The Dissertation Committee for Karin Comer Knudson
certifies that this is the approved version of the following dissertation:

Recovery of Continuous Quantities from Discrete and Binary Data with Applications to Neural Data

Committee:

Rachel Ward, Supervisor

Jonathan Pillow, Supervisor

Constantine Caramanis

Peter Mueller

Sujay Sanghavi

Yen-Hsi Richard Tsai

Recovery of Continuous Quantities from Discrete and Binary Data with Applications to Neural Data

by

Karin Comer Knudson, B.A., M.A.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2014

Dedicated to my family, mentors, and teachers.

Acknowledgments

I am deeply grateful to the many people who have worked with me, supported me, and inspired me during my graduate studies, and to whom I owe the opportunity to write this thesis. First and foremost, I thank Jonathan Pillow and Rachel Ward for their guidance and advice, their consistent generosity with their time and attention, and for inspiring me by example to be a better scientist and mathematician. I also thank my fellow graduate students in the UT math department and the members of the NCC lab for many helpful conversations as well as for their sense of fun and the joy of discovery. I thank Rayan Saab, who collaborated in the work on one-bit norm compressive sensing with norm estimation, as well as Jake Yates and Alex Huk, who collaborated with me in applying the spike-sorting methods to neural data. I thank V.J. Uzzell and E.J. Chichilnisky for the retinal data in Chapter 4. I also acknowledge the tremendous professors of the Williams College math department, and my senior thesis advisor, Colin Adams, in particular.

Lastly, I thank my friends and family for their patience and love. I thank my parents and grandparents for all they have taught me and for their unwavering encouragement, kindness, and love; Max for being the best brother I could ever have; and Colin, for lighting up my life.

Recovery of Continuous Quantities from Discrete and Binary Data with Applications to Neural Data

Karin Comer Knudson, Ph.D.
The University of Texas at Austin, 2014

Supervisors: Rachel Ward
Jonathan Pillow

We consider three problems, motivated by questions in computational neuroscience, related to recovering continuous quantities from binary or discrete data or measurements in the context of sparse structure. First, we show that it is possible to recover the norms of sparse vectors given one-bit compressive measurements, and provide associated guarantees. Second, we present a novel algorithm for spike-sorting in neural data, which involves recovering continuous times and amplitudes of events using discrete bases. This method, Continuous Orthogonal Matching Pursuit, builds on algorithms used in compressive sensing. It exploits the sparsity of the signal and proceeds greedily, achieving gains in speed and accuracy over previous methods. Lastly, we present a Bayesian method making use of hierarchical priors for entropy rate estimation from binary sequences.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

We briefly sketch the problems that will be considered, outline the structure of the thesis, and summarize results that appear in the remaining chapters.

The following chapters concern three problems in extracting continuous features of a signal from discrete measurements. We consider estimating the norms of sparse vectors given measurements that are quantized to one-bit (Chapter 2), inferring continuous-valued event times using discrete dictionaries with applications to the problem of *spike-sorting* in processing neural data (Chapter 3), and estimating entropy rate from binary sequences (Chapter 4). All of these problems are motivated at least to some extent by questions in neural coding or the processing of neural data, although they are applicable to non-neural data as well.

Several common themes emerge from our approaches to these problems. In each case, the underlying structure of the problem is nonlinear and dependent on continuous variables, yet we gain traction using certain assumptions of linearity and by discretization. In addition, each of the problems would be intractable or extremely computationally intensive without some prior infor-

mation about the structure of the data, and in each case, careful use of prior information will be of great importance. For example, the one-bit compressive sensing questions considered in Chapter 2 and spike-sorting of Chapter 3 would impossible without assumptions of sparsity. Including prior knowledge is important in reducing the complexity of each problem, and also in reducing the effect of the noise that is a ubiquitous feature of (neural) data.

Conveniently, we generally do have prior knowledge about a data source for a given application, and in particular, for neural data we are aided by the large body of experimental literature and physiological understanding. We incorporate such knowledge in the process of recovering relevant features of a signal in several different ways. For our entropy rate estimation, we proceed by assuming a certain hierarchical structure in the data, and specifying a prior for our probabilistic model. In the one-bit compressed sensing and spike-sorting contexts, we incorporate the assumptions into our recovery algorithm directly, but without explicitly specifying a prior.

Chapter 2 relates to the recently developed area of *one-bit compressive sensing*. Compressive sensing more broadly is applicable in Chapters 2 and 3. Compressive sensing, introduced in [17] and [24], typically involves the recovery of a sparse signal $\boldsymbol{x}$ from linear measurements $\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle$, where the number of measurements is less than the dimensionality of the signal (so that the associated linear system is underdetermined), but still sufficiently large compared to the sparsity of the signal so that recovery is possible via various methods. For example, *basis pursuit* is useful and popular recovery

method for sparse recovery, which consists of solving the convex optimization problem: minimize $\|\mathbf{z}\|_1$ subject to $\mathbf{Az} = \mathbf{y}$. Greedy and thresholding based methods can also be used to recover sparse vectors, and will be relevant to the development in Chapter 4.

Recent research has demonstrated that recovery is also possible when the observed measurements are quantized to $\text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle) \in \{\pm 1\}$; this is the domain of one-bit compressive sensing. Such severe quantization necessarily leads to a loss of information, but has the advantage of being extremely fast and easy to implement in hardware. Since the one-bit measurements give no information on the norm of $\boldsymbol{x}$, recovery methods in one-bit compressive sensing measurements typically assume that $\|\boldsymbol{x}\|_2 = 1$. We show in Chapter 2 that if one allows more generally for quantized *affine* measurements of the form $\text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle + b_i)$, and if such measurements are random, an appropriate choice of the affine shifts b_i allows norm recovery to be easily incorporated into existing methods for one-bit compressive sensing. Alternatively, if one is interested in norm estimation alone, we show that the fraction of measurements quantized to $+1$ (versus -1) can be used to estimate $\|\boldsymbol{x}\|_2$ through a single evaluation of the inverse Gaussian error function, providing a computationally efficient method for norm estimation from 1-bit compressive measurements. All of our recovery guarantees are *universal* over sparse vectors, in the sense that with high probability, one set of measurements will successfully estimate all sparse vectors simultaneously.

Unlike the subsequent chapters, Chapter 2 was not immediately moti-

vated by an explicit question in computational neuroscience. Still, it relates to broader issues of neural coding that are relevant to the remaining chapters. In the nervous system, signals are primarily propagated by neurons experiencing what is called an *action potential*, which is a rapid ($\sim$1 millisecond) event in which the electric potential across the cell membrane of the neuron rises and falls in a consistent pattern. Action potentials in neurons are also called *spikes*, and when a neuron experiences an action potential, we say that the neuron *fires*. We will also discuss *spike trains*, which are the temporal sequences of spikes that a neuron generates. Because the action potential of a neuron proceeds in a consistent way each time the neuron fires, it is usually assumed that information is not carried in the specific features of the action potential, but only in the pattern of firing times and rate of firing of neurons in a population. In particular, when time is discretized, a neuron's activity may be characterized by a binary sequence with ones in each time bin in which the neuron fired, and zeros in each time bin where it does not.

Neural coding therefore necessarily involves continuous quantities (features of natural scenes, color, sound qualities) encoded in binary neural responses (fires or does not fire). Sensory information, such as a visual scene, that neurons encode are also generally approximately sparse in some properly chosen basis. In this sense one-bit compressive sensing, which demonstrates one way in which binary measurements can encode sparse signals, is applicable to neural coding.

Chapter 3 treats sparsity in a different context than Chapter 2, and with

the goal of identifying the spikes of individual neurons from data recorded extracellularly. We consider signals that are a sparse combination of some translated copies of several known waveforms that may be superimposed and combined with noise. We seek to sort out the timings of each of the copies of the waveform that are present in the signal. We are motivated by the *spike-sorting* problem in processing neural data, where arrays of electrodes may be recording a voltage trace that includes firing of multiple neurons, each of which has its own stereotyped waveform. Of interest is not the aggregate voltage itself, but rather the times at which each neuron fires. The shape of a neuron's spike comes mostly from the shape of its dendritic arbor, its distance from the recording device, and its orientation relative to the electrode [35]. One seeks to use these differences in the waveforms of different neurons in order to identify and sort firing events, in the presence superimposed spikes and noise.

A typical approach to spike sorting is to identify candidate spikes first and then use a clustering method to categorize the waveforms by shape and assign them to different neurons. However, this approach introduces systematic errors when different neurons fire simultaneously (so that there are superimposed spikes). More principled might be to directly model the signal as the translated (and possibly scaled) copies of known waveforms, and then seek to recover them by exploiting sparsity. For example, one could use a dictionary of discretely translated copies of each waveform and see which translation best matches the signal. While the signal would be sparse in such a basis if

the timing of each event matched one of the discretely translations exactly, in general this discrete basis cannot represent the continuous time-shifts of the true signal. In [29], the authors use an augmented (but still discrete) dictionary to recover continuous time shifts in a method called Continuous Basis Pursuit, which leverages sparsity and follows a modified ℓ_1 minimization approach motived by basis pursuit. We build on this approach, introducing a different discrete basis that can better represent continuous time shifts, and using a greedy method that speeds up computation time considerably and avoids certain systematic errors of other methods when applied to neural data. Our greedy method, *continuous orthogonal matching pursuit* (COMP), gives improvements in accuracy and speed over other methods when applied to simulated and neural data.

In Chapter 4 we consider entropy rate estimation from binary sequences. Entropy rate is an information theoretic quantity that describes the maximum information that a sequence can be transmitting in bits per second. Entropy rate and related information theoretic quantities are of interest in computational neuroscience as quantifications of the information a neuron or population might be passing along. However, entropy rate by definition depends on infinitely long time dependencies, and there are many difficulties in accurately estimating it from limited data.

We use a hierarchical prior that is a special case of the hierarchical Dirichlet process, which has been used effectively in natural language processing. We model the data as a Markov chain, which must be of sufficient depth

to capture long time dependencies present in the data. Greater depth makes the model parameters more difficult to estimate from limited data; we address this by using the hierarchical prior to link transition probabilities from longer contexts to transition probabilities from shorter contexts. The entropy rate of a Markov chain can be calculated analytically, so we obtain estimates of the entropy rate as well as estimates of the transition probabilities, which may be of interest in their own right. We present a fully Bayesian and empirical Bayesian estimator, and show them to more accurate than existing methods when applied to simulated binary sequences and real neural spike train data.

In Chapter 5 we conclude and discuss directions for future research.

Chapter 2

One-Bit Compressive Sensing With Norm Estimation

2.1 Background

[1] The field of compressive sensing, as introduced in [17], [24], and [13], concerns the approximation of a sparse (or approximately sparse) vector $\boldsymbol{x} \in \mathbb{R}^n$ from linear measurements of the form

$$y_i = \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle, \quad i = 1, 2, \ldots, m.$$

The interesting case is when the linear system is underdetermined, when $m < n$. Although the signal $\boldsymbol{x}$ is in $\mathbf{R}^n$, if it has no more than $s < m$ nonzero entries (that is, if the *sparsity* of $\boldsymbol{x}$ is s), then recovery is still possible under some conditions on s, n, m and $\mathbf{A}$ (where the matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ has rows $\boldsymbol{a}_i$).

Arbitrary measurement matrices $\mathbf{A}$ do not in general provide enough information about the signal, so the choice of $\mathbf{A}$ matters. Moreover, to be of practical use, reconstruction algorithms must be accurate but also computationally tractable: in the noiseless case, there exists a matrix $\mathbf{A} \in \mathbb{R}^{2s \times N}$ such

[1] Material in this chapter is based on work by the author, Rayan Saab, and Rachel Ward [44]

that for any s-sparse $\mathbf{x}$ one can perform exact recovery by finding the sparsest signal for for which 2.1 holds, but this problem is known to be NP-hard. Nevertheless, tractable recovery via several different recovery algorithms can be guaranteed with high probability for classes of random matrices $\mathbf{A}$ given that m is sufficiently large ($m \geq Cs\log(n/s)$) for C a constant independent of n, m, s). In particular, $\mathbf{A}$ can be chosen to have entries that are independent and identically distributed according to a Gaussian or Bernoulli distribution. Moreover, recovery error remains controlled with measurements that are subject to noise and when the signal $\boldsymbol{x}$ is only approximately (rather than exactly) sparse. Compressive sensing has found application in image processing, magnetic resonance imaging (MRI), statistics, computational neuroscience, and machine learning, among other areas (see, e.g. [32] [Chapter 1]).

In this chapter, we focus on the related problem of *one-bit compressive sensing* as introduced in [11]. The goal in one-bit compressive sensing is to approximate a sparse (or almost sparse) vector $\boldsymbol{x} \in \mathbb{R}^n$ from one-bit quantized measurements of the form

$$y_i = \text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle), \quad i = 1, 2, \ldots, m, \tag{2.1}$$

where $\text{sign}(t) = 1$ when $t \geq 0$ and $\text{sign}(t) = -1$ when $t < 0$.

This problem is of practical as well as theoretical interest. In any compressive sensing architecture, measurements y_i must be quantized to a finite number of bits to allow processing using digital computers, so quantization is necessary to consider. In practice a comparator (one-bit quantizer) is easy

to build, fast, and consumes relatively little power, so the extreme quantization of one-bit measurements may even be *preferable* in situations where finer quantization is expensive relative to additional measurements. One-bit measurements can carry added benefits such as robustness to certain nonlinearities in the signal acquisition process (saturation, for example). Additionally, recent research indicates that in some settings, recovery from one-bit measurements may out-perform multi-bit compressed sensing [46].

