Compressed Representations of Signals and Models

By

Jonathan Ashbrock

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Mathematics

May 14, 2021

Nashville, Tennessee

Approved:

Alexander Powell, Ph.D.

Douglas Hardin, Ph.D.

Akram Aldroubi, Ph.D.

Mike Neamtu, Ph.D.

David Smith, Ph.D.

ACKNOWLEDGMENTS

I would first like to thank my entire dissertation committee, Professors Hardin, Aldroubi, Neamtu, and Smith, and in particular my advisor Dr. Alex Powell. Dr. Powell taught me everything about a subject I knew nothing about. He taught me that small progress is progress. He taught me that big progress doesn't just happen, it is the result of multiple instances of small progress. He taught me how to consider complex problems in the simplest cases to gain insights in intricate settings. He also taught me to how to be both careful and explicit with what I read and write. Lastly, he taught me the importance of being a person in addition to being an academic. In a field so reliant on collaboration, strong personal relationships are paramount.

I want to thank a few particular people for special support and encouragement over these last five years. First, I want to thank my sisters for teaching me that asking for help is something to be admired rather than stigmatized. Nobody has all the answers, asking others to show you the way leads to success.

I want to thank my friends from both Vanderbilt and from Dayton for reminding me that there is life outside of academics to be enjoyed.

I want to thank Josie for teaching me that all you have to do to be happy is to choose to be.

I want to thank my in-laws for constant support of both myself and Maureen.

I want to thank my parents for teaching me that the most important thing in life is to pursue something you care about so much you don't mind doing it from the moment you wake up. Maybe more importantly, my parents taught me that intellect is not a substitute for hard-work. The key to any success I have found is due to the example they have (and continue to) set. Lastly, thank you for all the other small ways you have helped me over the last 26 years.

And lastly, of course, I wanted to thank my wife Maureen. Listing everything you have done for me would likely result in this thesis going over some maximum page limit. I'll highlight a few, though. Thank you for patiently listening to me talk about math when I just needed to hear something said out loud. Thank you for teaching me how to teach - I am a good teacher because of you. Thank you for reminding me when it is time to live my life instead of living on my computer. Thank you for telling me that if I have an idea to do something, I have the power to make it happen. Thank you for keeping me sane over the course of these last five years.

Many people talk about the stress, anxiety, and difficulty of a PhD program. Luckily because of the support of everyone above and many others, my time spent at Vanderbilt these last five years was quite enjoyable. I am confident that without the support I have received I would not have been able to say such a thing. So, for one last time, thank you to everyone for helping me along the way.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER 1

Introduction

1.1 A Mathematical Theory of Compression

Digital data is doubly finite. Not only are we constrained to store finite length sequences, each element in the sequence can only take one of finitely many values. So, while signals are typically understood to exist as vectors in some vector space, only a finite subset of that vector space is accessible. The problem of data compression is, at its core, finding efficient representations of large classes of signals in a way that maximizes our finite allotment of memory.

Every digitally stored signal is accompanied with an understanding of what the doubly finite sequence represents. The process of converting from stored digital data to a meaningful object is called *reconstruction*. In the simplest case, one may understand the sequence of numbers to represent the coefficients in a basis expansion of a vector space. However, one could consider other - e.g. non-linear or redundant linear - reconstruction methods for our stored data. In these alternate settings, the mathematical theory of compression is extremely rich and presents many avenues for finding highly condensed representations.

One of the central objects of study is the *quantization alphabet*. Recalling that each element in our sequence can take only one of finitely many values, the *quantization alphabet* is the set of these possible values. For example, if we are working with binary data, the quantization alphabet is the set $\mathcal{A} = \{0, 1\}$. Every element in a binary sequence must either be a 0 or a 1.

Now, if $\mathcal{V}$ is some vector space, reconstruction of a signal from stored digital data is a rule that maps $\varphi : \mathcal{A}^m \to \mathcal{V}$. In this language, non-linear and redundant linear reconstruction refer to properties of the reconstructing function φ. We study non-linear and redundant linear representations throughout this thesis.

This thesis is divided into three main parts. In the first two Chapters we discuss compression of an already attained signal while in the final Chapter we discuss the design of the acquisition process when the signal is known to already be compressed. In Chapter 2 we study compression with non-linear reconstruction in order to quantize neural networks. In Chapter 3 we study compression with redundant linear quantization by developing theory of the existence of highly structured frames. In Chapter 4 we turn our attention away

from finding compressed representations to investigate a new technique for acquiring a signal that is known to be sparse.

The focus of Chapter 2 is the study of quantization of neural networks. The stored parameter vector for a neural network is a list of the weights and biases that define the network's action. Neural networks historically have been trained with stochastic gradient descent which results in networks whose weights are floating point. However, it is generally understood that neural networks are heavily *over parameterized* meaning most of the weights are redundant or otherwise superfluous. Because of this fact, we are motivated to study ways to compute highly compressed neural networks. This research direction is not novel - many researchers have proposed various neural network quantization algorithms - but our approach to the problem is. We propose a gradient descent alternative for lattice constrained optimization of differentiable functions: stochastic Markov gradient descent (SMGD). In addition to achieving experimental success in quantizing neural networks, SMGD is among the first such algorithms to have theoretical convergence analysis.

Frame theory and quantization of redundant linear representations is the content of Chapter 3 of this thesis. While frames are precisely defined objects, finite frames for $\mathbb{R}^d$ are nothing more than finite spanning sets. If $f^1, \dots, f^m \in \mathbb{R}^d, m > d$ are the frame vectors, the map

$$(x_1, \dots x_m) \mapsto \sum_{i=1}^{m} x_i f^i$$

is a redundant linear representation of a vector in $\mathbb{R}^d$. If we wish to find quantized representations, our goal is to find $(x_1, \dots, x_m)$ in $\mathcal{A}^m$, the finite set of digital signals, which is close to a fixed signal after reconstruction. Many techniques to solve this problem use error diffusion to exploit the inherent linear structure. We analyze the frames directly and show that a highly structured class of frames exists that allows for extremely efficient error diffusion.

Finally, in Chapter 4 we turn our attention to the very closely related problem of efficient acquisition of compressed signals. This work belongs to the field of compressed sensing. A vector is *sparse* with respect to a basis if most of the coefficients are zero when expanded in that basis. Sparse signals are important because they are a naturally compressed representation - we need only store the locations and values of the non-zero entries - and because many important real world signal classes are naturally sparse when expanded in the correct basis. Sparse signals have advantageous properties when it comes to signal acquisition. In general, a vector in $\mathbb{R}^m$ requires m linear measurements to acquire/recover. However, if we know the vector

we are measuring is sparse, we may use far fewer than m measurements to determine the vector uniquely. We introduce a new tool to be used in compressed sensing that allows for more accurate ad efficient signal acquisition.

CHAPTER 2

Compression of Neural Networks

2.1 Introduction

Our first topic of study is the quantization of neural networks. Neural networks are a widely used tool for classification and regression tasks, [25, 24]. Given training data $\{x_j\}_{j=1}^m \subset \mathbb{R}^d$ and a set of labels $\{l_j\}_{j=1}^m \subset \mathbb{R}$, the general goal is to learn a function y that explains the training set by

$$y(x_j) \approx l_j.$$

Neural networks address this by using a specially structured output function $y(x) = y(x, w)$ that is parametrized by a high-dimensional vector $w \in \mathbb{R}^n$ of weights and biases. In a standard feedforward neural network, y is an iterated composition of nonlinear activations and affine maps [17]. More generally, when the training data consists of objects with particular structure, such as images or time series, the output function y may incorporate additional components such as convolutional neurons [25] or feedback [20].

The universal approximation Theorem [10] and later advances, e.g., [11, 34, 36, 37], provide a theoretical foundation for neural networks, and show that weight parameters w can be selected so that the network output y expresses a wide class of input-output relationships. While neural networks enjoy approximation-theoretic power, the large size of the network weight set w creates nontrivial practical challenges during implementation:

- Nonconvexity of the cost function leads to non-unique minima during training.
- Slow training times can occur due to the large number of network parameters.
- Large networks yield slow signal propagation and consequently slow classification.
- Large amounts of memory are needed to store the network parameters.

These computational burdens have motivated the study of *quantized neural networks*. In a standard neural network, the weight parameters w are full-precision floating point numbers. Instead, quantized neural

networks use weight parameters that are intentionally represented using only a small number of bits. For example, in the extreme case of binary neural networks, each weight only contains a single bit of information and so is constrained to take one of only two possible values.

It has been shown recently, [9, 38, 31], that quantized neural networks can match the state-of-the-art performance obtained by comparably-sized full-precision neural networks. Somewhat paradoxically, over-parametrization creates computational challenges for implementing neural networks, but it also provides flexibility which allows heavily quantized, even one-bit, networks to perform well. Moreover, the use of low-bit neural networks reduces the memory requirements needed to store network parameters, can be used to speed up signal propagation through networks [31], and can also be viewed as having a regularizing effect.

In this work, we address two aspects of the neural network quantization program. First, we introduce a method, stochastic Markov gradient descent (SMGD), that produces neural networks that are low-memory during training as well as at run-time. For comparison, in [22] network weights are quantized at run-time but the method requires storage of full-precision auxiliary weights for the parameter update step during training which results in increased train-time memory requirements. Later works, [38], produce smaller run-time memory requirements by quantizing gradients and activations. Moreover, this has the effect of faster training because quantized gradients and activations allow access to bitwise operations during both the forward and the backward pass. We place particular emphasis on the memory requirements during the training phase since existing quantization methods typically require increased memory requirements during network learning. Our method is the first to our knowledge that allows training of highly-accurate networks while memory is constrained at both train and run-time.

Secondly, the theoretical understanding of quantized neural networks is still being developed. The problem of neural network quantization forces one to solve a discrete optimization problem in extremely high dimensions rather than a continuous problem. This high dimensionality disallows the use of many standard discrete optimization techniques. Therefore, existing methods often involve an ad hoc blend of gradient-based methods and discrete optimization techniques. For example, [16] uses k-means to cluster similar weights together before quantization. The algorithm in [9] quantizes weights during the forward pass while applying the gradient descent update to full-precision, pre-quantized weights. More recent methods [38, 22] generally involve a mild variation on this last idea to achieve goals including quantized gradients, activations, or to apply these ideas to recurrent neural networks. The work in [35] quantizes neural networks using an approach based on blended coarse gradient descent. Towards a more theoretically robust understanding,

the work in [21] incorporates quantization error directly into the cost function. Our approach is based on a simple probabilistic variation of stochastic gradient descent, and proves theoretical performance guarantees which are highly coincident with their counterparts in traditional stochastic gradient descent. These results give us intuition for how the networks learn and yield evidence for the effectiveness of stochastic Markov gradient descent as a tool for quantizing neural networks.

The main contributions of this Chapter are:

- We introduce stochastic Markov gradient descent (SMGD) for producing neural networks whose weights are fully quantized during both training and at run-time, allowing one to learn accurate networks in low-memory environments, see Section 2.3.
- We prove theoretical performance guarantees for SMGD in a general setting and draw strong comparisons to comparable results for stochastic gradient descent, see Theorems 2.4.1 and 2.5.1.
- We numerically validate the SMGD algorithm and show that it performs well on various benchmark sets for image classification, see Section 2.7
- We highlight the setting where memory is constrained during training, and show there are instances where networks trained by SMGD can outperform full-precision networks in a bit-for-bit comparison.

The remainder of the Chapter is organized as follows. Section 2.2 covers some necessary background information on gradient descent and neural network training. Section 2.3 introduces the SMGD algorithm and gives a brief discussion of intuition for why the algorithm works. Section 2.4 proves our first main results on the behavior of the cost function $f(x^t)$ under iterations of SMGD, see Theorem 2.4.1. Section 2.5 proves our next main results on rates of convergence for the iterates x^t of SMGD in the special case of strongly convex cost functions f, see Theorem 2.5.1. Section 2.6 collects several corollaries of our main results to illustrate the performance of SMGD in the non-stochastic setting, i.e., when we have access to the gradient itself. Section 2.7 contains numerical results which show that SMGD performs well in various settings.

2.2 Background: stochastic gradient descent

Neural network training is the process of using labelled training data $\{(x_j, l_j)\}_{j=1}^m$ to determine a good choice of network parameters w. Training is typically formulated as a minimization problem

$$\min_{w \in \mathbb{R}^n} f(w), \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a cost function associated to the network and training data. In machine learning in particular, f is often of the form

$$f(w) = \frac{1}{m} \sum_{i=1}^{m} f^i(w), \tag{2.2}$$

where $f^i(w)$ measures an error between the label l_i and the network output $y(x_i, w)$ for the ith piece of training data.

The backpropogation algorithm allows efficient computation of $\nabla f^i(w)$, a portion of the gradient of our cost function, which in turn opens the toolbox of gradient-based methods for model selection. Standard gradient descent is an iterative method which addresses (2.1) by making updates in the direction of the negative gradient. However, because of the form (2.2), even if ∇f^i may be efficiently computed, it may be slow to compute the entirety of ∇f if m is relatively large. In practice, m is often extremely large as we have access to larger and larger data sets to learn from.

Given a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, we say that a stochastic function $G : \mathbb{R}^n \to \mathbb{R}$ is an *unbiased estimator* of ∇f if $\mathbb{E}[G(x)] = \nabla f(x)$ where the expectation is with respect to the realization of G. In the case when f is of the form (2.2), typical examples of unbiased estimators G are:

- *Uniform*. Draw i uniformly at random from $\{1, \cdots, m\}$ and let $G(x) = \nabla f^i(x)$.
- *Mini-batch estimates*. Draw k distinct integers $i_1, \cdots, i_k$ uniformly at random without replacement from $\{1, \cdots, m\}$, and let $G(x) = \frac{1}{k} \sum_{j=1}^{k} \nabla f^{i_j}(x)$.

Stochastic gradient descent (SGD) addresses the minimization problem (2.1) by updating the parameter vector w^t at step t with the following the iteration

$$w^{t+1} = w^t - \lambda \, G^t(w^t), \tag{2.3}$$

where $G^t(w^t)$ is an unbiased estimate of $\nabla f(w^t)$ at iterate t of SGD. We consider the case when the learning

rate $\lambda \in (0, \infty)$ is constant, but it is also common to vary the learning at each iteration. Convergence properties of stochastic gradient descent are well-studied, especially for machine learning, e.g., [13, 23, 27, 28].

2.3 Stochastic Markov Gradient Descent

In this Chapter, our goal is to minimize a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ constrained to a scaled lattice $\alpha\mathbb{Z}^n$ given access to unbiased estimators of the gradient ∇f. Our approach, stochastic Markov gradient descent (SMGD), generalizes the least squares Markov gradient descent algorithm that was introduced for digital halftoning in [33], and is a variant of SGD where additional randomness is employed to allow the iterates to remain on the lattice. Throughout the remainder of this work, we let G^t denote an unbiased estimator of the gradient at step t. We generally use subscripts to denote coordinates of vectors, so that x_i^t denotes the ith coordinate of $x^t \in \mathbb{R}^n$ and $G^t(x^t)_i$ denotes the ith coordinate of the unbiased estimator $G^t(x^t)$ of $\nabla f(x^t)$.

The stochastic Markov gradient descent algorithm is described below.

Stochastic Markov Gradient Descent (SMGD)
Input: $f : \mathbb{R}^n \to \mathbb{R}$, stepsize α, initial $x^0 \in \alpha\mathbb{Z}^n$, number of iterations T, normalizer $\eta > 0$
Output: $x^T \in \alpha\mathbb{Z}^n$, an estimate of the minimizer
for $t = 1, \ldots, T$ iterations **do**
  Compute an unbiased estimator $G^t(x^t)$ of the gradient vector $\nabla f(x^t)$
  for each coordinate x_i^t **do**
    Let Δ_i^t be a Bernoulli random variable with $\mathbb{P}[\Delta_i^t = 1] = \min(\left|G^t(x^t)_i\right|/\eta, 1)$
    Update $x_i^{t+1} = x_i^t - \alpha \cdot sgn(G^t(x^t)_i)\Delta_i^t$
  end for
end for

We shall make the following probabilistic assumptions for SMGD throughout the Chapter:

- We assume that G is an unbiased estimator for ∇f, and that $\{G^t\}_{t=1}^T$ are independent identically distributed versions of G.
- We assume that each unbiased estimator G^t is independent of x^t, so that

$$\mathbb{E}[G^t(x^t)|x^t] = \nabla f(x^t). \tag{2.4}$$

Our analysis will require a slightly stronger independence assumption than (2.4). Let $\mathcal{E}^t$ denote the

event $\|G^t(x^t)\|_\infty \leqslant \eta$. We assume further that

$$\mathbb{E}[G^t(x^t)|x^t, \mathcal{E}^t] = \nabla f(x^t). \tag{2.5}$$

- We assume that the conditional distribution of Δ_i^t given G^t and x^t is a Bernoulli distribution with

$$\mathbb{P}[\Delta_i^t = 1 \,|\, G^t, x^t] = \min(\left|G^t(x^t)_i\right|/\eta, 1). \tag{2.6}$$

Standard stochastic gradient descent (2.3) makes updates that move non-discretely in the negative gradient direction $-\nabla f(x^t)$ in expectation. However, SGD does not in general produce solutions x^t that are constrained to the lattice $\alpha\mathbb{Z}^n$. To remain constrained to the lattice $\alpha\mathbb{Z}^n$, one should only make discrete updates in each direction. Therefore, SMGD instead updates each coordinate of x^t by a fixed amount *with some probability* chosen so that the expected update remains in the same direction as SGD. To see this, note that if $\mathcal{E}^t$ is the event that $\|G^t(x^t)\|_\infty \leqslant \eta$ then, by (2.5) and (2.6), one has

$$\begin{aligned}
\mathbb{E}[x_i^{t+1} \,|\, x^t, \mathcal{E}^t] &= \mathbb{E}[x_i^t - \alpha \cdot \operatorname{sgn}\left(G^t(x^t)_i\right) \Delta_i^t \,|\, x^t, \mathcal{E}^t] \\
&= \mathbb{E}\Big[\mathbb{E}[x_i^t - \alpha \cdot \operatorname{sgn}\left(G^t(x^t)_i\right) \Delta_i^t \,|\, x^t, G^t, \mathcal{E}^t] \,\Big|\, x^t, \mathcal{E}^t\Big] \\
&= \mathbb{E}\Big[x_i^t - \alpha \cdot \operatorname{sgn}\left(G^t(x^t)_i\right) \frac{|G^t(x^t)_i|}{\eta} \,\Big|\, x^t, \mathcal{E}^t\Big] \\
&= x_i^t - \frac{\alpha}{\eta}\mathbb{E}\Big[G^t(x^t)_i \,\Big|\, x^t, \mathcal{E}^t\Big] \\
&= x_i^t - \frac{\alpha}{\eta}\frac{\partial f}{\partial x_i}(x^t).
\end{aligned} \tag{2.7}$$

In view of (2.7), SMGD can be seen as a modification of SGD that keeps iterates x^t on $\alpha\mathbb{Z}^n$ by introducing extra noise at each update step. For this reason, the majority of our error analysis will occur conditioned on the event $\mathcal{E}^t$ which led to the interpretation (2.7). There are similarities between the lattice resolution α in SMGD and the learning rate in standard SGD; we shall see that some convergence properties of SMGD rely on α in the same way that SGD relies on the learning rate, e.g., see Theorem 2.5.1.

Stochastic Markov gradient descent follows the nomenclature used for least squares Markov gradient descent in [33]. In particular, since the estimators G^t are independent, SMGD generates a random walk on $\alpha\mathbb{Z}^n$ that is a Markov process.

2.4 Error estimates: cost function bounds

This Section presents Theorems that control how much the cost function f decreases at each iteration of SMGD. Our first main Theorem, Theorem 2.4.1, provides an upper bound on the expected value of $f(x^{t+1})$ in terms of gradient information. We assume that the gradient of f is L-Lipschitz continuous, i.e., $\|\nabla f(x) - \nabla f(y)\| \leqslant L\|x - y\|$.

Theorem 2.4.1. *Suppose the cost function $f : \mathbb{R}^n \to \mathbb{R}$ has L-Lipschitz gradient ∇f. Suppose G^t are independent versions of an unbiased estimator G for ∇f. Let $\mathcal{E}^t$ denote the event that $\|G^t(x^t)\|_\infty \leqslant \eta$. The iterate x^{t+1} of SMGD satisfies*

$$\mathbb{E}\left[f(x^{t+1}) \,|\, x^t, \mathcal{E}^t\right] \leqslant f(x^t) + \frac{L\alpha^2}{2\eta}\mathbb{E}\left[\|G(x^t)\|_1 \,|\, x^t, \mathcal{E}^t\right] - \frac{\alpha}{\eta}\|\nabla f(x^t)\|_2^2. \tag{2.8}$$

The proof of Theorem 2.4.1 is given in Section 2.4.1. Section 2.4.2 gives further insight into Theorem 2.4.1 in the special case of gradient estimators using mini-batches.

2.4.1 Proof of first main theorem

Gradient-based methods implicitly approximate cost functions by linear surrogates and use this approximation to move towards a minimum. Lipschitz continuity of the gradient is a frequent assumption in SGD literature because it controls the quality of linear approximation. We shall use the following standard lemma, e.g., [30].

Lemma 2.4.2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable and suppose that $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ is $L-$Lipschitz. If $D_p f(x)$ denotes the directional derivative of f in the direction p at x, then*

$$\left|f(x) + \|p\|_2 D_{p/\|p\|_2} f(x) - f(x+p)\right| \leqslant \frac{L\|p\|_2^2}{2}. \tag{2.9}$$

We use a specific case of Lemma 2.4.2 where p is of a form applicable to SMGD. Recall that the SMGD iterates x^t are defined component-wise by

$$x_i^{t+1} = x_i^t - \alpha \operatorname{sgn}(G^t(x^t)_i)\Delta_i^t. \tag{2.10}$$

Since each $\Delta_i \in \{0, 1\}$ is a Bernoulli random variable, let $\Omega^t = \{i \in \{1, 2, \cdots, n\} : \Delta_i \neq 0\}$ denote

the set of indices for which Δ_i is nonzero. Namely, Ω^t contains the indices of the coordinates in which x^t undergoes an update, and (2.10) can be written in vector form as $x^{t+1} = x^t + u^t$, where

$$u^t = -\alpha \sum_{i \in \Omega^t} \operatorname{sgn}\left(G^t(x^t)_i\right) e^i \tag{2.11}$$

and $\{e^i\}_{i=1}^n$ is the canonical basis for $\mathbb{R}^n$.

Corollary 2.4.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable everywhere and suppose that $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ is $L-$Lipschitz. The iterates x^t of SMGD satisfy*

$$f(x^{t+1}) \leqslant f(x^t) + \frac{L\alpha^2|\Omega^t|}{2} - \alpha \sum_{i \in \Omega^t} sgn\left(G^t(x^t)_i\right) \frac{\partial f}{\partial x_i}(x^t). \tag{2.12}$$

Proof. Apply Lemma 2.4.2 with $x = x^t$ and $p = u^t$ and note that $\|u^t\|_2^2 = \alpha^2|\Omega^t|$. □

For the proof of Theorem 2.4.1, we need two lemmas that compute conditional expectations of the terms in (2.12).

