})$ is the clean recovery. Learning the residual mapping is a popular approach in deep learning-based image restoration. The structure of DnCNN is shown in Figure 4.1.

We also construct a simple convolutional encoder-decoder model for denoising and call it **SimpleCNN**. SimpleCNN consists of 4 convolutional layers, with ReLU and mean-square-error (MSE) loss and does not utilize any pooling or (batch) normalization. The structure of SimpleCNN is shown in Figure 4.2.

We remark that realSN and the theory of this work is applicable to other deep denoisers. We use SimpleCNN to show that realSN is applicable to any CNN denoiser.

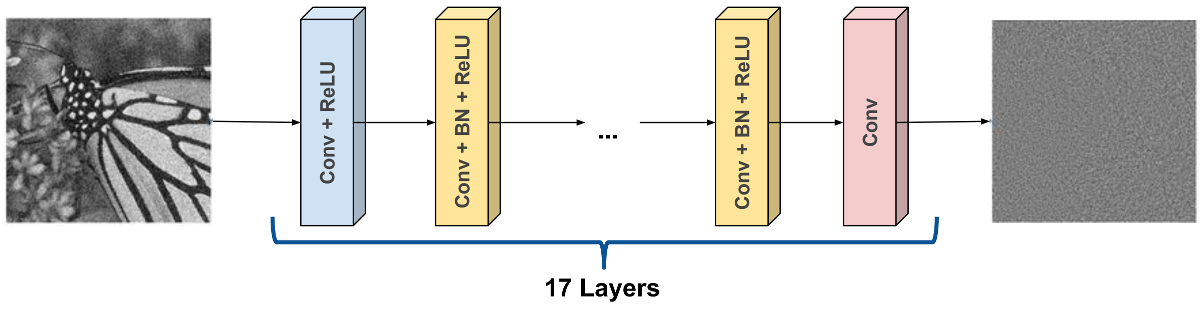


Figure 4.1: DnCNN Network Architecture

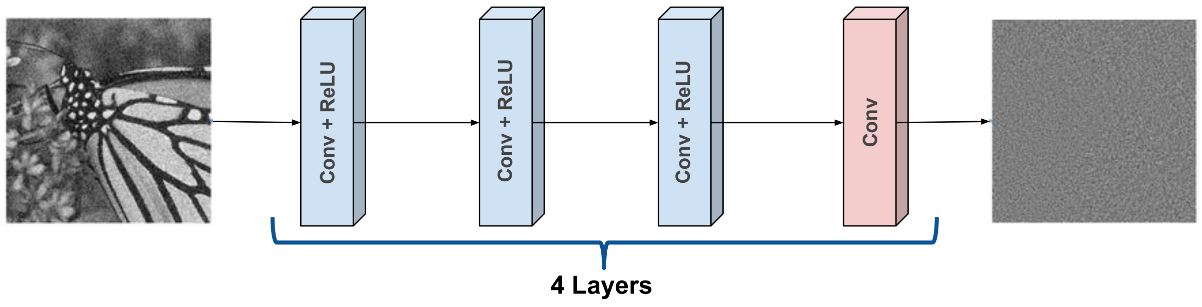


Figure 4.2: SimpleCNN Network Architecture

4.4.2 Lipschitz constrained deep denoising

Denote the denoiser (SimpleCNN or DnCNN) as $H(\mathbf{y}) = \mathbf{y} - R(\mathbf{y})$, where y is the noisy input and R is the residual mapping, i.e., $R(\mathbf{y}) = \mathbf{y} - H(\mathbf{y}) = (I - H)(\mathbf{y})$. Enforcing Assumption (A) is equivalent to constraining the Lipschitz constant of $R(\mathbf{y})$. We propose a variant of the spectral normalization (**SN**) [MKK18] for this.

Spectral normalization. [MKK18] proposed to normalize the spectral norm of each layer-wise weight (with ReLU non-linearity) to one. Provided that we use 1-Lipschitz nonlinearities (such as ReLU), the Lipschitz constant of a layer is upper-bounded by the spectral norm of its weight, and the Lipschitz constant of the full network is bounded by the product of spectral norms of all layers [GFP18]. To avoid the prohibitive cost of singular value decomposition (SVD) every iteration, SN approximately computes the largest singular values of weights using a small number of power iterations.

Given the weight matrix $\mathbf{W}_l \in \mathbb{R}^{m\times n}$ of the l-th layer, vectors $\mathbf{u}_l \in \mathbb{R}^m, \mathbf{v}_l \in \mathbb{R}^m$ are initialized randomly and maintained in the memory to estimate the leading first left and right singular vector of $\mathbf{W}_l$ respectively. During each forward pass of the network, SN is applied to all layers $1 \le l \le L$ following the two-step routine:

1. Apply one step of the power method to update $\mathbf{u}_l, \mathbf{v}_l$:

$$\mathbf{v}_l \leftarrow \mathbf{W}_l^T \mathbf{u}_l \ / \ \|\mathbf{W}_l^T \mathbf{u}_l\|_2, \quad \mathbf{u}_l \leftarrow \mathbf{W}_l \mathbf{v}_l \ / \ \|\mathbf{W}_l \mathbf{v}_l\|_2$$

2. Normalize $\mathbf{W}_l$ with the estimated spectral norm:

$$\mathbf{W}_l \leftarrow \mathbf{W}_l / \sigma(\mathbf{W}_l), \text{ where } \sigma(\mathbf{W}_l) = \mathbf{u}_l^T \mathbf{W}_l \mathbf{v}_l$$

While the basic methodology of SN suits our goal, the SN in [MKK18] uses a convenient but inexact implementation for convolutional layers. A convolutional layer is represented by a four-dimensional kernel $\mathbf{K}_l$ of shape $(C_{\text{out}}, C_{\text{in}}, h, w)$, where h, w are kernel's height and width. SN reshapes $\mathbf{K}_l$ into a two-dimensional matrix $\tilde{\mathbf{K}}_l$ of shape $(C_{\text{out}}, C_{\text{in}} \times h \times w)$ and regards $\tilde{\mathbf{K}}_l$ as the matrix $\mathbf{W}_l$ above. This relaxation **underestimates** the true spectral norm of the convolutional operator (Corollary 1 of [TSS18]) given by

$$\sigma(\mathbf{K}_l) = \max_{\mathbf{x}\neq 0} \|\mathbf{K}_l * \mathbf{x}\|_2 / \|\mathbf{x}\|_2,$$

where $\mathbf{x}$ is the input to the convolutional layer and $*$ is the convolutional operator. This issue is not hypothetical. When we trained SimpleCNN with SN, the spectral norms of the layers were 3.01, 2.96, 2.82, and 1.31, i.e., SN failed to control the Lipschitz constant below 1.

Real spectral normalization. We propose an improvement to SN for convolutional[1] layers, called the **real spectral normalization** (realSN), to more accurately constrain the network's Lipschitz constant and thereby enforce Assumption (A).

In realSN, we directly consider the convolutional linear operator $\mathcal{K}_l : \mathbb{R}^{C_{\text{in}}\times h\times w} \to \mathbb{R}^{C_{\text{out}}\times h\times w}$, where h, w are input's height and width, instead of reshaping the convolution

[1]We use stride 1 and zero-pad with width 1 for convolutions.

kernel $\mathbf{K}_l$ into a matrix. The power iteration also requires the conjugate (transpose) operator $\mathcal{K}_l^*$. It can be shown that $\mathcal{K}_l^*$ is another convolutional operator with a kernel that is a rotated version of the forward convolutional kernel; the first two dimensions are permuted and the last two dimensions are rotated by 180 degrees [LCW19]. Instead of two vectors $\mathbf{u}_l, \mathbf{v}_l$ as in SN, realSN maintains $\mathbf{U}_l \in \mathbb{R}^{C_{\text{out}} \times h \times w}$ and $\mathbf{V}_l \in \mathbb{R}^{C_{\text{in}} \times h \times w}$ to estimate the leading left and right singular vectors respectively. During each forward pass of the neural network, realSN conducts:

1. Apply one step of the power method with operator $\mathcal{K}_l$:

$$\mathbf{V}_l \leftarrow \mathcal{K}_l^*(\mathbf{U}_l) \ / \ \|\mathcal{K}_l^*(\mathbf{U}_l)\|_2,$$
$$\mathbf{U}_l \leftarrow \mathcal{K}_l(\mathbf{V}_l) \ / \ \|\mathcal{K}_l(\mathbf{V}_l)\|_2.$$

2. Normalize the convolutional kernel $\mathbf{K}_l$ with estimated spectral norm:

$$\mathbf{K}_l \leftarrow \mathbf{K}_l / \sigma(\mathcal{K}_l), \text{ where } \sigma(\mathcal{K}_l) = \langle \mathbf{U}_l, \mathcal{K}_l(\mathbf{V}_l) \rangle$$

By replacing $\sigma(\mathcal{K}_l)$ with $\sigma(\mathbf{K}_l)/c_l$, realSN can constrain the Lipschitz constant to any upper bound $C = \prod_{l=1}^{L} c_l$. Using the highly efficient convolution computation in modern deep learning frameworks, realSN can be implemented simply and efficiently. Specifically, realSN introduces three additional one-sample convolution operations for each layer in each training step. When we used a batch size of 128, the extra computational cost of the additional operations is mild.

4.4.3 Implementation details

We refer to SimpleCNN and DnCNN regularized by realSN as **RealSN-SimpleCNN** and **RealSN-DnCNN**, respectively. We train them in the setting of Gaussian denoising with known fixed noise levels $\sigma = 5, 15, 40$. We used $\sigma = 5, 15$ for CS-MRI and single photon imaging, and $\sigma = 40$ for Poisson denoising. The regularized denoisers are trained to have Lipschitz constant (no more than) 1. The training data consists of images from the BSD500 dataset, divided into 40×40 patches. The CNN weights were initialized in the same way as

[ZZC17]. We train all networks using the ADAM optimizer for 50 epochs, with a mini-batch size of 128. The learning rate was 10^{-3} in the first 25 epochs, then decreased to 10^{-4}. On an Nvidia GTX 1080 Ti, DnCNN took 4.08 hours and realSN-DnCNN took 5.17 hours to train, so the added cost of realSN is mild.

4.5 Poisson denoising: validating the theory

Consider the Poisson denoising problem, where given a true image $\mathbf{x}_{\text{true}} \in \mathbb{R}^M$, we observe independent Poisson random variables $y_i \sim \text{Poisson}((\mathbf{x}_{\text{true}})_i)$, so $y_i \in \mathbb{N}$, for $i = 1, \ldots, M$. For details and motivation for this problem setup, see [RGE16].