In [58], Plan and Vershynin provided the first computationally tractable method for reconstructing effectively sparse vectors from one-bit measurements of the form (2.1) via convex optimization (cf. [57]). In this chapter we will use the following result from [58]:

Theorem 1 (Theorem 1.1 of [58])**.** *Let $\boldsymbol{a}_i \in \mathbb{R}^n$, $i = 1, ..., m$, be random vectors with independent and identically distributed standard Gaussian entries and suppose $m > C\delta^{-5} s \log^2(2n/s)$. With probability exceeding $1 - C\exp(-c\delta m)$, the following holds for every $\boldsymbol{x} \in \mathbb{R}^n$ with $\|\boldsymbol{x}\|_2 = 1$ and $\|\boldsymbol{x}\|_1 \leq \sqrt{s}$:*

The solution $\boldsymbol{x}^\sharp$ to the optimization problem

$$\min_{\boldsymbol{x}' \in \mathbb{R}^n} \|\boldsymbol{x}'\|_1 \text{ subject to } \sum_{i=1}^{m} |\langle \boldsymbol{a}_i, \boldsymbol{x}' \rangle| = m \text{ and } \operatorname{sign}\langle \boldsymbol{a}_i, \boldsymbol{x}' \rangle = \operatorname{sign}\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle,\ i \in [m] \tag{2.2}$$

satisfies

$$\left\| \boldsymbol{x} - \frac{\boldsymbol{x}^\sharp}{\|\boldsymbol{x}^\sharp\|_2} \right\|_2 \leq \delta.$$

Above, C and c are universal constants, independent of all other parameters.

Other recovery algorithms have been explored methods outside of convex optimization based approaches, including greedy methods in [10] and [71] and other iterative methods in [11], [39], and [51] and a trust region method in [47]. Fundamental bounds on the best achievable reconstruction error from noiseless one-bit measurements are discussed in [40].

2.2 Outline and Contributions of Chapter

A limitation of Theorem 1, and in prior results treating the one-bit compressive sensing problem ([11], [10], [40], [57], [58], [71]) is that $\|\boldsymbol{x}\|_2 = 1$ must be assumed *a priori* to guarantee any accuracy in the reconstructed solution. If one considers only quantized linear measurements $y_i = \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle$, then such an assumption *must* be made: quantized linear measurements give no information about the magnitude of the underlying vector $\boldsymbol{x}$.

We are concerned with the scenario where the norm of $\boldsymbol{x}$ is not known a priori, and must be estimated along with the direction from one-bit compressive measurements. Because measurements of the form $\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle$ give no information about the norm, we consider the reconstruction of $\boldsymbol{x} \in \mathbb{R}^n$ from more general one-bit measurements of the form

$$y_i = \text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle + b_i), \quad i = 1, 2, \ldots, m, \tag{2.3}$$

where $\boldsymbol{b} = (b_i)_{i=1}^m$ is known (such affine shifts should not incur any additional difficulties in hardware design). We note that the application of affine shifts before quantization was recently shown to produce more accurate reconstruc-

tions in the area of one-bit matrix completion [22], but towards a different purpose. Affine shifts are used for effective recovery from one-bit compressive measurements in [41]; the approach of this work differs from ours in that it uses affine shifts that are chosen adaptively, and recovery results are numerical rather than theoretical. For reconstructing $\boldsymbol{x}$ from the measurements (2.3), we propose two algorithms: an *augmented convex programming approach*, and an *empirical distribution function approach*.

In the augmented convex programming approach, the shifts b_i are standard Gaussian variables and the measurement vectors $\boldsymbol{a}_i$ have i.i.d. standard Gaussian entries. We can then re-write the affine measurements (2.3) as augmented linear measurements

$$y_i = \text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle + b_i) = \text{sign}(\langle \tilde{\boldsymbol{a}}_i, \tilde{\boldsymbol{x}} \rangle),$$

where $\tilde{\boldsymbol{a}}_i = (\boldsymbol{a}_i, b_i)$ and $\tilde{\boldsymbol{x}} \in \mathbb{R}^{n+1}$ is given by $\tilde{\boldsymbol{x}} = (\boldsymbol{x}, 1)$. We use a standard 1-bit compressed sensing recovery method, such as that of [58] (2.2) to give us an estimate $x^\sharp$ of $\tilde{x}$, albeit without magnitude information. Defining $[n] := \{1, ..., n\}$ and denoting by x_T the restriction of x to $T \subset [n]$, we note that if each ratio $x_j^\sharp / \tilde{x}_j$ is roughly the same then the ratio of the norms $\|\boldsymbol{x}_{[n]}^\sharp\|_2 / \|\boldsymbol{x}\|_2$ should be close to the *known* ratio $x_{n+1}^\sharp / \tilde{x}_{n+1} = x_{n+1}^\sharp / 1$. Rearranging, this gives

$$\|\boldsymbol{x}\|_2 \approx \frac{\|\boldsymbol{x}_{[n]}^\sharp\|_2}{x_{n+1}^\sharp}.$$

In Section 2.4.1, we formalize this intuition and derive theoretical guarantees for this method.

For the empirical distribution function approach, on the other hand, the shifts b_i are all set to a common, non-random value τ, and we use a method based on the empirical cumulative distribution function to estimate the norm of $\boldsymbol{x}$. This method is motivated by the observation that $\langle \boldsymbol{a_i}, \boldsymbol{x} \rangle$ is a Gaussian random variable with mean zero and standard deviation of $\|\boldsymbol{x}\|$, which is the quantity we wish to estimate. Thus, the fraction of measurements $\text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle - \tau)$ that are negative should approximate the cumulative distribution function of a $\mathcal{N}(0, \|\boldsymbol{x}\|^2)$ random variable evaluated at τ. The accuracy of the empirical cumulative distribution function (empirical cdf or EDF) is guaranteed by the Dvoretzky-Keifer-Wolfowitz (DKW) Inequality from [26], and we use the value of the empirical cdf at τ to obtain an estimate for $\|\boldsymbol{x}\|_2$. Section 2.4.1 presents our theoretical results on this method.

Both methods assume a known upper bound on the norm of $\boldsymbol{x}$ and the EDF method further assumes a known lower bound on $\|\boldsymbol{x}\|_2$. For each method we present sufficient conditions on m for *universal* sparse signal recovery to hold with high probability (to within a desired accuracy $\delta > 0$). We show that the performance of the augmented convex programming approach scales like $\|\boldsymbol{x} - \boldsymbol{x}^\sharp\|_2 \lesssim 1/m^{1/5}$, similar to the theoretical rate given in [58] in the case where $\|\boldsymbol{x}\|_2 = 1$ is assumed. We show that the EDF method is guaranteed to do at least this well, and in certain regimes even achieves the scaling $\|\boldsymbol{x} - \boldsymbol{x}^\sharp\|_2 \lesssim 1/m^{1/2}$.

We find that empirically, the performance of both methods scales like $\|\boldsymbol{x} - \boldsymbol{x}^\sharp\|_2 \lesssim 1/m$, matching a known lower bound for the performance for one-

bit compressive sensing [40]. The numerical experiments suggest that the EDF method is more sensitive to the choice of parameters such as the lower and upper bounds on $\|\boldsymbol{x}\|_2$. At the same time, for norm estimation alone, the EDF method is much more computationally efficient than solving a convex program, requiring only a single evaluation of the inverse Gaussian error function, and not requiring one to store the measurement matrix $\boldsymbol{A}$ but rather just the proportion of measurements which are quantized to -1 (versus $+1$).

The structure of the chapter is as follows. In Section 2.3 we present necessary lemmas. In Section 2.4.1 and 2.4.2, we present each of the two approaches and associated theoretical guarantees. Finally, in Section 2.5 we include numerical experiments comparing the accuracy of each norm recovery method.

2.3 Preliminaries

Throughout, we use C, c, C_1, etc. to denote absolute constants whose values may change from line to line. For integer n we denote $[n] = \{1, 2, \ldots, n\}$. Vectors are written in bold italics, e.g. $\boldsymbol{x}$, and their coordinates written in plain text so that the ith component of $\boldsymbol{x}$ is x_i.

To prove our main results, we will need a few lemmas. The first lemma is a simple geometric inequality concerning the norm of the difference between two vectors.

Lemma 2. *Consider vectors $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^n$ and positive scalars $t_1, t_2, \alpha, \eta \in \mathbb{R}$*

satisfying

$t_1 \geq \alpha > \eta, \quad \|\boldsymbol{x}_1\|_2^2 + t_1^2 = 1, \quad \|\boldsymbol{x}_2\|_2^2 + t_2^2 \leq 1, \quad \text{and} \quad \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2 + (t_1 - t_2)^2 \leq \eta^2$. *Then*

$$\left\| \frac{\boldsymbol{x}_1}{t_1} - \frac{\boldsymbol{x}_2}{t_2} \right\|_2^2 \leq \frac{4\eta^2}{\alpha^2(\alpha - \eta)^2}$$

Proof. First, define $\varepsilon = 1 - \|\boldsymbol{x}_2\|_2^2 - t_2^2$. Since

$$\sqrt{\|\boldsymbol{x}_1\|_2^2 + t_1^2} - \sqrt{\left(\|\boldsymbol{x}_2\|_2^2 + t_2^2\right)} \leq \sqrt{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2 + (t_1 - t_2)^2} \leq \eta,$$

it follows that $1 - \sqrt{1 - \varepsilon} \leq \eta$, so $(1 - \eta)^2 \leq 1 - \varepsilon$ and finally $\varepsilon \leq 2\eta - \eta^2 \leq 2\eta$. Also by the reverse triangle inequality, $t_2 \geq t_1 - \eta \geq \alpha - \eta$.

Next we note that $\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2 + (t_1 - t_2)^2 \leq \eta^2$ implies

$$\begin{aligned} -2\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle &\leq \eta^2 - (t_1 - t_2)^2 - \|\boldsymbol{x}_1\|_2^2 - \|\boldsymbol{x}_2\|_2^2 \\ &= \eta^2 - (t_1^2 + \|\boldsymbol{x}_1\|_2^2) - (t_2^2 + \|\boldsymbol{x}_2\|_2^2) + 2t_1 t_2 \\ &= \eta^2 - 2 + \varepsilon + 2t_1 t_2 \end{aligned}$$

Now,

$$\begin{aligned} \left\| \frac{\boldsymbol{x}_1}{t_1} - \frac{\boldsymbol{x}_2}{t_2} \right\|_2^2 &= \frac{\|\boldsymbol{x}_1\|^2}{t_1^2} + \frac{\|\boldsymbol{x}_2\|^2}{t_2^2} - 2\frac{\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle}{t_1 t_2} \\ &\leq \frac{1 - t_1^2}{t_1^2} + \frac{1 - \varepsilon - t_2^2}{t_2^2} + \frac{\eta^2 - 2 + \varepsilon + 2t_1 t_2}{t_1 t_2} \\ &= \frac{(t_1 - t_2)^2 - \varepsilon t_1^2 + \eta^2 t_1 t_2 + \varepsilon t_1 t_2}{t_1^2 t_2^2} \\ &= \frac{(t_1 - t_2)^2}{t_1^2 t_2^2} + \frac{\eta^2 t_1 t_2}{t_1^2 t_2^2} + \frac{\varepsilon t_1 (t_2 - t_1)}{t_1^2 t_2^2} \\ &\leq \frac{4\eta^2}{t_1^2 t_2^2} \end{aligned}$$

where in the final inequality we use that $0 < t_1 \leq 1$, $0 < t_2 \leq 1$, $t_2 - t_1 < \eta$, and $\varepsilon < 2\eta$. Over the range $\alpha \leq t_1 \leq 1$, and $t_1 - \eta \leq t_2 < 1$, this expression attains its maximum at $t_1 = \alpha,\ t_2 = \alpha - \eta$. Substituting these values for t_1 and t_2 results in the bound stated in the lemma.

□

The next lemma gives a bound on the variation of a function with an inverse dependence on the Gaussian error function.