Lemma 2.4.4. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a cost function and suppose G^t are unbiased estimators of ∇f. Let $\mathcal{E}^t$ denote the event that $\|G^t(x^t)\|_\infty \leqslant \eta$. Then SMGD satisfies*

$$\mathbb{E}\left[|\Omega^t| \mid x^t, \mathcal{E}^t\right] = \frac{1}{\eta}\mathbb{E}\left[\|G^t(x^t)\|_1 \mid x^t, \mathcal{E}^t\right].$$

Proof. Let Δ_i^t be the Bernoulli random variable with parameter $\frac{1}{\eta}\left|G^t(x^t)_i\right|$, as in the definition of SMGD. Observe that $|\Omega^t| = \sum_{i=1}^n \Delta_i^t$, so that by (2.5) we may expand

$$\begin{aligned}
\mathbb{E}\left[|\Omega^t| \,|x^t, \mathcal{E}^t\right] = \sum_{i=1}^n \mathbb{E}\left[\Delta_i^t \mid x^t, \mathcal{E}^t\right] &= \sum_{i=1}^n \mathbb{E}\Big[\mathbb{E}[\Delta_i^t \mid G^t, x^t] \,\Big|\, x^t, \mathcal{E}^t\Big] \\
&= \sum_{i=1}^n \mathbb{E}\left[\frac{1}{\eta}\left|G^t(x^t)_i\right| \,\middle|\, x^t, \mathcal{E}^t\right] \\
&= \frac{1}{\eta}\mathbb{E}\left[\sum_{i=1}^n \left|G^t(x^t)_i\right| \,\middle|\, x^t, \mathcal{E}^t\right] \\
&= \frac{1}{\eta}\mathbb{E}\left[\|G^t(x^t)\|_1 \mid x^t, \mathcal{E}^t\right].
\end{aligned}$$

□

Lemma 2.4.5. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a cost function and suppose G^t are unbiased estimators of ∇f. Let $\mathcal{E}^t$ denote the event that $\|G^t(x^t)\|_\infty \leqslant \eta$. Then*

$$\mathbb{E}\left[\sum_{i\in\Omega^t} sgn\left(G^t(x^t)_i\right)\frac{\partial f}{\partial x_i}(x^t)\;\middle|\; x^t, \mathcal{E}^t\right] = \frac{1}{\eta}\|\nabla f(x^t)\|_2^2. \tag{2.13}$$

Proof. Let Δ_i^t be the Bernoulli random variable with parameter $\frac{1}{\eta}\left|G^t(x^t)_i\right|$, as in the definition of SMGD. Recall that $|\Omega^t| = \sum_{i=1}^n \Delta_i^t$, and compute

$$\begin{aligned}
\mathbb{E}\left[\sum_{i\in\Omega^t} \operatorname{sgn}\left(G(x^t)_i\right)\;\frac{\partial f}{\partial x_i}(x^t)\;\middle|\; x^t, \mathcal{E}^t\right] &= \mathbb{E}\left[\mathbb{E}\left[\sum_{i\in\Omega^t} \operatorname{sgn}\left(G^t(x^t)_i\right)\frac{\partial f}{\partial x_i}(x^t)\;\middle|\; G^t, x^t, \mathcal{E}^t\right]\;\middle|\; x^t, \mathcal{E}^t\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[\sum_{i=1}^n \Delta_i^t \cdot \operatorname{sgn}\left(G^t(x^t)_i\right)\frac{\partial f}{\partial x_i}(x^t)\;\middle|\; G^t, x^t, \mathcal{E}^t\right]\;\middle|\; x^t, \mathcal{E}^t\right] \\
&= \mathbb{E}\left[\sum_{i=1}^n \left(\operatorname{sgn}\left(G^t(x^t)_i\right)\frac{\partial f}{\partial x_i}(x^t)\right)\mathbb{E}[\Delta_i^t \mid G^t, x^t, \mathcal{E}^t]\;\middle|\; x^t, \mathcal{E}^t\right] \\
&= \mathbb{E}\left[\frac{1}{\eta}\sum_{i=1}^n G^t(x^t)_i\frac{\partial f}{\partial x_i}(x^t)\;\middle|\; x^t, \mathcal{E}^t\right] \\
&= \mathbb{E}\left[\frac{1}{\eta}\langle G^t(x^t), \nabla f(x^t)\rangle\;\middle|\; x^t, \mathcal{E}^t\right] \\
&= \frac{1}{\eta}\langle \mathbb{E}[G^t(x^t)|x^t, \mathcal{E}^t], \nabla f(x^t)\rangle \\
&= \frac{1}{\eta}\langle \nabla f(x^t), \nabla f(x^t)\rangle = \frac{1}{\eta}\|\nabla f(x^t)\|^2.
\end{aligned} \tag{2.14}$$

To reach step (2.14), recall that $\mathbb{E}[G^t(x^t)|x^t, \mathcal{E}^t] = \nabla f(x^t)$ by the assumption (2.5). □

We are now ready to prove Theorem 2.4.1.

Proof of Theorem 2.4.1. We have $\mathbb{E}\left[\|G^t(x^t)\|_1 \mid x^t, \mathcal{E}^t\right] = \mathbb{E}\left[\|G(x^t)\|_1 \mid x^t, \mathcal{E}^t\right]$ since G^t are identically distributed versions of G. Take conditional expectations on both sides of (2.12) in Corollary 2.4.3, and then apply Lemmas 2.4.4 and 2.4.5 to obtain

$$\begin{aligned}
\mathbb{E}[f(x^{t+1}) \mid x^t, \mathcal{E}^t] &\leqslant \mathbb{E}\left[f(x^t) + \frac{L\alpha^2|\Omega^t|}{2} - \alpha\sum_{i\in\Omega^t}\operatorname{sgn}\left(G^t(x^t)_i\right)\frac{\partial f}{\partial x_i}(x^t)\;\middle|\; x^t, \mathcal{E}^t\right] \\
&\leqslant f(x^t) + \frac{L\alpha^2}{2\eta}\mathbb{E}\left[\|G^t(x^t)\|_1 \mid x^t, \mathcal{E}^t\right] - \frac{\alpha}{\eta}\|\nabla f(x^t)\|^2 \\
&= f(x^t) + \frac{L\alpha^2}{2\eta}\mathbb{E}\left[\|G(x^t)\|_1 \mid x^t, \mathcal{E}^t\right] - \frac{\alpha}{\eta}\|\nabla f(x^t)\|^2.
\end{aligned}$$

□

2.4.2 Cost function bounds for mini-batch estimators

Theorem 2.4.1 depends heavily on the expected ℓ_1 norm of the gradient estimator $\mathbb{E}[\|G^t(x^t)\|_1 \mid x^t, \mathcal{E}^t]$, where $\mathcal{E}^t$ is the event that $\|\nabla G^t(x^t)\|_1 \leqslant \eta$. This Section studies the quantity $\mathbb{E}[\|G^t(x^t)\|_1 \mid x^t, \mathcal{E}^t]$ for the special case when G^t are *minibatch gradient estimators*. For simplicity, we focus on the case when $\mathcal{E}^t$ occurs almost surely, so that $\mathbb{E}[\|G^t(x^t)\|_1 \mid x^t, \mathcal{E}^t] = \mathbb{E}[\|G^t(x^t)\|_1 \mid x^t]$. Since G^t is independent of x^t, we proceed by deriving estimates for $\mathbb{E}[\|G^t(x)\|_1]$ with fixed $x \in \mathbb{R}^n$.

Mini-batch estimates are a commonly used technique to improve neural network training [24, 9, 22]. This Section only considers cost functions of the special form $f = \frac{1}{m}\sum_{i=1}^m f^i$ where each f^i is differentiable. A mini-batch estimator of size k selects k distinct indices $\{i_j\}_{j=1}^k$ uniformly at random from $\{1, 2, \cdots, m\}$ and then defines $G = \frac{1}{k}\sum_{j=1}^k \nabla f^{i_j}$ as an unbiased estimator of ∇f. With slight abuse of notation, let G_k denote a minibatch estimator of size k.

The following Theorem provides bounds on $\mathbb{E}[\|G^t(x)\|_1]$ for mini-batch estimates. It will be convenient to introduce some notation for the proof. Let $[m]$ denote the set $\{1, \ldots, m\}$, and let A^k denote the collection of all subsets of size k of a given subset $A \subset [m]$. For example, $[m]^k$ consists of all subsets of $\{1, \ldots, m\}$ containing k elements. We also let A^c denote the complement of A in $[m]$.

Theorem 2.4.6. *Fix a cost function $f = \frac{1}{m}\sum_{i=1}^m f^i$ where each f^i is differentiable. Let $G_k = \frac{1}{k}\sum_{j=1}^k \nabla f^{i_j}$ be the mini-batch estimator of size k for ∇f. Let $\|\cdot\|$ be any norm on $\mathbb{R}^n$. Given $x \in \mathbb{R}^n$, $\mathbb{E}\left[\|G_k(x)\|\right]$ is non-increasing in k and satisfies the bound*

$$\mathbb{E}\left[\|G_k(x)\|\right] \leqslant \frac{m}{k}\|\nabla f(x)\| + \frac{m-k}{k}\mathbb{E}\left[\|G_{m-k}(x)\|\right]. \tag{2.15}$$

Proof. We first show that $\mathbb{E}\left[\|G_k(x)\|\right]$ is non-increasing in k, by showing that $\mathbb{E}\left[\|G_k(x)\|\right] \leqslant \mathbb{E}\left[\|G_{k-1}(x)\|\right]$. By the definition of mini-batch estimates one has

$$\mathbb{E}\left[\|G_k(x)\|\right] = \frac{1}{\binom{m}{k}} \sum_{A \in [m]^k} \left\| \frac{1}{k}\sum_{i \in A} \nabla f^i(x) \right\|.$$

Fix any $A \in [m]^k$ and notice

$$\sum_{i \in A} \nabla f^i(x) = \sum_{B \in A^{k-1}} \sum_{i \in B} \frac{1}{k-1} \nabla f^i(x)$$

because for each index $i \in A$, there are exactly $k-1$ subsets $B \in A^{k-1}$ containing i. Therefore,

$$\begin{aligned} \mathbb{E}\left[\|G_k(x)\|\right] &= \frac{1}{\binom{m}{k}} \sum_{A \in [m]^k} \left\| \frac{1}{k} \sum_{B \in A^{k-1}} \sum_{i \in B} \frac{1}{k-1} \nabla f^i(x) \right\| \\ &\leqslant \frac{1}{\binom{m}{k}} \frac{1}{k} \sum_{A \in [m]^k} \sum_{B \in A^{k-1}} \left\| \sum_{i \in B} \frac{1}{k-1} \nabla f^i(x) \right\|. \end{aligned} \tag{2.16}$$

One has that

$$\sum_{A \in [m]^k} \sum_{B \in A^{k-1}} \left\| \sum_{i \in B} \frac{1}{k-1} \nabla f^i(x) \right\| = (m-k+1) \sum_{B \in [m]^{k-1}} \left\| \sum_{i \in B} \frac{1}{k-1} \nabla f^i(x) \right\|. \tag{2.17}$$

To see this, note that the double sum $\sum_{A \in [m]^k} \sum_{B \in A^{k-1}}$ sums over each set B of size $k-1$ once for each size k set A which contains B. There are $m-(k-1)$ elements of $[m]$ that can be added to B to get a size k set. Therefore, each B shows up $m-k+1$ times in this double summation, and (2.17) follows.

Combining (2.16) and (2.17), gives

$$\begin{aligned} \mathbb{E}\left[\|G_k(x)\|\right] &\leqslant \frac{m-k+1}{k\binom{m}{k}} \sum_{B \in [m]^{k-1}} \left\| \sum_{i \in B} \frac{1}{k-1} \nabla f^i(x) \right\| \\ &= \frac{(m-k+1)\binom{m}{k-1}}{k\binom{m}{k}} \mathbb{E}\left[\|G_{k-1}(x)\|\right]. \end{aligned}$$

A computation shows that $\frac{(m-k+1)\binom{m}{k-1}}{k\binom{m}{k}} = 1$ and the desired bound $\mathbb{E}\left[\|G_k(x)\|\right] \leqslant \mathbb{E}\left[\|G_{k-1}(x)\|\right]$ follows.

It remains to prove the bound (2.15). Using the triangle inequality and $f = \frac{1}{m}\sum_{i=1}^m f^i$ gives

$$\begin{aligned} \mathbb{E}\left[\|G_k\|\right] &= \frac{1}{\binom{m}{k}} \sum_{A \in [m]^k} \left\| \frac{1}{k} \left(\sum_{i \in A} \nabla f^i(x) \right) \right\| \\ &\leqslant \frac{1}{\binom{m}{k}} \sum_{A \in [m]^k} \left\| \frac{1}{k} \left(m \nabla f(x) - \sum_{i \in A} \nabla f^i(x) \right) \right\| + \frac{m}{k} \|\nabla f(x)\| \\ &= \frac{1}{\binom{m}{k}} \sum_{A \in [m]^k} \left\| \frac{1}{k} \sum_{i \in A^c} \nabla f^i(x) \right\| + \frac{m}{k} \|\nabla f(x)\| \end{aligned}$$

$$= \frac{m-k}{k} \cdot \frac{1}{\binom{m}{k}} \sum_{A \in [m]^k} \left\| \frac{1}{m-k} \sum_{i \in A^c} \nabla f^i(x) \right\| + \frac{m}{k} \|\nabla f(x)\|. \tag{2.18}$$

Since $\binom{m}{k} = \binom{m}{m-k}$ one has

$$\begin{aligned} \mathbb{E}\left[\|G_{m-k}(x)\|\right] &= \frac{1}{\binom{m}{m-k}} \sum_{A \in [m]^{m-k}} \left\| \frac{1}{m-k} \sum_{i \in A} \nabla f^i(x) \right\| \\ &= \frac{1}{\binom{m}{k}} \sum_{A \in [m]^k} \left\| \frac{1}{m-k} \sum_{i \in A^c} \nabla f^i(x) \right\|. \end{aligned} \tag{2.19}$$

Combining (2.18) and (2.19) gives (2.15) and completes the proof. □

While the above Theorem may be difficult to parse, we offer two main insights related to Theorem 2.4.6. Recall that the quantity we are bounding controls the performance of SMGD so a smaller value for $\mathbb{E}\left[\|G_k\|\right]$ conceivably implies better algorithm performance. With this in mind, because the expected value is non-increasing in k, choosing a larger mini-batch never worsens the performance. Second, as k approaches m the value $\mathbb{E}\left[\|G_k\|\right]$ approaches $\|\nabla f\|$, the optimal value.

2.5 Error estimates: rates of convergence

In this Section we analyze the *rate of convergence* for SMGD when the cost function is assumed to be strongly convex. A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is strongly convex with parameter μ, or simply $\mu-$strongly convex, provided that, for every $x, y \in \mathbb{R}^n$,

$$\langle \nabla f(x) - \nabla f(y), x - y \rangle \geqslant \mu \|x - y\|_2^2. \tag{2.20}$$

It follows from the Cauchy-Schwartz inequality that a μ-strongly convex differentiable function f satisfies

$$\|\nabla f(x) - \nabla f(y)\|_2 \geqslant \mu \|x - y\|_2. \tag{2.21}$$

Strongly convex functions are well-studied in optimization and it is known that a differentiable strongly convex function attains a unique minimum, see e.g., [27].

Our next main result provides bounds on how fast the iterates of SMGD x^t approach the minimizer x^* of the cost function.

Theorem 2.5.1. *Suppose the cost function $f : \mathbb{R}^n \to \mathbb{R}$ is μ-strongly convex and has L-Lipschitz gradient ∇f. Suppose G^t are independent versions of an unbiased estimator G for ∇f, and that G is L-Lipschitz continuous almost surely. Let $\mathcal{E}^t$ denote the event that $\|G^t(x^t)\|_\infty \leqslant \eta$. The iterates x^t of SMGD satisfy*

$$\mathbb{E}\left[\|x^{t+1} - x^*\|_2^2 \,|\, x^t, \mathcal{E}^t\right] \leqslant \left(1 - \frac{2\alpha\mu}{\eta}\right)\|x^t - x^*\|_2^2 + \frac{L\alpha^2\sqrt{n}}{\eta}\|x^t - x^*\|_2 + \frac{\alpha^2}{\eta}\mathbb{E}\left[\|G^t(x^*)\|_1 \,|\, x^t, \mathcal{E}^t\right]. \tag{2.22}$$

Proof. Let $u^t = \sum_{i=1}^n -\alpha \cdot \operatorname{sgn}\left(G^t(x^t)_i\right)\Delta_i^t$ be the random vector defined in (2.11), so that the SMGD iteration may be written as $x^{t+1} = x^t + u^t$. Thus,

$$\begin{aligned}\mathbb{E}[\|x^{t+1} - x^*\|_2^2 \,|\, x^t, \mathcal{E}^t] &= \mathbb{E}[\|x^t - x^* + u^t\|_2^2 \,|\, x^t, \mathcal{E}^t] \\ &= \|x^t - x^*\|_2^2 + 2\mathbb{E}[\langle x^t - x^*, u^t\rangle \,|\, x^t, \mathcal{E}^t] + \mathbb{E}[\langle u^t, u^t\rangle \,|\, x^t, \mathcal{E}^t]. \end{aligned} \tag{2.23}$$

Note that

$$\begin{aligned}\mathbb{E}[\langle x^t - x^*, u^t\rangle \,|\, x^t, \mathcal{E}^t] &= \sum_{i=1}^n (x^t - x^*)_i \,\mathbb{E}[u_i^t \,|\, x^t, \mathcal{E}^t] \\ &= -\alpha \sum_{i=1}^n (x^t - x^*)_i \,\mathbb{E}[\operatorname{sgn}(G^t(x^t))_i)\Delta_i^t \,|\, x^t, \mathcal{E}^t]. \end{aligned} \tag{2.24}$$

Recalling the definition of Δ^t in (2.6) and using (2.5) gives

$$\begin{aligned}\mathbb{E}[\operatorname{sgn}(G^t(x^t)_i)\Delta_i^t \,|\, x^t, \mathcal{E}^t] &= \mathbb{E}[\mathbb{E}[\operatorname{sgn}(G^t(x^t)_i)\Delta_i^t \,|\, x^t, \mathcal{E}^t, G^t] \,|\, x^t, \mathcal{E}^t] \\ &= \mathbb{E}[\operatorname{sgn}(G^t(x^t)_i)\frac{|G^t(x^t)_i|}{\eta} \,|\, x^t, \mathcal{E}^t] \\ &= \frac{1}{\eta}\frac{\partial f}{\partial x_i}(x^t). \end{aligned} \tag{2.25}$$

Combining (2.24) and (2.25) gives

$$\mathbb{E}[\langle x^t - x^*, u^t\rangle \,|\, x^t, \mathcal{E}^t] = \frac{-\alpha}{\eta}\langle x^t - x^*, \nabla f(x^t)\rangle. \tag{2.26}$$

Next note that

$$\begin{aligned}\mathbb{E}[\langle u^t, u^t\rangle \,|\, x^t, \mathcal{E}^t] &= \mathbb{E}[\mathbb{E}[\langle u^t, u^t\rangle \,|\, x^t, G^t, \mathcal{E}^t] \,|\, x^t, \mathcal{E}^t] \\ &= \alpha^2 \sum_{i=1}^{n} \mathbb{E}[\mathbb{E}[(\Delta_i^t)^2 \,|\, x^t, G^t, \mathcal{E}^t] | x^t, \mathcal{E}^t] \\ &= \alpha^2 \sum_{i=1}^{n} \mathbb{E}[\frac{|G^t(x^t)_i|}{\eta} \,|\, x^t, \mathcal{E}^t] \\ &= \frac{\alpha^2}{\eta}\mathbb{E}[\|G^t(x^t)\|_1 \,|\, x^t, \mathcal{E}^t]. \end{aligned} \tag{2.27}$$

Combining (2.23), (2.26), (2.28), and using that $\nabla f(x^*) = 0$ gives

$$\begin{aligned}\mathbb{E}[\|x^{t+1} - x^*\|_2^2 \,|\, x^t, \mathcal{E}^t] &= \|x^t - x^*\|_2^2 - \frac{2\alpha}{\eta}\langle x^t - x^*, \nabla f(x^t)\rangle + \frac{\alpha^2}{\eta}\mathbb{E}[\|G^t(x^t)\|_1 \,|\, x^t, \mathcal{E}^t] \\ &= \|x^t - x^*\|_2^2 - \frac{2\alpha}{\eta}\langle x^t - x^*, \nabla f(x^t) - \nabla f(x^*)\rangle + \frac{\alpha^2}{\eta}\mathbb{E}[\|G^t(x^t)\|_1 \,|\, x^t, \mathcal{E}^t] \\ &\leqslant \|x^t - x^*\|_2^2 - \frac{2\alpha}{\eta}\langle x^t - x^*, \nabla f(x^t) - \nabla f(x^*)\rangle \\ &+ \frac{\alpha^2}{\eta}\mathbb{E}[\|G^t(x^t) - G^t(x^*)\|_1 \,|\, x^t, \mathcal{E}^t] + \frac{\alpha^2}{\eta}\mathbb{E}[\|G^t(x^*)\|_1 \,|\, x^t, \mathcal{E}^t]. \end{aligned} \tag{2.28}$$

Applying strong convexity and Hölder's inequality in (2.28) gives

$$\begin{aligned}\mathbb{E}[\|x^{t+1} - x^*\|_2^2 \,|\, x^t, \mathcal{E}^t] &\leqslant \|x^t - x^*\|_2^2 - \frac{2\alpha\mu}{\eta}\|x^t - x^*\|_2^2 \\ &+ \frac{\alpha^2\sqrt{n}}{\eta}\mathbb{E}[\|G^t(x^t) - G^t(x^*)\|_2 \,|\, x^t, \mathcal{E}^t] + \frac{\alpha^2}{\eta}\mathbb{E}[\|G^t(x^*)\|_1 \,|\, x^t, \mathcal{E}^t]. \end{aligned} \tag{2.29}$$

Finally, since G is L-Lipschitz, (2.29) yields

$$\mathbb{E}[\|x^{t+1} - x^*\|_2^2 \,|\, x^t] \leqslant \left(1 - \frac{2\alpha\mu}{\eta}\right)\|x^t - x^*\|_2^2 + \frac{\alpha^2\sqrt{n}L}{\eta}\|x^t - x^*\|_2 + \frac{\alpha^2}{\eta}\mathbb{E}[\|G^t(x^*)\|_1 \,|\, x^t, \mathcal{E}^t].$$

□

Theorem 2.5.1 can be viewed as an analogue for SMGD of the convergence results for SGD in [28]. Changing notation to match our own, the work in [28] shows that, under similar assumptions as Theorem

2.5.1, standard SGD with learning rate of γ satisfies

$$\mathbb{E}[\|x^{t+1} - x^*\|_2^2 | x^t] \leqslant (1 - 2\gamma\mu)\|x^t - x^*\|_2^2 + 2\gamma^2 L\|x^t - x^*\|_2^2 + 2\gamma^2\mathbb{E}[\|G(x^*)\|_2^2]. \tag{2.30}$$

This illustrates that the learning rate γ for SGD plays an analogous role as the lattice resolution α for SMGD. It is also worth noting some differences between (2.22) and (2.30). The middle term in (2.22) is a squared norm $\|x^t - x^*\|_2^2$ whereas the middle term in (2.30) is not squared; unlike SGD this means that SMGD errors will generally not decrease exponentially fast until saturation. Moreover, the third terms in (2.22) and (2.30) reflect the different dependences of SMDG and SGD on the choice of unbiased estimator for ∇f.

2.6 Error bounds: the non-stochastic setting

In this Section we consider the special case of SMGD where the unbiased gradient estimator G^t is the non-stochastic estimate $G = \nabla f$. We shall refer to this special case of SMGD as *Markov gradient descent (MGD)*.

The following result is a corollary of Theorem 2.8.

Corollary 2.6.1. *Suppose the cost function $f : \mathbb{R}^n \to \mathbb{R}$ has L-Lipschitz gradient ∇f. Let $\mathcal{E}^t$ denote the event $\|\nabla f(x^t)\|_\infty \leqslant \eta$. The iterate x^{t+1} of MGD satisfies*

$$\mathbb{E}\left[f(x^{t+1}) \,|\, x^t, \mathcal{E}^t\right] \leqslant f(x^t) + \frac{L\alpha^2}{2\eta}\|\nabla f(x^t)\|_1 - \frac{\alpha}{\eta}\|\nabla f(x^t)\|_2^2.$$

The following consequence of Corollary 2.6.1 shows that iterates $f(x^{t+1})$ of the cost function decrease in expectation when the gradient $\nabla f(x^t)$ has sufficiently large norm.