For the objective function $f(\mathbf{x})$, we use the negative log-likelihood given by $f(\mathbf{x}) = \sum_{i=1}^{M} \ell(x_i; y_i)$, where

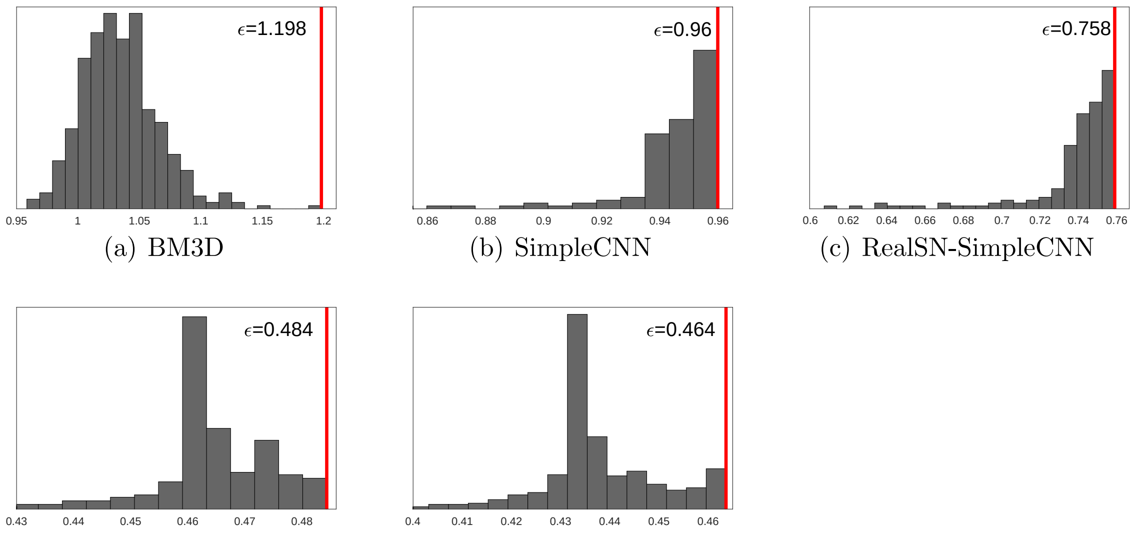
$$\ell(x; y) = \begin{cases} -y\log(x) + x & \text{for } y > 0,\ x > 0 \\ 0 & \text{for } y = 0,\ x \geq 0 \\ \infty & \text{otherwise.} \end{cases}$$

We can compute $\text{Prox}_{\alpha f}$ elementwise with

$$\text{Prox}_{\alpha f}(x) = (1/2)\left(x - \alpha + \sqrt{(x-\alpha)^2 + 4\alpha y}\right).$$

The gradient of f is given by $\partial f/\partial x_i = -y_i/x_i + 1$ for $x_i > 0$ for $i = 1, \ldots, M$. We set $\partial f/\partial x_i = 0$ when $x_i = 0$, although, strictly speaking, $\partial f/\partial x_i$ is undefined when $y_i > 0$ and $x_i = 0$. This does not seem to cause any problems in the experiments. Since we force the denoisers to output nonnegative pixel values, PNP-FBS never needs to evaluate $\partial f/\partial x_i$ for negative x_i.

For H_σ, we choose BM3D, SimpleCNN with and without realSN, and DnCNN with and without realSN. Note that these denoisers are designed or trained for the purpose of **Gaussian denoising**, and here we integrate them into the PnP frameworks for Poisson denoising. We scale the image so that the peak value of the image, the maximum mean of the Poisson random variables, is 1. The y-variable was initialized to the noisy image for



(a) BM3D (b) SimpleCNN (c) RealSN-SimpleCNN

(d) DnCNN (e) RealSN-DnCNN

Figure 4.3: Histograms for experimentally verifying Assumption (A). The x-axis represents values of $\|(I - H_\sigma)(\mathbf{x}) - (I - H_\sigma)(\mathbf{y})\|/\|\mathbf{x} - \mathbf{y}\|$ and the y-axis represents the frequency. The vertical red bar corresponds to the maximum value.

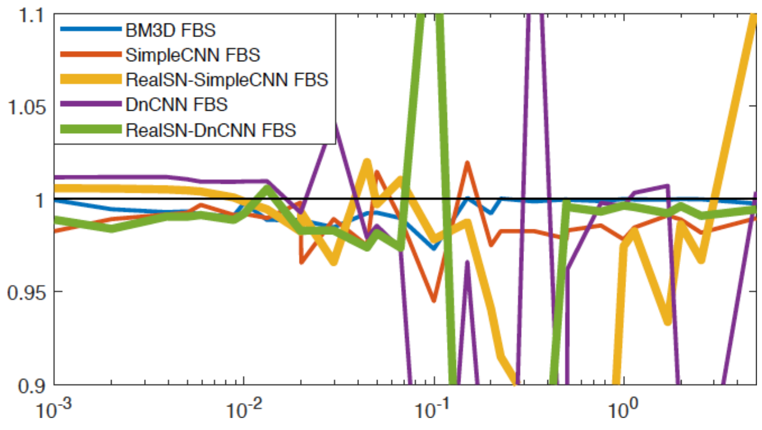
PnP-FBS and PnP-ADMM, and the u-variable was initialized to 0 for PnP-ADMM. We use the test set of 13 images in [CWE17].

Convergence. We first examine which denoisers satisfy Assumption (A) with small ε. In Figure 4.3, we run PnP iterations of Poisson denoising on a single image (flag of [RGE16]) with different models, calculate $\|(I - H_\sigma)(\mathbf{x}) - (I - H_\sigma)(\mathbf{y})\|/\|\mathbf{x} - \mathbf{y}\|$ between the iterates and the limit, and plot the histogram. The maximum value of a histogram, marked by a vertical red bar, lower-bounds the ε of Assumption (A). Remember that Corollary 1 requires $\varepsilon < 1$ to ensure convergence of PnP-ADMM. Figure 4.3(a) proves that BM3D violates this assumption. Figures 4.3(b) and 4.3(c) and Figures 4.3(d) and 4.3(e) respectively illustrate that RealSN indeed improves (reduces) ε for SimpleCNN and DnCNN.

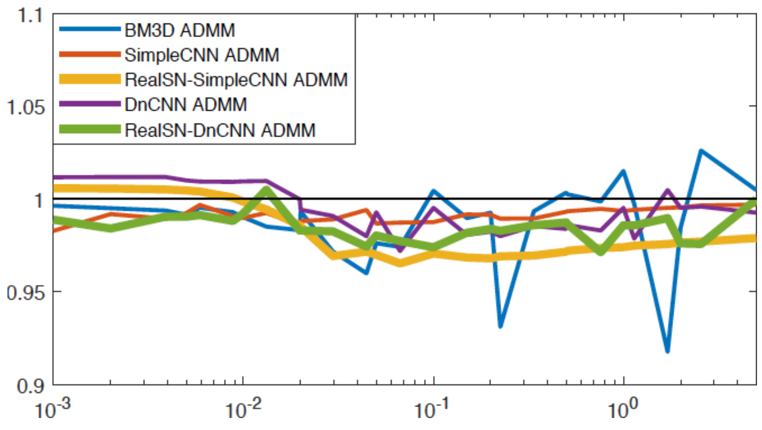
Figure 4.4 experimentally validates Theorems 8 and 9, by examining the average (geometric mean) contraction factor (defined in Section 4.3) of PnP-FBS and ADMM[2] iterations over a

[2]We compute the contraction factor of the equivalent PnP-DRS.

range of step sizes. Figure 4.4 qualitatively shows that PnP-ADMM exhibits more stable convergence than PnP-FBS. Theorem 8 ensures PnP-FBS is a contraction when α is within an interval and Theorem 9 ensures PnP-ADMM is a contraction when α is large enough. We roughly observe this behavior for the denoisers trained with RealSN.



(a) PnP-FBS



(b) PNP-ADMM

Figure 4.4: Average contraction factor of 500 iterations for the Poisson denoising experiment. The x-axis represents the value of α and y-axis represents the contraction factor. Although lower means faster convergence, a smoother curve means the method is easier to tune and has more stable convergence.

Empirical performance. Our theory only concerns convergence and says nothing about the recovery performance of the output the methods converge to. We empirically verify that the PnP methods with RealSN, for which we analyzed convergence, yield competitive denoising results. We fix $\alpha = 0.1$ for all denoisers in PNP-ADMM, and $\alpha = 0.0125$ in PNP-FBS. For deep learning-based denoisers, we choose $\sigma = 40/255$. For BM3D, we choose $\sigma = \sqrt{\gamma\alpha}$ as suggested in [RGE16] and use $\gamma = 1$. Table 4.1 compares the PnP methods with BM3D, RealSN-DnCNN, and RealSN-SimpleCNN plugged in. In both PnP methods, one of the two denoisers using RealSN, for which we have theory, outperforms BM3D. It is interesting to obverse that the PnP performance does not necessarily hinge on the strength of the plugged in denoiser and that different PnP methods favor different denoisers. For example, RealSN-SimpleCNN surpasses the much more sophisticated RealSN-DnCNN under PnP-FBS. However, RealSN-DnCNN leads to better, and overall best, denoising performance when plugged into PnP-ADMM.

Table 4.1: Average PSNR performance (in dB) on Poisson denoising (peak = 1) on the testing set in [CWE17].

	BM3D	RealSN-DnCNN	RealSN-SimpleCNN
PNP-ADMM	23.4617	**23.5873**	18.7890
PNP-FBS	18.5835	22.2154	**22.7280**

4.6 More applications

We now apply PnP on two imaging problems and show that RealSN improves the reconstruction of PnP.

Single photon imaging. Consider single photon imaging with quanta image sensors (QIS) [Fos11, CL14, EC16] with the model

$$\mathbf{z} = \mathbf{1}(\mathbf{y} \geq 1), \quad \mathbf{y} \sim \text{Poisson}(\alpha_{sg}\mathbf{G}\mathbf{x}_{\text{true}})$$

where $\mathbf{x}_{\text{true}} \in \mathbb{R}^M$ is the underlying image, $\mathbf{G} : \mathbb{R}^M \to \mathbb{R}^{MK}$ duplicates each pixel to K pixels, $\alpha_{sg} \in \mathbb{R}$ is sensor gain, K is the oversampling rate, $\mathbf{z} \in \{0, 1\}^{MK}$ is the observed binary photons. We want to recover $\mathbf{x}_{\text{true}}$ from $\mathbf{z}$. The likelihood function is

$$f(\mathbf{x}) = \sum_{j=1}^{n} -K_j^0 \log(e^{-\alpha_{sg}x_j/K}) - K_j^1 \log(1 - e^{-\alpha_{sg}x_j/K}),$$

where $K_j^1 = \sum_{i=1}^K z_{(j-1)K+i}$ is the number of ones in the j-th unit pixel, $K_j^0 = \sum_{i=1}^K 1 - z_{(j-1)K+i}$ is the number of zeros in the j-th unit pixel. The gradient of $f(\mathbf{x})$ is given by $\partial f/\partial x_j = (\alpha_{sg}/K)(K_j^0 - K_j^1/(e^{\alpha_{sg}x_j/K} - 1))$ and the proximal operator of f is given in [CL14].