Lemma 3. *Let* $\operatorname{erf} : \mathbb{R} \to [-1, 1]$ *be the Gaussian error function, and define* $h : [0, 1] \to \mathbb{R}$ *by*

$$h(u) = \begin{cases} \frac{1}{\operatorname{erf}^{-1}(2u-1)}, & u \in (0, 1) \\ 0, & u \in \{0, 1\} \end{cases}$$

For $\eta > 0$ *and* $a, b \in \left[\frac{1}{2} + \eta, \frac{1}{2}(\operatorname{erf}(1) + 1)\right)$*, we have*

$$|h(a) - h(b)| \leq \left|h'(\tfrac{1}{2} + \eta)(b - a)\right|.$$

Proof. As the derivative of the inverse error function is

$$\frac{d}{du} \operatorname{erf}^{-1}(u) = \frac{1}{2}\sqrt{\pi} \exp\left(\left(\operatorname{erf}^{-1}(u)\right)^2\right),$$

the derivative of h is given by

$$h'(u) = \frac{-\sqrt{\pi} \exp\left([\operatorname{erf}^{-1}(2u - 1)]^2\right)}{[\operatorname{erf}^{-1}(2u - 1)]^2},$$

which is negative and increasing on the interval $\left(\frac{1}{2}, \frac{1}{2}(\operatorname{erf}(1) + 1)\right)$. Thus, for any $\eta > 0$ and a, b in $\left[\frac{1}{2} + \eta, \frac{1}{2}(\operatorname{erf}(1) + 1)\right)$, we have $|h(a) - h(b)| \leq |h'(\frac{1}{2} + \eta)||b - a|$. □

2.4 Main Results

Here we describe and give guarantees for two methods by which the norm of an unknown vector $\boldsymbol{x} \in \mathbb{R}^n$ can estimated, possibly along with the direction, from one-bit compressive measurements $y_i = \text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle + b_i)$. The first method augments the convex program (1) to retrieve norm as well as directional information about the unknown vector, and inherits the error guarantees for that approach. The second method estimates the norm directly from the measured proportion $\#\{i : y_i = -1\}/m$, using the shift $b_i = \tau$ and the Gaussianity of $\boldsymbol{a}_i$ to obtain a consistent estimator for $\|\boldsymbol{x}\|_2$ which is analyzed using the Dvoretzky-Keifer-Wolfowitz inequality. This method is very efficient to implement compared to the convex programming approach, requiring only a single evaluation of the inverse Gaussian error function. At the same time, it is less robust to parameter uncertainty, as can be seen from numerical experiments.

2.4.1 Augmented Convex Programming

Our first main result is a bound on the accuracy of approximating a $\boldsymbol{x} \in \mathbb{R}^n$ with assumed structural constraint $\|\boldsymbol{x}\|_1/\|\boldsymbol{x}\|_2 \leq \sqrt{s}$ from one-bit measurements

$$y_i = \text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle + b_i), \quad i = 1, 2, \ldots, m. \tag{2.4}$$

Note that the constraint $\|\boldsymbol{x}\|_1/\|\boldsymbol{x}\|_2 \leq \sqrt{s}$ describes a class of "approximately" s-sparse vectors. For reconstruction, we consider an augmented version of the convex program (2.2) specified below.

Theorem 4. *For $i = 1, ..., m$, let the random vectors $\boldsymbol{a}_i \in \mathbb{R}^n$ be independent and identically distributed with $\mathcal{N}(0,1)$ entries and let b_i be independent $\mathcal{N}(0,R)$ scalars. If for some specified $\delta < \min\{1, R/2\}$,*

$$m \geq C\delta^{-5}R^5 s \log^2\left(\frac{2n}{s}\right),$$

then with probability exceeding $1 - C\exp\left(-c\frac{\delta m}{R}\right)$ the following holds uniformly for all vectors $\boldsymbol{x} \in \mathbb{R}^n$ with $\|\boldsymbol{x}\|_1 \leq \sqrt{s}\|\boldsymbol{x}\|_2$ and $\|\mathbf{x}\|_2 \leq R$: the solution $(\boldsymbol{x}^\sharp, t^\sharp)$ to the optimization problem

$$\min_{\boldsymbol{z}\in\mathbb{R}^n, u\in\mathbb{R}} \|(\boldsymbol{z}, u)\|_1 \quad \textit{subject to} \sum_{i=1}^m |\langle \boldsymbol{a}_i, \boldsymbol{z}\rangle + ub_i| = m$$

$$\textit{and} \quad \operatorname{sign}(\langle \boldsymbol{a}_i, \boldsymbol{z}\rangle + ub_i) = \operatorname{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle + b_i), \quad i \in [m]$$

satisfies

$$\|R\boldsymbol{x}^\sharp/t^\sharp - \boldsymbol{x}\|_2 \leq 4\sqrt{2}\delta.$$

Proof. Let $\eta = \delta/(R\sqrt{2})$. Having $m \geq C\eta^{-5}s\log^2\left(\frac{2n}{s}\right) \geq C'\eta^{-5}(s+1)\log^2\left(\frac{2(n+1)}{s+1}\right)$ implies by Theorem 1 that

$$\left\| \frac{(\boldsymbol{x}, R)}{\sqrt{\|\boldsymbol{x}\|_2^2 + R^2}} - \frac{(\boldsymbol{x}^\sharp, t^\sharp)}{\sqrt{\|\boldsymbol{x}^\sharp\|_2^2 + t^{\sharp 2}}} \right\|_2 \leq \eta$$

with high probability on the draw of the matrix, uniformly for all $\boldsymbol{x}$ satisfying the assumptions of the theorem. Since $\|\boldsymbol{x}\|_2 \leq R$ by assumption, we choose $\alpha = \frac{1}{\sqrt{2}} \leq t_1 := \frac{R}{\sqrt{\|\boldsymbol{x}\|_2^2+R^2}}$ and apply Lemma 2 with $\boldsymbol{x}_1 = \frac{\boldsymbol{x}}{\sqrt{\|\boldsymbol{x}\|_2^2+R^2}}$, $\boldsymbol{x}_2 = \frac{\boldsymbol{x}^\sharp}{\sqrt{\|\boldsymbol{x}^\sharp\|_2^2+t^{\sharp 2}}}$, and $t_2 = \frac{t^\sharp}{\sqrt{\|\boldsymbol{x}^\sharp\|_2^2+t^{\sharp 2}}}$. With our choice of η this yields

$$\begin{aligned}\|\boldsymbol{x} - R\boldsymbol{x}^\sharp/t^\sharp\|_2^2 &= R^2\|\boldsymbol{x}/R - \boldsymbol{x}^\sharp/t^\sharp\|_2^2 \\ &\leq R^2 \frac{4\delta^2}{R^2(1/\sqrt{2} - \delta/(R\sqrt{2}))^2} \\ &= \frac{8\delta^2}{(1-\delta/R)^2} \\ &\leq 32\delta^2.\end{aligned}$$

To obtain the last inequality above, we used the assumption $\delta < R/2$. It follows that

$$\|\boldsymbol{x} - R\boldsymbol{x}^\sharp/t^\sharp\|_2 \leq 4\sqrt{2}\delta. \tag{2.5}$$

□

A few remarks are in order.

Remark 5 (Alternative scaling of shift)**.** *Recovery in an analogous fashion is also possible if for $\tau \geq \delta$ we sample $b_i \sim \mathcal{N}(0,\tau)$ instead of $b_i \sim \mathcal{N}(0,R)$, as would be necessary if R is unknown a priori. The proof runs similarly, but we now estimate $\boldsymbol{x}$ with $\tau\boldsymbol{x}^\sharp/t^\sharp$ and apply Lemma 1 with $\eta = \frac{\delta}{\sqrt{R^2+\tau^2}} \leq \alpha = \frac{\tau}{\sqrt{R^2+\tau^2}} \leq t_1 = \frac{\tau}{\sqrt{\|\boldsymbol{x}\|_2^2+\tau^2}}$, $\boldsymbol{x}_1 = \frac{\boldsymbol{x}}{\sqrt{\|\boldsymbol{x}\|_2^2+\tau^2}}$, $\boldsymbol{x}_2 = \boldsymbol{x}^\sharp$, and $t_2 = t^\sharp$. Then the error bound (2.5) is replaced with the conclusion that (with high probability) $\|\boldsymbol{x} - \tau\boldsymbol{x}^\sharp/t^\sharp\|_2 \leq \frac{2\delta}{(\tau-\delta)}\sqrt{R^2+\tau^2}$.*

Remark 6 (Tightness)**.** *For fixed n, s, and R, the parameter $\lambda := \delta^{-5}$ plays the role of an oversampling parameter and appears in the rate of decay of the*

reconstruction error as $\|\boldsymbol{x} - R\boldsymbol{x}^\sharp/t^\sharp\|_2 \lesssim \lambda^{-1/5}$. *Compared to the known lower bound of* $\|\boldsymbol{x} - \boldsymbol{x}^\sharp/t^\sharp\|_2 \gtrsim \lambda^{-1}$ *for the 1-bit compressive sensing problem in the case* $\|\boldsymbol{x}\|_2 = 1$ *and* $\|\boldsymbol{x}\|_0 \leq s$, *this rate is suboptimal [40]. On the other hand, this rate matches the error rate achievable using the convex optimization method* (2.2).

Remark 7 (Alternative reconstruction methods)**.** *The above theorem can be easily adapted to alternate reconstruction methods and inherits their associated error decay rates. For example, using the non-uniform recovery method of [57], one obtains an error of* δ *at number of measurements* $m \gtrsim \delta^{-4}R^4 s \log n/s$. *This improves the dependence of the number of measurements on* δ, R, *and* $\log n$ *at the expense of losing the uniform recovery guarantee.*

2.4.2 Estimating $\|\boldsymbol{x}\|_2$ Using the Empirical Distribution Function

In this section, we consider an alternate approach to 1-bit compressive sensing with built-in norm estimation, where now we estimate $\|\boldsymbol{x}\|_2$ given measurements $\boldsymbol{y} = \text{sign}(\mathbf{A}\mathbf{x} - \mathbf{b})$ with constant (non-random) $\boldsymbol{b} = \boldsymbol{\tau} = (\tau, ..., \tau) \in \mathbb{R}^m$ and $\tau \neq 0$. Unlike the previous approach, the method in this section only approximates the norm of $\boldsymbol{x}$, and gives no information about its direction.

We consider m measurement vectors $\boldsymbol{a}_i \in \mathbb{R}^n$ whose entries $a_{i,j}$ are i.i.d. $\mathcal{N}(0, 1)$. Note that $\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle \sim \mathcal{N}(0, \|\boldsymbol{x}\|_2^2)$, and so $\|\boldsymbol{x}\|_2$ is the standard deviation of $\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle$. Since we only have access to the signs of the samples $\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle - \tau$, and not the samples themselves, we cannot simply estimate $\|\boldsymbol{x}\|_2$ via the sample standard deviation of $\{\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle\}_{i=1}^m$. Instead, we will make use of the *empirical*

cumulative distribution function defined by

$$F_m(\tau) := \frac{\sharp\{i : y_i = -1\}}{m}, \tag{2.6}$$

which gives the proportion of the m measurements $\{\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle\}_{i=1}^m$ satisfying $\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle \leq \tau$. As m increases, the random variable $F_m(\tau)$ will approach $F(\tau) = \frac{1}{2}(1+\mathrm{erf}\frac{\tau}{\|\boldsymbol{x}\|_2\sqrt{2}})$, where F is the cumulative distribution function of a Gaussian random variable with mean 0 and variance $\|\boldsymbol{x}\|_2^2$. Indeed, the empirical distribution function $F_m(\tau)$ is a consistent estimator of $F(\tau)$. We note that for $F(\tau) \neq \frac{1}{2}$, we may invert the expression for $F(\tau)$ to get $\|\boldsymbol{x}\|_2 = \frac{\tau}{\sqrt{2}\,\mathrm{erf}^{-1}(2F(\tau)-1)}$, which motivates, as an approximation of $\|\boldsymbol{x}\|_2$, the estimator

$$\Lambda = \Lambda_m(\tau) := \frac{\tau}{\sqrt{2}\,\mathrm{erf}^{-1}(2F_m(\tau) - 1)}. \tag{2.7}$$

To help estimate the accuracy of Λ as an approximation to $\|\boldsymbol{x}\|_2$, we turn to the Dvoretzky-Keifer-Wolfowitz Inequality [26], which gives the following bound on the difference between the cumulative distribution function and empirical cumulative distribution function.

Proposition 8 (Dvoretzky-Keifer-Wolfowitz [26])**.** *Let $X_1, X_2, ..., X_m$ be i.i.d. random variables with cumulative distribution function F, and let F_m be the associated empirical cumulative density function $F_m(\tau) := \frac{1}{m}\sum 1_{X_i \leq \tau}$. Then for any $\gamma > 0$,*

$$\mathrm{Prob}\left(\sup_{\tau} |F_m(\tau) - F(\tau)| > \gamma\right) \leq 2\exp\left(-2m\gamma^2\right).$$

The DKW inequality will allow us to bound the accuracy of $\Lambda_m(\tau)$ as an estimate of $\|\boldsymbol{x}\|_2$.