Corollary 2.6.2. *Suppose the cost function $f : \mathbb{R}^n \to \mathbb{R}$ has L-Lipschitz gradient ∇f. Let $\mathcal{E}^t$ denote the event $\|\nabla f(x^t)\|_\infty \leqslant \eta$. If $x \in \mathbb{R}^n$ satisfies $\frac{L\alpha}{2}\|\nabla f(x)\|_1 < \|\nabla f(x)\|_2^2$, then the iterate x^{t+1} of MGD satisfies*

$$\mathbb{E}\left[f(x^{t+1}) \,|\, x^t = x, \mathcal{E}^t\right] < f(x). \tag{2.31}$$

In particular, if $x \in \mathbb{R}^n$ satisfies $\|\nabla f(x)\|_2 > \frac{L\alpha\sqrt{n}}{2}$, then (2.31) *holds.*

Proof. It suffices to note that if $\|\nabla f(x)\|_2 > \frac{L\alpha\sqrt{n}}{2}$, then the Cauchy-Schwarz inequality implies

$$\frac{L\alpha}{2}\|\nabla f(x)\|_1 \leqslant \frac{L\alpha\sqrt{n}}{2}\|\nabla f(x)\|_2 < \|\nabla f(x)\|_2^2.$$

□

The next result gives conditions for expected decrease of the cost function under the assumption of strong convexity.

Corollary 2.6.3. *Suppose the cost function $f : \mathbb{R}^n \to \mathbb{R}$ is $\mu-$strongly convex and has $L-$Lipschitz gradient ∇f. Let x^* denote the unique minimizer of f. Given a tolerance level $\varepsilon > 0$, suppose that*

$$\alpha < \left(\frac{4\varepsilon\mu}{L^2 n}\right)^{1/2}. \tag{2.32}$$

Let $\mathcal{E}^t$ denote the event $\|\nabla f(x^t)\|_\infty \leqslant \eta$. If $x \in \mathbb{R}^n$ satisfies $f(x) - f(x^*) > \varepsilon$, then the iterate x^{t+1} of MGD satisfies

$$\mathbb{E}\left[f(x^{t+1}) \,|\, x^t = x, \mathcal{E}^t\right] < f(x).$$

Proof. Assume that $x \in \mathbb{R}^n$ satisfies $f(x) - f(x^*) > \varepsilon$. It suffices to prove that $\|\nabla f(x)\|_2 > \frac{L\alpha\sqrt{n}}{2}$, since the result then follows from Corollary 2.6.2. We consider two cases depending on whether $\|x - x^*\|$ is large or small.

Case 1. Suppose that $\|x - x^*\|_2 > \frac{L\alpha\sqrt{n}}{2\mu}$. Applying (2.21) and $\nabla f(x^*) = 0$ yields

$$\|\nabla f(x)\|_2 = \|\nabla f(x) - \nabla f(x^*)\|_2 \geqslant \mu\|x - x^*\|_2 > \frac{L\alpha\sqrt{n}}{2}.$$

Case 2. Suppose that $\|x - x^*\|_2 \leqslant \frac{L\alpha\sqrt{n}}{2\mu}$. Define the function $g(r) = f(x^* + r\frac{x-x^*}{\|x-x^*\|})$, the restriction of f to the line containing both x^t and x^*. Observe that g is a strictly convex function of the single variable r with unique minimizer at $r = 0$. Moreover, observe that $g'(r)$ is the directional derivative of f at the point $x^* + ru$ in the direction $u = \frac{x-x^*}{\|x-x^*\|_2}$. Because g is convex, we know that this directional derivative is larger than the slope of the secant line of g between 0 and r. Thus, using the Cauchy-Schwarz inequality, we have

$$\|\nabla f(x)\|_2 \geqslant \langle \nabla f(x), u\rangle = D_u f(x) > \frac{f(x) - f(x^*)}{\|x - x^*\|} > \frac{2\mu\varepsilon}{L\alpha\sqrt{n}}. \tag{2.33}$$

Rewriting (2.32) in terms of ε gives

$$\varepsilon > \frac{L^2\alpha^2 n}{4\mu}. \tag{2.34}$$

Combining (2.33) and (2.34) gives $\|\nabla f(x)\|_2 > \frac{L\alpha\sqrt{n}}{2}$.

□

The remainder of this Section address rates of convergence for MGD. The next result is a corollary of Theorem 2.5.1, and holds since $\nabla f(x^*) = 0$.

Corollary 2.6.4. *Suppose the cost function $f : \mathbb{R}^n \to \mathbb{R}$ is $\mu-$strongly convex and has $L-$Lipschitz gradient ∇f. Let $\mathcal{E}^t$ denote the event $\|\nabla f(x^t)\|_\infty \leqslant \eta$. Let x^* denote the unique minimizer of f. The iterate x^{t+1} of MGD satisfies*

$$\mathbb{E}\left[\|x^{t+1} - x^*\|_2^2 \,|\, x^t, \mathcal{E}^t\right] \leqslant \left(1 - \frac{2\alpha\mu}{\eta}\right)\|x^t - x^*\|_2^2 + \frac{L\alpha^2\sqrt{n}}{\eta}\|x^t - x^*\|_2. \tag{2.35}$$

Corollary 2.6.4 can be used to provide conditions under which the error $\|x^{t+1} - x^*\|$ for MGD decreases in expectation.

Corollary 2.6.5. *Suppose the cost function $f : \mathbb{R}^n \to \mathbb{R}$ is $\mu-$strongly convex and has $L-$Lipschitz gradient ∇f. Let $\mathcal{E}^t$ denote the event $\|\nabla f(x^t)\|_\infty \leqslant \eta$. Let x^* denote the unique minimizer of f. If $x \in \mathbb{R}^n$ satisfies $\|x - x^*\|_2 > \frac{L\alpha\sqrt{n}}{2\mu}$, then the iterate x^{t+1} of MGD satisfies*

$$\mathbb{E}\left[\|x^{t+1} - x^*\|_2^2 \,|\, x^t = x, \mathcal{E}^t\right] < \|x - x^*\|_2^2.$$

Proof. By Corollary 2.6.4, we have

$$\mathbb{E}\left[\|x^{t+1} - x^*\|_2^2 \,|\, x^t = x, \mathcal{E}^t\right] \leqslant \|x - x^*\|_2^2\left(1 - \frac{2\alpha\mu}{\eta} + \frac{L\alpha^2\sqrt{n}}{\eta\|x - x^*\|_2}\right).$$

In particular, $\mathbb{E}\left[\|x^{t+1} - x^*\|_2^2 \mid x^t = x, \mathcal{E}^t\right] < \|x - x^*\|_2^2$ holds whenever

$$\frac{2\alpha\mu}{\eta} > \frac{L\alpha^2\sqrt{n}}{\eta\|x - x^*\|_2}. \tag{2.36}$$

Since (2.36) is equivalent to $\|x - x^*\|_2 > \frac{L\alpha\sqrt{n}}{2\mu}$, this completes the proof. □

The following example shows that the conditions $\|\nabla f(x^t)\| > \frac{L\alpha\sqrt{n}}{2}$ and $\|x^t - x^*\|_2 > \frac{L\alpha\sqrt{n}}{2}$ in Corollaries 2.6.2 and 2.6.5 cannot be weakened.

Example 1. *Fix a lattice $\alpha\mathbb{Z}^n$. Define the function $f : \mathbb{R}^n \to \mathbb{R}$ by $f(x_1, \ldots, x_n) = \sum_{i=1}^n (x_i - \frac{\alpha}{2})^2$. The unique minimizer of f is $x^* = (\frac{\alpha}{2}, \ldots, \frac{\alpha}{2})$. Since $\nabla f(x_1, \ldots x_n) = (2x_1 - \alpha, \cdots, 2x_n - \alpha)$, it follows that ∇f is $2-$Lipschitz and f is $2-$strongly convex.*

Define $\mathcal{S} = \{(x_1, \cdots, x_n) \in \alpha\mathbb{Z}^n : \text{ each } x_i \in \{0, \alpha\}\}$. Note if $x \in \mathcal{S}$ then $f(x) = \frac{n\alpha^2}{4}$. Further note that if $x^t = 0$, then the next iterate of Markov gradient descent satisfies $x^{t+1} \in \mathcal{S}$ because $\frac{\partial f}{\partial x_i}(0) < 0$ for all i. Therefore, $\mathbb{E}\left[\|x^{t+1} - x^*\|_2^2 \,|\, x^t = 0\right] = \|x^t - x^*\|_2^2$ and $\mathbb{E}\left[f(x^{t+1}) - f(x^t) \,|\, x^t = 0\right] = 0$. This shows that the conclusions of Corollaries 2.6.2 and 2.6.5 do not hold. However, observe that

$$\|\nabla f(0)\|_2 = \left(\sum_{i=1}^n (-\alpha)^2\right)^{1/2} = \alpha\sqrt{n} = \frac{L\alpha\sqrt{n}}{2}$$

and

$$\|x^t - x^*\|_2 = \left(\sum_{i=1}^n \frac{\alpha}{2}^2\right)^{1/2} = \frac{\alpha\sqrt{n}}{2} = \frac{L\alpha\sqrt{n}}{2\mu}.$$

In particular, the conditions $\|\nabla f(x^t)\| > \frac{L\alpha\sqrt{n}}{2}$ and $\|x^t - x^*\|_2 > \frac{L\alpha\sqrt{n}}{2}$ in Corollaries 2.6.2 and 2.6.5 are tight.

2.7 Experiments and Numerical Validation

In this Section we validate the use of SMGD for training quantized neural networks with three experiments. First, we demonstrate the accuracy of SMGD-trained networks on the standard MNIST and CIFAR-10 datasets. Second, we compare SMGD to SGD while holding the amount of memory constant during training. Finally, we show the effect that the quality of gradient estimators has on SMGD training by altering minibatch sizes.

2.7.1 Performance of SMGD on MNIST and CIFAR-10

Our first experiment uses SMGD to train quantized networks with identical architectures as in [9]. These experiments validate that SMGD can perform well on some data sets but may not be optimal in other settings.

Method	MNIST	CIFAR-10
Binary Connect	0.96	11.4
SMGD (4-bit)	1.59	27
SMGD (1-bit)	6.97	-

Table 2.1: Test errors of SMGD versus BinaryConnect on MNIST and CIFAR-10.

We compare 1-bit and 4-bit versions of SMGD for neural network quantization to the performance of the 1-bit BinaryConnect method [9] on the MNIST and CIFAR10 datasets. For the MNIST dataset, we use a feed-forward neural network with 3 hidden layers of 4096 neurons. We use no preprocessing, the ReLU non-linearity, and the softmax output layer. We note that BinaryConnect uses an L2-SVM output layer, batch normalization, and dropout to improve performance while we omit these because the effects of these techniques are not included in our theoretical results. Including these techniques would likely further improve the competitiveness of SMGD. The first column in Table 2.1 shows the test errors for the MNIST dataset. It is important to emphasize that since SMGD is memory-constrained during training, it is expected that BinaryConnect will outperform SMGD, but the performance of SMGD becomes competitive when more bits are allowed.

On the CIFAR-10 dataset, we use a convolutional architecture which is identical to that in [22]. We observe that while SMGD can perform well on MNIST, it struggles on CIFRAR-10. This could be improved by incorporating advanced techniques such as dropout and SVM output during training, but we suspect that SMGD generally performs worse than other quantization algorithms in this setting. In particular, we failed to find a good parameter configuration of α, η to successfully train a 1-bit SMGD network on CIFAR-10. However, we again emphasize that SMGD weights are quantized during training so a true apples-to-apples comparison does not highlight the usefulness of SMGD. The results of our first set of experiments are summarized in Table 2.1.

2.7.2 Performance of SMGD: memory utilization during training

Our second experiment highlights the motivation for using SMGD: the network is compressed during training as well as at run time. This is in contrast to the existing techniques that we are aware of which require full precision during training. Moreover, many other neural network quantization techniques, e.g., [9], require more memory during training than a full precision network trained with SGD. To study this

issue, we compare a quantized network trained with SMGD and a full precision network trained with SGD where the memory during training is held approximately constant.

Training a network requires the storage of the weights and intermediate neural outputs as well as computation and storage of partial derivatives. The weights and partial derivatives take up an overwhelming amount of this memory, so let us compute how much savings SMGD provides in this area. SMGD requires q bits per weight and 2 bits to store each partial derivative after quantization. Computing the partial derivatives takes an additional 32 bits per weight when we use mini-batches as we must aggregate the full-precision gradient over many input signals before quantization. However, in the online setting where we process only one image at a time, we can compute the partial derivatives one-by-one. So, in the setting without mini batches we require only $2 + q$ bits-per-weight to train our network. When we use mini batches this number is $32 + q$.

Full-precision networks, on the other hand, require full-precision for weights and partial derivatives leading to 64 bits-per-weight. We recall that other quantization methods typically require more memory because they store both auxilliary and quantized weights. Therefore, other methods generally require at least $\frac{64}{2+q}$ times more memory during online training than an SMGD network. Therefore, for a fixed amount of memory, one can use a network that is approximately $\frac{64}{2+q}$ times larger than the full-precision networks which allows for better accuracy in a memory-constrained environment.

The details of our second experiment are as follows. First, we trained a full-precision neural network with a batch size of 1 on the MNIST data set to determine a baseline performance. Then, we compute the size of the SMGD-trained network that requires the same amount of memory and train that network for the same number of epochs as the full-precision network. The results of these experiments for $q = 4, 5, 6$ bit quantization are shown in Figure 2.1. While not included in the figure, the result for $q = 3$ bits is still favorable, but the results degrade for $q = 2$ and $q = 1$ bit networks on these small architectures.

The motivation for SMGD is consistent with the fact that training neural networks is not a one size fits all problem. The choice of training method should be dependent on the setting in which the learning occurs. We offer that SMGD may be best implemented in the 'memory-constrained during training' environment while other quantization methods are better in other constrained settings. Table 2.2 itemizes some recommendations regarding best training practices under various constraints on the resulting network and training process.

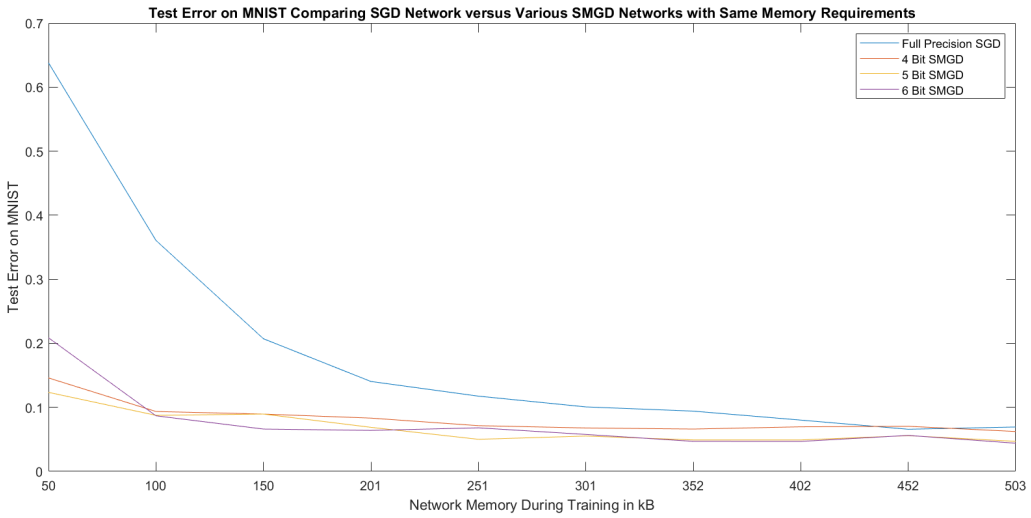


Figure 2.1: Comparison of test accuracy of various training methods each using approximately the same memory to store the weights.

Constraints	**Methods**
None	SGD, AdaGrad [13], Adam [23]
Unconstrained during training; memory constrained at run-time	BinaryConnect [9], QNN [22]
Time constrained during both run and test-time	XNOR [31], QNN [22]
Memory constrained during training	SMGD

Table 2.2: Network training methods that are suitable under different constraints.

2.7.3 Effect of minibatch size on SMGD

Our final experiment highlights the effect of increased minibatch size and illustrates the improvements suggested by Theorem 2.4.6 together with Theorem 2.4.1. We trained a network using SMGD and with increasing mini-batch sizes. The experiment illustrates that as mini-batch size increases SMGD achieves better training error until it saturates. Moreover, we see that while increasing the mini-batch size improves the performance of SMGD, there are diminishing returns as the batch size grows. The results of this experiment are contained in Figure 2.2.

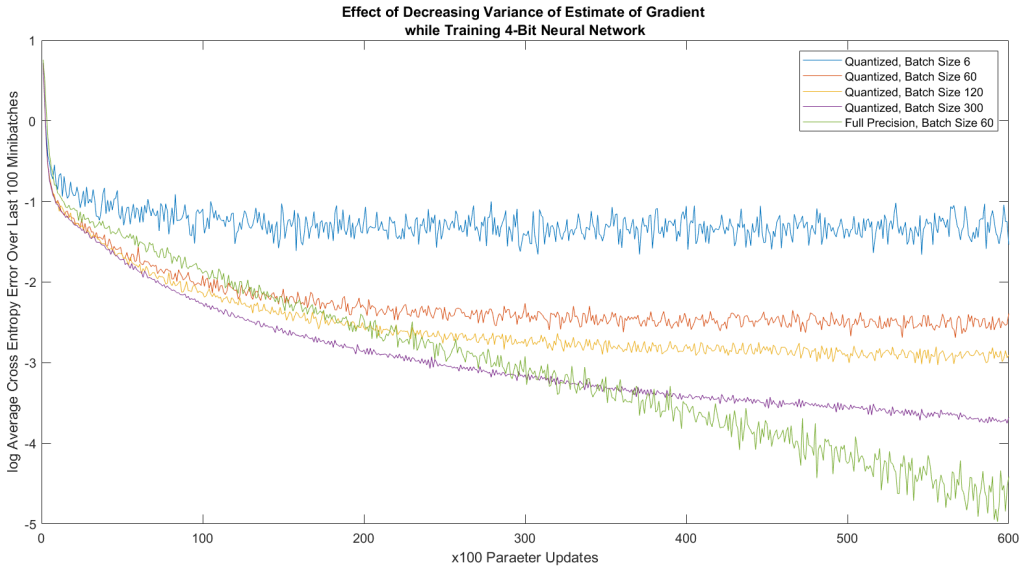


Figure 2.2: Training errors for various batch sizes trained with SMGD on identical network architectures

CHAPTER 3

Compression of Sampled Signals

3.1 Quantization in Frame Theory

In this Chapter we turn our focus to signals that are measured and reconstructed using a redundant linear representation system. In particular, we use the language of and prove results in finite frame theory in order to develop highly accurate and efficient quantization schemes in this setting. Frames for Hilbert spaces are collections $\{f^i\}_{i\in\Omega}$ of vectors that generalize bases. Informally, frames are redundant representation systems that trade the linear independence of bases for more flexibility in our choice of representation.

For a fixed frame $\{f^i\}_{i\in\Omega}$, knowledge of the sequence of inner products, $\left(\langle v, f^i\rangle\right)_{i\in\Omega}$, is enough to reconstruct the vector v perfectly. Thus, one may store v by storing this sequence of inner products. Because our focus is digital signals, we consider only the case where $|\Omega|$ is finite. Here, a finite frame for $\mathcal{H} = \mathbb{R}^d$ is nothing more than a (generally over-complete) spanning set.

This redundancy is what gives frames many of their desirable properties and allows them to be applied throughout signal processing and harmonic analysis in advantageous ways. For instance, frame representations are superior to basis expansions when we may expect erasures [18], when we must quantize the coefficients [2] into a fixed alphabet, or if any other distortions of the stored coefficients are likely.

One is naturally led to ask: given a frame $\{f^i\}_{i=1}^N$, how can we realize the reconstruction map $\left(\langle v, f^i\rangle\right)_{i=1}^N \mapsto v$. It is known that a dual frame - a set $\{g^i\}_{i=1}^N$ so that $v = \sum_{i=1}^N \langle v, f^i\rangle g^i$ - always exists. In this setting, the frame $\{f^i\}_{i=1}^N$ is called the *analysis frame* and $\{g^i\}_{i=1}^N$ is called the *synthesis frame*. If the frame $\{f^i\}_{i=1}^N$ is redundant, there are infinitely many choices of sets $\{g^i\}_{i=1}^N$ that we can use for reconstruction.

In this Chapter, we study the existence of a class of highly structured frames - called dynamical frames - and show their utility in the quantization problem. We prove two main results in this Chapter. First, we show that every finite, redundant frame for $\mathbb{R}^d$ has infinitely many dual frames that are dynamical. Second, we show that when these frames are used in an error-diffusion quantization scheme, we can expect reconstruction errors on the order of ρ^N where N is the frame size and $\rho < 1$.

This Chapter is organized as follows. First, we develop the necessary background information for frame theory in Section 3.2. Then, we spend Sections 3.3 and 3.4 characterizing dynamical duals and showing every finite frame has infinitely many dynamical dual frames. Finally in Section 3.5 we develop our quantization scheme, prove error bounds, and show experiments verifying our result.

3.2 Background: frame theory

A frame for a Hilbert space $\mathcal{H}$ is any collection of vectors $\{f^i\}_{i\in\Omega}$ for which there exist constants $A, B > 0$ so that

$$A\|v\|_2^2 \leqslant \sum_{i\in\Omega} |\langle v, f^i\rangle|^2 \leqslant B\|v\|_2^2$$

for every vector $v \in \mathcal{H}$. In the case where $\mathcal{H}$ is finite dimensional and Ω is finite, a frame is nothing more than a spanning set of vectors. It is common to identify a frame in this setting with the full rank matrix F whose columns are $f^1, \ldots, f^N$.

In the matrix terminology, a dual frame is simply a frame $\{g^i\}_{i=1}^N$ whose matrix G satisfies $GF^* = I$ where F^* denotes the transpose of F. We often refer to the *canonical dual frame* to F which is the Moore-Penrose inverse of F given by the formula $(FF^*)^{-1}F$. When the frame F is redundant - when it is not just a basis - there are infinitely many choices of frames G that are dual to F. It is precisely this fact that allows us to study advantageous dual synthesis frames for use in quantization. For reference, given two dual frames F and G, when we write the relation $v = \sum_{i=1}^N \langle v, f^i\rangle g^i$, we say F is the *analysis frame* while G is the *synthesis frame*.

3.2.1 Frame Quantization

In a very general sense, a quantizer is a function $Q : \mathbb{R}^N \to \mathcal{A}^N$ for some quantization alphabet $\mathcal{A} \subset \mathbb{R}$. Then, for analysis and synthesis frames F and G (resp.), the process of measuring a signal, quantizing the measurements for storage, and reconstruction is realized by the composition GQF^*. The goal in quantization is to pick the quantizer so that GQF^* is as close to the identity as possible.

In general the method of measurement - the matrix F - is fixed. However, one is generally free to choose both the method of reconstruction and the method of quantization. Ideally, the function Q and the matrix G will be chosen in conjunction with one another in a way that propagates as little error as possible.

In addition to the quantizer Q and the synthesis frame G needing to behave well together, it is also

important that the output of Q may be realizable relatively quickly. The simplest possible Q operates by rounding each coefficient to the nearest entry in the quantization alphabet $\mathcal{A}$. This is called *memoryless scalar quantization* or *pulse code modulation* and is very well studied. Here, Q is very quick to compute but the error induced can be quite large.