We compare PnP-ADMM and PnP-FBS respectively with the denoisers BM3D, RealSN-DnCNN, and RealSN-SimpleCNN. We take $\alpha_{sg} = K = 8$. The y-variable was initialized to K^1 for PnP-FBS and PnP-ADMM, and the u-variable was initialized to 0 for PnP-ADMM. All deep denoisers used in this experiment were trained with fixed noise level $\sigma = 15$. We report the PSNRs achieved at the 50th iteration, the 100th iteration, and the best PSNR values achieved within the first 100 iterations.

Table 4.2 reports the average PSNR results on the 13 images used in [CWE17]. Experiments indicate that PnP-ADMM methods constantly yields higher PNSR than the PnP-FBS counterparts using the same denoiser. The best overall PSNR is achieved using PnP-ADMM with RealSN-DnCNN, which shows nearly 1dB improvement over the result obtained with BM3D. We also observe that deep denoisers with RealSN make PnP converges more stably.

Compressed sensing MRI. Magnetic resonance imaging (MRI) is a widely-used imaging technique with a slow data acquisition. Compressed sensing MRI (CS-MRI) accelerates MRI by acquiring less data through downsampling. PnP is useful in medical imaging as we do not have a large amount of data for end-to-end training: we train the denoiser H_σ on natural images, and then "plug" it into the PnP framework to be applied to medical images. CS-MRI is described mathematically as

$$\mathbf{y} = \mathcal{F}_p \mathbf{x}_{\text{true}} + \varepsilon_e,$$

where $\mathbf{x}_{\text{true}} \in \mathbb{C}^M$ is the underlying image, $\mathcal{F}_p : \mathbb{C}^M \to \mathbb{C}^k$ is the linear measurement model, $\mathbf{y} \in \mathbb{C}^k$ is the measured data, and $\varepsilon_e \sim N(0, \sigma_e I_k)$ is measurement noise. We want to recover $\mathbf{x}_{\text{true}}$ from $\mathbf{y}$. The objective function is

$$f(\mathbf{x}) = (1/2)\|\mathbf{y} - \mathcal{F}_p \mathbf{x}\|^2.$$

Table 4.2: Average PSNR (in dB) of single photon imaging task on the test set of [CWE17].

PnP-FBS, $\alpha = 0.005$			
Average PSNR	BM3D	RealSN-DnCNN	RealSN-SimpleCNN
Iteration 50	28.7933	27.9617	29.0062
Iteration 100	29.0510	27.9887	29.0517
Best Overall	**29.5327**	28.4065	29.3563

PnP-ADMM, $\alpha = 0.01$			
Average PSNR	BM3D	RealSN-DnCNN	RealSN-SimpleCNN
Iteration 50	30.0034	31.0032	29.2154
Iteration 100	30.0014	31.0032	29.2151
Best Overall	30.0474	**31.0431**	29.2155

The gradient of $f(\mathbf{x})$ is given in [LZC16] and the proximal operator of $f(\mathbf{x})$ is given in [Eks16]. We use BM3D, SimpleCNN and DnCNN, and their variants by RealSN for the PnP denoiser H_σ.

We take $\mathcal{F}_p$ as the Fourier k-domain subsampling (partial Fourier operator). We tested random, radial, and Cartesian sampling [Eks16] with a sampling rate of 30%. The noise level σ_e is taken as 15/255.

We compare PnP frameworks with zero-filling, total-variation (TV) [LSL05], RecRF [YZY10], and BM3D-MRI [Eks16] [3]. The parameters are taken as follows. For TV, the regularization parameter λ is taken as the best one from $\{a \times 10^b, a \in \{1, 2, 5\}, b \in \{-5, -4, -3, -2, -1, 0, 1\}\}$. For RecRF, the two parameters λ, μ are both taken from the above sets and the best results are reported. For BM3D-MRI, we set the "final noise level (the noise level in the last iteration)" as $2\sigma_e$, which is suggested in their MATLAB library. For PnP methods with H_σ as BM3D, we set $\sigma = 2\sigma_e$, take $\alpha \in \{0.1, 0.2, 0.5, 1, 2, 5\}$ and report the best results. For PNP-ADMM with H_σ as deep denoisers, we take $\sigma = \sigma_e = 15/255$

[3]Some recent deep-learning based methods [SLX16, KLT16, MMB17, ZG18b] are not compared here because we assume we do not have enough medical images for training.

Table 4.3: CS-MRI results (30% sample with additive Gaussian noise $\sigma_e = 15$) in PSNR (dB).

Sampling approach		Random		Radial		Cartesian	
Image		Brain	Bust	Brain	Bust	Brain	Bust
Zero-filling		9.58	7.00	9.29	6.19	8.65	6.01
TV [LSL05]		16.92	15.31	15.61	14.22	12.77	11.72
RecRF [YZY10]		16.98	15.37	16.04	14.65	12.78	11.75
BM3D-MRI [Eks16]		17.31	13.90	16.95	13.72	14.43	12.35
PnP-FBS	BM3D	19.09	16.36	18.10	15.67	14.37	12.99
	DnCNN	19.59	16.49	18.92	15.99	14.76	14.09
	RealSN-DnCNN	**19.82**	**16.60**	**18.96**	**16.09**	**14.82**	**14.25**
	SimpleCNN	15.58	12.19	15.06	12.02	12.78	10.80
	RealSN-SimpleCNN	17.65	14.98	16.52	14.26	13.02	11.49
PnP-ADMM	BM3D	19.61	**17.23**	18.94	**16.70**	14.91	13.98
	DnCNN	19.86	17.05	19.00	16.64	14.86	14.14
	RealSN-DnCNN	**19.91**	17.09	**19.08**	16.68	**15.11**	**14.16**
	SimpleCNN	16.68	12.56	16.83	13.47	13.03	11.17
	RealSN-SimpleCNN	17.77	14.89	17.00	14.47	12.73	11.88

and $\alpha = 2.0$ uniformly for all the cases. For PNP-FBS with H_σ as deep denoisers, we take $\sigma = \sigma_e/3 = 5/255$ and $\alpha = 0.4$ uniformly. All deep denoisers are trained on BSD500 [MFT01], a natural image data set; no medical image is used in training. The **y**-variable was initialized to the zero-filled solution for PnP-FBS and PnP-ADMM, and the **u**-variable was initialized to 0 for PnP-ADMM. Table 4.3 reports our results on CS-MRI, from which we can confirm the effectiveness of PnP frameworks. Moreover, using RealSN-DnCNN seems to the clear winner over all. We also observe that PnP-ADMM generally outperforms PnP-FBS when using the same denoiser, which supports Theorems 8 and 9.

4.7 Conclusion

In this work, we analyzed the convergence of PnP-FBS and PnP-ADMM under a Lipschitz assumption on the denoiser. We then presented real spectral normalization a technique to enforce the proposed Lipschitz condition in training deep learning-based denoisers. Finally, we validate the theory with experiments.

Appendix 4.A Preliminaries

For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$, write $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^T \mathbf{y}$ for the inner product. We say a function $f : \mathbb{R}^M \to \mathbb{R} \cup \{\infty\}$ is convex if

$$f(\theta x + (1-\theta)y) \le \theta f(\mathbf{x}) + (1-\theta) f(\mathbf{y})$$

for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$ and $\theta \in [0,1]$. A convex function is closed if it is lower semi-continuous and proper if it is finite somwhere. We say f is μ-strongly convex for $\mu > 0$ if $f(\mathbf{x}) - (\mu/2)\|\mathbf{x}\|^2$ is a convex function. Given a convex function $f : \mathbb{R}^M \to \mathbb{R} \cup \{\infty\}$ and $\alpha > 0$, define its proximal operator $\mathrm{Prox}_f : \mathbb{R}^M \to \mathbb{R}^M$ as

$$\mathrm{Prox}_{\alpha f}(\mathbf{z}) = \operatorname*{arg\,min}_{\mathbf{x} \in \mathbb{R}^M} \left\{ \alpha f(\mathbf{x}) + (1/2)\|\mathbf{x} - \mathbf{z}\|^2 \right\}.$$

When f is convex, closed, and proper, the $\arg\min$ uniquely exists, and therefore Prox_f is well-defined. An mapping $T : \mathbb{R}^M \to \mathbb{R}^M$ is L-Lipschitz if

$$\|T(\mathbf{x}) - T(\mathbf{y})\| \le L\|\mathbf{x} - \mathbf{y}\|$$

for all $\mathbf{x}, \mathbf{y}, \in \mathbb{R}^M$. If T is L-Lipschitz with $L \le 1$, we say T is nonexpansive. If T is L-Lipschitz with $L < 1$, we say T is a contraction. A mapping $T : \mathbb{R}^M \to \mathbb{R}^M$ is θ-averaged for $\theta \in (0,1)$, if it is nonexpansive and if

$$T = \theta R + (1-\theta) I,$$

where $R : \mathbb{R}^M \to \mathbb{R}^M$ is another nonexpansive mapping.

Lemma 8 (Proposition 4.35 of [BC17]). *$T : \mathbb{R}^M \to \mathbb{R}^M$ is θ-averaged if and only if*

$$\|T(\mathbf{x}) - T(\mathbf{y})\|^2 + (1-2\theta)\|\mathbf{x} - \mathbf{y}\|^2 \le 2(1-\theta)\langle T(\mathbf{x}) - T(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$*.*

Lemma 9 ([OY02, CY15])**.** *Assume* $T_1 : \mathbb{R}^M \to \mathbb{R}^M$ *and* $T_2 : \mathbb{R}^M \to \mathbb{R}^M$ *are* θ_1 *and* θ_2*-averaged, respectively. Then* T_1T_2 *is* $\frac{\theta_1+\theta_2-2\theta_1\theta_2}{1-\theta_1\theta_2}$*-averaged.*

Lemma 10. *Let* $T : \mathbb{R}^M \to \mathbb{R}^M$*.* $-T$ *is* θ*-averaged if and only if* $T \circ (-I)$ *is* θ*-averaged.*

Proof. The lemma follows from the fact that

$$T \circ (-I) = \theta R + (1-\theta)I \quad \Leftrightarrow \quad -T = \theta(-R) \circ (-I) + (1-\theta)I$$

for some nonexpansive R and that nonexpansiveness of R and implies nonexpansivenes of $-R \circ (-I)$. $\square$

Lemma 11 ([THG18])**.** *Assume* f *is* μ*-strongly convex and* ∇f *is* L*-Lipschitz. Then for any* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$*, we have*

$$\|(I - \alpha\nabla f)(\mathbf{x}) - (I - \alpha\nabla f)(\mathbf{y})\| \leq \max\{|1-\alpha\mu|, |1-\alpha L|\}\|\mathbf{x} - \mathbf{y}\|.$$

Lemma 12 (Proposition 5.4 of [Gis17])**.** *Assume* f *is* μ*-strongly convex, closed, and proper. Then*

$$-(2\mathrm{Prox}_{\alpha f} - I)$$

is $\frac{1}{1+\alpha\mu}$*-averaged.*

References. The notion of proximal operator and its well-definedness were first presented in [Mor65]. The notion of averaged mappings were first introduced in [BBR78]. The idea of Lemma 10 relates to "negatively averaged" operators from [Gis17]. Lemma 11 is proved in a weaker form as Theorem 3 of [Pol87] and in Section 5.1 of [RB16]. Lemma 11 as stated is proved as Theorem 2.1 in [THG18].