Lemma 9. *Fix $0 < \delta < 1/5$, and let $\boldsymbol{x} \in \mathbb{R}^n$ be such that $r \leq \|\boldsymbol{x}\|_2 \leq R$ for known positive constants r and R. Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ be a matrix with independent identically distributed $\mathcal{N}(0,1)$ entries. Set $\tau = r$ and compute $F_m(\tau)$ from $\boldsymbol{y} = \operatorname{sign}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{\tau})$ via (2.6). If*

$$m \geq \pi e \frac{R^2}{r^2} \delta^{-2} \log \frac{2}{\varepsilon},$$

then with probability at least $1 - \varepsilon$ it holds that

$$|F(\tau) - F_m(\tau)| < \frac{\delta}{\sqrt{2\pi}} \frac{r}{R}$$

and

$$F(\tau) \text{ and } F_m(\tau) \in \left[\frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{(1-\delta)r}{\sqrt{2}R}\right)\right) \ , \ \frac{1}{2}\Big(1 + \operatorname{erf}(1)\Big)\right].$$

Proof. By Proposition 8, we have for any choice of $\gamma > 0$, that $|F(\tau) - F_m(\tau)| \leq \gamma$ with probability at least $1 - 2\exp(-2m\gamma^2)$. Set $\tau = r$ and note that

$$F(\tau) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{\tau}{\|\boldsymbol{x}\|_2\sqrt{2}}\right)\right) \in \left[\frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{r}{\sqrt{2}R}\right)\right) \ , \ \frac{1}{2}\Big(1 + \operatorname{erf}\left(\frac{1}{\sqrt{2}}\right)\Big)\right].$$

For $\delta < 1$, set

$$\eta = \frac{1}{2}\operatorname{erf}\left(\frac{(1-\delta)r}{\sqrt{2}R}\right)$$

and

$$\gamma = \frac{1}{2}\left(\operatorname{erf}\left(\frac{r}{\sqrt{2}R}\right) - \operatorname{erf}\left(\frac{(1-\delta)r}{\sqrt{2}R}\right)\right).$$

Noting that

$$\frac{d\,\mathrm{erf}(x)}{dx} = \frac{2}{\sqrt{\pi}}\exp\left(-x^2\right), \tag{2.8}$$

we have for $0 \le a \le b$

$$(b-a)\frac{2}{\sqrt{\pi}}\exp\left(-b^2\right)\boldsymbol{x} \le \mathrm{erf}(b) - \mathrm{erf}(a) \le (b-a)\frac{2}{\sqrt{\pi}}\exp\left(-a^2\right). \tag{2.9}$$

Consequently

$$\frac{\delta}{\sqrt{2\pi e}}\frac{r}{R} \le \gamma \le \frac{\delta}{\sqrt{2\pi}}\frac{r}{R}.$$

By the DKW inequality, with probability exceeding

$$1 - 2\exp(-2m\gamma^2) \ge 1 - 2\exp\left(-\frac{\delta^2 r^2}{\pi e R^2}m\right),$$

we have

$$F_m(\tau) \in \left[\frac{1}{2}+\eta,\ \ \frac{1}{2}+\frac{1}{2}\,\mathrm{erf}\left(\frac{1}{\sqrt{2}}\right)+\frac{1}{2}\left(\mathrm{erf}\left(\frac{r}{\sqrt{2}R}\right)-\mathrm{erf}\left(\frac{(1-\delta)r}{\sqrt{2}R}\right)\right)\right].$$

This yields the conclusion of the lemma provided

$$2\gamma = \mathrm{erf}\left(\frac{r}{\sqrt{2}R}\right) - \mathrm{erf}\left(\frac{(1-\delta)r}{\sqrt{2}R}\right) \le \mathrm{erf}\left(1\right) - \mathrm{erf}\left(\frac{1}{\sqrt{2}}\right),$$

which holds when $\delta \le 1/5$, and hence $\gamma \le \frac{\delta r}{\sqrt{2\pi}R} \le \frac{1}{5\sqrt{2\pi}} \le \frac{1}{2}\,\mathrm{erf}\left(1\right) - \mathrm{erf}\left(\frac{1}{\sqrt{2}}\right)$.

Theorem 10. *Fix $0 < \delta < \frac{2\sqrt{e}}{5}R$, and let $\boldsymbol{x} \in \mathbb{R}^n$ be such that $r \le \|\boldsymbol{x}\|_2 \le R$ for known strictly positive constants r and R. Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ be a matrix with independent identically distributed $\mathcal{N}(0,1)$ entries. Set $\tau = r$ and compute $F_m(\tau)$ and $\Lambda = \Lambda_m(\tau)$ from $\boldsymbol{y} = \mathrm{sign}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{\tau})$ via (2.6) and (2.7) respectively. If*

$$m \ge 4\pi e^2 \frac{R^4}{r^2}\delta^{-2}\log\frac{2}{\varepsilon},$$

then with probability at least $1-\varepsilon$ *it holds that*

$$\Big| \|\boldsymbol{x}\|_2 - \Lambda_m(\tau) \Big| \leq \delta.$$

Proof. Define the function $h : [0,1] \to \mathbb{R}$ as in Lemma 3 by

$$h(u) = \begin{cases} \frac{1}{\mathrm{erf}^{-1}(2u-1)}, & u \in (0,1) \\ 0, & u \in \{0,1\} \end{cases}$$

Then $|\|\boldsymbol{x}\|_2 - \Lambda_m(\tau)| = \frac{\tau}{\sqrt{2}} |h(F(\tau)) - h(F_m(\tau))|$. If $F(\tau)$ and $F_m(\tau)$ are in $\big[.5+\eta, \frac{1}{2}(\mathrm{erf}(1)+1)\big)$ for some $\eta > 0$, then by Lemma 3 we have:

$$|\|\boldsymbol{x}\|_2 - \Lambda_m(\tau)| = \frac{\tau}{\sqrt{2}} |h(F(\tau)) - h(F_m(\tau))| \leq \frac{\tau}{\sqrt{2}} |h'(.5+\eta)||F(\tau) - F_m(\tau)|.$$

Indeed, provided $\delta_0 := \frac{1}{2R\sqrt{e}}\delta < 1/5$, by Proposition 9 we have that $|F(\tau) - F_m(\tau)| \leq \gamma := \frac{\delta_0}{\sqrt{2\pi}} \frac{r}{R}$, and that $F_m(\tau)$ and $F(\tau)$ do satisfy the assumptions of Lemma 3 with probability at least $1-\epsilon$. So, using Lemma 3 and the definitions of γ, τ we conclude that

$$\begin{aligned} |\|\boldsymbol{x}\|_2 - \Lambda_m(\tau)| &= \frac{\tau}{\sqrt{2}} |h(F(\tau)) - h(F_m(\tau))| \\ &\leq \frac{r}{\sqrt{2}} \cdot \frac{\delta_0}{\sqrt{2\pi}} \frac{r}{R} \cdot \frac{2\sqrt{\pi} R^2 \exp\Big(\big((1-\delta_0) r / (\sqrt{2} R)\big)^2 \Big)}{(1-\delta_0)^2 r^2} \\ &\leq 2R\sqrt{e}\delta_0 \\ &= \delta. \end{aligned}$$

□

The previous theorem gave a bound for norm estimation for a particular fixed $\boldsymbol{x}$, and we assumed no particular structural constraints on $\boldsymbol{x}$. We now provide a *universal* norm estimation bound akin to Theorem 4 for the class of s-sparse vectors: $\boldsymbol{x} \in \mathbb{R}^n$ satisfying $\|\boldsymbol{x}\|_0 \leq s$, $r \leq \|\boldsymbol{x}\|_2 \leq R$.

Theorem 11. *For $i = 1, \ldots, m$, let the random vectors $\boldsymbol{a}_i \in \mathbb{R}^n$ have i.i.d. $\mathcal{N}(0,1)$ entries. Fix $0 < \delta < 1/5$ and positive constants $r \leq R$. If*

$$m \geq C_1 \frac{R^4}{r^2} \delta^{-2} s \log\left(\frac{Rn}{\delta r s}\right)$$

then with probability exceeding $1 - C_2 \exp(-c_3 m)$,

$$\left| \|\boldsymbol{x}\|_2 - \Lambda_m\left(\frac{3r}{5}\right) \right| \leq 2\delta$$

holds uniformly for all vectors $\boldsymbol{x} \in \mathbb{R}^n$ in the set $\{r \leq \|\boldsymbol{x}\|_2 \leq R\} \cap \{\|\boldsymbol{x}\|_0 \leq s\}$. Here, $\Lambda_m(\frac{3r}{5})$ is the estimator computed from $\boldsymbol{y} = \operatorname{sign}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{\tau})$ via (2.7) with $\tau = \frac{3r}{5}$.

Proof. The idea of the proof is to first show that Theorem 10 holds uniformly over a sufficiently fine net of points contained in the set of s-sparse vectors of bounded norm. We then leverage concentration properties of the matrix $\boldsymbol{A}$ and monotonicity of the function h as defined in Lemma 3 to extend to a bound which holds uniformly for s-sparse vectors with norm bounded by R.

It will be helpful below to define F_m and Λ now as functions of more than one argument: $F_m(\tau, \boldsymbol{z}) := \frac{\#\{i : \langle \boldsymbol{a}_i, \boldsymbol{z} \rangle < \tau\}}{m}$ and $\Lambda_m(\tau, \boldsymbol{z}) := \frac{\tau}{\sqrt{2}\mathrm{erf}^{-1}(2F_m(\tau, \boldsymbol{z}) - 1)}$.

First, consider a finite set of points $\mathcal{Q}$ such that

$$\mathcal{Q} \subseteq \mathcal{S} := \{\boldsymbol{x} \in \mathbb{R}^n : \|x\|_2 \leq R, \|x\|_0 \leq s\},$$

and

$$\min_{\boldsymbol{q}\in\mathcal{Q}} \|\boldsymbol{x}-\boldsymbol{q}\|_2 \le \xi/4 \qquad \text{for each } \boldsymbol{x}\in\mathcal{S}. \tag{2.10}$$

By a well-known result in the literature on covering numbers (see, e.g.,[32][Appendix C.2]) such a set exists. Let B_2^n denote the unit Euclidean ball in $\mathbb{R}^n$. Given a fixed s-dimensional linear subspace T of $\mathbb{R}^n$, there exists a finite set of points $\mathcal{Q}_T$ in $B_2^n \cap T$ such that $\max_{\boldsymbol{x}\in B_2^n\cap T}\min_{q\in\mathcal{Q}_T}\|\boldsymbol{x}-\boldsymbol{q}\|_2 \le \xi/4$, and such that $\#(\mathcal{Q}_T) \le (12/\xi)^s$. Picking such a set of points for each of the $\binom{n}{s}$ $\le (\frac{ne}{s})^s$ s-dimensional linear subspaces T whose union is $\{\boldsymbol{x}\in\mathbb{R}^n : \|\boldsymbol{x}\|_0 \le s\}$, and rescaling, we arrive at a set of points $\mathcal{Q}$ in $\mathcal{S}$ of size $\#(\mathcal{Q}) \le (\frac{ne}{s})^s(12R/\xi)^s$ satisfying (2.10).

Now note that there exists a constant C so that with probability exceeding $1-2\exp(-\frac{\delta^2}{2C}m)$, where $m > 2C\delta^{-2}s\log(n/s)$, the normalized matrix $\frac{1}{\sqrt{m}}\boldsymbol{A}\in\mathbb{R}^{m\times n}$ has the *restricted isometry property* of order $2s$ at level δ, that is,

$$(1-\delta)\|\boldsymbol{x}\|_2 \le \frac{1}{\sqrt{m}}\|\boldsymbol{A}\boldsymbol{x}\|_2 \le (1+\delta)\|\boldsymbol{x}\|_2 \qquad \forall \boldsymbol{x} : \|\boldsymbol{x}\|_0 \le 2s$$

For more details on the restricted isometry property, we refer the reader to [[32], Theorem 9.2].

We now condition on the event that $\boldsymbol{A}$ has the restricted isometry property. Let $\mathcal{Q}' = \{P(\boldsymbol{q}) : \boldsymbol{q}\in\mathcal{Q}\}$ where P projects radially by $P(\boldsymbol{q}) = \boldsymbol{q}\cdot\max\{\frac{5r}{3}, \|\boldsymbol{q}\|_2\}$. For any $\boldsymbol{x}\in\mathcal{S}\cap\{\|\boldsymbol{x}\|_2 \ge \frac{5r}{3}\}$ consider the point $\boldsymbol{q}\in\mathcal{Q}'$ realizing (2.10). Note that $\boldsymbol{x}-\boldsymbol{q}$ is $2s$-sparse for each $\boldsymbol{x},\boldsymbol{q}\in\mathcal{S}$. Let $m^* =$

$\#\{i : |\langle \boldsymbol{a}_i, \boldsymbol{x} - \boldsymbol{q}\rangle|^2 \geq r^2/4\}$.We have then

$$\begin{aligned} m^* r^2/4 &\leq \|\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{q})\|_2^2 \\ &\leq m(1+\delta)^2 \|\boldsymbol{x} - \boldsymbol{q}\|_2^2 \\ &\leq m(1+\delta)^2 \xi^2/16. \end{aligned}$$

It follows that $m^* \leq \frac{\xi^2}{r^2} m$. Moreover,

$$\begin{aligned} m \cdot F_m(r, \boldsymbol{x}) &= \sharp\{i : \langle \boldsymbol{a}_i, \boldsymbol{x}\rangle \leq r\} \\ &\geq \sharp\{i : \langle \boldsymbol{a}_i, \boldsymbol{q}\rangle \leq r/2 \text{ and } \langle \boldsymbol{a}_i, \boldsymbol{x} - \boldsymbol{q}\rangle \leq r/2\} \\ &= m - \sharp\{i : \langle \boldsymbol{a}_i, \boldsymbol{q}\rangle \geq r/2 \text{ or } \langle \boldsymbol{a}_i, \boldsymbol{x} - \boldsymbol{q}\rangle \geq r/2\} \\ &\geq m - \big(m(1 - F_m(r/2, \boldsymbol{q})) + m^*\big) \\ &\geq m \left(F_m(r/2, \boldsymbol{q}) - \frac{\xi^2}{r^2}\right) \end{aligned}$$

So, repeating this calculation for the upper bound we have

$$F_m(r/2, \boldsymbol{q}) - \frac{\xi^2}{r^2} \leq F_m(r, \boldsymbol{x}) \leq F_m(3r/2, \boldsymbol{q}) + \frac{\xi^2}{r^2}.$$

We will now choose ξ small enough to obtain

$$F_m(r/3, \boldsymbol{q}) \leq F_m(r, \boldsymbol{x}) \leq F_m(5r/3, \boldsymbol{q}). \tag{2.11}$$

In particular, for the right-hand side inequality to hold we desire

$$\begin{aligned} \xi^2/r^2 &\leq F_m(5r/3, \boldsymbol{q}) - F_m(3r/2, \boldsymbol{q}) \\ &= \big(F_m(5r/3, \boldsymbol{q}) - F(5r/3, \boldsymbol{q})\big) + \big(F(5r/3, \boldsymbol{q}) - F(3r/2, \boldsymbol{q})\big) \\ &+ \big(F(3r/2, \boldsymbol{q}) - F_m(3r/2, \boldsymbol{q})\big). \end{aligned}$$

Using Lemma 9 to control the first and third terms and (2.9) for the middle term, this is achieved if

$$\xi^2/r^2 \leq \frac{r}{6R\sqrt{2\pi e}} - \frac{\delta}{\sqrt{2\pi}}\frac{19r}{6R}.$$

So if we choose $\delta < c_1 < \frac{1}{19\sqrt{e}}$ small enough for the right hand side to be positive, it suffices to choose $\xi \leq c_2 r\sqrt{r/R}$ for (2.11) to hold. Here $c_2 = \sqrt{\frac{1-19c_1\sqrt{e}}{6\sqrt{2\pi e}}}$ is a sufficiently small constant and is also sufficient to obtain the left hand side of (2.11).