At the other extreme, given a synthesis frame G one could define $Q(v)$ to be the minimizer of $\|v - Gx\|_2$ over all vectors $x \in \mathcal{A}^N$. This Q requires solving a discrete optimization problem at each step and so is quite impractical. Though Q is slow to compute here, the induced error is very small.

A good goal would be to find a compromise between these two extremes: a quantization scheme Q that is both easy to compute and that has good error properties. One of the most commonly used ideas is error-diffusion and $\Sigma\Delta$ quantization. In general, these techniques operate by rounding a coefficient with a scalar quantizer then propagating some portion of the induced error onto the not yet quantized coefficients.

Error diffusion has been known for some time to achieve small quantization error for specific types of frames. However, recently it was discovered how to choose a synthesis frame G (given a fixed analysis frame F) that interacts in an extremely efficient way with an error-diffusion quantizer [2, 19]. In addition to sporting high-quality quantization in a variety of settings, these so-called Sobolev dual frames have helped motivative the search for synthesis frames with structure. This Chapter introduces another entirely different class of structured dual frames - the dynamical dual frames - that enjoy success in reconstructing signals that have been quantized using an error-diffusion scheme.

3.2.2 Dynamical Sampling

Dynamical Sampling is the process of recovering a time-varying signal via samples over both space and time. In this field, frames are of the form $\{T^i f^0\}_{i\in\Omega}$ where $T : \mathcal{H} \to \mathcal{H}$ is a linear operator. We will refer to frames that can be written in this way as *dynamical frames* or *dynamical dual frames*, depending on context.

Traditionally, research in dynamical sampling has concerned itself with questions about the properties T and Ω must have for $\{T^i f^0\}_{i\in\Omega}$ to be a frame for $\mathcal{H}$, [1]. Complete answers to these questions are known for finite dimensional $\mathcal{H}$ and partial answers exist in general. Recently, though, the authors in [6, 7, 8] have been interested in the reverse question: Given a frame $\{f^i\}_{i=1}^N$ for an infinite dimensional and separable $\mathcal{H}$ can we determine whether or not there exists a T, f^0 so that $f^i = T^i f^0$ for every $i \in \Omega$. That is, can we tell if a frame is a dynamical frame without knowing a priori what linear operator generated the frame vectors.

We study the finite dimensional analogue of this question in this Chapter. What properties must a finite frame $F = \{f^i\}_{i=1}^N$ for $\mathbb{R}^d$ have in order for us to guarantee (and construct!) the existence of a T and f^0 so that $T^i f^0 = f^i$. Moreover, because we are interested in quantization, we care to determine, given a frame F, whether we can find a dual frame to F that is a dynamical frame.

3.3 Characterization of Dynamical Frames

Our eventual characterization of dynamical frames relies on a property that the kernel of their frame matrix must have. In the coming analysis, we use the rank-nullity theorem and consequences time and time again. Recall that the dimension of a matrix's kernel plus the dimension of the matrix's image must equal the dimension of the domain. Throughout this Section, we will freely refer to the kernel of a frame to mean the kernel of the synthesis frame matrix. The first results we need are some technical facts about dynamical frames.

Lemma 3.3.1. *Let $T : \mathbb{R}^d \to \mathbb{R}^d$ be linear and $\{T^i f^0\}_{i=1}^N$ be a frame for $\mathbb{R}^d$. Then every d consecutive frame vectors is a basis for $\mathbb{R}^d$. Moreover, none of the frame vectors are 0.*

Proof. First notice that every frame vector is in the range of T so that the range of T spans $\mathbb{R}^d$ and T is invertible. Thus, it suffices to show that $Tf^0, \ldots, T^d f^0$ is a basis for $\mathbb{R}^d$ because every set of d consecutive frame vectors is translation of this set by T^n which is invertible. Because T is invertible, each frame vector is non-zero. Now, let $k \geqslant 1$ be the maximal index so that $Tf^0, \ldots, T^k f^0$ is linearly independent. Let $S = \text{span}\{Tf^0, \ldots, T^k f^0\}$. We will show first that S is invariant under T, that $T(S) = S$. Notice that by choice of the index k, we can find coefficients $\alpha_1, \ldots, \alpha_{k+1}$ so that

$$\sum_{i=1}^{k+1} \alpha_i T^i f^0 = 0.$$

Because $T^1 f^0, \ldots T^k f^0$ are linearly independent, then the coefficient α_{k+1} is non-zero so that $T^{k+1} f^0$ is in the set S, the span of $Tf^0, \ldots T^k f^0$. Therefore, for any $v \in S$, we can find $(\beta_i)_{i=1}^N$ so that

$$Tv = T\left(\sum_{i=i}^{k} \beta_i T^i f^0\right) = \sum_{i=i}^{k} \beta_i T^{i+1} f^0$$

which, because $Tf^0, \ldots T^{k+1} f^0$ are all in the subspace S, shows that Tv is in S. Thus, $T(S) \subset S$.

However, because T is invertible it preserves dimension of subspaces, so $T(S) \subset S$ implies that $T(S) = S$. It follows from $T(S) = S$ that $T^n(S) = S$ so that every frame vector $T^i f^0$ is in S. Therefore, $S = \mathbb{R}^d$ and the set $Tf^0, \dots T^k f^0$ must be a basis with $k = d$. □

For a vector x, we let x_i refer to coordinate i. In our characterization of dynamical frames, we often need to make use of vectors in a very particular form. Therefore, we give these vectors a name here.

.

Definition 3.3.2. *A vector $x \in \mathbb{R}^N$ is said to be d-suitable in $\mathbb{R}^N$ provided $x_1 \neq 0$, $x_{d+1} \neq 0$ and $x_i = 0$ for every $i = d+2, \dots, N$.*

$d-$Suitable vectors are ones whose support is $\{1, d+1\} \cup \Omega$ where $\Omega \subset \{2, \dots d\}$. We point out that Lemma 3.3.1 proves that there are $d-$suitable vectors in the kernel of F. We will show that the kernel is actually the linear span of $d-$suitable vectors and their right-shifts. Moreover, if a frame's kernel has this property, we will show later this implies the frame is dynamical. Throughout the remainder of this Chapter, we let L and R be the left and right shift operators given by

$$R(v) = (0, v_1, \dots, v_{N-1})^*$$
$$L(v) = (v_2, \dots, v_N, 0)^*$$

Now, R^j is the $j-$fold composition of R and R^0 is the identity map. First, some technical facts working towards the goal of characterizing dynamical frames by their kernels.

Lemma 3.3.3. *Let $F = \{f^i\}_{i=1}^N$ be a frame for $\mathbb{R}^d$ with $N > d$. Let $b \in \mathbb{R}^N$ be $d-$suitable and Ker $F = span\{R^{i-1}b : 1 \leqslant i \leqslant N-d\}$.*

1. *If $v \in$ Ker F has $v_N = 0$, then the right shift Rv of v is also in Ker F. If $v_1 = 0$, then the left-shift Lv of v is in Ker F.*
2. *Every d consecutive frame vectors form a basis.*
3. *If $1 \leqslant i < j < N$ and $f^i = f^j$ then $f^{i+1} = f^{j+1}$.*

Proof. **Statement 1)** Because b is $d-$suitable, the only vector in $\{R^{i-1}b : 1 \leqslant i \leqslant N-d\}$ whose support contains N is $R^{N-d-1}b$ and the only vector whose support contains 1 is $R^0 b = b$. Suppose that $v \in$ Ker

F and that $v_N = 0$. Note that $\{R^{i-1}b : 1 \leqslant i \leqslant N-d\}$ is a basis for KerF because it spans the kernel and the kernel, by rank-nullity, must be $N-d$ dimensional. Then, expanding b in terms of the kernel basis $\{R^{i-1}b : 1 \leqslant i \leqslant N-d\}$, the basis coefficient on $R^{N-d-1}b$ must be 0 so that we may write

$$\begin{aligned} Rv &= R\left(\sum_{i=1}^{N-d-1} \alpha_i R^{i-1} b\right) \\ &= \sum_{i=1}^{N-d-1} \alpha_i R^i b \\ &= \sum_{i=2}^{N-d} \alpha_{i-1} R^{i-1} b \in \text{ Ker } F. \end{aligned}$$

For the closure under the left shift, we suppose $v \in KerF$ and $v_1 = 0$. Then, identically, expanding v in terms of $\{R^{i-1}b, 1 \leqslant i \leqslant N-d\}$ forces the coefficient on $R^0 b$ to be 0.

$$\begin{aligned} Lv &= L\left(\sum_{i=2}^{N-d} \alpha_i R^{i-1} b\right) \\ &= LR \sum_{i=1}^{N-d-1} \alpha_{i+1} R^{i-1} b. \end{aligned}$$

Notice that L is a left-inverse to R on the subspace of vectors with last coordinate 0. Notice further that $\sum_{i=1}^{N-d-1} \alpha_{i+1} R^{i-1} b$ is in this subspace so that Lv remains in Ker F.

Statement 2) We start by showing $f^1, \ldots, f^d$ is a basis. Let $S = \text{span}\{f^1, \ldots, f^d\}$. Because b is in the kernel of F and $b_{d+1} \neq 0$, we have the formula

$$f^{d+1} = \frac{-1}{b_{d+1}} \sum_{i=1}^{d} b_i f^i$$

so that f^{d+1} is in S. Then, because $Rb \in$ Ker F by Statement 1, we can represent

$$f^{d+2} = \frac{-1}{b_{d+1}} \sum_{i=1}^{d} b_i f^{i+1}$$

which remains in S because $f^{d+1} \in S$. Repeating this process for each shift R^{i-1} of b shows that every frame vector is in S so that $S = \mathbb{R}^d$ and $f_1, \ldots f_d$ is a basis.

We now show this is true for every d consecutive frame vectors. Fix an index k and a vector v supported on a subset of $\{k, \ldots, k+d-1\}$ so that $\sum_{i=1}^{d} v_i f^{k-1+i} = 0$. By part 1 of this Theorem, $L^{k-1}v$ remains

in Ker F and its support is now contained in $\{1, \dots, d\}$. However, because $f^1, \dots f^d$ is a basis, $v = 0$. Therefore $f^k, \dots, f^{k+d-1}$ is also a basis.

Statement 3) Let $\delta_1, \dots, \delta_N$ denote the canonical basis for $\mathbb{R}^N$. Suppose that there exists $1 \leqslant i < j < N$ so that $f^i = f^j$. Then $v = \delta_i - \delta_j \in$ Ker F and observe that $v_N = 0$. Thus by part 1 of this Lemma, $Rv = \delta_{i+1} - \delta_{j+1} \in$ Ker F and so $f^{i+1} = f^{j+1}$. □

With these technical Lemmas in hand we may prove that the dynamical frames may be characterized by their kernels.

Theorem 3.3.4. *Let $F = \{f^i\}_{i=1}^N$ be a frame for $\mathbb{R}^d$ with $N > d$. There exists a T and f^0 so that $T^i f^0 = f^i$ if and only if $KerF = span\{R^{i-1}b : 1 \leqslant i \leqslant N - d\}$ for some $d-$suitable b.*

Proof. ($\Rightarrow$) Consider a frame $\{T^i f^0\}_{i=1}^N$. Because by cardinality $T^1 f^0, \dots T^{d+1} f^0$ cannot be linearly independent in $\mathbb{R}^d$, we can find a non-zero vector b in Ker F supported on a subset of $\{1, \dots d+1\}$.

Observation 1: $b_1, b_{d+1} \neq 0$. This follows from Lemma 3.3.1. If b_1 were 0, then there would exist a non-trivial linear combination of $f^2, \dots f^{d+1}$ that is zero. The same argument shows b_{d+1} cannot be zero.

Observation 2: The set $\{R^{i-1}b\}_{i=1}^{N-d}$ is linearly independent. Notice that $b_{d+1} \neq 0$ but $b_i = 0$ for $i > d+1$. Then, $R^{i-1}b$ cannot be in the span of $\{R^{j-1}b\}_{j<i}$ so long as $i \leqslant N - d$.

Now we show that each $R^{i-1}b$ is in Ker F for each $i \in \{1, \dots N-d\}$. Notice that, writing F as a matrix,

$$FR^{i-1}b = \sum_{k=1}^{d+1} b_k T^{k+i-1} f^0 = T^{i-1} \left(\sum_{k=1}^{d+1} b_k T^k f^0 \right)$$
$$= T^{i-1} (Fb) = 0.$$

Notice that F is a frame matrix so its kernel has dimension $N-d$. Then, because $\{R^{i-1}b, 1 \leqslant i \leqslant N-d\}$ is linearly independent and each is in the kernel, these vectors span the entire kernel of F.

($\Leftarrow$) Suppose F is a frame and the associated matrix has kernel given by Ker $F =$ span$\{R^{i-1}b : 1 \leqslant i \leqslant N - d\}$ where b is $d-$suitable. Define the map $T_0 : \{f_1, \dots f_{N-1}\} \to \mathbb{R}^d$ by $T_0 f_i = f_{i+1}$ and observe that this is well-defined because Lemma 3.3.3 provides that $f_i = f_j$ guarantees $f_{i+1} = f_{j+1}$. Again by Lemma 3.3.3, $f_1, \dots f_{N-1}$ must span $\mathbb{R}^d$ because $N - 1 \geqslant d$. We show that this T_0 extends to a well-defined linear

map over all of $\mathbb{R}^d$. In particular, we want

$$\sum_{i=1}^{N-1} a_i f^i = \sum_{i=1}^{N-1} b_i f^i \Rightarrow \sum_{i=1}^{N-1} a_i f^{i+1} = \sum_{i=1}^{N-1} b_i f^{i+1}$$

Towards this end, define $c_i = a_i - b_i$ for $1 \leqslant i \leqslant N-1$ and $c_N = 0$. Then, the vector $c = (c_i)_{i=1}^N$ is in Ker F. By Lemma 3.3.3 the right shift Rc remains in the kernel of F because $c_N = 0$. Thus,

$$\sum_{i=1}^{N-1} (a_i - b_i) f^{i+1} = \sum_{i=1}^{N-1} c_i f^{i+1} = FRc = 0.$$

Therefore T_0 can be extended to the linear map T defined on $\mathbb{R}^d$. Because $f^2, \dots f^N$ are in the range of T and $f^2, \dots f^N$ span $\mathbb{R}^d$ (by Lemma 3.3.3 and $N \geqslant d+1$), T is invertible. So, define $f^0 = T^{-1} f^1$. Therefore $f^i = T^i f^0$ for each i and so the frame F is dynamical. □

We have shown a characterization Theorem for dynamical frames. Our next goal is to study when a given frame has a dynamical dual frame. However, before turning to that question we will continue a bit longer down this path to explore dynamical frames a bit longer.

First, we have an $O(d^2 N)$ algorithm to compute whether a given input frame is dynamical or not. Other than this algorithm, one could also compute the candidate linear operator T by looking at the first $d+1$ columns of F and then seeing if the remaining columns are generated by T. Both have the same complexity.

The basis for the following boxed algorithm is that, if a frame is dynamical, we know what its kernel must look like. The kernel must be translates of of a $d-$suitable vector. In this algorithm, first we compute what b, the $d-$suitable vector must be by expanding $f^{d+1} = \sum_{i=1}^d b_i f^i$. Then, we verify if each of the right translates $R^{i-1} b$ of b remains in the kernel by verifying that $f^{d+k} = \sum_{i=1}^d b_i f^{k+i-1}$.

Input: Frame vectors $f^1, \dots, f^N$
Output: Yes, if $F = \{f^i\}_{i=1}^N$ is a dynamical frame, No otherwise
IF $f^1, \dots, f^d$ are linearly dependent, output NO
ELSE compute b_i so that $f^{d+1} = \sum_{i=1}^d b_i f^i$
IF each $b_i = 0$, output NO
ELSE verify that $f^{d+k} = \sum_{i=1}^d b_i f^{k+i-1}$ for every $2 \leqslant k \leqslant N-d$. If this is true, output YES. Otherwise, output NO

We can also prove easily that the canonical dual of a dynamical frame is dynamical.

Corollary 3.3.5. *If F is a dynamical frame, then the canonical dual to F is a dynamical frame.*

Proof. The canonical dual to F has the form $(FF^*)^{-1}F$. Therefore the canonical dual and F have the same kernel. Because dynamical frames are characterized by their kernels by Theorem 3.3.4, F being dynamical forces its canonical dual to be dynamical. □

If F is a dynamical frame generated by the matrix T and the vector f^0 and $S = FF^*$ is the *frame operator*, then the frame with columns generated by the operator $S^{-1}TS$ and vector $g^0 = S^{-1}f^0$ is the canonical dual to F and is dynamical. To see why, simply consider the canonical dual frame, G, which has the formula $G = S^{-1}F$. Then, the columns of G are $S^{-1}Tf^0, S^{-1}T^2f^0, \dots S^{-1}T^Nf^0$. Notice that $\left(S^{-1}TS\right)^n = S^{-1}T^nS$ so that $\left(S^{-1}TS\right)^n g^0 = S^{-1}T^nSg^0 = S^{-1}T^nf^0$. Therefore the matrix with columns $(S^{-1}TS)g^0, \dots, (S^{-1}TS)^ng^0$ is precisely the canonical dual, G, to F.

3.4 All Redundant Frames have Infinitely Many Dynamical Dual Frames

The following Theorem is a general case of the classification of frames with dynamical duals. Whenever we have a class of frames that is characterized by their kernels, we may determine the existence of a dual frame by searching for a subspace satisfying two conditions. This Theorem is not much more than an application of the relationship between a linear operator's range and the kernel of its adjoint.

Theorem 3.4.1. *Suppose there exists properties P_1, P_2 so that a frame F of size N for $\mathbb{R}^d$ has property P_1 if and only if its kernel satisfies property P_2. A frame F of size N for $\mathbb{R}^d$ has a dual frame with property P_1 if and only if there exists a subspace V of $\mathbb{R}^N$ that satisfies both of the following*

1. $\mathbb{R}^N = V +$ *Range* F^* *and* $V \cap$ *Range* $F^* = \{0\}$
2. V *has property* P_2

Proof. ($\Rightarrow$) Let G be a dual satisfying P_1. Then $V = \text{Ker } G$ satisfies P_2. Since both G and F are frames, dim Ker $G = N - d$ and dim Range$F^* = d$. Since $GF^* = I_d$, it follows that Range $F^* \cap$ Ker $G = \{0\}$. Then Ker $G +$ Range $F^* = \mathbb{R}^N$.

($\Leftarrow$). Let P be the projection onto $V^\perp$. Notice that Ker $P = (V^\perp)^\perp = \overline{V} = V$ because V is finite dimensional. Because $V \cap$ Range $F^* = \{0\}$ by assumption and Ker $F^* = \{0\}$, the map PF^* is injective from $\mathbb{R}^d \to \mathbb{R}^N$. Thus, let $A : \mathbb{R}^N \to \mathbb{R}^d$ be a left inverse to PF^*. We define $G = AP$ to be the candidate dual frame to F. Notice that $GF^* = I$. Using rank-nullity we see that the dimension of the kernel of G

is the same as the dimension of P. Now, because Ker $P \subset$ Ker G it follows that Ker $G =$ Ker $P = V$. Therefore G has property P_1. $\square$

Now, the characterization of those frames which have dynamical dual frames follows as a direct corollary of Theorem 3.4.1 and 3.3.4.

Corollary 3.4.2. *Let F be a frame of size $N > d$ for $\mathbb{R}^d$. Then F has a dynamical dual frame if and only if there exists a subspace V of $\mathbb{R}^N$ so that*

1. $\mathbb{R}^N = V +$ *Range* F^* *and* $V \cap$ *Range* $F^* = \{0\}$.
2. $V =$ *span*$\{R^{i-1}b : 1 \leqslant i \leqslant N-d\}$ *for some* $d-$*suitable vector* b.

We remark, yet again by rank-nullity, that $V \cap$Range $F^* = \{0\}$ for free because V is $N-d$ dimensional, F is rank d, and together they span $\mathbb{R}^N$. This Theorem is constructive. Knowing the subspace V, we may construct the dynamical dual frame to F by following the steps outlined in the proof of Theorem 3.4.1. Let $P_{V^\perp}$ be the projection onto $V^\perp$. Then notice that

$$\left(FP_{V^\perp}F^*\right)^{-1} FP_{V^\perp} \tag{3.1}$$

is a dual frame to F with kernel V. For us, though, we know more about the space V. Let b be a $d-$suitable vector that generates the kernel of F. Then, letting N_b be the matrix whose columns are $b, Rb, \ldots R^{N-d-1}b$, the matrix $N_b(N_b^*N_b)^{-1}N_b^*$ is the projection onto the range of N_b. In particular, $N_b(N_b^*N_b)^{-1}N_b^* = P_V$ because it is self-adjoint, idempotent, and has the desired range. Finally, because $P_V + P_{V^\perp} = I$, we can write $P_{V^\perp} = I - N_b(N_b^*N_b)^{-1}N_b^*$. Therefore, knowing the $d-$suitable vector b, the dynamical dual frame associated to this b is given by

$$\left(F(I - N_b(N_b^*N_b)^{-1}N_b^*)F^*\right)^{-1} F(I - N_b(N_b^*N_b)^{-1}N_b^*). \tag{3.2}$$

Now, the existence of dynamical duals boils down to: when does a $d-$suitable b exist so that $V =$span$\{R^{i-1}b : 1 \leqslant i \leqslant N-d\}$ together with Range F^* span $\mathbb{R}^N$? In particular, We define the $N \times N$ matrix $M_{F,b}$ whose first d columns are F^* and the next $N-d$ columns are $\{R^{i-1}b : 1 \leqslant i \leqslant N-d\}$. If we can find a $d-$suitable vector $b \in \mathbb{R}^N$ so that $M_{F,b}$ is invertible, then F has a dynamical dual frame. This follows because the invertibility of $M_{F,b}$ is equivalent to its columns spanning $\mathbb{R}^N$. This forces Range

F^* and $\{R^{i-1}b : 1 \leqslant i \leqslant N-d\}$ to satisfy Corollary 3.4.2. Moreover, we can use this b to compute the dynamical dual with formula (3.2).

The question of whether or not a $d-$suitable x always exists is *almost* answered by Theorem 1 of [32]. We restate that result here in the language we use for completeness.

Lemma 3.4.3. *Let $F = \{f^i\}_{i=1}^N, N > d$ be a frame for $\mathbb{R}^d$. Then, there exists an x supported on a subset of $\{1, \ldots, d+1\}$ so that $M_{F,x}$ is invertible.*

We point out that this is quite close to what we need, but this result does not force x to be nonzero in coordinates 1 and $d+1$. However, a simple appeal to measure theory lets us show that, not only can we find a $d-$suitable x for which $M_{F,x}$ is invertible, in fact we can find uncountably many of them.

Theorem 3.4.4. *Let $d \in \mathbb{N}, N > d$ and F be a frame of size N for $\mathbb{R}^d$. Then F has infinitely many distinct dynamical dual frames.*

Proof. First, define

$$\iota : \mathbb{R}^{d+1} \to \mathbb{R}^N \text{ by } \iota : (x_1, \ldots, x_{d+1}) \mapsto (x_1, \ldots, x_{d+1}, 0, \ldots, 0)$$

and

$$\varphi : \mathbb{R}^{d+1} \to \mathbb{R} \text{ by } \varphi : (x_1, \ldots, x_{d+1}) \mapsto \det(M_{F,\iota(x)}).$$

Notice that the result in Lemma 3.4.3 guarantees the existence of some x_0 so that $\varphi(x_0) \neq 0$. We observe that φ is a polynomial in the variables $x_1, \ldots x_{d+1}$. The zero-set of a polynomial has Lebesgue measure 0 so $A = \{x : \varphi(x) \neq 0\}$ is a Lebesgue co-null subset of $\mathbb{R}^{d+1}$. Moreover, the set $U = \{x : x_1, x_{d+1} \neq 0\}$ is also co-null so that the intersection $S = U \cap A$ is co-null. Moreover, each vector $x \in S$ induces a $d-$suitable vector $\iota(x)$ so that $M_{F,\iota(x)}$ is invertible. Thus, F has a dynamical dual. All that is left to show that there are infinitely many dynamical duals.