Appendix 4.B Proofs of main results

4.B.1 Equivalence of PNP-DRS and PNP-ADMM

We show the standard steps that establish equivalence of PNP-DRS and PNP-ADMM. Starting from PNP-DRS, we substitute $\mathbf{z}^{(k)} = \mathbf{x}^{(k)} + \mathbf{u}^{(k)}$ to get

$$\begin{aligned}
\mathbf{x}^{(k+1/2)} &= \mathrm{Prox}_{\alpha f}(\mathbf{x}^{(k)} + \mathbf{u}^{(k)}) \\
\mathbf{x}^{(k+1)} &= H_\sigma(\mathbf{x}^{(k+1/2)} - (\mathbf{u}^{(k)} + \mathbf{x}^{(k)} - \mathbf{x}^{(k+1/2)})) \\
\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \mathbf{x}^{(k)} - \mathbf{x}^{(k+1/2)}.
\end{aligned}$$

We reorder the iterations to get the correct dependency

$$\begin{aligned}
\mathbf{x}^{(k+1/2)} &= \mathrm{Prox}_{\alpha f}(\mathbf{x}^{(k)} + \mathbf{u}^{(k)}) \\
\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \mathbf{x}^{(k)} - \mathbf{x}^{(k+1/2)} \\
\mathbf{x}^{(k+1)} &= H_\sigma(\mathbf{x}^{(k+1/2)} - \mathbf{u}^{(k+1)}).
\end{aligned}$$

We label $\tilde{\mathbf{y}}^{(k+1)} = \mathbf{x}^{(k+1/2)}$ and $\tilde{\mathbf{x}}^{(k+1)} = \mathbf{x}^{(k)}$

$$\begin{aligned}
\tilde{\mathbf{x}}^{(k+1)} &= H_\sigma(\tilde{\mathbf{y}}^{(k)} - \mathbf{u}^{(k)}) \\
\tilde{\mathbf{y}}^{(k+1)} &= \mathrm{Prox}_{\alpha f}(\tilde{\mathbf{x}}^{(k+1)} + \mathbf{u}^{(k)}) \\
\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \tilde{\mathbf{x}}^{(k+1)} - \tilde{\mathbf{y}}^{(k+1)},
\end{aligned}$$

and we get PNP-ADMM.

4.B.2 Convergence analysis

Lemma 13. *$H_\sigma : \mathbb{R}^M \to \mathbb{R}^M$ satisfies Assumption (A) if and only if*

$$\frac{1}{1+\varepsilon} H_\sigma$$

is nonexpansive and $\frac{\varepsilon}{1+\varepsilon}$-averaged.

Proof. Define $\theta = \frac{\varepsilon}{1+\varepsilon}$, which means $\varepsilon = \frac{\theta}{1-\theta}$. Clearly, $\theta \in [0, 1)$. Define $G = \frac{1}{1+\varepsilon} H_\sigma$, which means $H_\sigma = \frac{1}{1+\theta}G$. Then

$$\begin{aligned}
&\underbrace{\|(H_\sigma - I)(\mathbf{x}) - (H_\sigma - I)(\mathbf{y})\|^2 - \frac{\theta^2}{(1-\theta)^2}\|\mathbf{x}-\mathbf{y}\|^2}_{\text{(TERM A)}} \\
&\quad = \frac{1}{(1-\theta)^2}\|G(\mathbf{x}) - G(\mathbf{y})\|^2 + \left(1 - \frac{\theta^2}{(1-\theta)^2}\right)\|\mathbf{x}-\mathbf{y}\|^2 - \frac{2}{1-\theta}\langle G(\mathbf{x}) - G(\mathbf{y}), \mathbf{x}-\mathbf{y}\rangle \\
&\quad = \frac{1}{(1-\theta)^2}\Bigg(\underbrace{\|G(\mathbf{x}) - G(\mathbf{y})\|^2 + (1-2\theta)\|\mathbf{x}-\mathbf{y}\|^2 - 2(1-\theta)\langle G(\mathbf{x}) - G(\mathbf{y}), \mathbf{x}-\mathbf{y}\rangle}_{\text{(TERM B)}}\Bigg).
\end{aligned}$$

Remember that Assumption (A) corresponds to (TERM A) ≤ 0 for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$. This is equivalent to (TERM B) ≤ 0 for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$, which corresponds to G being θ-averaged by Lemma 8. $\square$

Lemma 14. *$H_\sigma : \mathbb{R}^M \to \mathbb{R}^M$ satisfies Assumption (A) if and only if*

$$\frac{1}{1+2\varepsilon}(2H_\sigma - I)$$

is nonexpansive and $\frac{2\varepsilon}{1+2\varepsilon}$-averaged.

Proof. Define $\theta = \frac{2\varepsilon}{1+2\varepsilon}$, which means $\varepsilon = \frac{\theta}{2(1-\theta)}$. Clearly, $\theta \in [0,1)$. Define $G = \frac{1}{1+2\varepsilon}(2H_\sigma - I)$, which means $H_\sigma = \frac{1}{2(1-\theta)}G + \frac{1}{2}I$. Then

$$\begin{aligned}
&\underbrace{\|(H_\sigma - I)(\mathbf{x}) - (H_\sigma - I)(\mathbf{y})\|^2 - \frac{\theta^2}{4(1-\theta)^2}\|\mathbf{x}-\mathbf{y}\|^2}_{\text{(TERM A)}} \\
=&\frac{1}{4(1-\theta)^2}\|G(\mathbf{x}) - G(\mathbf{y})\|^2 + \left(\frac{1}{4} - \frac{\theta^2}{4(1-\theta)^2}\right)\|\mathbf{x}-\mathbf{y}\|^2 \\
&- \frac{1}{2(1-\theta)}\langle G(\mathbf{x}) - G(\mathbf{y}), \mathbf{x}-\mathbf{y}\rangle \\
=&\frac{1}{4(1-\theta)^2}\Bigg(\underbrace{\|G(\mathbf{x}) - G(\mathbf{y})\|^2 + (1-2\theta)\|\mathbf{x}-\mathbf{y}\|^2 - 2(1-\theta)\langle G(\mathbf{x}) - G(\mathbf{y}), \mathbf{x}-\mathbf{y}\rangle}_{\text{(TERM B)}}\Bigg).
\end{aligned}$$

Remember that Assumption (A) corresponds to (TERM A) ≤ 0 for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$. This is equivalent to (TERM B) ≤ 0 for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$, which corresponds to G being θ-averaged by Lemma 8. $\square$

Proof of Theorem 8. In general, if operators T_1 and T_2 are L_1 and L_2-Lipschitz, then the composition T_1T_2 is (L_1L_2)-Lipschitz. By Lemma 11, $I - \alpha\nabla f$ is $\max\{|1-\alpha\mu|, |1-\alpha L|\}$-Lipschitz. By Lemma 13, H_σ is $(1+\varepsilon)$-Lipschitz. The first part of the theorem following from composing the Lipschitz constants. The restrictions on α and ε follow from basic algebra. $\square$

Proof of Theorem 9. By Lemma 12,

$$-(2\mathrm{Prox}_{\alpha f} - I)$$

is $\frac{1}{1+\alpha\mu}$-averaged, and this implies

$$(2\mathrm{Prox}_{\alpha f} - I) \circ (-I)$$

is also $\frac{1}{1+\alpha\mu}$-averaged, by Lemma 10. By Lemma 14,

$$\frac{1}{1+2\varepsilon}(2H_\sigma - I)$$

is $\frac{2\varepsilon}{1+2\varepsilon}$-averaged. Therefore,

$$\frac{1}{1+2\varepsilon}(2H_\sigma - I)(2\mathrm{Prox}_{\alpha f} - I) \circ (-I)$$

is $\frac{1+2\varepsilon\alpha\mu}{1+\alpha\mu+2\varepsilon\alpha\mu}$-averaged by Lemma 9, and this implies

$$-\frac{1}{1+2\varepsilon}(2H_\sigma - I)(2\mathrm{Prox}_{\alpha f} - I)$$

is also $\frac{1+2\varepsilon\alpha\mu}{1+\alpha\mu+2\varepsilon\alpha\mu}$-averaged, by Lemma 10.

Using the definition of averagedness, we can write

$$(2H_\sigma - I)(2\mathrm{Prox}_{\alpha f} - I) = -(1+2\varepsilon)\left(\frac{\alpha\mu}{1+\alpha\mu+2\varepsilon\alpha\mu}I + \frac{1+2\varepsilon\alpha\mu}{1+\alpha\mu+2\varepsilon\alpha\mu}R\right)$$

where R is a nonexpansive operator. Plugging this into the PNP-DRS operator, we get

$$\begin{aligned} T &= \frac{1}{2}I - \frac{1}{2}(1+2\varepsilon)\left(\frac{\alpha\mu}{1+\alpha\mu+2\varepsilon\alpha\mu}I + \frac{1+2\varepsilon\alpha\mu}{1+\alpha\mu+2\varepsilon\alpha\mu}R\right) \\ &= \underbrace{\frac{1}{2(1+\alpha\mu+2\varepsilon\alpha\mu)}}_{=A} I - \underbrace{\frac{(1+2\varepsilon\alpha\mu)(1+2\varepsilon)}{2(1+\alpha\mu+2\varepsilon\alpha\mu)}}_{=B} R, \end{aligned} \tag{4.2}$$

where define the coefficients A and B for simplicity. Clearly, $A > 0$ and $B > 0$. Then we have

$$\begin{aligned}
\|T\mathbf{x} - T\mathbf{y}\|^2 &= A^2\|\mathbf{x} - \mathbf{y}\|^2 + B^2\|R(\mathbf{x}) - R(\mathbf{y})\|^2 - 2\langle A(\mathbf{x} - \mathbf{y}), B(R(\mathbf{x}) - R(\mathbf{y}))\rangle \\
&\leq A^2 \left(1 + \frac{1}{\delta}\right) \|\mathbf{x} - \mathbf{y}\|^2 + B^2 (1 + \delta) \|R(\mathbf{x}) - R(\mathbf{y})\|^2 \\
&\leq \left(A^2 \left(1 + \frac{1}{\delta}\right) + B^2 (1 + \delta)\right) \|\mathbf{x} - \mathbf{y}\|^2
\end{aligned}$$

for any $\delta > 0$. The first line follows from plugging in equation 4.2. The second line follows from applying Young's inequality to the inner product. The third line follows from nonexpansiveness of R.