Since $r/3$ and $5r/3$ are both lower bounds for $\|\boldsymbol{q}\|_2$, and since we have invoked Lemma 9, we have by Lemma 3 that $t \to h(t)$ is decreasing over the range $t \in [F_m(r/3, \boldsymbol{q}), F_m(5r/3, \boldsymbol{q})]$. It follows that

$$|\|\boldsymbol{q}\|_2 - \Lambda_m(r, \boldsymbol{x})| \leq \max\left\{|\|\boldsymbol{q}\|_2 - \Lambda_m(r/3, \boldsymbol{q})|\,, |\|\boldsymbol{q}\|_2 - \Lambda_m(5r/3, \boldsymbol{q})|\right\}$$

and hence

$$\begin{aligned}
|\|\boldsymbol{x}\|_2 - \Lambda_m(r, \boldsymbol{x})| &\leq |\|\boldsymbol{x}\|_2 - \|\boldsymbol{q}\|_2| + |\|\boldsymbol{q}\|_2 - \Lambda_m(r, \boldsymbol{x})| \\
&\leq |\|\boldsymbol{x}\|_2 - \|\boldsymbol{q}\|_2| + \max\{|\|\boldsymbol{q}\|_2 - \Lambda_m(r/3, \boldsymbol{q})|\,, \qquad (2.12)\\
&\quad |\|\boldsymbol{q}\|_2 - \Lambda_m(5r/3, \boldsymbol{q})|\} \\
&\leq \xi/4 + 2R\sqrt{e}\delta \\
&\leq 3R\sqrt{e}\cdot\delta. \qquad (2.13)
\end{aligned}$$

The last inequality is obtained by setting $\xi \leq 4R\sqrt{e}\delta$.

To obtain the bound on m and the probability, note that invoking Lemma 9 for each individual q, requires

$$m \geq \pi e \frac{R^2}{r^2}\delta^{-2}\log\left(\frac{2}{\varepsilon}\right)$$

and yields a probability of failure less than ε. By our choice of ξ, $\sharp(\Omega) \leq (\frac{ne}{s})^s C_0^s \delta^{-s} (R/r)^{3s/2}$ for some positive constant C_0, so we pick

$$m \geq C_1 \frac{R^2}{r^2} \delta^{-2} s \log(\frac{R}{r} \frac{n}{s\delta})$$

. Consequently by a union bound, and by accounting for the probability that $\boldsymbol{A}$ satisfies the RIP, our probability of failure is less than $C_2 \exp(-c_3 m)$. The full statement of the theorem follows by rescaling δ and r. $\square$

As noted above, this method of norm estimation does not give us an estimate of the direction $\boldsymbol{x}$ itself; it only yields an estimate of the norm. In order to recover $\boldsymbol{x}$, we could easily combine the estimated norm with an estimate of $\boldsymbol{x}/\|\boldsymbol{x}\|_2$ recovered as in Theorem 1.

Corollary 12. *Let $\boldsymbol{x} \in \mathbb{R}^n$ be such that $0 < r \leq \|\boldsymbol{x}\|_2 \leq R$. Let $\delta > 0$ and choose τ as in Theorem 10. Suppose we have $m = m_1 + m_2$ random Gaussian vectors and we collect m_1 measurements of the form $y_i = \operatorname{sign}(\langle \boldsymbol{a_i}, \boldsymbol{x} \rangle - \tau)$, and m_2 measurements $y_i = \operatorname{sign}(\langle \boldsymbol{a_i}, \boldsymbol{x} \rangle)$. Suppose Λ is calculated from the m_1 measurements as in Theorem 10 and let $\boldsymbol{x}^\sharp$ be the solution to the optimization problem in Theorem 1. If $m_1 \geq C_0 \delta^{-5} R^5 \left(s \log^2(\frac{n}{s}) + \log(C/\varepsilon) \right)$ and $m_2 \geq 4\pi e^2 \frac{R^4}{r^2} \delta^{-2} \log(4/\varepsilon)$, then with probability at least $1 - \varepsilon$ it holds that $\|\Lambda \boldsymbol{x}^\sharp - \boldsymbol{x}\|_2 \leq \delta$.*

Proof. By Theorem 1, we use a convex optimization problem to obtain $\boldsymbol{x}^\sharp$ such that

$$\left\| \boldsymbol{x}^\sharp - \boldsymbol{x}/\|\boldsymbol{x}\|_2 \right\|_2 \leq \frac{\delta}{2} R$$

with probability at least $1-\varepsilon/2$, using only the first $m_1 \geq C\delta^{-5}R^5(s\log(\frac{2n}{s}))$ measurements. With the remaining $m_2 \geq C_{R,r}\delta^{-2}\log(8/\varepsilon)$ measurements, we calculate Λ and have by Theorem 10 that with probability at least $1-\varepsilon/2$, we have $|\|\boldsymbol{x}\|_2 - \Lambda| \leq \frac{\delta}{2}$. Hence, with probability at least $1-\varepsilon \leq (1-\varepsilon/2)^2$, we have

$$\begin{aligned}\|\Lambda\boldsymbol{x}^\sharp - \boldsymbol{x}\|_2 &\leq \left\|\Lambda\boldsymbol{x}^\sharp - \|\boldsymbol{x}\|_2\boldsymbol{x}^\sharp\right\|_2 + \left\|\boldsymbol{x}^\sharp\|\boldsymbol{x}\|_2 - \boldsymbol{x}\right\| \\ &\leq \delta/2 + \delta\|\boldsymbol{x}\|_2/(2R) \\ &\leq \delta.\end{aligned}$$

□

2.5 Numerical Experiments

Here we test the performance of the two proposed methods for 1-bit compressive sensing with norm estimation. In all experiments, we consider s-sparse vectors $\boldsymbol{x} \in \mathbb{R}^n$ with $n = 300$ and $s = 10$ that are constructed by a uniform draw from the set $\mathcal{S} = \{\boldsymbol{x} : r < \|\boldsymbol{x}\|_2 < R, \|\boldsymbol{x}\|_0 < s\}$ for r = 10, R = 20. We estimate $\|\boldsymbol{x}\|_2$ in two ways: (1) using the approximation $\|\hat{\boldsymbol{x}}\|$ produced as in Theorem 4, and (2) using the Gaussian empirical cumulative distribution function (EDF) as in Theorem 10 (Figures 2.1a, 2.2a). The first estimation method is labeled as PV_{aug}, because it precedes by applying an augmented version of the optimization problem (2.2) of Plan and Vershynin [58] as in Theorem 4. In a second set of experiments, we estimate $\boldsymbol{x}$ itself, rather than just its norm $\|\boldsymbol{x}\|_2$, with first $\hat{\boldsymbol{x}}$ as in Theorem 4 (PV_{aug}), and

second by partitioning the measurements into two sets, estimating the norm using one set according to the EDF method described in Theorem 10, and estimating the direction using the remaining measurements, as in Corollary 12. (Figures 2.1b, 2.12b).

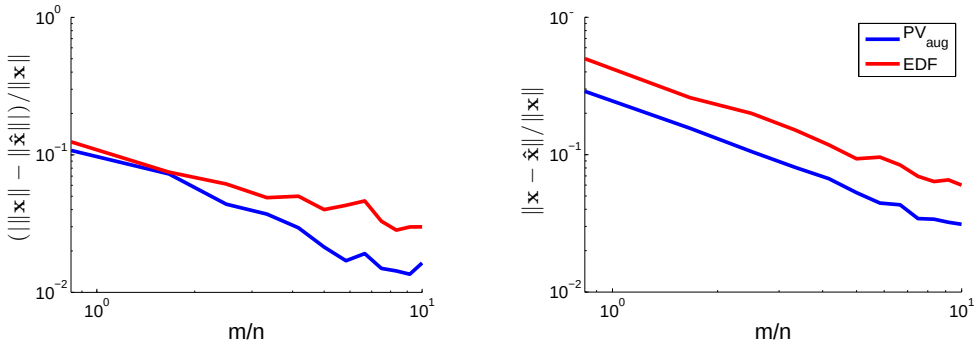

Figure 2.1: Error of the reconstructed norm (left) and reconstructed signal (right) for values of the number of measurements m. Here, τ is held at r. Results are averaged over 40 trials.

In Figure 2.1 we plot recovery error for various values of m/n. Note that the *oversampled* regime $m > n$, while uninteresting in classical compressive sensing setting (with no quantization), is still potentially useful in the one-bit setting, particularly when measurements are fast or cheap relative to finer quantization. For each value of m/n we report the average error over a 40 trials in estimating each of $\boldsymbol{x}$ and $\|\boldsymbol{x}\|_2$. The PV_{aug} method outperforms the EDF method in the plotted regime, at the cost of more computation time.

We also explore the effect of the choice of threshold τ on the accuracy of recovery for both methods. In the EDF method, each measurement is quantized according to whether it is above or below the same threshold τ.

In the PV_{aug} method, we consider for the same parameter τ the thresholds $b_i \sim \mathcal{N}(0, \tau)$. In this case, the expected norm of $\boldsymbol{b}$ equals the norm of the threshold vector $\boldsymbol{\tau} = (\tau, \tau, ..., \tau)$ used in the EDF method.

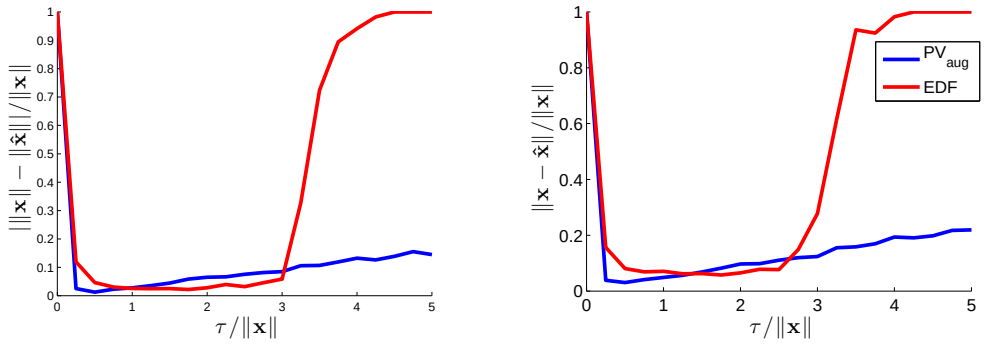

Figure 2.2: Error of the reconstructed norm (left) and reconstructed signal (right), and for values of the thresholding parameter τ relative to $\|\boldsymbol{x}\|_2$. Here m/n=6. Results are averaged over 40 trials.

We expect reconstruction to be poor when τ is too large or too small relative to the true norm of $\boldsymbol{x}$. As τ goes to zero, the proportion of measurements y_i that are -1, $\frac{|\{i:y_i=-1\}|}{m}$ will approach $\frac{1}{2}$ for either fixed τ or random $\mathcal{N}(0.\tau)$ thresholds. On the other hand as τ gets large $\frac{|\{i:y_i=-1\}|}{m}$ will go to 0 for the deterministically thresholded measurements, and to $\frac{1}{2}$ for thresholds $\sim \mathcal{N}(0.\tau)$. The poor performance at these two extremes yields the U-shaped error graphs in Figure 2.2. We find that the EDF method may slightly outperform the PV_{aug} method at the optimal choice of τ, but that PV_{aug} is more robust, its error increasing more gradually as τ is increased away from $\|\boldsymbol{x}\|_2$.

Chapter 3

Continuous Orthogonal Matching Pursuit

3.1 Background

[1] We consider signals that can be represented as the sparse sum of translated and scaled copies of some known waveforms whose timing and amplitudes are of interest. From the aggregate signal one may seek to sort out the identities, amplitudes, and translations of the waveforms of which the signal is composed.