Let u, v be $d-$suitable in $\mathbb{R}^N$ not co-linear so that $M_{F,u}$ and $M_{F,v}$ are invertible. We let G_u and G_v be the two dynamical frames dual to F with kernels given by span$\{u, Ru, \ldots, R^{N-d-1}u\}$ and span$\{v, Rv, \ldots, R^{N-d-1}v\}$, respectively. Showing that G_u and G_v are distinct proves there are infinitely many dynamical duals to F because there are infinitely many choice of u and v. If we can simply demon-

strate that $u \notin \text{Ker } G_v$, then we are done because G_u, G_v having different kernels suffices. Suppose that

$$u = \sum_{i=1}^{N-d} \alpha_i R^{i-1} v \in \text{ Ker } G_v.$$

Notice that $u_1 \neq 0$ because it is $d-$suitable. Because for $i > 1$, the first coordinate of $R^{i-1}v$ is zero, we have that $\alpha_1 \neq 0$. We argue the remaining α_i must be 0. The only vector of $\{R^{i-1}v : i \leqslant i \leqslant N-d\}$ supported in coordinate N is $i = N-d$. Since $u_N = 0$, this implies $\alpha_{N-d} = 0$. We can repeat the same argument to show that $\alpha_2, \ldots \alpha_{N-d} = 0$. Therefore, $u = \alpha_1 v$. But, this is a contradiction because we assumed u and v were not co-linear. Therefore $u \notin \text{Ker } G_v$ and so $G_u \neq G_v$. □

In light of the proof of the above Theorem, we can construct a dynamical dual of a given frame very quickly almost surely. Indeed, select a random $v \in \mathbb{R}^{d+1}$ according to any probability measure that is absolutely continuous with respect to Lebesgue measure. Then, $M_{F,\iota(v)}$ is almost surely invertible. Then, use formula (3.2) to quickly compute a dynamical dual to F.

3.5 Quantization with Dynamical Duals

Having shown that every frame has a rich space of dynamical dual frames, we turn our eye to explore the application of dynamical dual frames to the frame quantization problem so we may explore the utility of these objects for digital data storage.

Throughout this Section we let F be a fixed finite analysis frame for $\mathbb{R}^d$ and $G = \{T^i g^0\}_{i=1}^N$ be a dynamic dual (synthesis) frame to F. The quantization problem for frames is as follows. Fix a finite (often, very small) subset $\mathcal{A}$ of $\mathbb{R}$. Given a vector $x \in \mathbb{R}^N$, we compute a new vector $Q(x) \in \mathcal{A}^N$ so that the reconstruction error $\|GQ(x) - Gx\|_2$ is as small as possible. That is, we want to be able to store a full-precision vector using a finite amount of information while distorting the reconstructed signal as little as possible.

We denote the measurements of the signal s as $F^*s = x = (x_i)_{i=1}^N$. The main idea of a $\Sigma\Delta$ quantization scheme is dispersal of quantization error. The action of moving, e.g., x_1 from $\mathbb{R}$ into $\mathcal{A}$ induces an error when we reconstruct the signal using G in the direction of g^1. In general, some of this error is in the span of the next synthesis frame vector. Therefore, by adjusting x_2 to absorb some of this error we can reduce the total quantization error. An order-r $\Sigma\Delta$ scheme will seek to recoup more of the error by dispersing it onto

the next r frame vectors instead of just the next 1.

The first reason for using dynamical synthesis frames is that the quantization error may be perfectly dispersed precisely because consecutive frame vectors form a basis for the ambient space. After quantizing a coefficient, we may *perfectly* disperse the error onto the next d frame vectors because they are guaranteed to be a basis. This perfect dispersal property is precisely the result of Lemma 3.3.1.

Dynamical frames are not the only frames for which every d consecutive vectors are a basis; there are others frames that allow perfect dispersal. However, in general dispersing error requires the computationally expensive task of matrix inversion. For most frames, we would have to do this matrix inversion at every step. The second reason for using dynamical frames is that, because of their special structure, we only need to perform the matrix inversion once before we perform any quantization.

Let us develop this idea a bit more formally. If $[d] = \{1, \ldots, d\}$, then we use $G_{[d]}$ to denote the submatrix of G containing only the first d columns. Because G is dynamical, $G_{[d]}$ is invertible. Recalling that g^0 is the generating vector so that $g^i = T^i g^0$, we define the vector $(\alpha_i)_{i=1}^d$ by

$$G_{[d]}^{-1} g^0 = \alpha. \tag{3.3}$$

Now, the error generated by quantizing coefficient i is a vector in the direction of $T^i g^0$. So, to disperse the quantization error from coefficient i, we need to represent $T^i g^0$ in terms of $T^{i+1} g^0, \ldots, T^{i+d} g^0$. Now observe by (3.3),

$$\sum_{j=1}^{d} \alpha_i T^{i+j} g^0 = T^i \left(\sum_{j=1}^{d} \alpha_i T^j g^0 \right) = T^i \left(G_{[d]} \alpha \right) = T^i g^0. \tag{3.4}$$

That is, the parameter vector α tells us how to represent $T^i g^0$ in terms of $T^{i+1} g^0, \ldots T^{i+d} g^0$ and that we can use the same α for each dispersal problem. So, if the quantization error is $\beta T^i g^0$, then adding $\beta \alpha_j$ to coefficient x_{i+j} results in perfect dispersal. Below, we present formally the dynamical quantization scheme based on the preceding discussion.

In the following, the state variables $w_1, \ldots, w_d$ contain the accumulated dispersal of prior errors. Moreover, we let $\mathcal{Q}' : \mathbb{R} \to \mathcal{A}$ be the scalar quantizer, mapping a real number to the nearest element of $\mathcal{A}$ (with ties broken arbitrarily).

Dynamical Quantization Scheme (DQ)

Input: Measurement vector $(x_i)_{i=1}^N$, dispersal vector $(\alpha_i)_{i=1}^d$

Output: Quantized representation $(q_i)_{i=1}^N$

Initialize $w_i = 0$ for each $i = 1, \ldots, d$.

Set $q_1 = \mathcal{Q}'(x_1)$

for $i = 1, \ldots, d$ **do**

Set $w_i = (x_1 - q_1) \cdot \alpha_i$

end for

for $j = 2, \ldots N$ **do**

Set $q_j = \mathcal{Q}'(x_j + w_1)$

for $i = 1, \ldots, d-1$ **do**

Set $w_i = w_{i+1} + (x_j + w_1 - q_j) \cdot \alpha_i$

end for

Set $w_d = (x_j + w_1 - q_j) \cdot \alpha_d$

end for

The remainder of this Section is a discussion of error bounds for this quantization scheme. Our main result is that, under conditions only on the number of levels in the quantization alphabet and the function T, we can achieve error bounded above by a constant times $\|T\|_{op}^N$ where N is the frame size and T is the operator generating the dynamical frame. Thus, for frames with $\|T\|_{op} < 1$, we can achieve quite small error.

3.5.1 Analysis of Error

The first Lemma is the formal proof that the quantization scheme results in perfect dispersal. The vector S_k below resembles the reconstruction after iteration k but includes terms corresponding to what would be the $N+1, \ldots, N+d+1$ dynamical dual frame vectors. The reason for including these extra terms is to ease the analysis later in this Section. For clarity and ambiguity, we introduce an additional index on the state variables w_i. Specifically, we let w_i^j denote the value of state variable i at the completion of the loop iteration in which x_j is quantized. Then, the update of the state variables in the DQ algorithm looks like

$$w_i^j = w_{i+1}^{j-1} + (x_j + w_1^{j-1} - q_j) \cdot \alpha_i \tag{3.5}$$

$$w_d^j = (x_j + w_1^{j-1} - q_j) \cdot \alpha_d \tag{3.6}$$

where w_i^j is interpreted to be 0 for $j \leqslant 0$.

Lemma 3.5.1. *Fix a frame F with dynamical dual $\{T^i g^0\}_{i=1}^N$. Define $(x_i)_{i=1}^N = F^* s$ and for $N < i < N + d + 1$ define $x_i = 0$. Define the partially quantized reconstruction*

$$S_k = \sum_{i=1}^{k} q_i T^i g^0 + \sum_{i=k+1}^{k+d} (x_i + w_{i-k}^k) T^i g^0 + \sum_{i=k+d+1}^{N+d+1} x_i T^i g^0.$$

Then, for every $k = 1, \ldots, N$ we have $S_k = s$.

Proof. We will show this is true by inducting on k. Let $k = 1$. Then, $w_i^1 = (x_1 - q_1) \cdot \alpha_i$. Expanding the formula for S_1 we see

$$\begin{aligned} S_1 &= q_1 T g^0 + \sum_{i=2}^{d+1} \left(x_i + w_{i-1}^1\right) T^i g^0 + \sum_{i=d+2}^{N+d+1} x_i T^i g^0 \\ &= q_1 T g^0 + \sum_{i=2}^{d+1} w_{i-1}^1 T^i g^0 + \sum_{i=2}^{N} x_i T^i g^0 \end{aligned}$$

because $x_{N+1}, \ldots x_{N+d+1} = 0$. Then,

$$\begin{aligned} S_1 &= q_1 T g^0 + \sum_{i=1}^{d} (x_1 - q_1) \alpha_i T^{i+1} g_0 + \sum_{i=2}^{N} x_i T^i g_0 \\ &= q_1 T g^0 + (x_1 - q_1) T^1 g^0 + \sum_{i=2}^{N} x_i T^i g_0 \qquad (3.7) \\ &= \sum_{i=1}^{N} x_i T^i g^0 = s \end{aligned}$$

where equation (3.7) is by equality (3.4). Now, fix $k \in \{2, \ldots, N\}$. For this Lemma, it suffices to show the equality $S_k = S_{k-1}$. Notice that their difference can be expressed

$$\begin{aligned} S_k - S_{k-1} = &\sum_{i=1}^{k} q_i T^i g^0 + \sum_{i=k+1}^{k+d} (x_i + w_{i-k}^k) T^i g^0 + \sum_{i=k+d+1}^{N+d+1} x_i T^i g^0 \\ &- \sum_{i=1}^{k-1} q_i T^i g^0 - \sum_{i=k}^{k+d-1} (x_i + w_{i-k+1}^{k-1}) T^i g^0 - \sum_{i=k+d}^{N+d+1} x_i T^i g^0 \\ = &\, q_k T^k g^0 + (x_{k+d} + w_d^k) T^{k+d} g^0 - (x_k + w_1^{k-1}) T^k g^0 \end{aligned}$$

$$+ \sum_{i=k+1}^{k+d-1} (w_{i-k}^k - w_{i-k+1}^{k-1}) T^i g^0 - x_{k+d} T^{k+d} g^0$$
$$= (q_k - x_k - w_1^{k-1}) T^k g^0 + \sum_{i=1}^{d-1} (w_i^k - w_{i+1}^{k-1}) T^{k+i} g^0 + w_d^k T^{k+d} g^0 \tag{3.8}$$

Now, notice that for $i \in \{2, \ldots, d-1\}$, using equation (3.5) gives $w_i^k - w_{i+1}^{k-1} = (x_k + w_1^{k-1} - q_k) \cdot \alpha_i$. Moreover, $w_d^k = (x_k + w_1^{k-1} - q_k) \cdot \alpha_d$. Substituting these into (3.8) we can continue,

$$\begin{aligned} S_k - S_{k-1} &= (q_k - x_k - w_1^{k-1}) T^k g^0 + \sum_{i=1}^{d} (x_k + w_1^{k-1} - q_k) \cdot \alpha_i T^{k+i} g^0 \\ &= (q_k - x_k - w_k^{k-1}) T^k v^0 + (x_k + w_k^{k-1} - q_k) T^k \left(\sum_{i=1}^{d} \alpha_i T^i g^0 \right) \\ &= 0 \end{aligned}$$

because $\sum_{i=1}^d \alpha_i T^i g^0 = g^0$ by equation (3.5.1). □

The vector S_N has a close relationship to the vector we reconstruct after quantization. We care about minimizing the error between the original signal s and the reconstruction GQF^*s. In particular, because $S_N = s$ by Lemma (3.4),

$$s - GQF^*s = S_N - \sum_{i=1}^{N} q_i T^i g_0 = \sum_{i=N+1}^{N+d} (w_{i-N}^N) T^i g^0. \tag{3.9}$$

Our main result, Theorem 3.5.2, argues that if $\|T\|_{op} < 1$, then this error will be quite small. This seems clear and indeed if we can argue that the state variables stay small in magnitude, then this fact is *immediate*. The main work in Theorem 3.5.2 is showing that if the quantization alphabet $\mathcal{A}$ is big enough, then this is indeed the case.

Theorem 3.5.2. *Let F be a frame for $\mathbb{R}^d$ with dynamical dual frame $\{T^i g^0\}_{i=1}^N$ and let α be defined as in* (3.3)*. Suppose that the original signal s is such that $\|F^*s\|_\infty < r$ and that $\mathcal{A} = \{-m, \ldots, m\}$ so that $m \geqslant r + \frac{\|\alpha\|_1}{2}$. Suppose that $\|T\|_{op} < 1$. Then, the dynamic quantizer $Q : \mathbb{R}^N \to \mathcal{A}^N$ gives reconstruction error bounded above by*

$$\|s - GQF^*s\|_2 \leqslant \left(\frac{d}{2} \|g^0\|_2 \cdot \|\alpha\|_1 \right) \|T\|_{op}^{N+1}. \tag{3.10}$$

Proof. The proof technique is to show that the variables w_i^k are uniformly bounded so that we may substitute this upper bound into (3.9). Fix a measurement vector $x = F^* s$ as above. We will show by induction over k that

$$|w_i^k| \leqslant \frac{1}{2} \sum_{\ell=i}^{d} |\alpha_\ell| \tag{3.11}$$

Start with $k = 1$. We first point out if $v \in [-m - \frac{1}{2}, m + \frac{1}{2}]$ and $\mathcal{Q}' : \mathbb{R} \to \mathcal{A}$ is the scalar quantizer associated to $\mathcal{A}$, then $|\mathcal{Q}'(v) - v| \leqslant \frac{1}{2}$. Therefore, $|x_1 - q_1| \leqslant \frac{1}{2}$. Then, by definition

$$|w_i^1| = |(x_1 - q_1) \cdot \alpha_i| \leqslant \frac{1}{2}|\alpha_i| \leqslant \frac{1}{2} \sum_{\ell=i}^{d} |\alpha_\ell|.$$

Now we let k be arbitrary and assume (3.11) holds for w_i^j with $j < k$. Notice first that $|x_k + w_1^{k-1}| \leqslant r + \frac{\|\alpha\|_1}{2}$ because $\|x\|_\infty = \|F^* s\|_\infty < r$ so that

$$|x_k + w_1^{k-1} - \mathcal{Q}'(x_k + w_1^{k-1})| = |x_k + w^{k-1} - q_k| \leqslant \frac{1}{2}$$

Then, by using the formulas in (3.5) and (3.6), we can see that, for $i \in \{1, \dots d-1\}$, the following inequalities hold

$$\begin{aligned}
|w_i^k| &\leqslant |w_{i+1}^{k-1}| + |x_k + w_1^{k-1} - q_k| \cdot |\alpha_i| \leqslant \frac{1}{2} \sum_{\ell=i+1}^{d} |\alpha_\ell| + \frac{1}{2}|\alpha_i| \leqslant \frac{1}{2} \sum_{\ell=1}^{d} |\alpha_\ell| \\
|w_d^k| &\leqslant |x_k + w_1^{k-1} - q_k| \cdot |\alpha_d| \leqslant \frac{1}{2}|\alpha_d|.
\end{aligned}$$

Thus, the state variables are uniformly bounded. We then use (3.9) to compute:

$$\begin{aligned}
\|s - GQF^* s\|_2 &= \| \sum_{i=1}^{d} w_i^N T^{N+i} g^0 \|_2 \\
&= \|T^N \left(\sum_{i=1}^{d} w_i^N T^i g^0 \right) \|_2 \\
&\leqslant \|T^N\|_{op} \cdot \frac{\|\alpha\|_1}{2} \cdot \| \sum_{i=1}^{d} T^i g^0 \|_2 \\
&\leqslant \|T\|_{op}^N \cdot \frac{\|\alpha\|_1}{2} \sum_{i=1}^{d} \|T^i g^0\|_2.
\end{aligned} \tag{3.12}$$

Now, because $\|T\|_{op} < 1$ we know that $\|T^i g^0\|_2 < \|T\|_{op}\|g^0\|_2$ for every i. Thus, we can replace $\sum_{i=1}^{d} \|T^i g^0\|_2 \leqslant d\|T\|_{op}\|g^0\|_2$. Substituting this into (3.12) completes the proof. □

The above Theorem is not as general as possible in terms of the alphabet $\mathcal{A}$ used. Indeed, the same proof technique will show that, using an alphabet $\mathcal{A} = \delta \cdot \{-m, -m+1, \ldots, m\}$, then if $\delta m \geqslant r + \frac{\delta\|\alpha\|_1}{2}$ the bound (3.10) holds with $\frac{d}{2}$ replaced by $\frac{d\delta}{2}$.

One of the key assumptions of Theorem 3.5.2 is that T, the matrix generating the dynamical dual frame to F, satisfies $\|T\|_{op} < 1$. From a computational perspective, it would be desirable not only that every F had a dual with this property but that we could find those duals quickly. However, the following counterexample exhibits a frame F for which *every* dynamical dual frame is generated by T with $\|T\|_{op} > 1$.

Example 2. *Let $\{e^i\}_{i=1}^d$ be the canonical basis for $\mathbb{R}^d$. Define $F = \{f^i\}_{i=1}^N$ in the following way. Let $f^1, \ldots f^{N-d}$ each be the zero vector. Then, we let $f^{N-d+1}, \ldots, f^N$ be the vectors $10e^1, e^2, \ldots, e^d$, respectively. Then, if G is a dual frame to F, we may see from the relation $GF^* = I_d$ that the final d columns of G must be the vectors $\frac{1}{10}e^1, e^2, \ldots e^d$. Thus, if T generates a dynamical dual to F, then we must have that $T(\frac{1}{10}e^1) = e^2$ so that $\|T\|_{op} \geqslant 10$.*

While this counterexample may preclude us from using dynamical quantization for arbitrary frames, hope is not lost. Recall that frames are not, in general, inherently *ordered* objects. However, when we enforce a dynamical structure on a frame we also enforce an ordering. The prior example required a very particular frame in a very particular order to go through. We are motivated, then, to ask whether every frame admits an ordering so that it has a dynamical dual frame arising from an operator with $\|T\|_{op} < 1$.

Moreover, we would like to know whether Example 2 is typical or atypical. Does *almost every* frame have dynamical duals that are useful in quantization? If not *almost every*, then do *most*? Finally, we want to better understand the relationship between the $d-$suitable vectors $b \in \mathbb{R}^N$ that we use to generate dynamical frames and the operator norm of the matrix T that generates the dynamical frame. Understanding this relationship will likely help answer the two previously posed questions.

3.5.2 Numerical Verification

While the previous Section showed that not every frame allows us to use dynamical quantizaton effectively, we still can show it works in some cases. We ran the following experiment and show the outcome

below. We began by generating random operators T until one had operator norm less than 1. Then, we selected a random vector v_0 and for each frame size between 3 and 300 generated the frame $G = \{T^i v_0\}_{i=1}^N$ and used as measurement matrix the canonical dual to G (which is also dynamical!). Then, we performed quantization on the vector $(\pi, e)^T$. Shown below is a log plot containing four data sequences: the theoretical error upper bound (including constants) from Theorem 3.5.2 and the real reconstruction error along with the lines corresponding to $\frac{1}{n}$ and $\frac{1}{n^2}$ decay.

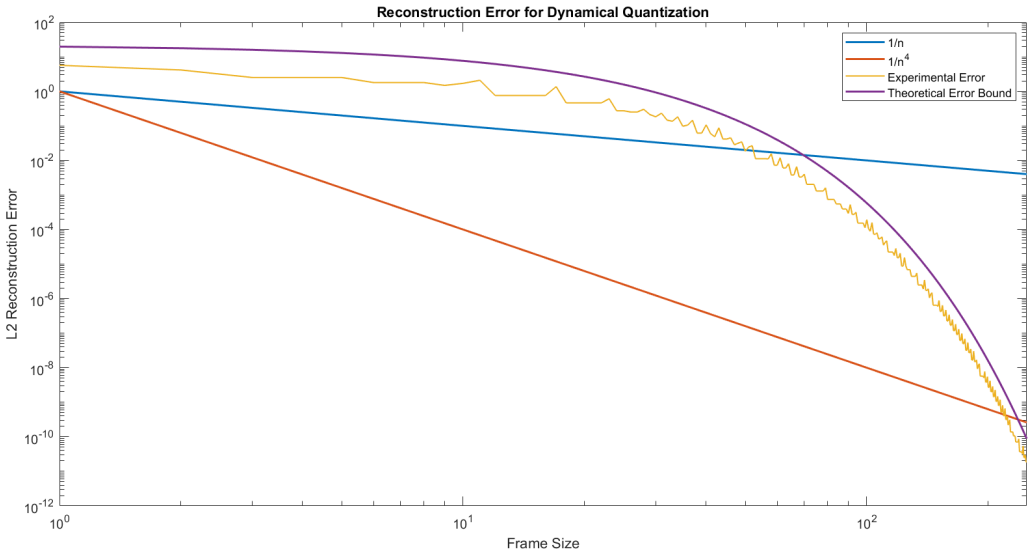


Figure 3.1: Reconstruction Error for the Dynamical Quantization Algorithm

We see that our algorithm does indeed achieve exponential error decay at a rate very close to (but never larger than!) the theoretical upper error bound.

3.5.3 Commentary

While this new line of research into dynamical dual frames does indeed have applications to quantization, we are obliged to highlight some potential shortcomings. One of the most glaring issues is that our algorithm is not flexible in the sense that we are not free to choose the memory requirements of our algorithm.

Traditional $\Sigma\Delta$ algorithms allow specification of the order r that determines memory requirements. While in a computer having relatively large r is not strictly prohibitive, many $\Sigma\Delta$ applications are actually accomplished with circuitry. The order of dynamical quantization is the dimension of the ambient space.

For many applications - such as images and audio signals - this ambient dimension can be extremely large. Ideally one could define an alternate quantization scheme using dynamical frames that allows flexibility in selecting the order.

CHAPTER 4

Recovery of Compressed Signals

4.1 The Compressed Sensing Problem

The two preceding Chapters were primarily focused on finding compressed representations when the original was known to arbitrary precision. However, it is crucially important in the design of acquisition system we to study the problem of *how* to efficiently measure signals that we know are compressible.

Real world signals often contain relatively small amounts of information compared to their dimension. This property manifests in the following widely observed fact: many signal classes, when expanded in the appropriate basis, are very sparse. Sparse signals inherently are simpler to store, transmit, and process than dense ones. Because of the utility of sparse signals, it is of particular interest to find sparse representations in useful bases when they exist.

The field of Compressed Sensing (CS) was developed precisely to study the concepts of representation, acquisition, and recovery of sparse signals. One central question in CS is this: Given a vector of possibly noisy measurements, how may we find the sparsest signal that explains these measurements (up to, perhaps, some error tolerance level). More precisely, letting $\|x\|_0$ denote the number of non-zero elements of the vector x, then given a tolerance ϵ, we wish to solve

$$\min_{x \in \mathbb{C}^d} \|x\|_0 \qquad \text{subject to } \|Ax - y\|_2 < \epsilon \tag{4.1}$$

when given both A and y.