Finally, we optimize the bound. It is a matter of simple calculus to see

$$\min_{\delta>0} \left\{A^2 \left(1 + \frac{1}{\delta}\right) + B^2 (1 + \delta)\right\} = (A + B)^2.$$

Plugging this in, we get

$$\|T\mathbf{x} - T\mathbf{y}\|^2 \leq (A + B)^2\|\mathbf{x} - \mathbf{y}\|^2 = \left(\frac{1 + \varepsilon + \varepsilon\alpha\mu + 2\varepsilon^2\alpha\mu}{1 + \alpha\mu + 2\varepsilon\alpha\mu}\right)^2 \|\mathbf{x} - \mathbf{y}\|^2,$$

which is the first part of the theorem.

The restrictions on α and ε follow from basic algebra. $\square$

CHAPTER 5

Conclusions

In this final chapter, we summarize the main results in this dissertation.

In Chapter 2, we have proposed two efficient online convolutional dictionary learning methods: Modified SGD and Surrogate-Splitting. Both of them have a theoretical convergence guarantee and show better performance on both time and memory usage than the state-of-the-arts.

Chapter 3 presents multi-fold contributions in advancing the theoretical understanding of LISTA, a fast sparse coding solver learned from data.

- We give a result on asymptotic coupling between the two weight matrices in LISTA. This result leads us to eliminating one of them, thus reducing the number of trainable parameters. This elimination still retains the theoretical and experimental performance of LISTA.
- Furthermore, we introduce a thresholding scheme for support selection, which is extremely simple to implement and significantly boosts the practical convergence.
- ISTA is generally sublinearly convergent before its iterates settle on a support. We prove that, however, there exists a sequence of parameters that makes LISTA (and its variants) converge linearly since its first iteration. The learned parameters are interpretable.
- Based on the above theoretical results, we show that the layer-wise weights in LISTA need not being learned from data. That is based on decoupling LISTA training into a *data-free* analytic optimization stage (coherence minimization) followed by a lighter-weight *data-driven* learning stage without compromising the linear convergence rate.

The new scheme, called Analytic LISTA, provides important insights into the working mechanism of LISTA. Experiments shows ALISTA to perform comparably with previous LISTA models with much lighter-weight training.

- We extend the above discussions and conclusions to convolutional sparse coding, and introduce an efficient algorithm to solve the convolutional version of coherence minimization.

To our best knowledge, this is the first attempt to establish a theoretical convergence rate (upper bound) of LISTA directly. Our proofs do not rely on any indirect resemblance, e.g., to AMP [BS16] or PGD [GEB18]. The theories are supported by extensive simulation experiments, and substantial performance improvements are observed.

Chapter 4 presents the convergence analysis of two Plug-and-play methods, Plug-and-play forward-backward splitting (PNP-FBS) and PNP-ADMM. For the analysis, we assume the denoiser (regularizer) satisfies a certain Lipschitz condition that corresponds to the denoiser being close to the identity map, which is reasonable when the estimated noise level is small. In particular, we do not assume that the denoiser is nonexpansive or differentiable since most denoisers do not have such properties. Under the assumption, we show that the PnP methods are contractive iterations. We then propose real spectral normalization (realSN), a technique based on [MKK18] for more accurately constraining deep learning-based denoisers in their training to satisfy the proposed Lipschitz condition. Finally, we present experimental results validating our theory.

REFERENCES

[AAB15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.", 2015. Software available from tensorflow.org.

[AE08] Michal Aharon and Michael Elad. "Sparse and redundant modeling of image content using an image-signature-dictionary." *SIAM Journal on Imaging Sciences*, **1**(3):228–247, 2008.

[AEB06] Michal Aharon, Michael Elad, and Alfred Bruckstein. "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation." *IEEE Transactions on Signal Processing*, **54**(11):4311–4322, 2006.

[Ahl79] Lars V. Ahlfors. *Complex analysis: an introduction to the theory of analytic functions of one complex variable.* McGraw-Hill, 1979.

[AMJ18] Hemant K Aggarwal, Merry P Mani, and Mathews Jacob. "MoDL: Model-based deep learning architecture for inverse problems." *IEEE transactions on medical imaging*, **38**(2):394–405, 2018.

[AO18] Jonas Adler and Ozan Öktem. "Learned primal-dual reconstruction." *IEEE transactions on medical imaging*, **37**(6):1322–1332, 2018.

[BBR78] J. B. Bailion, R. E. Bruck, and S. Reich. "ON THE ASYMPTOTIC BEHAVIOR OF NONEXPANSIVE MAPPINGS AND SEMIGROUPS IN BANACH SPACES." *Houston Journal of Mathematics*, **4**(1), 1978.

[BC17] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces.* Springer New York, 2nd edition, 2017.

[BCM05] A. Buades, B. Coll, and J.-M. Morel. "A Non-Local Algorithm for Image Denoising." In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[BCS18] Gregery T. Buzzard, Stanley H. Chan, Suhas Sreehari, and Charles A. Bouman. "Plug-and-Play Unplugged: Optimization-Free Reconstruction Using Consensus Equilibrium." *SIAM Journal on Imaging Sciences*, **11**(3):2001–2020, 2018.

[BCW18] Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. "Can We Gain More from Orthogonality Regularizations in Training Deep Networks?" In *Advances in Neural Information Processing Systems*, pp. 4266–4276, 2018.

[BD09] Thomas Blumensath and Mike E. Davies. "Iterative hard thresholding for compressed sensing." *Applied and Computational Harmonic Analysis*, **27**(3):265 – 274, 2009.

[BDS19] Andrew Brock, Jeff Donahue, and Karen Simonyan. "Large Scale GAN Training for High Fidelity Natural Image Synthesis." In *International Conference on Learning Representations*, 2019.

[BEL13] Hilton Bristow, Anders Eriksson, and Simon Lucey. "Fast convolutional sparse coding." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 391–398, 2013.

[Ber99] Dimitri P Bertsekas. *Nonlinear programming.* Athena Scientific, 1999.

[BFT17] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. "Spectrally-normalized margin bounds for neural networks." In *Advances in Neural Information Processing Systems*, pp. 6240–6249, 2017.

[Bil08] Patrick Billingsley. *Probability and measure.* John Wiley & Sons, 2008.

[BL08] Kristian Bredies and Dirk A Lorenz. "Linear convergence of iterative soft-thresholding." *Journal of Fourier Analysis and Applications*, **14**(5-6):813–837, 2008.

[Bot99] Léon Bottou. "On-line Learning and Stochastic Approximations." In David Saad, editor, *On-line Learning in Neural Networks*, pp. 9–42. Cambridge University Press, 1999.

[BRE16] A. Brifman, Y. Romano, and M. Elad. "Turning a denoiser into a super-resolver using plug and play priors." *2016 IEEE International Conference on Image Processing*, pp. 1404–1408, 2016.

[BS16] Mark Borgerding and Philip Schniter. "Onsager-corrected deep learning for sparse linear inverse problems." In *Signal and Information Processing (GlobalSIP), 2016 IEEE Global Conference on*, pp. 227–231. IEEE, 2016.

[BSR17] Mark Borgerding, Philip Schniter, and Sundeep Rangan. "AMP-inspired Deep Networks for Sparse Linear Inverse Problems." *IEEE Transactions on Signal Processing*, **65**(16):4293–4308, 2017.

[BT97] Dimitris Bertsimas and John N Tsitsiklis. *Introduction to linear optimization.* Athena Scientific Belmont, MA, 1997.

[BT09] Amir Beck and Marc Teboulle. "A fast iterative shrinkage-thresholding algorithm for linear inverse problems." *SIAM journal on imaging sciences*, **2**(1):183–202, 2009.

[CDS98] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. "Atomic decomposition by basis pursuit." *SIAM Journal on Scientific Computing*, **20**(1):33–61, 1998.

[CF17] Il Yong Chun and Jeffrey A. Fessler. "Convolutional Dictionary Learning: Acceleration and Convergence." *IEEE Transactions on Image Processing*, 2017.

[Cha19] Stanley H. Chan. "Performance Analysis of Plug-and-Play ADMM: A Graph Signal Processing Perspective." *IEEE Transactions on Computational Imaging*, 2019.

[CL14] Stanley H Chan and Yue M Lu. "Efficient image reconstruction for gigapixel quantum image sensors." In *Signal and Information Processing (GlobalSIP), 2014 IEEE Global Conference on*, pp. 312–316. IEEE, 2014.

[CLP17] JH Rick Chang, Chun-Liang Li, Barnabas Poczos, and BVK Vijaya Kumar. "One network to solve them all—solving linear inverse problems using deep projection models." In *2017 IEEE International Conference on Computer Vision*, pp. 5889–5898. IEEE, 2017.

[CLW18] Xiaohan Chen, Jialin Liu, Zhangyang Wang, and Wotao Yin. "Theoretical linear convergence of unfolded ISTA and its practical weights and thresholds." In *Advances in Neural Information Processing Systems*, pp. 9061–9071, 2018.

[CPR13] Rakesh Chalasani, Jose C Principe, and Naveen Ramakrishnan. "A fast proximal method for convolutional sparse coding." In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pp. 1–5. IEEE, 2013.

[CWE17] S. H. Chan, X. Wang, and O. A. Elgendy. "Plug-and-Play ADMM for Image Restoration: Fixed-Point Convergence and Applications." *IEEE Transactions on Computational Imaging*, **3**(1):84–98, 2017.

[CY15] P. L. Combettes and I. Yamada. "Compositions and convex combinations of averaged nonexpansive operators." *Journal of Mathematical Analysis and Applications*, **425**(1):55–70, 2015.

[Dan66] John M. Danskin. "The theory of max-min, with applications." *SIAM Journal on Applied Mathematics*, **14**(4):641–664, 1966.

[DBE16] Y. Dar, A. M. Bruckstein, M. Elad, and R. Giryes. "Postprocessing of Compressed Images via Sequential Denoising." *IEEE Transactions on Image Processing*, **25**(7):3044–3058, 2016.

[DE03] David L Donoho and Michael Elad. "Optimally sparse representation in general (nonorthogonal) dictionaries via l1 minimization." *Proceedings of the National Academy of Sciences*, **100**(5):2197–2202, 2003.

[DFK07] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. "Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering." *IEEE Transactions on Image Processing*, **16**(8):2080–2095, 2007.