Concretely, we suppose that we have a signal y that can be modeled as:

$$y(t) = \sum_{n=1}^{N_f} \sum_{j=1}^{J} a_{n,j} f_n(t - \tau_{n,j}) \tag{3.1}$$

We assume that the waveforms f_n are known and we seek to recover positive amplitudes $a_{n,j}$ and event times $\tau_{n,j}$. We may additionally assume the presence of an additive noise term $\eta(t)$. Signals of this form have been studied in, for example, [69], [56], [29], and [28].

Of particular interest here is the applicability of this recovery problem to the task of performing *spike-sorting* on extracellular recordings of neural

[1] Material in this chapter is based on work by the author with Jacob Yates, Alex Huk, and Jonathan Pillow to appear in [45]

activity (see also Chapter 1). The voltage trace from an electrode (or array of electrodes) used to measure neural activity in vivo may be recording from a population of neurons, each of which produces many instances of its own stereotyped action potential waveform. The timing of each action potential and which neuron it corresponds to are assumed to be carry the information that is transmitted, and thus must be extracted from the raw data before questions of neural coding can be considered. One would thus like to decompose the analog voltage trace into a list of the timings and amplitudes of action potentials (spikes) for each neuron.

Recovering amplitudes a and times τ for each instance of each waveform from an observed signal y as in (3.1) is difficult in part because of the nonlinearity of the dependence of y on τ. Moreover, in most applications we will not have access to $y(t)$ for arbitrary t but rather have a vector of possibly noisy observations at discrete time points. One way to simplify the problem is to discretize τ, for example by considering only a finite set of evenly spaced time shifts $\tau_{n,j} \in \{\Delta, 2\Delta..., N_\Delta \Delta\}$ and approximating the signal as

$$y \approx \sum_{n=1}^{N_f} \sum_{j=1}^{J} a_{n,j} f_n(t - i_{n,j}\Delta),\ i_{n,j} \in 1, ..., N_\Delta \tag{3.2}$$

Once discretized in this way, the problem is one of sparse recovery: we seek to represent the observed signal with a sparse linear combination of elements of a finite dictionary $\{f_{n,j}(t) : n \in 1, ..., N_f\ ,\ j \in 1, ..., N_\Delta\}$ where $f_{n,j}(t) := f_n(t - j\Delta)$. Framing the problem as sparse recovery, one can then bring tools from compressed sensing to bear. However, the discretization also introduces several

new difficulties. First, with this choice of dictionary, we can only approximate the true translation τ by values on a discrete grid. Choosing Δ becomes a difficult task: small values of Δ allow us to more closely approximate τ, but they correspond to larger dictionaries (thus requiring more computation) and also highly coherent dictionaries (impeding the effectiveness of sparse recovery algorithms).

A previously introduced algorithm that uses techniques of sparse recovery and returns accurate and continuous-valued estimates of a and τ is Continuous Basis Pursuit (CBP) [29], which we describe in Section 3.3. CBP proceeds (roughly speaking) by augmenting the discrete dictionary $f_{n,j}(t)$ with other carefully chosen basis elements, and then solving a convex optimization problem inspired by basis pursuit denoising. Basis pursuit denoising and other convex optimization or ℓ_1 minimization based methods have been effective in the realm of sparse recovery and compressed sensing. However, greedy methods have also been used with great success, and in the regimes considered, and we will draw on the success of these greedy methods in our own approach. We extend ideas introduced in CBP to provide a new method for recovering the desired time shifts τ and amplitudes a that leverages the speed and tractability of solving the discretized problem while still ultimately producing continuous valued estimates of τ, and partially circumventing the problem of too much coherence.

3.2 Outline and Contributions of Chapter

In this chapter we present fast methods for recovering the identities, amplitudes, and translations of waveforms present in signals of the form (3.1). The method proceeds first by greedily identifying rough estimates of the component waveforms and then refining the estimates, moving iteratively between these steps in a process analogous to the well known Orthogonal Matching Pursuit algorithm [67].. We also draw on Continuous Basis Pursuit (CBP), which we extend in several ways: by selecting a subspace that optimally captures translated copies of the waveforms, replacing the convex optimization problem with a greedy framework, and moving to the Fourier domain to more precisely estimate time shifts. We test the resulting method, which we call Continuous Orthogonal Matching Pursuit (COMP) on simulated and neural data, where it shows gains over CBP in both speed and accuracy.

Broadly speaking, our approach has three parts. First, we augment the discretized basis in one of several ways. We draw on [29] for two of these choices, but also present another choice of basis that is in some sense optimal. Second, we greedily select candidate time bins of size Δ in which we suspect an event has occurred. Finally, we move from this rough, discrete-valued estimate of timing τ to continuous-valued estimates of τ and a. We iterate the second and third steps, greedily adding candidate time bins and updating our estimates of τ and a until a stopping criterion is reached.

The structure of the paper is as follows. In Section 3.3 we describe the method of Continuous Basis Pursuit (CBP), which our method builds upon.

In Section 3.4 we develop Continuous Orthogonal Matching Pursuit (COMP). In Section 3.5 we present the performance of our method on real and simulated data.

3.3 Continuous Basis Pursuit

Continuous Basis Pursuit (CBP) ([29], [28], [30] [27]) is a method for recovering the time shifts and amplitudes of waveforms present in a signal of the form (3.1). A key element of CBP is augmenting or replacing the set $\{f_{n,j}(t)\}$ with certain additional dictionary elements, such as translated copies of the waveforms' derivatives. These additional dictionary elements are chosen to smoothly interpolate the one dimensional manifold traced out by $f_{n,j}(t-\tau)$ as τ varies in $(-\Delta/2, \Delta/2)$.

The benefit of a dictionary that is expanded in this way is twofold. First, it increases the ability of the dictionary to represent shifted copies of the waveform $f_n(t-\tau)$ without introducing as much correlation as would be introduced by simply using a finer discretization (decreasing Δ)–an advantage because dictionaries with smaller coherence are generally better suited for sparse recovery techniques. Second, one can move from recovered coefficients in this augmented dictionary to estimates $a_{n,j}$ and *continuous*-valued estimates of $\tau_{n,j}$.

In general there are three ingredients for CBP: basis elements, an interpolator with corresponding function Φ, and a convex constraint set, C. There are K basis elements $\{g_{n,j,k} = g_{n,k}(t-j\Delta)\}_{k=1}^{k=K}$, for each waveform and width-

Δ time bin, which together can be used to linearly interpolate $f_{n,j}(t-\tau), |\tau| < \Delta/2$. The function Φ maps from amplitude a and time shift τ to K-tuples of coefficients $\Phi(a,\tau) = (c_{n,j}^{(1)}, ..., c_{n,j}^{(K)})$, so $af_{n,j}(t-\tau) \approx \sum_{k=1}^{K} c_{n,j}^{(k)} g_{n,j,k}$. The convex constraint set C is for K-tuples of coefficients of $\{g_{n,j,k}\}_{k=1}^{k=K}$ and corresponds to the requirement that $a > 0$ and $|\tau| < \Delta/2$. If the constraint region corresponding to these requirements is not convex (e.g. in the polar basis discussed below), its convex relaxation is used.

As a concrete example, let us first consider the dictionary augmented with shifted copies of each waveform's derivative: $\{f'_{n,j}(t) := f'_n(t - j\Delta)\}$. Assuming f_n is sufficiently smooth, we have from the Taylor expansion that for small τ, $af_{n,j}(t-\tau) \approx af_{n,j}(t) - a\tau f'_{n,j}(t)$. If we recover a representation of y as $c_1 f_{n,j}(t) + c_2 f'_{n,j}(t)$, then we can estimate the amplitude a of the waveform present in y as c_1, the time shift τ as $-c_2/c_1$. Hence, we estimate $y \approx c_1 f_{n,j}(t + c_2/c_1) = c_1 f_n(t - j\Delta + c_2/c_1)$. Note that the estimate of the time shift τ varies *continuously* with c_1, c_2. In contrast, using shifted copies of the waveforms only as a basis would not allow for a time shift estimate off of the grid $\{j\Delta\}_{j=1}^{j=N_\Delta}$.

Of course, once a suitable dictionary is chosen, one must still recover coefficients (i.e. c_1, c_2 in the example above). Motivated by the assumed sparsity of the signal (i.e. y is the sum of relatively few shifted copies of waveforms, and so the coefficients of most dictionary elements will be zero), the CBP algorithm draws on the method of basis pursuit denoising, which has been effective in the compressive sensing setting and elsewhere [65],[19]. In a typical compressive sensing setting, basis pursuit denoising produces the following

estimate of sparse $\mathbf{x}$ given that $\mathbf{x}$ satisfies the underdetermined system $\mathbf{Ax} = \mathbf{y}$:

$$\text{argmin}_{\mathbf{z}} \|\mathbf{Az} - \mathbf{y}\|_2 + \lambda \|\mathbf{z}\|_1.$$

The ℓ_1 penalty encourages sparsity, pushing most of the estimated coefficients to zero and reflecting the belief that $\mathbf{x}$ is sparse, with higher λ encouraging greater sparsity.

Analogously, CBP (with a first order Taylor basis) recovers coefficients using the following convex optimization problem:

$$\begin{aligned} \text{minimize}_{\mathbf{c^{(1)}},\mathbf{c^{(2)}}} & \left\| \sum_{n=1}^{N_f} (\mathbf{F_n c_n^{(1)}} + \mathbf{F'_n c_n^{(2)}}) - \mathbf{y} \right\|_2^2 + \lambda \sum_{n=1}^{N_f} \left\| \mathbf{c}_n^{(1)} \right\|_1 \\ & \text{subject to } c_{n,i}^{(1)} \geq 0 \text{ and } |c_{n,i}^{(2)}| \leq \frac{\Delta}{2} c_{n,i}^{(1)} \ \forall\, n, i \end{aligned} \tag{3.3}$$

Here we denote by $\mathbf{F}$ the matrix with columns $\{f_{n,j}(\mathbf{t})\}$ and $\mathbf{F'}$ the matrix with columns $\{f'_{n,j}(\mathbf{t})\}$. Then, for each (n, j) such that $c_{n,j}^{(1)} \neq 0$, one estimates that there is a waveform in the shape of f_n with amplitude $\hat{a} = c_{n,j}^{(1)}$ and time shift $j\Delta - \hat{\tau} = j\Delta - c_{n,j}^{(2)}/c_{n,j}^{(1)}$ present in the signal. Since the $c_{n,j}^{(1)}$ terms become the estimated amplitudes, the ℓ_1 penalty on these terms (but not on $c_{n,j}^{(2)}$) encourages sparsity. The inequality constraints in the optimization problem ensure first that we only recover positive amplitudes $\hat{a}$, and second that estimates $\hat{\tau}$ satisfy $|\hat{\tau}| < \Delta/2$. Requiring $\hat{\tau}$ to fall in this range keeps the estimated τ in the time bin represented by $f_{n,j}$ and also in the regime where they Taylor approximation to $f_{n,j}(t - \tau)$ is more accurate. Because the objective function and the constraints are convex, (3.3) is a convex optimization problem.

One variant of CBP uses this interpolation based on a first-order Taylor expansion ($K = 2$). However, better results in [29] are obtained for a second order Taylor interpolation($K = 3$), and the best results come from their polar interpolator, which represents each manifold of time-shifted waveforms $f_{n,j}(t-\tau), |\tau| \leq \Delta/2$ as an arc of the circle that is uniquely defined to pass through $f_{n,j}(t), f_{n,j}(t-\Delta/2)$, and $f_{n,j}(t+\Delta/2)$. Letting the radius of the circle on which the arc lies be r, and its angle be 2θ we follow [29] and represent points on this arc by linear combinations of functions w, u, v: $f(t-\tau) \approx w(t) + r\cos(\frac{2\tau}{\Delta}\theta)u(t) + r\sin(\frac{2\tau}{\Delta})v(t)$. The functions w, u and v are themselves linear combinations of $f_{n,j}(t), f_{n,j}(t-\Delta/2)$, and $f_{n,j}(t+\Delta/2)$ given by $[w(t), u(t), v(t)]^T = \mathbf{A}^{-1}[f_{-\frac{\Delta}{2}}, f, f_{-\frac{\Delta}{2}}]^T$ for matrix $\mathbf{A} := \begin{pmatrix} 1 & r\cos(\theta) & -r\sin(\theta) \\ 1 & r & 0 \\ 1 & r\cos(\theta) & r\sin(\theta) \end{pmatrix}$.

The Taylor and polar bases both consist of shifted copies of elements chosen in order to linearly interpolate the curve in function space defined by $f_n(t-\tau)$ as τ varies from $-\Delta/2$ to $\Delta/2$. CBP is defined more generally given other choices of interpolator as well. Let $\mathbf{G}_{n,k}$ be the matrix whose columns are $g_{n,j,k}(\mathbf{t})$ for $j \in 1, ..., N_\Delta$. With basis elements, interpolator, and corresponding convex constraint set C in place, one proceeds to estimate coefficients in the chosen basis by solving:

$$\text{argmin}_{\mathbf{c}} \left\| \mathbf{y} - \sum_{n=1}^{N_f} \sum_{k=1}^{K} \mathbf{G}_{n,k}\mathbf{c}_n^{(k)} \right\|_2^2 + \lambda \| \sum_{n=1}^{N_f} \mathbf{c}_n^{(1)} \|_1 \tag{3.4}$$

$$\text{subject to } (\mathbf{c}_{n,j}^{(1)}, ..., \mathbf{c}_{n,j}^{(K)}) \in C \ \forall (n,j) \tag{3.5}$$

Table 3.1: Basis choices (see also [29], Table 1.)