While the field of CS is mainly concerned with finding the sparse representations, it is necessary to study closely related questions. For instance, it is well known that (4.1) is NP-Hard for general matrices A and every $\epsilon \geqslant 0$ [15]. However, with additional structure forced upon A, namely the restricted isometry property (RIP), the problem is actually tractable.

With this knowledge that the problem *can* be solved, research quickly turned to the question of *how* to solve (4.1). The list of techniques includes, yet is not limited to, iterative hard thresholding (IHT) [4, 5], (orthogonal) matching pursuit [12], hard thresholding pursuit [14], and CoSaMP [29]. Each of these

algorithms solve (4.1) provided A has sufficiently strong RIP.

Many of the above techniques, particularly those which are *thresholding* based, rely crucially on the action of the hard thresholding operator. For instance, IHT simply alternates a gradient descent step followed by hard thresholding. The hard thresholding operator acts by taking in an arbitrary vector and returning the nearest $s-$sparse vector. However, doing so makes no use of y, the measurement vector. This leads us to ask: Can we find a better thresholding technique?

In this Chapter, we answer this question in the affirmative. We propose the Look Ahead Thresholding (LAT) technique which will be shown in a variety of settings to achieve better results than simply using hard thresholding. In particular, we modify IHT to use our new thresholding rule and propose the Iterative Look Ahead Thresholding (ILAT) algorithm. We prove that ILAT has comparable worst case performance to IHT in terms of the required RIP to converge. Moreover, we show both experimentally and theoretically that look ahead thresholding excels when used in compressed sensing.

The remainder of this Chapter is organized as follows. Section 4.2 will present the necessary background information for compressed sensing, develop our new thresholding rule, and present the ILAT algorithm. Section 4.3 analyzes the worst-case behavior of ILAT in relation to the RIP of the sensing matrix. Section 4.4 contains an average case analysis showing the power of look ahead thresholding. Finally, Section 4.5 shows in a few different experiments that this technique performs exceedingly well in practice.

4.2 Background: compressed sensing

Let us begin with the necessary priors from compressed sensing while simultaneously standardizing our notation. A vector $x \in \mathbb{R}^d$ is said to be $s-$sparse if it has at most s non-zero entries. We note that most signals of interest are only sparse in a particular basis so one may similarly define sparsity with respect to some other orthonormal basis such as wavelets, fourier, etc.

The process of 'taking a measurement' is computing an inner product against a known vector. So, the transformation from signal to a vector of measurements is realized by matrix multiplication. We let $x^* \in \mathbb{R}^d$ be the $s-$sparse signal to recover. Given access to the vector of measurements $y = Ax^*$ in $\mathbb{R}^m$, the goal is to recover quickly and uniquely the $s-$sparse vector x^*. Notice that if $m \geqslant d$ and A is invertible, we can recover x^* perfectly from y. In practice, we want to minimize the necessary number of measurements so we study at length the case when $m < d$.

At the other extreme, notice that if we knew the support of x^*, then s linearly independent measurements would suffice to recover x^*. While we cannot know the support of every measured signal, knowing that the signal is sparse inherently reduces the complexity of x^*. Therefore, we hope to still be able to get away with using m measurements with $s < m < d$.

Two of the main questions, then, in compressed sensing are:

1. What does the matrix A need to look like for x^* to be uniquely determined amongst all $s-$sparse vectors by $y = Ax^*$?

2. Knowing A and y, how can we procedurally reconstruct the solution x^*?

The first question is very well-studied and traditionally answered by enforcing A to have the restricted isometry property.

Definition 4.2.1. *A matrix A has s^{th} restricted isometry constant $\delta_s = \delta_s(A)$ provided that δ_s is the smallest $\delta \geqslant 0$ so that for every $s-$sparse vector x we have*

$$(1 - \delta)\|x\|_2^2 \leqslant \|Ax\|_2^2 \leqslant (1 + \delta)\|x\|_2^2.$$

We say informally that A has the restricted isometry property (RIP) if δ_s is relatively small for s relatively large. Essentially, though A cannot possibly be an isometry when $m < d$, if A has RIP of order s then every subset of s columns of A form an approximate isometry.

We let $\|\cdot\|_{op}$ be the operator norm of a matrix relative to the Euclidean norms on both the domain and range. Every matrix A trivially has $\delta_s(A) \leqslant \max\{1, \|A\|_{op}^2 - 1\}$ so we need to ask: How strong does the RIP of A need to be to guarantee that x^* is uniquely determined by y? In theory, recovery is possible if the matrix A is injective on the set of $s-$sparse vectors. Notice that if x, y are both $s-$sparse, their difference is $2s-$sparse. Then, if $\delta_{2s} < 1$, it follows that $A(x - y) \neq 0$ by definition. So, in fact the condition $\delta_{2s} < 1$ is sufficient for the map $x^* \mapsto Ax^*$ to have *an* inverse on the $s-$sparse signal space.

While $\delta_{2s} < 1$ is sufficient *in theory* for recovery of $s-$sparse vectors, in practice this is not enough. Notice that the set of sparse vectors is not a subspace of $\mathbb{R}^d$, so the inverse map $Ax^* \mapsto x^*$ is not linear. Thus, most compressed sensing techniques require much stronger versions of the restricted isometry property in order to be able to find x^* quickly, robustly, and stably.

Unfortunately, the restricted isometry property is a delicate thing. Constructing matrices of a given size and with suitable RIP is difficult or impossible to do deterministically. Luckily, though, large enough random matrices with entries chosen i.i.d. from a Bernoulli or other sub-Gaussian distribution tend to have sufficiently strong RIP [15]. That is, we do not know necessarily how to construct RIP matrices but we do know that using random matrices with sufficiently many rows will suffice with high probability.

Having answered question (1) above, we can now turn our attention to question (2): how exactly can we perform the non-linear inversion of the matrix A on the set of $s-$sparse vectors? There are countless techniques and algorithms, one of which we discuss in depth in the next Section.

4.2.1 Iterative Hard Thresholding

Iterative Hard Thresholding was first introduced by Blumensath and Davies in [4] as a method of sparse dictionary approximation and by the same authors shortly after in [5] for compressed sensing. There are many ways to interpret IHT but we prefer to understand it as alternating a gradient descent step followed by a hard thresholding step.

To develop the gradient descent setting, given the measurements y and matrix A, define the cost function

$$C(x) = \|y - Ax\|_2^2 = \|A(x^* - x)\|_2^2 \tag{4.2}$$

to describe how well the vector x explains the observed measurements y. A routine calculation shows the gradient of C is $\nabla C(x) = -2A^*(y - Ax) = -2A^*A(x^* - x)$. In general, a thresholder is any function that outputs *some* $s-$sparse projection of the input vector. In particular, we define H_s to be the *hard thresholding operator* which acts on a vector by retaining only the s largest entries in magnitude while setting the rest to zero (ties can be broken arbitrarily). Then, given a starting point x^0 and using a 'step-size' of $1/2$, IHT is defined by the iteration

1. $a^{t+1} = x^t - \frac{1}{2}\nabla C(x^t) = x^t + A^*(y - Ax^t)$
2. $x^{t+1} = H_s(a^{t+1})$.

The inspiration here is that gradient descent helps move towards a zero of C while the thresholding ensures that our iterates are actually sparse. And, under the conditions $\delta_{2s} < 1$ the only point having both these properties is x^*. However, because it is possible that H_s 'undoes' enough of the progress of the

gradient step, we are not necessarily guaranteed to converge to a point where $\nabla C(x^t) = 0$. If we specify a stronger RIP, though, we can guarantee convergence.

For IHT to work, we need x^t to stay *about as close* to x^* as a^t is. That is, H_s needs to not undo too much of the gradient progress. The following crucially important fact from [5] gets us most of the way there:

$$\|x^* - x^t\|_2 = \|x^* - H_s(a^t)\|_2 \leqslant 2\|x^* - a^t\|_2. \tag{4.3}$$

Then, because $x^* - a^t = (I - A^*A)(x^* - x^{t-1})$ and noting that the RIP forces A^*A to be close to the identity when applied to sparse vectors (see Lemma 4.3.2), we can show $\|x^* - x^t\|_2 \leqslant \rho\|x^* - x^{t-1}\|_2$ for $\rho < 1$ so that (x^t) converges to x^*.

This fact generalizes. Suppose we have an algorithm that alternates $a^t = x^{t-1} + A^*(y - Ax^{t-1})$ then chooses x^t to be *some* thresholding of a^t (not necessarily using H_s). Then, if

$$\|x^* - x^t\|_2 \leqslant k\|x^* - a^t\|_2 \tag{4.4}$$

for some universal k, the same Lemma 4.3.2 lets us specify which RIP of A (namely $\delta_{2s} < 1/k^2$) suffices to prove convergence when normalizing $\|A\|_{op} = 1$. This is the proof technique we pursue in Section 4.3.

Here is a final thought to motivate our campaign for alternative thresholding. If P_* is the projection onto the support of x^*, then

$$\|x^* - P_* a^t\|_2 \leqslant \|x^* - a^t\|_2.$$

That is, inequality (4.4) is satisfied for $k = 1$. Comparing this to the hard thresholding requirement $k = 2$, this leaves quite a bit of room for improvement. While we cannot use P_* because the support is in general unknown, maybe we can use some of the information content from the measurements y to pick a thresholding that does better than $k = 2$.

4.2.2 Look Ahead Thresholding

In this Section we will describe how our novel look ahead thresholder chooses which coordinates to keep and which to kill. Along the way, we will show the thought process that led to the definition of look

ahead thresholding.

Fix a vector $z \in \mathbb{R}^d$. Then, $H_s(z)$ is the $s-$sparse vector $H(z)$ minimizing $\|z - H(z)\|_2$. However, per the discussion preceding inequality (4.4), the ideal thresholder H for sparse recovery is the one which minimizes $\|x^* - H(z)\|_2$. That is, it is preferable to threshold a vector z in a way so that the result is as close to the solution x^* as possible. While we cannot minimize $\|x^* - H(z)\|_2$ directly because we do not know x^*, we develop look ahead thresholding to get a good approximation.

Notice that the value of the cost C at a particular thresholding $H(z)$ of z is

$$C(H(z)) = \|y - A(H(z))\|_2^2 = \|A(x^* - H(z))\|_2^2.$$

Then, because $x^* - H(z)$ is $2s-$sparse, if A has RIP it is an approximate isometry so that $C(H(z))$ is close to $\|x^* - H(z)\|_2^2$. Moreover, C is something we can work with unlike $\|x^* - H(z)\|_2^2$.

Unfortunately, C is a non-separable quadratic and there are $\binom{d}{s}$ possible $s-$sparse projections of a point z. It is computationally intractable to find the $H(z)$ minimizing C. We address this issue by introducing a surrogate $f_{\eta,z}$ which is a separable quadratic related closely to the Taylor series of C at z. The idea for using $f_{\eta,z}$ was inspired by the work in [26] which searched for sparse representations for neural networks.

First let us simply define the surrogate

$$f_{\eta,z}(x) = C(z) + \sum_{i=1}^{d} \left(2\eta \frac{\partial C}{\partial x_i}(z)(x_i - z_i) + (x_i - z_i)^2 \right). \tag{4.5}$$

We claim this function is quite closely related to the Taylor series for C at z. Indeed, first assume we have normalized the columns of A to have $\|A_i\|_2 = 1$ so that $\frac{\partial^2 C}{\partial x_i^2} = 1$. Then the function $f_{\eta,z}$ is formed from the Taylor series for C by first ignoring the off-diagonal terms of the Hessian then multiplying the linear terms by a tunable constant. Ignoring the off-diagonal terms is necessary for separability and adding the weight to the gradient term lets us control the algorithm's performance more carefully.

Now we can define the look ahead thresholding rule. If z is the point to threshold, we pick the thresholding of z that minimizes $f_{\eta,z}$. More precisely, first let P_Ω be the projection onto the coordinate axes indexed by $\Omega \subset \{1, \ldots, d\}$. Then, the look ahead thresholding function $H_{s,\eta}$ has the action defined by

$$H_{s,\eta}(z) = \underset{P_\Omega z, |\Omega| = s}{\operatorname{argmin}} f_{\eta,z}(P_\Omega z). \tag{4.6}$$

We recall that the original problem (4.1) can be solved exactly if given unlimited time. Therefore, it is important to emphasize that the action defined in (4.6) can be computed quickly. Notice that, for a particular projection $P_\Omega(z)$, the function $f_{\eta,z}$ takes the value

$$f_{\eta,z}(P_\Omega(z)) = C(z) + \sum_{i \notin \Omega} \left(2\eta \frac{\partial C}{\partial x_i}(z)(-z_i) + (-z_i)^2 \right). \tag{4.7}$$

Then we observe that the solution to (4.6) can be computed by picking Ω to contain precisely the s coordinates for which the values

$$2\eta \frac{\partial C}{\partial x_i}(z)(-z_i) + (z_i)^2 \tag{4.8}$$

are largest. Because computation of $\frac{\partial C}{\partial x_i}(z)$ is by far the most expensive step of (4.8), the cost to compute $H_{s,\eta}(z)$ is essentially the cost of computing the gradient $\nabla C(z)$.

Now, we promised a second motivation for the definition of the thresholding rule (4.6). This reformulation of $H_{s,\eta}$ is the basis for the name *look ahead thresholding*. Because this reformulation is used throughout the proofs in Sections 4.3 and 4.4, Lemma 4.3.3 proves this reformulation is correct. Moreover, this reformulation also lets us see that hard thresholding is a special case of look ahead thresholding with the weight η chosen to be zero.

Define the **look ahead point** $\ell_\eta = z - \eta \nabla C(z)$, the point that would be the next gradient descent step from z with step-size η. Then, look ahead thresholding defined by (4.6) picks the $s-$sparse projection of z that is closest to ℓ_η. Because for specific values of η, the point ℓ_η is provably closer to x^* than z is, we might expect this thresholding to find better projections. Indeed, this is the case.

To show how look ahead thresholding may be applied in practice, we define the Iterative Look Ahead Thresholding algorithm as an example. The only difference between ILAT and IHT is that the thresholding step is done using $H_{s,\eta}$ instead of H_s. That is, we use look ahead thresholding instead of hard thresholding to hopefully retain as much of the progress made by the gradient updates as possible while remaining $s-$sparse.

In Section 4.3 we show that this algorithm is guaranteed to converge under some RIP conditions that are, perhaps surprisingly, more restrictive than the conditions for IHT. However, convergence guarantees are worst case analyses and we argue for the use of look ahead thresholding both theoretically in Section 4.4 and experimentally in Section 4.5 by appealing to the average case. Let us end this Section by formally

presenting Iterative Look Ahead Thresholding.

Iterative Look Ahead Thresholding
Input: Matrix A, measurements y, sparsity s, iterations T
Output: Estimate x^T of $s-$sparse solution x^*.
Set $x^0 = 0 \in \mathbb{R}^d$
for $1 \leqslant t \leqslant T$ **do**
  Set $a^t = x^{t-1} + A^*(y - Ax^{t-1})$
  for $1 \leqslant i \leqslant d$ **do**
    Compute $s_i = 2\eta(-a_i^t)\frac{\partial C}{\partial x_i}(a^t) + (a_i^t)^2$
  end for
  Set $\Omega \subset \{1, \ldots, d\}$ the indices of the s largest values of s_i.
  Set $x^t = P_\Omega a^t$.
end for

4.3 Iterative Look Ahead Thresholding Converges with Suitable RIP

As is typical for compressed sensing, we present theorems of the form 'for sufficient restricted isometry constants, our algorithm is guaranteed to recover the solution x^*'. We prove theorems of this sort both in a noiseless environment and a corresponding result when the measurements may be corrupted. We begin first with a few technical facts that show up in our analysis time and again.

4.3.1 Technical Lemmas

We point out that in our analysis, we require that the measurement matrix A is normalized to have operator norm 1. This is slightly different from traditional CS literature which normalizes the columns of A to have expected squared length 1. However, in some settings normalizing via the operator norm has been observed to yield superior performance [3].

Lemma 4.3.1. *If A is an $m \times d$ matrix and $\|A\|_{op} = 1$, then $\|I - 2\eta A^*A\|_{op} \leqslant 1$ whenever $\eta \in [0, 1]$. Moreover, if $m < d$ then $\|I - 2\eta A^*A\|_{op} = 1$.*

Proof. We build a singular value decomposition for $I - 2\eta A^*A$ from an svd for A itself. Let the svd for A be $A = V\Sigma U$. Then notice that $I - 2\eta A^*A = U^*(I - 2\eta\Sigma^*\Sigma)U$. Notice first that because $\|A\|_{op} = 1$, the diagonal of $\Sigma^*\Sigma$ is in $[0, 1]$. Thus, the diagonal of $D = I - 2\eta\Sigma^*\Sigma$ is in $[-1, 1]$. Now, form $\tilde{D}$ and $\tilde{U}$ by first replacing the negative entries in D by their absolute values then multiplying the corresponding row in U by -1.

We point out three things. First, that we can write $I - 2\eta A^*A = U^*\tilde{D}\tilde{U}$. Second, the matrix $\tilde{U}$ remains a unitary. Finally, $\tilde{D}$ is a diagonal matrix with positive entries in $[0, 1]$. Therefore $U\tilde{D}\tilde{U}$ is an svd for $I - 2\eta A^*A$ so $\|I - 2\eta A^*A\|_{op} \leqslant 1$.

Now, because $m < d$ we can pick a nontrivial v in the kernel of A so $(I - 2\eta A^*A)v = v$. Therefore $\|I - 2\eta A^*A\|_{op} = 1$. □

The prior Lemma controlled the operator norm of $I - 2\eta A^*A$ over the entirety of its domain. The following Lemma controls the operator norm of $I - A^*A$ when restricted to $s-$sparse vectors. In this case we get much better performance in terms of the restricted isometry constants of A.

Lemma 4.3.2. *If $x \in \mathbb{R}^d$ is $s-$sparse and $\|A\|_{op} = 1$, then $\|(I - A^*A)x\|_2 \leqslant \sqrt{\delta_s}\|x\|_2$.*

Proof. Simply expand

$$\begin{aligned}\|(I - A^*A)x\|_2^2 &= \|x\|_2^2 + \|A^*Ax\|_2^2 - 2\langle x, A^*Ax\rangle \\ &= \|x\|_2^2 - \|Ax\|_2^2 + \|A^*Ax\|_2^2 - \|Ax\|_2^2. \end{aligned} \tag{4.9}$$

Now, because A^* has norm 1, $\|A^*(Ax)\|_2^2 - \|Ax\|_2^2 \leqslant 0$. By re-arranging the RIP condition we may also see

$$\|Ax\|_2^2 \geqslant (1 - \delta_s)\|x\|_2^2 \Rightarrow \delta_s\|x\|_2^2 \geqslant \|x\|_2^2 - \|Ax\|_2^2. \tag{4.10}$$

Then, substituting the reformulation in (4.10) into (4.9), our result follows by taking a square root. □

We remark that Lemma 4.3.2 is similar to the following reformulation of δ_s whose proof can be found in [15]. If A_Ω is the matrix formed from A by taking the columns indexed by Ω, then

$$\delta_s(A) = \sup_{|\Omega|=s} \|I - A_\Omega^* A_\Omega\|_{op}^2. \tag{4.11}$$

Finally, we will show that using the decision rule (4.6) to threshold a vector z is equivalent to picking the $s-$sparse thresholding of z which is closest to the look ahead point ℓ_η.

Lemma 4.3.3. *Fix a vector $z \in \mathbb{R}^d$, $\eta > 0$, measurement vector $y \in \mathbb{R}^m$, and an $m \times d$ matrix A. Let $C(z) = \|y - Az\|_2^2$ and define $\ell_\eta = z - \eta\nabla C(z)$. Then, the vector $H_{s,\eta}(z)$ is the closest $s-$sparse*

thresholding of z to ℓ_η.

Proof. Notice that for a generic vector x,

$$\begin{aligned}\|x-\ell_\eta\|_2^2 &= \sum_{i=1}^{d}\left(x_i-\left(z_i-\eta\frac{\partial C}{\partial x_i}(z)\right)\right)^2\\ &= \sum_{i=1}^{d}\left((x_i-z_i)+\eta\frac{\partial C}{\partial x_i}(z)\right)^2\\ &= \sum_{i=1}^{d}\left(2\eta\frac{\partial C}{\partial x_i}(z)(x_i-z_i)+(x_i-z_i)^2\right)+\eta^2\|\nabla C(z)\|_2^2. \end{aligned} \tag{4.12}$$

For an arbitrary index set Ω and projection $P_\Omega z$ we have that

$$\|P_\Omega z-\ell_\eta^t\|_2^2 = \sum_{i\notin\Omega}\left(2\eta\frac{\partial C}{\partial x_i}(z)(-z_i)+(z_i)^2\right)+\eta^2\|\nabla C(z)\|_2^2.$$

Now notice that, no matter the choice of Ω, the value

$$\sum_{i\notin\Omega}\left(2\eta\frac{\partial C}{\partial x_i}(z)(-z_i)+(z_i)^2\right)+\sum_{i\in\Omega}\left(2\eta\frac{\partial C}{\partial x_i}(z)(-z_i)+(z_i)^2\right)+\eta^2\|\nabla C(z)\|_2^2$$

is a constant. So, minimizing the sum indexed over $i\notin\Omega$ is equivalent to maximizing the sum indexed over $i\in\Omega$. Selecting Ω in this way is precisely the decision rule for $H_{s,\eta}(z)$ given by (4.8). □

4.3.2 Noiseless Convergence

In this Section we prove that Iterative Look Ahead Thresholding recovers exactly the exactly $s-$sparse solution x^* from the noiseless measurements $y=Ax$ provided $\delta_{2s}(A)$ is sufficiently small. Per the discussion preceding (4.4), if we can establish a universal k so that

$$\|x^*-x^t\|_2 = \|x^*-H_{s,\eta}(a^t)\|_2 \leqslant k\|x^*-a^t\|_2 \tag{4.13}$$

then invoking Lemma 4.3.2 and expanding the definition of a^t, it follows that

$$\|x^*-x^t\|_2 \leqslant k\sqrt{\delta_{2s}}\|x^*-x^{t-1}\|_2.$$

Then, we may specify an RIP constraint so that we recover x^* in the limit. The first step is to determine a suitable constant k.