[DKB17] Kevin Degraux, Ulugbek S. Kamilov, Petros T. Boufounos, and Dehong Liu. "Online Convolutional Dictionary Learning for Multimodal Imaging." In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pp. 1617–1621, Beijing, China, September 2017.

[DKE10] A. Danielyan, V. Katkovnik, and K. Egiazarian. "Image deblurring by augmented Lagrangian with BM3D frame prior." In *Workshop on Information Theoretic Methods in Science and Engineering*, pp. 16–18, 2010.

[DKE12] A. Danielyan, V. Katkovnik, and K. Egiazarian. "BM3D Frames and Variational Image Deblurring." *IEEE Transactions on Image Processing*, **21**(4):1715–1728, 2012.

[DWY18] W. Dong, P. Wang, W. Yin, and G. Shi. "Denoising Prior Driven Deep Neural Network for Image Restoration." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[DY16] Damek Davis and Wotao Yin. "Convergence rate analysis of several splitting schemes." In *Splitting Methods in Communication, Imaging, Science, and Engineering*, pp. 115–163. Springer, 2016.

[EA06] Michael Elad and Michal Aharon. "Image denoising via sparse and redundant representations over learned dictionaries." *IEEE Transactions on Image processing*, **15**(12):3736–3745, 2006.

[EAH99] Kjersti Engan, Sven Ole Aase, and J Hakon Husoy. "Method of optimal directions for frame design." In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, volume 5, pp. 2443–2446. IEEE, 1999.

[EC16] Omar A Elgendy and Stanley H Chan. "Image reconstruction and threshold design for quanta image sensors." In *2016 IEEE International Conference on Image Processing*, pp. 978–982. IEEE, 2016.

[EHJ04] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. "Least angle regression." *The Annals of Statistics*, **32**(2):407–499, 2004.

[EK15] K. Egiazarian and V. Katkovnik. "Single image super-resolution via BM3D sparse coding." In *2015 23rd European Signal Processing Conference*, pp. 2849–2853, 2015.

[Eks16] Ender M Eksioglu. "Decoupled algorithm for MRI reconstruction using nonlocal block matching model: BM3D-MRI." *Journal of Mathematical Imaging and Vision*, **56**(3):430–440, 2016.

[Ela07] Michael Elad. "Optimized projections for compressed sensing." *IEEE Transactions on Signal Processing*, **55**(12):5695–5702, 2007.

[ERK99] Kjersti Engan, Bhaskar D. Rao, and Kenneth Kreutz-Delgado. "Frame design using FOCUSS with method of optimal directions (MOD)." In *Proceedings of Norwegian Signal Processing Symposium (NORSIG)*, pp. 65–69, 1999.

[Fis65] Donald L Fisk. "Quasi-martingales." *Transactions of the American Mathematical Society*, **120**(3):369–389, 1965.

[Fos11] Eric Fossum. "The quanta image sensor (QIS): concepts and challenges." In *Imaging Systems and Applications*. Optical Society of America, 2011.

[FPR18] Alyson K Fletcher, Parthe Pandit, Sundeep Rangan, Subrata Sarkar, and Philip Schniter. "Plug-in estimation in high-dimensional linear inverse problems: A rigorous analysis." In *Advances in Neural Information Processing Systems*, pp. 7440–7449, 2018.

[Fuc05] Jean-Jacques Fuchs. "Recovery of exact sparse representations in the presence of bounded noise." *IEEE Transactions on Information Theory*, **51**(10):3601–3608, 2005.

[FWW19] Kai Fan, Qi Wei, Wenlin Wang, Amit Chakraborty, and Katherine Heller. "InverseNet: Solving Inverse Problems with Splitting Networks." *IEEE International Conference on Multimedia and Expo*, 2019.

[GEB18] Raja Giryes, Yonina C Eldar, Alex Bronstein, and Guillermo Sapiro. "Tradeoffs between convergence speed and reconstruction accuracy in inverse problems." *IEEE Transactions on Signal Processing*, 2018.

[GFP18] Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael Cree. "Regularisation of Neural Networks by Enforcing Lipschitz Continuity." *arXiv preprint arXiv:1804.04368*, 2018.

[GHO99] Gene H Golub, Per Christian Hansen, and Dianne P O'Leary. "Tikhonov regularization and total least squares." *SIAM journal on matrix analysis and applications*, **21**(1):185–194, 1999.

[Gis17] P. Giselsson. "Tight global linear convergence rate bounds for Douglas–Rachford splitting." *Journal of Fixed Point Theory and Applications*, **19**(4):2241–2270, 2017.

[GL10] Karol Gregor and Yann LeCun. "Learning fast approximations of sparse coding." In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pp. 399–406. Omnipress, 2010.

[GL13] Saeed Ghadimi and Guanghui Lan. "Stochastic first-and zeroth-order methods for nonconvex stochastic programming." *SIAM Journal on Optimization*, **23**(4):2341–2368, 2013.

[GW17] Cristina Garcia-Cardona and Brendt Wohlberg. "Subproblem Coupling in Convolutional Dictionary Learning." In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pp. 1697–1701, Beijing, China, September 2017.

[GZX15] Shuhang Gu, Wangmeng Zuo, Qi Xie, Deyu Meng, Xiangchu Feng, and Lei Zhang. "Convolutional Sparse Coding for Image Super-resolution." In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[HA07] Ke Huang and Selin Aviyente. "Sparse representation for signal classification." In *Advances in neural information processing systems*, pp. 609–616, 2007.

[HHW15] Felix Heide, Wolfgang Heidrich, and Gordon Wetzstein. "Fast and Flexible Convolutional Sparse Coding." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5135–5143, 2015.

[HST14] F. Heide, M. Steinberger, Y.-T. Tsai, M. Rouf, D. Pajak, D. Reddy, O. Gallo, J. Liu, W. Heidrich, K. Egiazarian, J. Kautz, and K. Pulli. "FlexISP: A Flexible Camera Image Processing Framework." *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2014)*, **33**(6), 2014.

[HTL10] Mark J. Huiskes, Bart Thomee, and Michael S. Lew. "New Trends and Ideas in Visual Concept Detection: The MIR Flickr Retrieval Evaluation Initiative." In *Proceedings of the international conference on Multimedia information retrieval (MIR)*, pp. 527–536. ACM, 2010.

[HYW18] J. He, Y. Yang, Y. Wang, D. Zeng, Z. Bian, H. Zhang, J. Sun, Z. Xu, and J. Ma. "Optimizing a Parameterized Plug-and-Play ADMM for Iterative Low-Dose CT Reconstruction." *IEEE Transactions on Medical Imaging*, pp. 1–13, 2018.

[HYZ08] Elaine T. Hale, Wotao Yin, and Yin Zhang. "Fixed-point continuation for ℓ_1-minimization: methodology and convergence." *SIAM Journal on Optimization*, **19**(3):1107–1130, 2008.

[HZR16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[IS15] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 448–456, 2015.

[ITW18] Daisuke Ito, Satoshi Takabe, and Tadashi Wadayama. "Trainable ISTA for Sparse Signal Recovery." *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, 2018.

[JJB82] Richard M. Johnstone, C. Richard Johnson, Robert R. Bitmead, and Brian D. O. Anderson. "Exponential convergence of recursive least squares with exponential forgetting factor." *Systems & Control Letters*, **2**(2):77–82, 1982.

[JMB10] Rodolphe Jenatton, Julien Mairal, Francis R. Bach, and Guillaume R. Obozinski. "Proximal methods for sparse hierarchical dictionary learning." In *Proceedings of the 27th international conference on machine learning (ICML)*, pp. 487–494, 2010.

[KB06] Charles Kervrann and Jérôme Boulanger. "Unsupervised patch-based image regularization and representation." In *European Conference on Computer Vision*, pp. 555–567. Springer, 2006.

[KHL20] Jian Kang, Danfeng Hong, Jialin Liu, Gerald Baier, Naoto Yokoya, and Begüm Demir. "Learning Convolutional Sparse Coding on Complex Domain for Interferometric Phase Restoration." *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[KHS19] Andreas Kofler, Markus Haltmeier, Tobias Schaeffter, Marc Kachelrieß, Marc Dewey, Christian Wald, and Christoph Kolbitsch. "Neural Networks-based Regularization for Large-Scale Medical Image Reconstruction." *arXiv*, pp. arXiv–1912, 2019.

[KLT16] Kuldeep Kulkarni, Suhas Lohit, Pavan Turaga, Ronan Kerviche, and Amit Ashok. "ReconNet: Non-iterative reconstruction of images from compressively sensed measurements." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 449–458, 2016.

[KMW17] U. S. Kamilov, H. Mansour, and B. Wohlberg. "A Plug-and-Play Priors Approach for Solving Nonlinear Imaging Inverse Problems." *IEEE Signal Processing Letters*, **24**(12):1872–1876, 2017.

[KS80] R. Kinderman and S.L. Snell. *Markov random fields and their applications*. American mathematical society, 1980.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks." In *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[KWB12] Shiva P. Kasiviswanathan, Huahua Wang, Arindam Banerjee, and Prem Melville. "Online l_1-dictionary learning with application to novel document detection." In *Advances in Neural Information Processing Systems*, pp. 2258–2266, 2012.

[LCW16] Yu Liu, Xun Chen, Rabab K. Ward, and Z. Jane Wang. "Image Fusion With Convolutional Sparse Representation." *IEEE Signal Processing Letters*, 2016.

[LCW19] Jialin Liu, Xiaohan Chen, Zhangyang Wang, and Wotao Yin. "ALISTA: Analytic Weights Are As Good As Learned Weights in LISTA." In *International Conference on Learning Representations*, 2019.

[LGW17] Jialin Liu, Cristina Garcia-Cardona, Brendt Wohlberg, and Wotao Yin. "Online Convolutional Dictionary Learning." In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pp. 1707–1711, Beijing, China, September 2017.

[LGW18] Jialin Liu, Cristina Garcia-Cardona, Brendt Wohlberg, and Wotao Yin. "First-and Second-Order Methods for Online Convolutional Dictionary Learning." *SIAM Journal on Imaging Sciences*, **11**(2):1589–1628, 2018.

[LLL18] Canyi Lu, Huan Li, and Zhouchen Lin. "Optimized projections for compressed sensing via direct mutual coherence minimization." *Signal Processing*, **151**:45–55, 2018.

[LRH19] Q. Lyu, D. Ruan, J. Hoffman, R. Neph, M. McNitt-Gray, and K. Sheng. "Iterative reconstruction for low dose CT using Plug-and-Play alternating direction method of multipliers (ADMM) framework." *Proceedings of SPIE*, **10949**, 2019.

[LS99] Michael S. Lewicki and Terrence J. Sejnowski. "Coding Time-varying Signals Using Sparse, Shift-invariant Representations." In M. J. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in neural information processing systems*, volume 11, pp. 730–736, 1999.