Interpolator	**Basis Vectors**	$\Phi(a,\tau)$	**C**
Taylor (K=2)	$\{f_{n,j}(\mathbf{t})\}$, $\{f'_{n,j}(\mathbf{t})\}$	$(a, -a\tau)$	$c^{(1)} \geq 0, \lvert c^{(2)}\rvert < c^{(1)}\frac{\Delta}{2}$
Taylor (K=3)	$\{f_{n,j}(\mathbf{t})\}$, $\{f'_{n,j}(\mathbf{t})\}$, $\{f''_{n,j}(\mathbf{t})\}$	$(a, -a\tau, a\frac{\tau^2}{2})$	$c^{(1)}, c^{(3)} > 0$, $\lvert c^{(2)}\rvert < c^{(1)}\frac{\Delta}{2}$ $\lvert c^{(3)}\rvert < c^{(1)}\frac{\Delta^2}{8}$
Polar	$\{\mathbf{w}_{n,j}\}$, $\{\mathbf{u}_{n,j}\}$, $\{\mathbf{v}_{n,j}\}$	$(a, ar\cos(\frac{2\tau}{\Delta}\theta)$, $ar\sin(\frac{2\tau}{\Delta}\theta))$	$c^{(1)} \geq 0$, $rc^{(1)}\cos(\theta) \leq c^{(2)} \leq rc^{(1)}$ $\sqrt{(c^{(2)})^2 + (c^{(3)})^2} \leq rc^{(1)}$
SVD	$\{\mathbf{u}^1_{n,j}\}...\{\mathbf{u}^K_{n,j}\}$.	(Section 3.4.1)	(Section 3.4.1)

One then maps back from each nonzero set of K recovered coefficients $c^{(1)}_{n,j}, ..., c^{(K)}_{n,j}$ to corresponding $\hat{a}_{n,j}$, $\hat{\tau}_{n,j}$ that represent the precise amplitude and timing of the nth waveform present in the jth time bin. This can be done by inverting Φ, if possible, or estimating

$$(\hat{a}_{n,j}, \hat{\tau}_{n,j}) = \text{argmin}_{a,\tau}\|\Phi(a,\tau) - (c^{(1)}_{n,j}, ..., c^{(K)}_{n,j})\|^2_2. \tag{3.6}$$

3.4 Continuous Orthogonal Matching Pursuit

We now present our method for recovery, which makes use of the idea of augmented bases presented above, but differs from CBP in several important ways. First, we introduce a different choice of basis that we find enables more accurate estimates. Second, we make use of greedy methods that iterates

between choosing basis vectors and estimating time shifts and amplitudes, rather than proceeding via a single convex optimization problem as CBP does. Lastly, we introduce an alternative to the step of mapping back from recovered coefficients via Φ that notably improves the accuracy of the recovered time estimates.

Greedy matching pursuit methods such as Orthogonal Matching Pursuit (OMP) [67] (and its variants StOMP[25] and ROMP[53]), Subspace Pursuit [21], and Compressive Sampling Matching Pursuit (CoSaMP)[52] have proven to be both fast and effective in the realm of compressed sensing. Since the number of iterations of these greedy methods tend to go as the sparsity (at least when the algorithms succeed), they tend to be extremely fast when a signal is very sparse. Moreover, the greedy methods we use eliminate the necessity of choosing a regularization constant λ, a choice which can vastly alter the effectiveness of CBP. (We still need to choose constants K and Δ.) Our method is most closely analogous to Orthogonal Matching Pursuit, but recovers continuous valued time estimates, we call it Continuous Orthogonal Matching Pursuit (COMP).

3.4.1 Choice of Finite Basis

We build upon [29] and, choosing as our basis N_Δ shifted copies of a set of K basis vectors for each waveform in such away that these K basis vectors can effectively linearly interpolate $f_n(t-\tau)$ for $|\tau| < \Delta/2$. In our method, as in Continuous Basis Pursuit, these basis vectors allow us to represent continuous

time shifts instead of discrete time shifts, and expand the descriptive power of our dictionary without introducing undue amounts of coherence. While previous work introduced Taylor and polar bases, we obtain the best recovery from a different basis, which we describe now.

The basis comes from a singular value decomposition of a matrix whose columns correspond to discrete points on the curve in function space traced out by $f_{n,j}(t-\tau)$ as we vary τ for $|\tau| < \Delta/2$. Within one time bin of size Δ, consider discretizing further into $N_\delta = \Delta/\delta$ time bins of size $\delta \ll \Delta$. Then, consider the matrix $\mathbf{F}_\delta$ whose columns come from these (slightly) shifted copies of the waveform, so that the i^{th} column of $\mathbf{F}_\delta$ is $f_{n,j}(\mathbf{t} - i\delta + \Delta/2)$ for a discrete vector of time points $\mathbf{t}$. Each column of this matrix is a discrete point on the curve traced out by $f_{n,j}(\mathbf{t}-\tau)$ as τ varies.

In choosing a basis, we seek the best choice of K vectors to use to linearly interpolate this curve. We might instead seek to solve the related problem of finding the best K vectors to represent these finely spaced points on the curve, in which case a clear choice for these K vectors is the first K left singular vectors of $\mathbf{F}_\delta$. This choice is optimal in the sense that the singular value decomposition yields the best rank-K approximation to a matrix. If $\mathbf{F}_\delta = \mathbf{U\Sigma V^T}$ is the singular value decomposition, $\mathbf{u^k}, \mathbf{v^k}$ are the columns of $\mathbf{U}$ and $\mathbf{V}$ respectively, and $s_k := \Sigma_{k,k}$, then $\|\mathbf{F}_\delta - \sum_{k=1}^{K} \mathbf{u}^k s_k (\mathbf{v}^k)^T\| \leq \|\mathbf{F} - \mathbf{A}\|$ for any rank-K matrix $\mathbf{A}$ and any unitarily invariant norm $\|\cdot\|$.

In order to use this SVD basis with CBP or COMP one must specify a convex constraint set in $\mathbb{R}^K$ for the coefficients of this basis. Since $af_{n,j}(t-$

$i\delta) = \sum_{k=1}^{K} a\mathbf{u}^k s_k v_i^k$ a reasonable and simply enforced constraint set would be to assume that the recovered coefficients $c^{(k)}$ corresponding to each basis vector $\mathbf{u}^k$, when divided by $c^{(1)}$ to account for scaling, be between $\min_i s_k v_i^k$ and $\max_i s_k v_i^k$. Or, more restrictively (but also more computationally intensive) we can choose the constraint set to be the convex hull (or an approximation to the convex hull) of the points $\{(s_1 v_i^1, .., s_K v_i^K)\}_{i=1}^{\Delta/\delta}$. Secondly, we need a way to recover a and τ. Again two reasonable choices present themselves: we can $\tau = i\delta$ and a, i to minimize $\sum_{k=1}^{K}(c^{(k)} - a s_k v_i^k)^2$, or we can interpolate to find a function Φ and then find an estimate of a and τ via (3.6).

In Figure 3.1, we compare the error between shifted copies of a sample waveform $f(t-\tau)$ for $|\tau| < 0.5$ and the best (least-squares) approximation of that waveform as a linear combination of $K = 3$ vectors from the Taylor, polar, and SVD bases. Note that the polar basis, as presented in [29] requires $K = 3$, while the Taylor and SVD basis can have $K \in \{2, 3, 4...\}$. The structure of the error as a function of the time shift τ reflects the structure of the bases. The Taylor approximation is chosen to be exactly accurate at $\tau = 0$ while the polar basis is chosen to be precisely accurate at $\tau = 0, \Delta/2, -\Delta/2$. The SVD basis gives the lowest mean error across time shifts.

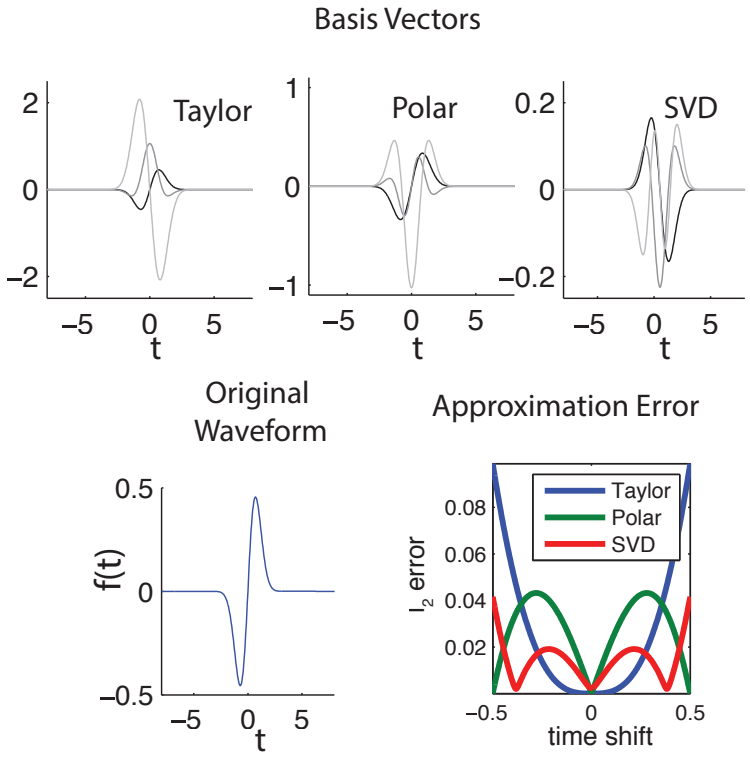

Figure 3.1: Using sample waveform $f(t) \propto t\exp(-t^2)$, we compare the error introduced by approximating $f(t-\tau)$ for varying τ with a linear combination of $K = 3$ basis vectors, from the Taylor, polar or SVD bases. The SVD basis introduces the least error on average over the shift τ. The average errors for the Taylor, polar, and SVD bases are 0.026, 0.027, and 0.014 respectively.

3.4.2 Greedy Recovery

Having chosen our basis, we then greedily recover the time bins in which an occurrence of each waveform appears to be present. We would like to build up a set of pairs (n, j) corresponding to an instance of the n^{th} waveform in the j^{th} time bin. (In our third step, we will refine the estimate within the chosen bins.)

Our greedy method is motivated by Orthogonal Matching Pursuit (OMP), which is used to recover a sparse solution $\mathbf{x}$ from measurements $\mathbf{y} = \mathbf{A}\mathbf{x}$. In OMP [67], one greedily adds a single dictionary element to an estimated support set S at each iteration, and then projects orthogonally to adjust the coefficients of all chosen dictionary elements. After initializing $S = \emptyset, \mathbf{x} = 0$, one repeats the following until a stopping criterion is reached:

$$\mathbf{r} = \mathbf{y} - \mathbf{A}\mathbf{x}$$

$$j = \text{argmax}_j\{|\langle \mathbf{a}_j, \mathbf{r}\rangle| \; s.t. \; j \in \{1, ...J\}\backslash S\}$$

$$S = S \cup \{j\}$$

$$x = \text{argmin}_{\mathbf{z}}\{||\mathbf{y} - \mathbf{A}\mathbf{z}||_2 \; s.t. \; z_i = 0 \; \forall \, i \notin S\}$$

The stopping criterion could be based on the ℓ_2 norm of the residual, or the number of indices already added to the support set, for example.

We adjust this method to choose at each step not a single additional element but rather a set of K associated basis vectors. S is again initialized to be empty, but at each step we add a time-bin/waveform pair (n, j), which

is associated with K basis vectors. In this way, we are adding K vectors at each step, instead of one as in OMP. We greedily add the next index (n, j) according to:

$$(n, j) = \text{argmin}_{m,i} \left\{ \min_{\mathbf{c}_{m,i}} \{ \| \sum_{i=1}^{k} c_{m,i}^{(k)} \mathbf{g}_{m,i}^{(k)} - \mathbf{r} \|_2^2 \; s.t. \, \mathbf{c}_{m,i} \in C \} \, , \, (m, i) \in S^c \right\} \tag{3.7}$$

Here $\{\mathbf{g}_{m,i}^{(k)}\}$ are the chosen basis vectors (Taylor, polar, or SVD), and C is the corresponding constraint set, as in Sections 3.3 and 3.4.1. We proceed by adding the chosen (n, j) pair to our set S.

In comparison with the greedy step in OMP, we note that choosing j as in (3.7) is certainly more costly, because we need to perform a constrained optimization over a K dimensional space for each n, j. If it were necessary to repeat this optimization for each of the $N_f \cdot N_\Delta$ possible indices each time we added an index, the algorithm would be slow. However, if we assume waves are localized in time, we need only update the results of the constrained optimization locally. That is, when we update the residual $\mathbf{r}$ by subtracting the newly identified waveform n in the j^{th} bin, the residual will only change in the bins at and possibly neighboring the jth bin, and so we need only update the quantity $\min_{\mathbf{c}_{n,j'}} \{ \| \sum_{i=1}^{k} c_{n,j'}^{(k)} \mathbf{g}_{n,j'}^{(k)} - \mathbf{r} \|_2^2 \, s.t. \, \mathbf{c}_{n,j'} \in C \, , \}$ for j' neighboring j (for all n).