Lemma 4.3.4. *Suppose we are given an $n \times d$ matrix A with $\|A\|_{op} = 1$ and $y = Ax^*$ for some $s-$sparse $x^* \in \mathbb{R}^d$. Let $0 \leqslant \eta \leqslant 1$ and the sequences $(x^t), (a^t)$ be defined by Iterative Look Ahead Thresholding with parameter η. Then,*

$$\|x^* - x^t\|_2 \leqslant (1 + \sqrt{1 + 4\eta^2})\|x^* - a^t\|_2.$$

Proof. Let $\ell_\eta = a^t - \eta \nabla C(a^t)$, the look ahead point from a^t. Let P be the projection onto the support of a best $s-$term approximation of ℓ_η. Because $x^t = H_{s,\eta}(a^t)$ is the closest $s-$sparse thresholding of a^t to ℓ_η, it follows that $\|x^t - \ell_\eta\|_2 \leqslant \|Pa^t - \ell_\eta\|_2$. Then we write

$$\begin{aligned} \|x^* - x^t\|_2 &\leqslant \|x^* - \ell_\eta\|_2 + \|x^t - \ell_\eta\|_2 \\ &\leqslant \|x^* - \ell_\eta\|_2 + \|Pa^t - \ell_\eta\|_2. \end{aligned} \tag{4.14}$$

Notice first that $\|x^* - \ell_\eta\|_2 = \|(I - 2\eta A^* A)(x^* - a^t)\|_2$. Because $0 \leqslant \eta \leqslant 1$ and through Lemma 4.3.1 we can say

$$\|x^* - \ell_\eta\|_2 \leqslant \|x^* - a^t\|_2. \tag{4.15}$$

Dealing now with the second term of (4.14) we see

$$\begin{aligned} \|Pa^t - \ell_\eta\|_2^2 &= \|Pa^t - P\ell_\eta\|_2^2 + \|P\ell_\eta - \ell_\eta\|_2^2 \\ &\leqslant \|a^t - \ell_\eta\|_2^2 + \|x^* - \ell_\eta\|_2^2 \qquad (4.16) \\ &\leqslant \|a^t - \ell_\eta\|_2^2 + \|x^* - a^t\|_2^2. \qquad (4.17) \end{aligned}$$

Above, inequality (4.16) is because projection is norm 1 and because $P\ell_\eta$ is a better $s-$term approximation of ℓ_η than x^* is. Now again because of the normalization $\|A\|_{op} = 1$,

$$\|a^t - \ell_\eta\|_2 = \|\eta \nabla C(a^t)\|_2 = \|2\eta A^* A(x^* - a^t)\|_2 \leqslant 2\eta \|x^* - a^t\|_2.$$

We finish working with inequality (4.17) by substituting to yield:

$$\|Pa^t - \ell_\eta\|_2^2 \leqslant (1 + 4\eta^2)\|x^* - a^t\|_2^2. \tag{4.18}$$

Finally, substituting (4.15) and (4.18) into (4.14) we see

$$\|x^t - x^*\|_2 \leqslant \|x^* - a^t\|_2 + \sqrt{1 + 4\eta^2}\|x^* - a^t\|_2$$

which is precisely the inequality given in the Lemma statement. □

Now, as argued above, the proof of our first main Theorem requires just an algebraic manipulation after appealing to Lemmas 4.3.4 and 4.3.2.

Theorem 4.3.5. *Suppose we are given an $m \times d$ matrix A with $\|A\|_{op} = 1$ and $y = Ax^*$ for some $s-$sparse $x^* \in \mathbb{R}^d$. Let $0 \leqslant \eta \leqslant 1$ and the sequences $(x^t), (a^t)$ be defined by Iterative Look Ahead Thresholding using with parameter η. If the restricted isometry constant δ_{2s} of A satisfies $\delta_{2s} < \frac{1}{\left(1+\sqrt{1+4\eta^2}\right)^2}$, then*

$$\|x^t - x^*\|_2 \leqslant \rho^t\|x^0 - x^*\|_2$$

where $\rho = \sqrt{\delta_{2s}}\left(1 + \sqrt{1 + 4\eta^2}\right) < 1$. In particular, (x^t) converges to x^*.

Proof. Because the conditions of Lemma 4.3.4 are satisfied, we have that $\|x^t - x^*\|_2 \leqslant (1+\sqrt{1 + 4\eta^2})\|a^t - x^*\|_2$. Then,

$$\begin{aligned}
\|x^t - x^*\|_2 &\leqslant (1 + \sqrt{1 + 4\eta^2})\|a^t - x^*\|_2 \\
&= (1 + \sqrt{1 + 4\eta^2})\|x^{t-1} - x^* + A^*A(x^* - x^{t-1})\|_2 \\
&= (1 + \sqrt{1 + 4\eta^2})\|(I - A^*A)(x^{t-1} - x^*)\|_2 \\
&\leqslant (1 + \sqrt{1 + 4\eta^2})\sqrt{\delta_{2s}}\|x^{t-1} - x^*\|_2
\end{aligned}$$

where the final inequality follows from Lemma 4.3.2 because $x^{t-1} - x^*$ is $2s$-sparse and $\|A\|_{op} = 1$. Therefore, by iteration we have that $\|x^t - x^*\|_2 \leqslant \rho^t\|x^0 - x^*\|_2$ for $\rho < 1$ so convergence is guaranteed. □

4.3.3 Noisy Convergence

Let us now turn our attention to the case where there is noise present. In particular, the signal x^* is no longer required to be exactly $s-$sparse and the measurements look like $y = Ax^* + e$ for e some noise term. We define the following auxiliary noise term which describes how much y is corrupted from the noiseless measurements of the best $s-$term approximation of x^*.

Definition 4.3.6. *For a vector $x^* \in \mathbb{R}^d$, $H_s(x^*)$ its best $s-$term approximation, and a corrupted measurement vector $y = Ax^* + e$, we define the error term $\tilde{e} = y - A(H_s(x^*)) = A(x^* - H_s(x^*)) + e$.*

We have the following useful relation for the gradient at a point in terms of $\tilde{e}$:

$$\begin{aligned}\eta\nabla C(z) = -2\eta A^*(y - Az) &= -2\eta A^*(Ax^* + e - Az) \\ &= -2\eta A^*A(H_s(x^*) - z) + A^*(A(x^* - H_s(x^*)) + e) \\ &= -2\eta A^*A(H_s(x^*) - z) - 2\eta A^*\tilde{e}. \end{aligned} \tag{4.19}$$

Before proving our noisy convergence Theorem, which proceeds very similarly to the previous Section, let us offer a comment on what convergence actually means here. Because the signal x^* is not exactly $s-$sparse and our algorithm returns sparse signals, at best we hope to recover the signal $H_s(x^*)$. However, because our measurements are corrupted by noise, even this cannot be expected. We do show, however, that (x^t) will converge to some small neighborhood of $H_s(x^*)$ and the size of the neighborhood depends only on δ_{2s}, $\|\tilde{e}\|_2$ and η.

Lemma 4.3.7. *Suppose we are given an $m \times d$ matrix A with $\|A\|_{op} = 1$ and $y = Ax^* + e$ for some $x^* \in \mathbb{R}^d$. Let $0 \leqslant \eta \leqslant 1$ and the sequences $(x^t), (a^t)$ be defined by Iterative Look Ahead Thresholding with parameter η. Then,*

$$\|x^t - H_s(x^*)\|_2 \leqslant (2 + 2\eta)\|a^t - H_s(x^*)\|_2 + 6\eta\|\tilde{e}\|_2.$$

Proof. Let $\ell_\eta = a^t - \eta\nabla C(a^t)$. First we will derive two bounds based on the equality (4.19). Because Lemma 4.3.1 guarantees $\|I - 2\eta A^*A\|_{op} \leqslant 1$,

$$\|H_s(x^*) - \ell_\eta\|_2 = \|H_s(x^*) - a^t + \eta\nabla C(a^t)\|_2 = \|(I - 2\eta A^*A)(H_s(x^*) - a^t) + 2\eta A^*\tilde{e}\|_2$$

$$\leqslant \|H_s(x^*) - a^t\|_2 + 2\eta\|\tilde{e}\|_2. \tag{4.20}$$

Second, directly from (4.19) we have

$$\|a^t - \ell_\eta\|_2 = \|\eta\nabla C(a^t)\|_2 \leqslant 2\eta\|H_s(x^*) - a^t\|_2 + 2\eta\|\tilde{e}\|_2. \tag{4.21}$$

Next we use the triangle inequality:

$$\|x^t - H_s(x^*)\|_2 \leqslant \|x^t - \ell_\eta\|_2 + \|H_s(x^*) - \ell_\eta\|. \tag{4.22}$$

Because the second term on the right of (4.22) is bounded by (4.20), let us work with the first term on the right. If P is the projection onto the support of $H_s(\ell_\eta)$, then because x^t is the closest $s-$sparse thresholding of a^t to ℓ_η, we also have that

$$\begin{aligned}\|x^t - \ell_\eta\|_2 \leqslant \|Pa^t - \ell_\eta\|_2 &\leqslant \|Pa^t - P\ell_\eta\|_2 + \|P\ell_\eta - \ell_\eta\|_2 \\ &\leqslant \|a^t - \ell_\eta\|_2 + \|H_s(x^*) - \ell_\eta\|_2,\end{aligned} \tag{4.23}$$

where the last term is because $P\ell_\eta$ is a better $s-$term approximation of ℓ_η than is $H_s(x^*)$. Substituting (4.23) into (4.22) yields

$$\|x^t - H_s(x^*)\|_2 \leqslant 2\|H_s(x^*) - \ell_\eta\|_2 + \|a^t - \ell_\eta\|_2. \tag{4.24}$$

Now, we again substitute (4.20) and (4.21) directly into (4.24) achieves the stated bound in this Lemma.

□

Again, our second main Theorem follows from Lemma 4.3.7 with some simple algebra.

Theorem 4.3.8. *Suppose we are given an $m \times d$ matrix A with $\|A\|_{op} = 1$ and $y = Ax^* + e$ for $x^* \in \mathbb{R}^d$. Let $0 \leqslant \eta \leqslant 1$ and the sequences $(x^t), (a^t)$ be defined by Iterative Look Ahead Thresholding with parameter η. If the restricted isometry constant δ_{2s} of A satisfies $\delta_{2s} < \frac{1}{(2+2\eta)^2}$, then the sequence (x^t) satisfies the recurrence relation*

$$\|H_s(x^*) - x^t\|_2 \leqslant \rho\|H_s(x^*) - x^{t-1}\|_2 + (2 + 8\eta)\|\tilde{e}\|_2$$

for $\rho = (2+2\eta)\sqrt{\delta_{2s}} < 1$.

Proof. By Lemma 4.3.7, we have $\|H_s(x^*) - x^t\|_2 \leqslant (2+2\eta)\|H_s(x^*) - a^t\|_2 + 6\eta\|\tilde{e}\|_2$. But notice that

$$\begin{aligned}
\|H_s(x^*) - x^t\|_2 &\leqslant (2+2\eta)\|H_s(x^*) - a^t\|_2 + 6\eta\|\tilde{e}\|_2 \\
&= (2+2\eta)\|H_s(x^*) - x^{t-1} - A^*(y - Ax^{t-1})\|_2 + 6\eta\|\tilde{e}\|_2 \\
&= (2+2\eta)\|H_s(x^*) - x^{t-1} - A^*(A(H_s(x^*) - x^{t-1} + x^* - H_s(x^*)) + e)\|_2 + 6\eta\|\tilde{e}\|_2 \\
&\leqslant (2+2\eta)\left(\|(I - A^*A)(H_s(x^*) - x^{t-1})\|_2 + \|A^*(\tilde{e})\|_2\right) + 6\eta\|\tilde{e}\|_2 \\
&\leqslant (2+2\eta)\sqrt{\delta_{2s}}\|H_s(x^*) - x^{t-1}\|_2 + (2+8\eta)\|\tilde{e}\|_2 \qquad (4.25)
\end{aligned}$$

where the last line is because of Lemma 4.3.2. □

4.4 Average Case Analysis

Our goal in defining a better thresholding rule H given an $s-$sparse vector to recover x^* is to minimize the distance $\|H(z) - x^*\|_2$. Moreover, we are interested in measuring the size of $\|H(z) - x^*\|_2$ relative to $\|z - x^*\|_2$. Recall that in the case of $H = H_s$, the hard thresholding operator, at best we can say that $\|H_s(z) - x^*\|_2 \leqslant 2\|z - x^*\|_2$ (see, e.g., inequality (21) of [5]). Even if we take the expected value of $\|H_s(z) - x^*\|$ over a random choice of measurement matrix A, this upper bound does not improve because H_s is independent of A.

However, for look ahead thresholding the average case is much better. In this Section we will show that taking expectation over random A with entries chosen i.i.d. from a Gaussian,

$$\mathbb{E}\left[\|H_{s,\eta}(z) - x^*\|_2\right] \leqslant \rho\|z - x^*\|_2 \qquad (4.26)$$

and that ρ can be taken strictly smaller than 2 for a range of η values. Now, during the proof of Lemma 4.3.4 we derive the following important inequality which is the basis of this Section's analysis. We recreate it here in the precise form we need for completeness.

Lemma 4.4.1. *Let $z \in \mathbb{R}^d$ be a vector, $x^* \in \mathbb{R}^d$ be $s-$sparse and A some $m \times d$ measurement matrix. Let*

$C(z) = \|A(x^* - z)\|_2^2$ and define $\ell_\eta = z - \eta\nabla C(z)$ to be the look ahead point from z. Then,

$$\|H_{s,\eta}(z) - x^*\|_2 \leqslant \sqrt{\|z - \ell_\eta\|_2^2 + \|\ell_\eta - x^*\|_2^2} + \|\ell_\eta - x^*\|_2. \tag{4.27}$$

Proof. Let P be the projection onto the support of a best $s-$term approximation of ℓ_η. Then, because x^* is a worse $s-$term approximation of ℓ_η than $P\ell_\eta$ is,

$$\begin{aligned}\|Pz - \ell_\eta\|_2^2 &= \|Pz - P\ell_\eta\|_2^2 + \|\ell_\eta - P\ell_\eta\|_2^2 \\ &\leqslant \|z - \ell_\eta\|_2^2 + \|\ell_\eta - x^*\|_2^2.\end{aligned}$$

Now, by choice $H_{s,\eta}(z)$ is at least as close to ℓ_η as Pz is. So, a triangle inequality leaves

$$\begin{aligned}\|H_{s,\eta}(z) - x^*\|_2 &\leqslant \|H_{s,\eta}(z) - \ell_\eta\|_2 + \|\ell_\eta - x^*\|_2 \\ &\leqslant \|Pz - \ell_\eta\|_2 + \|\ell_\eta - x^*\|_2 \\ &\leqslant \sqrt{\|z - \ell_\eta\|_2^2 + \|\ell_\eta - x^*\|_2^2} + \|\ell_\eta - x^*\|_2.\end{aligned}$$

□

The reason this upper bound is useful is because we can write each term on the right hand side of (4.27) as a transformation of a common vector:

$$z - \ell_\eta = -2\eta A^* A(x^* - z) \tag{4.28}$$

$$x^* - \ell_\eta = (I - 2\eta A^* A)(x^* - z). \tag{4.29}$$

Then, controlling the random behavior of A and the related matrices $I - 2\eta A^* A$ and $2\eta A^* A$ gives us our result. In the rest of this Section we compute the expected size of $\|z - \ell_\eta\|_2^2$ and $\|x^* - \ell_\eta\|_2^2$ over the random draw of A.

We first desire to answer: For random V, in what ways does the average size of $\|Vy\|_2^2$ depend on both $\|y\|_2^2$ and on V? It turns out that under some mild constraints which are satisfied in our setting (see Lemma 4.4.6), the important quantity is the expected Frobenius norm of the matrix in question.

Lemma 4.4.2. *Fix a vector $y \in \mathbb{R}^d$ and let V be an $m \times d$ random matrix with entries satisfying*

1. *The columns $V_i, 1 \leqslant i \leqslant d$ of V all have the same expected squared length.*

2. *Two distinct entries $V_{i,j}, V_{i,k}$ from the same row have $\mathbb{E}\left[V_{i,j}V_{i,k}\right] = 0$.*

Then, the expected value of $\|Vy\|_2^2$ is scaled by the Frobenius norm like:

$$\mathbb{E}\left[\|Vy\|_2^2\right] = \frac{1}{d} \cdot \mathbb{E}\left[\|V\|_F^2\right] \cdot \|y\|_2^2.$$

Proof. We begin by expanding the random variable of concern:

$$\begin{aligned}\|Vy\|_2^2 &= \sum_{i=1}^{m} \left(\sum_{j=1}^{d} V_{i,j}y_j\right)\left(\sum_{k=1}^{d} V_{i,k}y_k\right) \\ &= \sum_{i=1}^{m}\sum_{j=1}^{d}\sum_{k=1}^{d} V_{i,j}V_{i,k}y_jy_k. \end{aligned} \tag{4.30}$$

If $j \neq k$, then $\mathbb{E}\left[V_{i,j}V_{i,k}y_jy_k\right] = 0$ because of the assumption (2) on V. Therefore taking the expectation of both sides of (4.30) we are only left with

$$\begin{aligned}\mathbb{E}\left[\|Vy\|_2^2\right] &= \sum_{i=1}^{m}\sum_{j=1}^{d} \mathbb{E}\left(V_{i,j}^2y_j^2\right) \\ &= \sum_{j=1}^{d} y_j^2 \sum_{i=1}^{m} \mathbb{E}\left(V_{i,j}^2\right). \end{aligned} \tag{4.31}$$

Then, by the assumption (1) on the random matrix V, the terms $\sum_{i=1}^{m}\mathbb{E}(V_{i,j}^2)$ are constant for each column j. Letting this common squared expected length be denoted c, then we can continue (4.31) by

$$\mathbb{E}\left[\|Vy\|_2^2\right] = c\sum_{j=1}^{d} y_j^2 = c\|y\|_2^2. \tag{4.32}$$

Finally, it remains to show that the the constant c is equal to the expected squared Frobenius norm divided by the dimension of y. Well, the expected Frobenius norm can be written

$$\mathbb{E}\left[\|V\|_F^2\right] = \sum_{j=1}^{d}\sum_{i=1}^{m} V_{i,j}^2 = \sum_{j=1}^{d} c = cd$$

which completes the proof □

We remark that the probabilistic setting above is quite mild and is satisfied by most random matrix ensembles including any matrix with entries selected i.i.d. from a zero mean distribution. For the remainder of this Section we will work with A whose entries are drawn i.i.d. from a Gaussian normalized so the expected squared column length is 1. Before computing the expected Frobenius norms, though, we will isolate a few key probabilistic facts. Throughout the remainder of this Chapter we let $N(\mu, \sigma^2)$ denote the normal distribution with mean μ and variance σ^2.

Lemma 4.4.3. *Let $A_1, A_2, A_3 \in \mathbb{R}^m$ be random vectors with entries selected i.i.d. from $N(0, \frac{1}{m})$. Then,*

$$\mathbb{E}\left[\langle A_1, A_2\rangle^2\right] = \frac{1}{m} \tag{4.33}$$

and for $p = 3$ or $p = 1$ we have

$$\mathbb{E}\left[\langle A_1, A_2\rangle\langle A_1, A_p\rangle\right] = 0. \tag{4.34}$$

Moreover, the fourth moment of $\|A_i\|_2$ behaves like

$$\mathbb{E}\left[\|A_i\|_2^4\right] = 1 + \frac{2}{m}. \tag{4.35}$$

Proof. We expand the first expectation like

$$\begin{aligned}\mathbb{E}\left(\langle A_1, A_2\rangle^2\right) &= \mathbb{E}\left[\left(\sum_{k=1}^{m}(A_1)_k(A_2)_k\right)\cdot\left(\sum_{\ell=1}^{m}(A_1)_\ell(A_2)_\ell\right)\right] \\ &= \mathbb{E}\left[\sum_{k,\ell=1}^{m}(A_1)_k(A_2)_k(A_1)_\ell(A_2)_\ell\right].\end{aligned} \tag{4.36}$$

First notice that when $k \neq \ell$, the four terms are distinct and chosen mutually independently so expectation commutes with taking their product. Then, since they are each zero mean, the product of the four terms has zero expected value. Therefore, after taking the expectation of both sides of (4.36) the only terms contributing are $k = \ell$. Therefore,

$$\mathbb{E}\left[\langle A_1, A_2\rangle^2\right] = \sum_{k=1}^{m}\mathbb{E}\left[(A_1)_k^2(A_2)_k^2\right]$$

$$= \sum_{k=1}^{m} \mathbb{E}\left[(A_1)_k^2\right] \mathbb{E}\left[(A_2)_k^2\right] = \frac{1}{m}$$

where the last line is because $(A_1)_k$ and $(A_2)_k$ are independent.

Now for the second part of this Lemma. Identically to the previous part,

$$\mathbb{E}\left[\langle A_1, A_2\rangle\langle A_1, A_p\rangle\right] = \sum_{k,\ell=1}^{m} \mathbb{E}\left[(A_1)_k(A_2)_k(A_1)_\ell(A_p)_\ell\right].$$

Now, because the entries are mutually independent and $(A_2)_k$ is always distinct from $(A_1)_k, (A_1)_\ell$ and $(A_p)_\ell$ no matter if $p = 1$ or $p = 3$, then for every k, ℓ we have

$$\mathbb{E}\left[(A_1)_k(A_2)_k(A_1)_\ell(A_p)_\ell\right] = \mathbb{E}\left[(A_1)_k(A_1)_\ell(A_p)_\ell\right] \mathbb{E}\left[(A_2)_k\right] = 0$$

which finishes the second part of this Lemma.

Finally we will compute the fourth moment of $\|A_j\|_2$

$$\begin{aligned} \|A_j\|_2^4 &= \left(\sum_{i=1}^{m} (A_j)_i^2\right)\left(\sum_{k=1}^{m} (A_j)_i^2\right) \\ &= \sum_{i=1}^{m} (A_j)_k^4 + \sum_{i=1}^{m}\sum_{k\neq i} (A_j)_i^2 (A_j)_k^2. \end{aligned} \tag{4.37}$$

Now because $(A_j)_i, (A_j)_k$ are independent when $i \neq k$, taking expectations of both sides of (4.37) leaves us with

$$\begin{aligned} \mathbb{E}\left[\|A_j\|_2^4\right] &= \sum_{i=1}^{m} \mathbb{E}\left[(A_j)_i^4\right] + \sum_{i=1}^{m}\sum_{k\neq j} \mathbb{E}\left[(A_j)_i^2\right] \mathbb{E}\left[(A_j)_k^2\right] \\ &= m\left(\frac{3}{m^2}\right) + (m^2 - m)\left(\frac{1}{m^2}\right) \end{aligned}$$

where the last equality is in part because the fourth moment of a zero mean Gaussian is $3\sigma^4$. Cleaning up this expression finishes the proof. □

With all our tools ready, we can now compute the expected Frobenius norm of the matrix $I - 2\eta A^* A$.

Lemma 4.4.4. *Let A be an $m \times d$ random matrix with entries $A_{i,j}$ selected i.i.d. from $N(0, \frac{1}{m})$ and let*

$\eta > 0$. *Then the squared Frobenius norm of* $I - 2\eta A^* A$ *has expected value*

$$\mathbb{E}\left[\|I - 2\eta A^* A\|_F^2\right] = 4d\left[\left(\frac{d+m+1}{m}\right)\eta^2 - \eta + \frac{1}{4}\right].$$

Proof. The off diagonal entries of $I - 2\eta A^* A$, of which there are $d^2 - d$, all have the form $-2\eta\langle A_i, A_j\rangle$ for $i \neq j$. Then, by Lemma 4.4.3 the expected squared size of each off diagonal term is $\frac{4\eta^2}{m}$. Therefore the off-diagonal contribution to the squared Frobenius norm is $(d^2 - d)\frac{4\eta^2}{m}$.

The diagonal entries of $I - 2\eta A^* A$ are of the form $1 - 2\eta\|A_i\|_2^2$. So their squared expectation again by the Lemma 4.4.3 is

$$\begin{aligned}\mathbb{E}\left[(1 - 2\eta\|A_i\|_2^2)^2\right] &= 1 - 4\eta\mathbb{E}\left[\|A_i\|_2^2\right] + 4\eta^2\mathbb{E}\left[\|A_i\|_2^4\right] \\ &= 1 - 4\eta + 4\eta^2\left(1 + \frac{2}{m}\right)\end{aligned}$$

so the diagonal contribution to the expected squared Frobenius norm is $d\left(1 - 4\eta + 4\eta^2\left(1 + \frac{2}{m}\right)\right)$. Combining the diagonal and off-diagonal terms we get

$$\mathbb{E}\left[\|I - 2\eta A^* A\|_F^2\right] = d\left(1 - 4\eta + 4\eta^2 + \frac{8\eta^2}{m}\right) + (d^2 - d)\frac{4\eta^2}{m} = 4d\left[\left(\frac{d+m+1}{m}\right)\eta^2 - \eta + \frac{1}{4}\right].$$

□

In much the same way we now compute the expected Frobenius norm of $2\eta A^* A$.