[LSL05] Michael Lustig, Juan M Santos, Jin-Hyung Lee, David L Donoho, and John M Pauly. "Application of compressed sensing for rapid MR imaging." *SPARS,(Rennes, France)*, 2005.

[LZC16] Yunsong Liu, Zhifang Zhan, Jian-Feng Cai, Di Guo, Zhong Chen, and Xiaobo Qu. "Projected iterative soft-thresholding algorithm for tight frames in compressed sensing magnetic resonance imaging." *IEEE transactions on medical imaging*, **35**(9):2130–2140, 2016.

[LZZ18] Pengju Liu, Hongzhi Zhang, Kai Zhang, Liang Lin, and Wangmeng Zuo. "Multi-level Wavelet-CNN for Image Restoration." In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 886–895, 2018.

[Mal99] Stéphane Mallat. *A wavelet tour of signal processing.* Academic press, 1999.

[MB17] Thomas Moreau and Joan Bruna. "Understanding Trainable Sparse Coding with Matrix Factorization." In *ICLR*, 2017.

[MBP09] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. "Online dictionary learning for sparse coding." In *Proceedings of the 26th annual international conference on machine learning (ICML)*, pp. 689–696. ACM, 2009.

[MBP10] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. "Online learning for matrix factorization and sparse coding." *Journal of Machine Learning Research*, **11**(January):19–60, 2010.

[MBP12] Julien Mairal, Francis Bach, and Jean Ponce. "Task-driven dictionary learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **34**(4):791–804, 2012.

[MBP14] Julien Mairal, Francis Bach, and Jean Ponce. "Sparse Modeling for Image and Vision Processing." *Foundations and Trends in Computer Graphics and Vision*, **8**(2-3):85–283, 2014.

[MFT01] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. "A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics." In *Proceedings of the International Conference on Computer Vision*, volume 2, pp. 416–423, 2001.

[MKK18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. "Spectral Normalization for Generative Adversarial Networks." In *International Conference on Learning Representations*, 2018.

[MMB16] C. A. Metzler, A. Maleki, and R. G. Baraniuk. "From Denoising to Compressed Sensing." *IEEE Transactions on Information Theory*, **62**(9):5117–5144, 2016.

[MMB17] Christopher A Metzler, Ali Mousavi, and Richard G Baraniuk. "Learned D-AMP: Principled Neural Network based Compressive Image Recovery." In *Advances in Neural Information Processing Systems*, pp. 1770–1781, 2017.

[MMH17] T. Meinhardt, M. Moeller, C. Hazirbas, and D. Cremers. "Learning proximal operators: Using denoising networks for regularizing inverse imaging problems." In *2017 International Conference on Computer Vision*, p. 1799–1808, 2017.

[Mor65] J. J. Moreau. "Proximité et dualité dans un espace Hilbertien." *Bulletin de la Société Mathématique de France*, **93**:273–299, 1965.

[MSY16] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections." In *Advances in Neural Information Processing Systems*, pp. 2802–2810, 2016.

[OC18] Adam M Oberman and Jeff Calder. "Lipschitz regularized Deep Neural Networks converge and generalize." *arXiv preprint arXiv:1808.09540*, 2018.

[OMD10] Stanley Osher, Yu Mao, Bin Dong, and Wotao Yin. "Fast linearized Bregman iteration for compressive sensing and sparse denoising." *Communications in Mathematical Sciences*, **8**(1):93–111, 2010.

[Ono17] S. Ono. "Primal-Dual Plug-and-Play Image Restoration." *IEEE Signal Processing Letters*, **24**(8):1108–1112, 2017.

[OY02] N. Ogura and I. Yamada. "NON-STRICTLY CONVEX MINIMIZATION OVER THE FIXED POINT SET OF AN ASYMPTOTICALLY SHRINKING NONEXPANSIVE MAPPING." *Numerical Functional Analysis and Optimization*, **23**(1-2):113–137, 2002.

[PGM19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. "PyTorch: An imperative style, high-performance deep learning library." In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

[Pol87] B. T. Polyak. *Introduction to Optimization.* Optimization Software Inc., New York, 1987.

[PR18] Tobias Plötz and Stefan Roth. “Neural nearest neighbors networks.” In *Advances in Neural Information Processing Systems*, pp. 1095–1106, 2018.

[PRE16] Vardan Papyan, Yaniv Romano, and Michael Elad. “Convolutional neural networks analyzed via convolutional sparse coding.” *arXiv preprint arXiv:1607.08194*, 2016.

[PRE17] Vardan Papyan, Yaniv Romano, and Michael Elad. “Convolutional neural networks analyzed via convolutional sparse coding.” *Journal of Machine Learning Research*, **18**(83):1–52, 2017.

[PRS17] Vardan Papyan, Yaniv Romano, Jeremias Sulam, and Michael Elad. “Convolutional Dictionary Learning via Local Processing.” *arXiv preprint arXiv:1705.03239*, 2017.

[PSE17] Vardan Papyan, Jeremias Sulam, and Michael Elad. “Working locally thinking globally: Theoretical guarantees for convolutional sparse coding.” *IEEE Transactions on Signal Processing*, **65**(21):5687–5701, 2017.

[QJ16] Tran Minh Quan and Won-Ki Jeong. “Compressed sensing reconstruction of dynamic contrast enhanced MRI using GPU-accelerated convolutional sparse coding.” In *IEEE International Symposium on Biomedical Imaging (ISBI)*, pp. 518–521, April 2016.

[QSC18] Chen Qin, Jo Schlemper, Jose Caballero, Anthony N Price, Joseph V Hajnal, and Daniel Rueckert. “Convolutional recurrent neural networks for dynamic MR image reconstruction.” *IEEE transactions on medical imaging*, **38**(1):280–290, 2018.

[QW19] Haifeng Qian and Mark N. Wegman. “L2-Nonexpansive Neural Networks.” In *International Conference on Learning Representations*, 2019.

[RB16] E. K. Ryu and S. Boyd. “Primer on Monotone Operator Methods.” *Applied and Computational Mathematics*, **15**:3–43, 2016.

[RBE10] Ron Rubinstein, Alfred M. Bruckstein, and Michael Elad. “Dictionaries for sparse representation modeling.” *Proceedings of the IEEE*, **98**(6):1045–1057, 2010.

[REM17] Y. Romano, M. Elad, and P. Milanfar. “The Little Engine That Could: Regularization by Denoising (RED).” *SIAM Journal on Imaging Sciences*, **10**(4):1804–1844, 2017.

[RGE16] A. Rond, R. Giryes, and M. Elad. “Poisson inverse problems by the Plug-and-Play scheme.” *Journal of Visual Communication and Image Representation*, **41**:96–108, 2016.

[RLW19] Ernest Ryu, Jialin Liu, Sicheng Wang, Xiaohan Chen, Zhangyang Wang, and Wotao Yin. “Plug-and-Play Methods Provably Converge with Properly Trained Denoisers.” In *International Conference on Machine Learning*, pp. 5546–5557, 2019.

[ROF92] Leonid I Rudin, Stanley Osher, and Emad Fatemi. "Nonlinear total variation based noise removal algorithms." *Physica D: nonlinear phenomena*, **60**(1-4):259–268, 1992.

[RS19] E. T. Reehorst and P. Schniter. "Regularization by Denoising: Clarifications and New Interpretations." *IEEE Transactions on Computational Imaging*, **5**(1):52–67, 2019.

[RSA15] Oren Rippel, Jasper Snoek, and Ryan P. Adams. "Spectral representations for convolutional neural networks." In *Advances in Neural Information Processing Systems*, pp. 2449–2457, 2015.

[RW09] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009.

[SBL12] Laurent Sorber, Marc Van Barel, and Lieven De Lathauwer. "Unconstrained optimization of real functions in complex variables." *SIAM Journal on Optimization*, **22**(3):879–898, 2012.

[SBS15] Pablo Sprechmann, Alexander M Bronstein, and Guillermo Sapiro. "Learning efficient sparse and low rank models." *IEEE transactions on pattern analysis and machine intelligence*, **37**(9):1821–1833, 2015.

[SCH17] Jo Schlemper, Jose Caballero, Joseph V Hajnal, Anthony N Price, and Daniel Rueckert. "A deep cascade of convolutional neural networks for dynamic MR image reconstruction." *IEEE transactions on Medical Imaging*, **37**(2):491–503, 2017.

[SE10] Karl Skretting and Kjersti Engan. "Recursive Least Squares Dictionary Learning Algorithm." *IEEE Transactions on Signal Processing*, **58**(4):2121–2130, April 2010.

[SF18] M. Shi and L. Feng. "Plug-and-Play Prior Based on Gaussian Mixture Model Learning for Image Restoration in Sensor Network." *IEEE Access*, **6**:78113–78122, 2018.

[SG14] Konstantinos Slavakis and Georgios B. Giannakis. "Online dictionary learning from big data using accelerated stochastic approximation algorithms." In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 16–20, 2014.

[SG18] Hillel Sreter and Raja Giryes. "Learned convolutional sparse coding." In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2191–2195. IEEE, 2018.

[SLX16] Jian Sun, Huibin Li, Zongben Xu, et al. "Deep ADMM-Net for compressive sensing MRI." In *Advances in neural information processing systems*, pp. 10–18, 2016.

[SPL11] Zoltán Szabó, Barnabás Póczos, and András Lőrincz. "Online group-structured dictionary learning." In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2865–2872, 2011.

[SPR17] Jeremias Sulam, Vardan Papyan, Yaniv Romano, and Michael Elad. "Multi-Layer Convolutional Sparse Modeling: Pursuit and Dictionary Learning." *arXiv preprint arXiv:1708.08705*, 2017.

[SVB17] S. Sreehari, S. V. Venkatakrishnan, K. L. Bouman, J. P. Simmons, L. F. Drummy, and C. A. Bouman. "Multi-Resolution Data Fusion for Super-Resolution Electron Microscopy." *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017.

[SVW16] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, G. T. Buzzard, L. F. Drummy, J. P. Simmons, and C. A. Bouman. "Plug-and-Play Priors for Bright Field Electron Tomography and Sparse Interpolation." *IEEE Transactions on Computational Imaging*, **2**(4):408–423, 2016.

[SWK18] Yu Sun, Brendt Wohlberg, and Ulugbek S. Kamilov. "Plug-In Stochastic Gradient Method." *arXiv preprint arXiv:1811.03659*, 2018.

[SWK19] Y. Sun, B. Wohlberg, and U. S. Kamilov. "An Online Plug-and-Play Algorithm for Regularized Image Reconstruction." *IEEE Transactions on Computational Imaging*, 2019.

[SXL18] Yu Sun, Shiqi Xu, Yunzhe Li, Lei Tian, Brendt Wohlberg, and Ulugbek S. Kamilov. "Regularized Fourier Ptychography using an Online Plug-and-Play Algorithm." *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2018.

[TBF16] A. M. Teodoro, J. M. Bioucas-Dias, and M. A. T. Figueiredo. "Image restoration and reconstruction using variable splitting and class-adapted image priors." *IEEE International Conference on Image Processing*, 2016.

[TBF17] A. M. Teodoro, J. M. Bioucas-Dias, and M. A. T. Figueiredo. "Scene-Adapted plug-and-play algorithm with convergence guarantees." In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing*, pp. 1–6, 2017.

[TBF19] A. M. Teodoro, J. M. Bioucas-Dias, and M. A. T. Figueiredo. "A Convergent Image Fusion Algorithm Using Scene-Adapted Gaussian-Mixture-Based Denoising." *IEEE Transactions on Image Processing*, **28**(1):451–463, 2019.

[TBZ16] Shaozhe Tao, Daniel Boley, and Shuzhong Zhang. "Local linear convergence of ISTA and FISTA on the LASSO problem." *SIAM Journal on Optimization*, **26**(1):313–336, 2016.

[TDB18] Bahareh Tolooshams, Sourav Dey, and Demba Ba. "Scalable Convolutional Dictionary Learning with Constrained Recurrent Sparse Auto-encoders." *arXiv preprint arXiv:1807.04734*, 2018.

[TG19] Tom Tirer and Raja Giryes. "Image restoration by iterative denoising and backward projections." *IEEE Transactions on Image Processing*, **28**(3):1220–1234, 2019.

[THG18] A. B. Taylor, J. M. Hendrickx, and F. Glineur. "Exact Worst-Case Convergence Rates of the Proximal Gradient Method for Composite Convex Minimization." *Journal of Optimization Theory and Applications*, 2018.

[Tik63] Andrei Nikolaevich Tikhonov. "On the solution of ill-posed problems and the method of regularization." In *Doklady Akademii Nauk*, pp. 501–504. Russian Academy of Sciences, 1963.

[TSS18] Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. "Lipschitz-Margin Training: Scalable Certification of Perturbation Invariance for Deep Neural Networks." In *Advances in Neural Information Processing Systems*, pp. 6541–6550, 2018.

[TYG12] Ye Tang, Yu-Bin Yang, and Yang Gao. "Self-paced dictionary learning for image classification." In *Proceedings of the 20th ACM international conference on Multimedia*, pp. 833–836, 2012.

[Vaa00] Aad W. Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge University Press, 2000.

[VBW13] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. "Plug-and-Play priors for model based reconstruction." *2013 IEEE Global Conference on Signal and Information Processing*, pp. 945–948, 2013.

[vv16] Michal Šorel and Filip Šroubek. "Fast convolutional sparse coding using matrix inversion lemma." *Digital Signal Processing*, 2016.

[WC17] Xiran Wang and Stanley H Chan. "Parameter-free plug-and-play ADMM for image restoration." In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1323–1327. IEEE, 2017.

[WCZ16] Zhangyang Wang, Shiyu Chang, Jiayu Zhou, Meng Wang, and Thomas S Huang. "Learning a task-specific deep architecture for clustering." In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pp. 369–377. SIAM, 2016.

[WLC16] Zhangyang Wang, Ding Liu, Shiyu Chang, Qing Ling, Yingzhen Yang, and Thomas S Huang. "D3: Deep dual-domain based fast restoration of JPEG-compressed images." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2764–2772, 2016.

[WLH16] Zhangyang Wang, Qing Ling, and Thomas Huang. "Learning deep l0 encoders." In *AAAI Conference on Artificial Intelligence*, pp. 2194–2200, 2016.

[WMM10] John Wright, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S. Huang, and Shuicheng Yan. "Sparse representation for computer vision and pattern recognition." *Proceedings of the IEEE*, **98**(6):1031–1044, 2010.

[Woh14] Brendt Wohlberg. "Efficient Convolutional Sparse Coding." In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 7173–7177, May 2014.

[Woh16a] Brendt Wohlberg. "Boundary Handling for Convolutional Sparse Representations." In *Proceedings IEEE Conference Image Processing (ICIP)*, pp. 1833–1837, Phoenix, AZ, USA, September 2016.

[Woh16b] Brendt Wohlberg. "Convolutional Sparse Representation of Color Images." In *Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, pp. 57–60, Santa Fe, NM, USA, March 2016.

[Woh16c] Brendt Wohlberg. "Convolutional Sparse Representations as an Image Model for Impulse Noise Restoration." In *Proceedings of the IEEE Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, Bordeaux, France, July 2016.

[Woh16d] Brendt Wohlberg. "Efficient Algorithms for Convolutional Sparse Representations." *IEEE Transactions on Image Processing*, **25**(1):301–315, January 2016.

[Woh16e] Brendt Wohlberg. "Efficient Algorithms for Convolutional Sparse Representations." *IEEE Transactions on Image Processing*, **25**:301–315, 2016.

[Woh16f] Brendt Wohlberg. "SParse Optimization Research COde (SPORCO)." Software library available from `http://purl.org/brendt/software/sporco`, 2016.

[Woh17a] Brendt Wohlberg. "ADMM Penalty Parameter Selection by Residual Balancing." *Preprint arXiv:1704.06209*, 2017.

[Woh17b] Brendt Wohlberg. "SPORCO: A Python package for standard and convolutional sparse representations." In *Proceedings of the 15th Python in Science Conference*, pp. 1–8, Austin, TX, USA, July 2017.

[WYC16] Zhangyang Wang, Yingzhen Yang, Shiyu Chang, Qing Ling, and Thomas S. Huang. "Learning a deep ℓ_∞ encoder for hashing." In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 2174–2180. AAAI Press, 2016.

[WYG09] John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. "Robust face recognition via sparse representation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **31**(2):210–227, 2009.

[WYK18] Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. "Scalable Online Convolutional Sparse Coding." *IEEE Transactions on Image Processing*, 2018.

[WYY10] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. "Locality-constrained linear coding for image classification." In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3360–3367, 2010.

[WZC18] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. "Evaluating the Robustness of Neural Networks: An Extreme Value Theory Approach." In *International Conference on Learning Representations*, 2018.

[XWG16] Bo Xin, Yizhou Wang, Wen Gao, David Wipf, and Baoyuan Wang. "Maximal sparsity with deep networks?" In *Advances in Neural Information Processing Systems*, pp. 4340–4348, 2016.

[XY16] Yangyang Xu and Wotao Yin. "A fast patch-dictionary method for whole image recovery." *Inverse Problems and Imaging*, **10**(2):563–583, 2016.

[XZ13] Lin Xiao and Tong Zhang. "A proximal-gradient homotopy method for the sparse least-squares problem." *SIAM Journal on Optimization*, **23**(2):1062–1091, 2013.

[YST18] D. H. Ye, S. Srivastava, J. Thibault, K. Sauer, and C. Bouman. "Deep Residual Learning for Model-Based Iterative CT Reconstruction Using Plug-and-Play Framework." In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6668–6672, 2018.

[YWH10] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. "Image super-resolution via sparse representation." *IEEE Transactions on Image Processing*, **19**(11):2861–2873, 2010.

[YZY10] Junfeng Yang, Yin Zhang, and Wotao Yin. "A fast alternating direction method for TVL1-L2 signal reconstruction from partial Fourier data." *IEEE Journal of Selected Topics in Signal Processing*, **4**(2):288–297, 2010.

[ZC15] Hui Zhang and Lizhi Cheng. "Restricted strong convexity and its applications to convergence analysis of gradient-type methods in convex optimization." *Optimization Letters*, **9**(5):961–979, 2015.

[ZDD18] Joey Tianyi Zhou, Kai Di, Jiawei Du, Xi Peng, Hao Yang, Sinno Jialin Pan, Ivor W Tsang, Yong Liu, Zheng Qin, and Rick Siow Mong Goh. "SC2Net: Sparse LSTMs for Sparse Coding." In *AAAI Conference on Artificial Intelligence*, 2018.

[ZG18a] Jian Zhang and Bernard Ghanem. "ISTA-Net: Interpretable Optimization-Inspired Deep Network for Image Compressive Sensing." In *IEEE CVPR*, 2018.

[ZG18b] Jian Zhang and Bernard Ghanem. "ISTA-Net: Interpretable Optimization-Inspired Deep Network for Image Compressive Sensing." In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[ZHL17] Lufang Zhang, Yaohua Hu, Chong Li, and Jen-Chih Yao. "A new linear convergence result for the iterative soft thresholding algorithm." *Optimization*, **66**(7):1177–1189, 2017.

[ZJD12] Guangxiao Zhang, Zhuolin Jiang, and Larry S. Davis. "Online Semi-Supervised Discriminative Dictionary Learning for Sparse Representation." In *Proceedings of Asian Conference on Computer Vision (ACCV)*, pp. 259–273, 2012.

[ZKT10] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor, and Rob Fergus. "Deconvolutional networks." In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2528–2535, June 2010.

[ZKY15] Shengping Zhang, Shiva Kasiviswanathan, Pong C. Yuen, and Mehrtash Harandi. "Online Dictionary Learning on Symmetric Positive Definite Manifolds with Vision Applications." In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 3165–3173, 2015.

[ZP16] He Zhang and Vishal M. Patel. "Convolutional Sparse Coding-based Image Decomposition." In *Proceedings of British Machine Vision Conference (BMVC)*, York, UK, September 2016.

[ZP17] He Zhang and Vishal M. Patel. "Convolutional sparse and low-rank coding-based rain streak removal." In *Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2017.

[ZW11] D. Zoran and Y. Weiss. "From learning models of natural image patches to whole image restoration." In *2011 International Conference on Computer Vision*, pp. 479–486, 2011.

[ZWZ18] Hao Zhang, Jing Wang, Dong Zeng, Xi Tao, and Jianhua Ma. "Regularization strategies in statistical image reconstruction of low-dose x-ray CT: A review." *Medical physics*, **45**(10):e886–e907, 2018.

[ZXY15] Zheng Zhang, Yong Xu, Jian Yang, Xuelong Li, and David Zhang. "A survey of sparse representation: algorithms and applications." *IEEE Access*, **3**:490–530, 2015.

[ZZC17] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. "Beyond a Gaussian denoiser: Residual learning of deep cnn for image denoising." *IEEE Transactions on Image Processing*, **26**(7):3142–3155, 2017.

[ZZG17] K. Zhang, W. Zuo, S. Gu, and L. Zhang. "Learning Deep CNN Denoiser Prior for Image Restoration." In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2808–2817, 2017.

[ZZZ18] Kai Zhang, Wangmeng Zuo, and Lei Zhang. "FFDNet: Toward a fast and flexible solution for CNN based image denoising." *IEEE Transactions on Image Processing*, **27**(9):4608–4622, 2018.