Lemma 4.4.5. *Let* A *be an* $m \times d$ *random matrix with entries* $A_{i,j}$ *selected i.i.d. from* $N(0, \frac{1}{m})$ *and let* $\eta > 0$. *Then the squared Frobenius norm of* $2\eta A^* A$ *has expected size*

$$\mathbb{E}\left[\|2\eta A^* A\|_F^2\right] = 4d\left[\left(\frac{d+m+1}{m}\right)\eta^2\right].$$

Proof. We will compute the expected squared Frobenius norm for $A^* A$ and pick up the $4\eta^2$ by linearity. The $d^2 - d$ off-diagonal entries of $A^* A$ are precisely $\langle A_i, A_j\rangle$ for i and j distinct. Thus, by Lemma 4.4.3 these terms each contribute $\frac{1}{m}$. Moreover, the diagonal terms are $\|A_i\|_2^2$ which have expected squared size

$1 + \frac{2}{m}$. Thus we have

$$\mathbb{E}\left[\|A^*A\|_F^2\right] = (d^2 - d)\left(\frac{1}{m}\right) + d\left(1 + \frac{2}{m}\right) = d\left[\frac{d+m+1}{m}\right].$$

□

Finally, in order to use Lemma 4.4.2, we need to verify that the matrices $2\eta A^*A$ and $I - 2\eta A^*A$ satisfy its conditions. This Lemma does precisely that.

Lemma 4.4.6. *Let A be an $m \times d$ random matrix selected i.i.d. from $N(0, \frac{1}{m})$. Then the matrices $2\eta A^*A$ and $I - 2\eta A^*A$ each satisfy*

1. *The columns of each matrix have the same expected squared length, and*
2. *For $V = 2\eta A^*A$ or $V = I - 2\eta A^*A$, two distinct entries $V_{i,j}$ and $V_{i,k}$ from the same row of V have $\mathbb{E}[V_{i,j}V_{i,k}] = 0$.*

Proof. It is straightforward to see that both $I - 2\eta A^*A$ and $2\eta A^*A$ have columns with the same expected squared length by their definition. Let us verify condition (2) directly for each matrix.

Let us begin with $V = 2\eta A^*A$. Two distinct entries from row i of V are either

- $\langle A_i, A_j\rangle$ and $\langle A_i, A_k\rangle$ for $j \neq k$ and neither equal to i, or
- $\langle A_i, A_j\rangle$ and $\|A_i\|_2^2$ for $j \neq i$.

In both of these situations, the expectation of their products are handled by equation (4.34) of Lemma 4.4.3 taking first $p = k$ then $p = i$.

Now, let $V = I - 2\eta A^*A$. Akin to when $V = 2\eta A^*A$, two distinct entries from a row of V look like either

- $-2\eta\langle A_i, A_j\rangle$ and $-2\eta\langle A_i, A_k\rangle$ for $j \neq k$ and neither equal to i, or
- $1 - 2\eta\|A_i\|_2^2$ and $-2\eta\langle A_i, A_j\rangle$ for $j \neq i$.

Again, the first of these two possibilities is handled by equation (4.34) of Lemma 4.4.3. Now all that is left to observe is that:

$$\mathbb{E}\left[\left(1 - 2\eta\|A_i\|_2^2\right) \cdot \langle A_i, A_j\rangle\right] = \mathbb{E}\left[\langle A_i, A_j\rangle\right] - \mathbb{E}\left[2\eta\|A_i\|_2^2\langle A_i, A_j\rangle\right] = 0.$$

The first term is 0 by a simple expansion while the second term is zero again by Lemma 4.4.3.

□

Finally we can combine all the previous results from this Section to be able to say that in the average case, look ahead thresholding outperforms hard thresholding on arbitrary vectors.

Theorem 4.4.7. *Fix vectors z, x^* in $\mathbb{R}^d$ with x^* $s-$sparse and let A be an $m \times d$ random matrix with i.i.d. $N(0, \frac{1}{m})$ entries. Then,*

$$\mathbb{E}\left[\|H_{s,\eta}(z) - x^*\|_2\right] \leqslant \rho\|z - x^*\|_2 \tag{4.38}$$

for $\rho < 2$ whenever $0 < \eta < \frac{1}{2}\frac{m}{m+d+1}$.

Proof. From Lemma 4.4.1, we have that

$$\mathbb{E}\left[\|H_{s,\eta}(z) - x^*\|_2\right] \leqslant \mathbb{E}\left[\sqrt{\|z - \ell_\eta\|_2^2 + \|\ell_\eta - x^*\|_2^2} + \|\ell_\eta - x^*\|_2\right]. \tag{4.39}$$

First we will deal with the square root term. By substituting from Lemmas 4.4.4 and 4.4.5 and using Jensen's inequality we get

$$\begin{aligned}
\mathbb{E}\left[\sqrt{\|z - \ell_\eta\|_2^2 + \|\ell_\eta - x^*\|_2^2}\right] &\leqslant \left(\mathbb{E}\left[\|z - \ell_\eta\|_2^2 + \|\ell_\eta - x^*\|_2^2\right]\right)^{1/2} \\
&= \left(\mathbb{E}\left[\|2\eta A^*A\|_F^2 + \|I - 2\eta A^*A\|_F^2\right]\right)^{1/2} \cdot \|z - x^*\|_2 \\
&= \left(\frac{8(d+m+1)}{m}\eta^2 - 4\eta + 1\right)^{1/2} \cdot \|z - x^*\|_2.
\end{aligned} \tag{4.40}$$

Now the second term of (4.39) can be handled with just Lemma 4.4.5 to yield:

$$\begin{aligned}
\mathbb{E}\left[\|\ell_\eta - x^*\|_2\right] &\leqslant \left(\mathbb{E}\left[\|(I - A^*A)(x^* - z)\|_2^2\right]\right)^{1/2} \\
&= \left(\frac{4(d+m+1)}{m}\eta^2 - 4\eta + 1\right)^{1/2} \|z - x^*\|_2.
\end{aligned} \tag{4.41}$$

Noticing that (4.41) is dominated by (4.40) for every value of η, we can make the greatly simplifying

substitution that

$$\mathbb{E}\left[\|H_{s,\eta}(z) - x^*\|_2\right] \leqslant 2\left(\frac{8(d+m+1)}{m}\eta^2 - 4\eta + 1\right)^{1/2}\|z - x^*\|_2$$

and it is easy to verify that $\frac{8(d+m+1)}{m}\eta^2 - 4\eta + 1$ is less than 1 for precisely the specified values of eta. $\square$

We have two comments related to the analysis in this Section. First, after equation (4.41) we make a simplifying substitution to reduce the complexity greatly. If desired, we could use numerical software to expand the range of η values that give better average case performance.

Second, the result in Theorem 4.4.7 is an attempt to say that, when compared to hard thresholding, look ahead thresholding returns sparse vectors that are closer to the desired solution on average. However, when $H_{s,\eta}$ is used in practice, e.g. as a step in ILAT, the points to threshold are far from arbitrary. In fact, they rely quite heavily on the draw of the random matrix A. Perhaps surprisingly, the experimental results of the following Section seem to suggest that this interaction actually *improves* the performance of look ahead thresholding relative to hard thresholding. That is, the range of values of η for which ILAT outperforms IHT is much larger than predicted by Theorem 4.4.7. Certainly there is more to be understood here.

4.5 Experiments

While the last Section was theoretical justification for the use of look ahead thresholding over hard thresholding, this Section is experimental justification.

We perform three main experiments to support our claims. First, we show how well ILAT performs relative to IHT when the algorithms' iterations are held constant. Second, because ILAT takes extra computational power, we show that even when holding computations approximately constant we outperform IHT. Finally, we validate directly Theorem 4.4.7 by testing the thresholders' performances outside the framework of iterative algorithms.

For our first experiment, we record the percentage of exact recoveries for a range of sparsity values for both IHT and ILAT with a variety of η values. The measurement matrix we use is 128×256 i.i.d. Gaussian scaled to have operator norm 1. Figure 4.1 shows ILAT outperforming IHT for a range of η values.

We also tested our algorithm in the case where our measurements are corrupted by random noise. The performance metric used here is Euclidean distance between the true solution and the algorithm's output

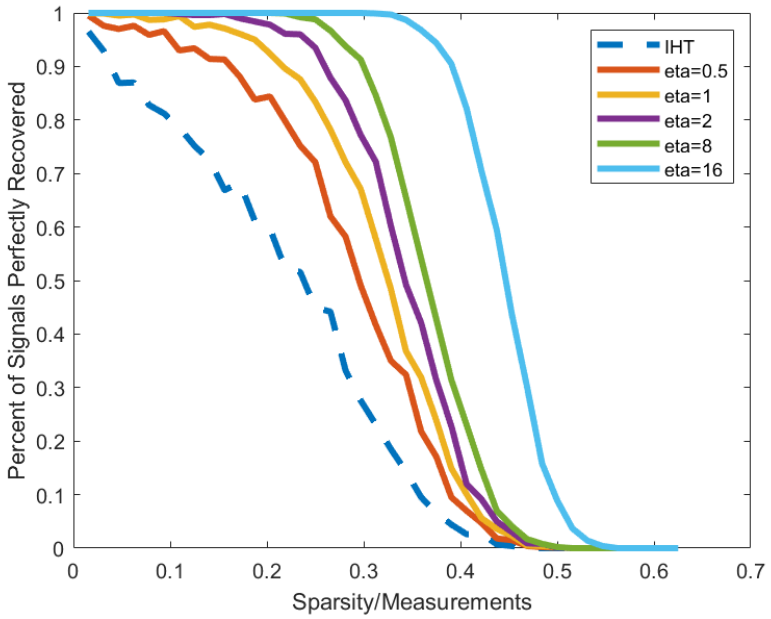
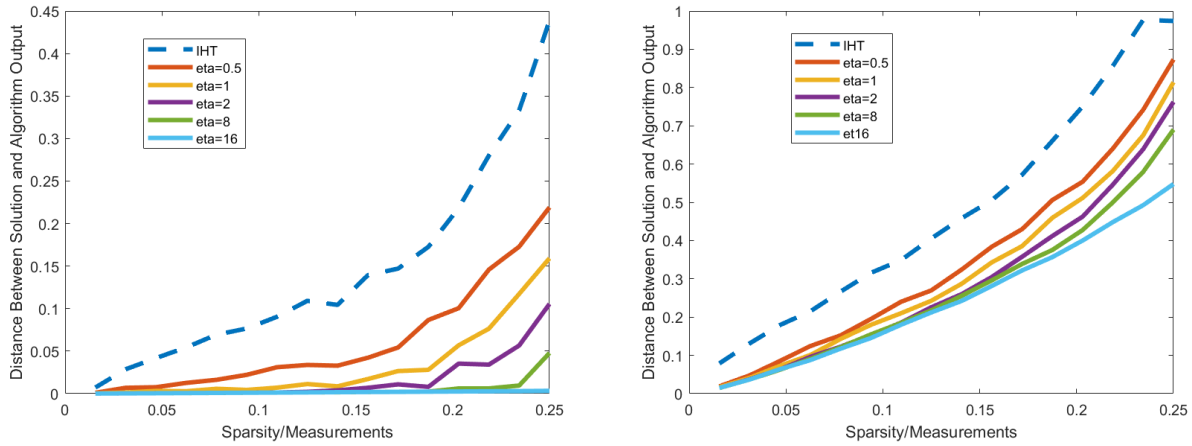


Figure 4.1: Percentage of signals perfectly reconstructed using IHT (dashed) and ILAT (solid, η increases to the right) using the same number of iterations

after 1000 iterations. We performed these experiments in the same settings as the above example. For Figure 4.2a, our measurements had a 30 dB SNR and in Figure 4.2b a 0 dB SNR. As above, Figure 4.2 show ILAT achieves smaller error than IHT.

The two previous experiments held the number of iterations constant. Since look ahead thresholding is more costly than hard thresholding, we need to ask whether ILAT is superior to IHT when we hold constant the amount of computations or run-time. Notice that the expensive step is the gradient computation and that ILAT requires two gradients per iteration while IHT requires only one. With this in mind, we compared the percentage of perfect recoveries for IHT against ILAT when IHT is allowed twice the number of iterations.

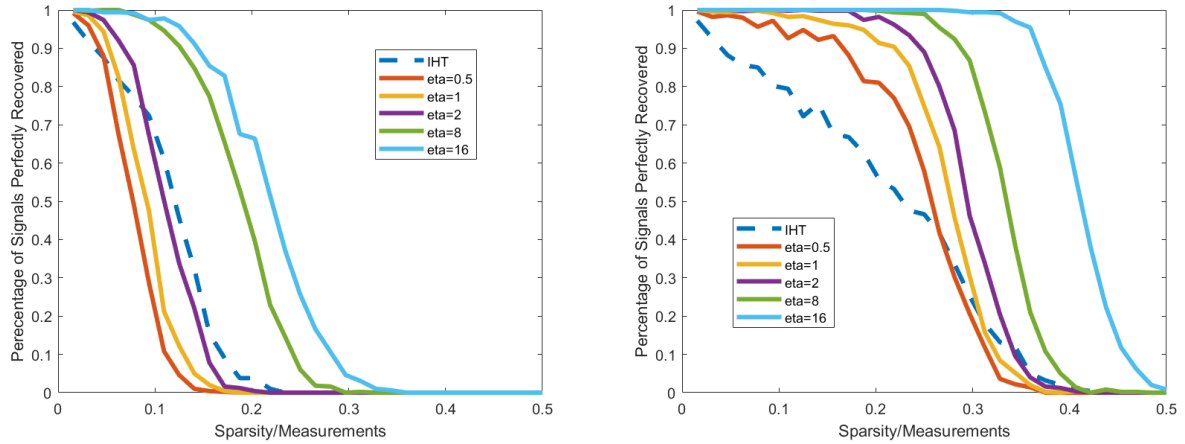
On the left in Figure 4.3a, IHT is given 100 iterations for recovery while ILAT is given 50 iterations for recovery. On the right in Figure 4.3b, the algorithms are given 500 and 250 iterations, respectively. Again, the entirety of Figure 4.3 shows ILAT exceeding the recovery ability of IHT for appropriately chosen values of η.



(a) Error with 30dB SNR

(b) Error with 0dB SNR

Figure 4.2: Reconstruction Error of IHT (dashed) and ILAT (solid, η increasing to the right)



(a) 100 Gradient Computations

(b) 500 Gradient Computations

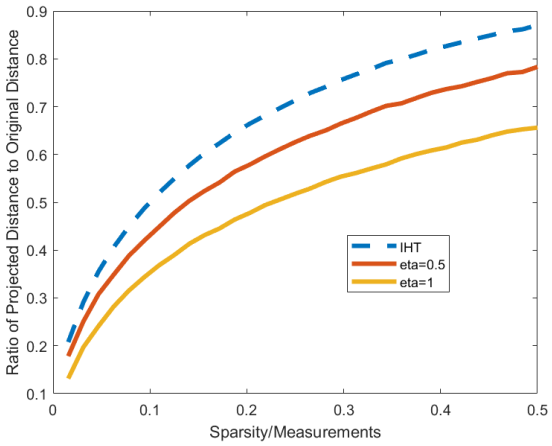
Figure 4.3: Percentage of signals perfectly reconstructed using IHT (dashed) and ILAT (solid, η increasing to the right). Here, IHT is allowed twice as many iterations as ILAT

For our final experiment, we compare look ahead thresholding directly to hard thresholding when they are not simply a step in another algorithm. This lets us directly validate the conclusions of Theorem 4.4.7.

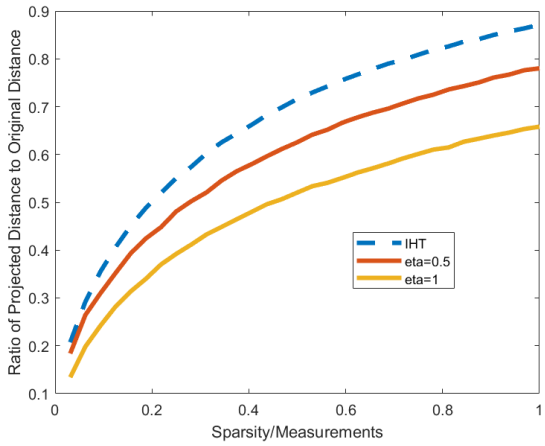
In the following, we generate a random sparse signal x^* where the support is chosen uniformly at random and the entries on that support are i.i.d. $N(0,1)$. Morevoer, we pick a dense vector z from $N(0,I)$ and the matrix A has entries chosen i.i.d. from $N(0,1)$ then normalized to have $\|A\|_{op} = 1$. Then, we measure how close the thresholdings $H_s(z)$ and $H_{s,\eta}(z)$ are to x^*.

In light of the statement of 4.4.7, Figure 4.4 below shows the plots for $\|H_{s,\eta}(z) - x^*\|_2/\|z - x^*\|_2$ and

$\|H_s(z) - x^*\|_2/\|z - x^*\|_2$. Note: here we only use $\eta = 0.5, 1$ because for $\eta = 2, 8, 16$, the performance is indistinguishable from the $\eta = 1$ case. Finally, on the left we use a 128×256 matrix while the right figure uses only a 64×256 measurement matrix to define $H_{s,\eta}$.



(a) Thresholding Error with 128 Measurements



(b) Thresholding Error with 64 Measurements

Figure 4.4: Distance from output $H_s(z)$ (dashed) and $H_{s,\eta}(z)$ (solid, η increasing to the right) to solution x^* divided by distance from z to x^*

4.6 Commentary

The main contribution of this Chapter is the introduction of an alternative thresholding rule to hard thresholding for compressed sensing. Our thresholder uses specific problem instance information, namely the vector of measurements and measurement matrix, to help choose which entries to retain. This stands in stark contrast to hard thresholding which makes no use of such information and, as we have shown, is therefore suboptimal.

Our thresholding rule is meant to act as a tool for use throughout compressed sensing. We investigated in-depth one of these cases by defining Iterative Look Ahead Thresholding, a variation on Iterative Hard Thresholding, which uses our new thresholding rule. We showed that the worst case performance of ILAT is comparable to IHT while the average case performance and experimental results exhibit greatly improved signal recovery. Even better, though our thresholding rule requires extra computational complexity, the amount of additional time is small and we have shown that even when computations are held approximately constant ILAT still excels.

There are still outstanding questions related to look ahead thresholding. First, our theoretical analysis

of Section 4.4 and experimental results of Section 4.5 both suggest improved performance relative to hard thresholding. However, the experimental results are stronger than the theoretical results, i.e. look ahead thresholding works for a much larger range of η values than predicted. Moreover, after explaining this disparity, we would like to understand how to pick the optimal value of η given the ambient dimension, number of measurements, sparsity level, and perhaps a signal model.

In addition to the questions we have about ILAT, we also encourage others to explore the utility of look ahead thresholding in a variety of different CS algorithms. In particular, we are aware that CoSaMP makes critical usage of a thresholding step so perhaps a more accurate thresholding rule could increase the algorithm's accuracy.

BIBLIOGRAPHY

[1] A. Aldroubi, C. Cabrelli, U. Molter, and S. Tang. Dynamical sampling. *Applied and Computational Harmonic Analysis*, 42(3):378 – 401, 2017.

[2] James Blum, Mark Lammers, Alexander M. Powell, and Özgür Yılmaz. Sobolev duals in frame theory and sigma-delta quantization. *Journal of Fourier Analysis and Applications*, 16(3):365–381, Jun 2010.

[3] T. Blumensath and M. E. Davies. Normalized iterative hard thresholding: Guaranteed stability and performance. *IEEE Journal of Selected Topics in Signal Processing*, 4(2):298–309, 2010.

[4] Thomas Blumensath and Mike Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14:629–654, 12 2008.

[5] Thomas Blumensath and Mike Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27:265–274, 11 2009.

[6] Ole Christensen and Marzieh Hasannasab. An open problem concerning operator representations of frames. *arXiv e-prints*, page arXiv:1705.00480, May 2017.

[7] Ole Christensen and Marzieh Hasannasab. Operator representations of frames: Boundedness, duality, and stability. *Integral Equations and Operator Theory*, 88(4):483–499, Aug 2017.

[8] Ole Christensen and Marzieh Hasannasab. Frame properties of systems arising via iterated actions of operators. *Applied and Computational Harmonic Analysis*, 46(3):664 – 673, 2019.

[9] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. BinaryConnect: Training deep neural networks with binary weights during propagations. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3123–3131. Curran Associates, Inc., 2015.

[10] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec 1989.

[11] I. Daubechies, R. DeVore, S. Foucart, B. Hanin, and G. Petrova. Nonlinear approximation and deep (relu) networks. arxiv:1905.02199, 05 2019.

[12] 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.

[13] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, July 2011.

[14] Simon Foucart. Hard thresholding pursuit: An algorithm for compressive sensing. *SIAM Journal on Numerical Analysis*, 49(6):2543–2563, 2011.

[15] Simon Foucart and Holger Rauhut. *A Mathematical Introduction to Compressive Sensing*. Birkhäuser Basel, 2013.

[16] Yunchao Gong, Liu Liu, Ming Yang, and Lubomir D. Bourdev. Compressing deep convolutional networks using vector quantization. arxiv:1412.6115, 2014.

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[18] Vivek Goyal, Jelena Kovaeevi, and Jonathan Kclncr. Quantized frame expansions with erasures. *Applied and Computational Harmonic Analysis*, 10, 12 2000.

[19] C. S. Güntürk, M. Lammers, A. M. Powell, R. Saab, and Ö. Yılmaz. Sobolev duals for random frames and $\sigma\delta$ quantization of compressed sensing measurements. *Foundations of Computational Mathematics*, 13(1):1–36, Feb 2013.

[20] Sepp Hochreiter and Jrgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[21] Lu Hou, Quanming Yao, and James T. Kwok. Loss-aware binarization of deep networks. arxiv:1611.01600, 2017.

[22] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *Journal of Machine Learning Research*, 18(187):1–30, 2018.

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arxiv:1412.6980, 2014.

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[25] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature Cell Biology*, 521(7553):436–444, 5 2015.

[26] Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 598–605. Morgan-Kaufmann, 1990.

[27] Eric Moulines and Francis R. Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 451–459. Curran Associates, Inc., 2011.

[28] D. Needell, N. Srebro, and R. Ward. Stochastic Gradient Descent, Weighted Sampling, and the Randomized Kaczmarz algorithm. arxiv:1310.5715, October 2013.

[29] Deanna Needell and Joel Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Communications of the ACM*, 53, 12 2010.

[30] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014.

[31] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. arxiv:1603.05279, 2016.

[32] Marek Rychlik. On completion of a linearly independent set to a basis with shifts of a fixed vector. *arXiv e-prints*, page arXiv:1905.11812, May 2019.

[33] Jianhong (Jackie). Shen. Least-squares halftoning via human vision system and markov gradient descent (ls-mgd): Algorithm and analysis. *SIAM Review*, 51(3):567–589, 2009.

[34] Z. Shen, H. Yang, and Zhang S. Deep network approximation characterized by number of neurons. *arxiv*, arxiv:1906.05497, 2020.

[35] P. Yin, S. Zhang, J. Lyu, S. Osher, Y. Qi, and J. Xin. Blended coarse gradient descent for full quantization of deep neural networks. *Research in the Mathematical Sciences*, 6(1), 2019.

[36] D.-X. Zhou. Deep distributed convolutional neural networks: universality. *Analysis and Applications*, 16(6):895–919, 2018.

[37] D.-X. Zhou. Universality of deep convolutional neural networks. *Applied and Computational Harmonic Analysis*, 48(2):787–794, 2020.

[38] Shuchang Zhou, Zekun Ni, Xinyu Zhou, He Wen, Yuxin Wu, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. arxiv:1606.06160, 2016.