We remark briefly on the choice of stopping criterion. In the event that the sparsity (i.e. the number of waveforms present in the signal) is known, one can simply continue adding (sets of K) basis vectors until the proper number

of waveforms have been identified. If the sparsity is unknown, as is the case in spike-sorting, we must choose a different stopping criterion. We could for example, stop when the ℓ_2 norm of the residual drops below a certain threshold, or when the change in ℓ_2 norm of the residual between successive iterations falls below a certain threshold. In practice below below we will choose our stopping criterion as follows.

We assume our signal is noisy, and model our observation y as:

$$y(t) = \sum_{n=1}^{N_f} \sum_{j=1}^{J} a_{n,j} f_n(t - \tau_{n,j}) + \eta(t) \tag{3.8}$$

evaluated at discrete time points, further that η is Gaussian and independent at each of the observed time points, with mean 0 and variance σ^2. Let us further assume that a priori, the probability of a waveform occurring in each bin is distributed as an independent Bernoulli trial with probability of success p. We assume that the prior probability of $(\tau, \mathbf{a})$ depends only on the length of each of these vectors (i.e. the number of waveforms that have been identified), so that $p(\tau, \mathbf{a}) = p^{|\mathbf{a}|}(1-p)^{N_\Delta - |\mathbf{a}|}$ where here $|\mathbf{a}| = |\tau|$ is the length of the vector of amplitudes of identified waveforms $\mathbf{a}$. Then comparing the logarithm of the posterior probabilities $\log p(\mathbf{a}, \tau|y) = \log p(y|\mathbf{a}, \tau) + \log p(\mathbf{a}, \tau) - \log p(y)$, and noting that at each iteration $|\mathbf{a}_{new}| = |\mathbf{a}_{old}|+1$, we have that $\log p(\tau_{new}, \mathbf{a}_{new}) - \log p(\tau_{old}, \mathbf{a}_{old})$ is equal to

$$\frac{1}{2\sigma^2}(\|\boldsymbol{r}_{old}\|_2^2 - \|\boldsymbol{r}_{new}\|_2^2) + \log(p) - \log(1-p). \tag{3.9}$$

Here $\boldsymbol{r}_{new}$ is the current residual and $\boldsymbol{r}_{old}$ is the residual after the previous iteration. We then continue while the posterior probability is increasing from

one iteration to the next, that is, while (3.9) is positive. It is straightforward to incorporate certain additional prior information into the stopping criterion if desired; for example, a prior over the amplitudes.

3.4.3 Estimating Time Shifts

Having greedily added a new waveform/timebin index pair (n, j), we next define our update step, which will correspond to the orthogonal projection in OMP. We present two alternatives, one of which most closely mirrors the corresponding step in OMP, the other of which works within the Fourier domain to obtain more accurate recovery.

To most closely follow the steps of OMP, at each iteration after updating S we update coefficients $\mathbf{c}$ according to:

$$\text{argmin}_c \left\| \sum_{(n,j)\in S} \sum_{k=1}^{K} c_{n,j}^{(k)} \mathbf{g}_{n,j}^{(k)} - \mathbf{y} \right\|_2^2 \text{ subject to } \mathbf{c}_{n,j} \in C \,\forall\, (n,j) \in S \qquad (3.10)$$

We alternate between the greedily updating S via (3.7), and updating $\mathbf{c}$ as in (3.10), at each iteration finding the new residual $\mathbf{r} = \sum_{(n,j)\in S} \sum_{k=1}^{K} c_{n,j}^{(k)} \mathbf{g}_{n,j}^{(k)} - \mathbf{y}$ until the ℓ_2 stopping criterion is reached. At this point, one maps back from the recovered coefficients $\{\mathbf{c}_{n,j}\}_{(n,j)\in S}$ to $\{a_{(n,j)}, \tau_{(n,j)}\}_{(n,j)\in S}$ as described in Section 3.3.

Alternatively we may replace the orthogonal projection step with a more accurate recovery of spike timings that involves working with the discrete Fourier transform of the signal. For this, we make use of the following property of the Fourier transform with respect to translation: $(f(t-\tau))^\wedge = e^{2\pi i \tau} \hat{f}$.

This allows us to estimate a, τ directly via:

$$\text{argmin}_{a,\tau} \|(\sum_{n,j \in S} a_{n,j} e^{2\pi i \boldsymbol{\omega} \tau_{n,j}} \hat{f}_{n,j}(\boldsymbol{\omega})) - \hat{y}(\boldsymbol{\omega})\|_2 \text{ subject to } |\tau_{n,j}| < \frac{\Delta}{2} (n,j) \in S \tag{3.11}$$

This is a nonlinear and non-convex constrained optimization problem. However, it can be solved reasonably quickly using, for example, trust region methods. The search space is dramatically reduced because τ has only $|S|$ entries, each of which are constrained to be small in absolute value. By searching directly for a, τ as in (3.11) we sacrifice convexity, but with the benefit of eliminating from this step the error of interpolation, i.e. the error introduced as we map back from $\mathbf{c}$ to a, τ using Φ^{-1} or a least squares estimation (recalling that our signal is only approximately equal to linear combinations of the basis functions with approximation error as in Figure 3.1),.

We note that if one can define $y(t)$ and $f_n(t-\tau)$ for any value of t and τ in the range of interest (for example by using some interpolation method), then one could also proceed by searching directly in the original time domain for a and τ that minimize the ℓ_2 error. In fact, such search using trigonometrical polynomials for the interpolation is equivalent to recovering based on the error in the frequency space as defined above.

Lastly, we observe that while we have bounded τ, it is easy and often helpful to add inequality constraints to a as well, for example requiring a to be in some interval around one, and we do impose this in our spike-sorting simulations and analysis in Section 3.5. Such a requirement effectively imposes

a uniform prior on a over the chosen interval.

3.5 Application to Data

We apply COMP and CBP for each choice of basis to simulated data and compare results. We then present a preliminary application of COMP and CBP to neural data and show how COMP is robust to a certain pathology of CBP-based spike sorting. In this section, COMP denotes the greedy method that includes direct estimation of a and τ during the update set as in (3.11). The convex optimization for CBP is implemented using the cvx package for MATLAB [38], [36].

3.5.1 Simulated Data

We simulate a signal y as the sum of time-shifted copies of two sample waveforms $f_1(t) \propto t\exp(-t^2)$ and $f_2(t) \propto e^{-t^4/16} - e^{-t^2}$ (Figure 3.2a). There are $s_1 = s_2 = 5$ shifted copies of f_1 and f_2, respectively. The time shifts are independently generated for each of the two waveforms using a Poisson process (which is then trimmed to maintain $s_1 = s_2 = 5$), and independent Guassian noise of variance σ^2 is added at each time point. Figure 3.2b shows an example of the type of noise-free signal ($\sigma = 0$), and noisy signal ($\sigma = .2$) on which each recovery method will be run.

We run CBP with the Taylor and polar bases, but also with our SVD basis, and we run COMP with all three choices of basis. We found $\lambda = .1$ to be an effective regularization parameter for CBP. One can also scale λ with

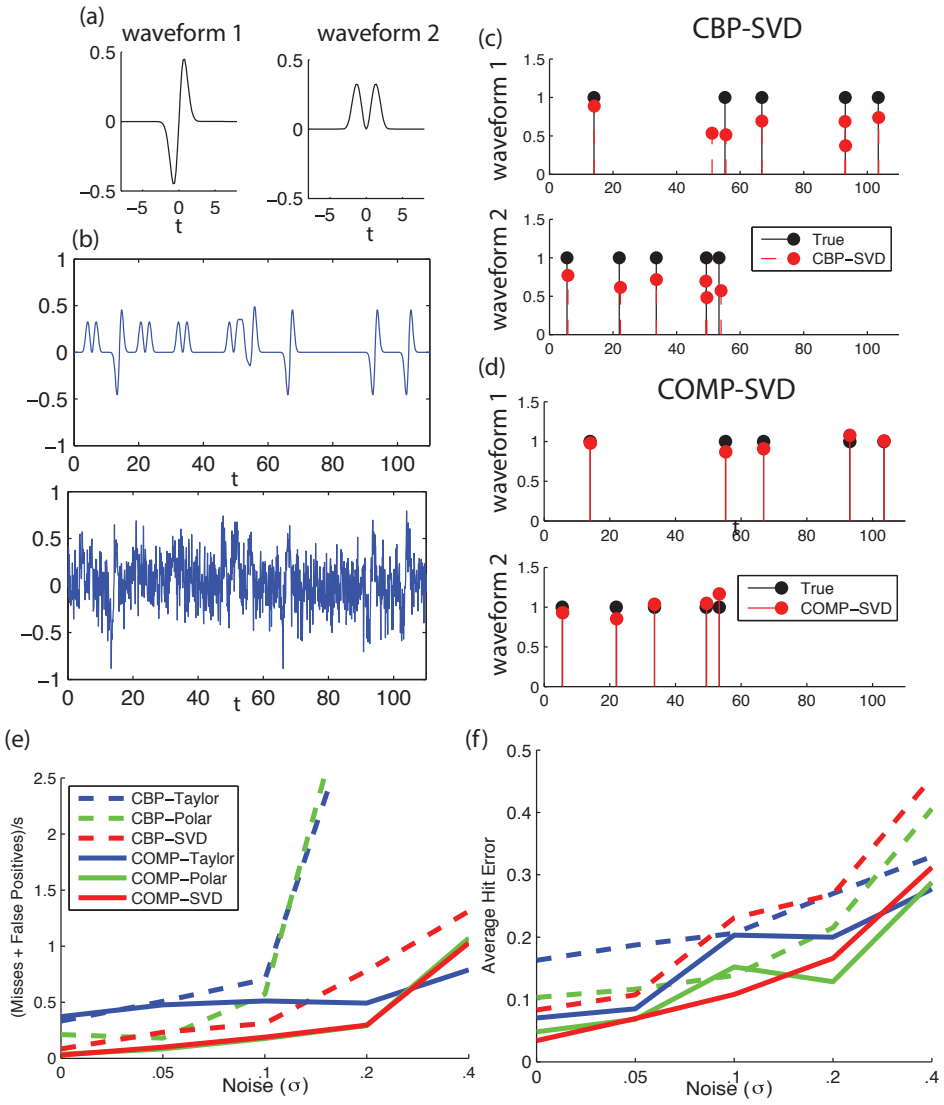

Figure 3.2: (a) Waveforms present in the signal. (b) A noise free example signal (top) and noisy signal (bottom) with $\sigma = .2$. (c) Recovery using CBP. (d) Recovery using COMP. (e) For each recovery method over different values of the standard deviation of the noise σ, misses plus false positives, divided by the total number of events present, $s = s_1 + s_2$. (f) Average distance between the true and estimated spike for each hit.

the known σ^2, but found this did not seem to result in improvements over the data and range of σ considered. Since we did not incorporate knowledge of σ into our choice of λ, we also did not incorporate it into our stopping criterion for COMP and instead used a bound on ℓ_2 norm of the difference in residuals from one iteration to the next (rather than incorporate additional information as in (3.9)). Moreover, since the Fourier update step of COMP imposes a lower bound on a, we also impose a thresholding step after recovery with CBP, discarding any recovered waveforms with amplitude less than .3. We find this thresholding generally improved the performance of the CBP algorithm by pruning false positives. In all results below, we use $K = 3$, since the polar basis requires 3 basis vectors per bin.

We categorize hits, false positive and misses based on whether a time shift estimate is within a threshold of $\epsilon = 1$ of the true value. The "average hit error" of Figure 3.2f, 3.3b is the average distance between the true and estimated event time for each estimate that is categorized as a hit. Results are averaged over 20 trials.

We compare CBP and COMP over different parameter regimes, varying the noise (σ) and the bin size (Δ). Figures 3.2e and 3.3a show misses plus false positives for each method, normalized by the total number of events present $s = s_1 + s_2$. Figures 3.2f and 3.3b show average distance between the true and estimated spike for each estimate categorized as a hit. The best performance across by both measures across nearly all parameter regimes considered is achieved by COMP using the SVD basis. COMP is more robust

to noise (Figure 3.2f), and also to increases in bin width Δ. (Since both algorithms increase in speed with higher Δ, robustness with respect to Δ is an advantage.) We also note a significant increase in CBP's robustness to noise when we implement it with our SVD basis rather than with the Taylor or polar basis (Figure 3.2e).

A significant advantage of COMP over CBP is its speed. In Figure 3.3c we compare the speed of COMP (solid) and CBP (dashed) algorithms for each basis. COMP yields vast gains in speed. The comparison is especially dramatic for small bin size Δ, where results are most accurate across all methods.

3.5.2 Neural Data

We now present recovery of spike times and sortings from neural data. Recordings were made using glass-coated tungsten electrodes in the lateral intraparietal sulcus (LIP) of a macaque monkey performing a motion discrimination task. In addition to demonstrating potential of COMP to bring improvements to the realm of spike-sorting, this section also demonstrates the resistance of COMP to a certain kind of error that recovery via CBP can systematically commit, and which is relevant to neural data.

Spike sorting using CBP may turn out to be unsatisfactory on this data because one waveform resembles a scaled copy of another waveform (Figure 3.4a). The first waveform (in blue) is similar in shape but larger in amplitude that the second waveform (in red). This is problematic for CBP or any other ℓ_1 minimization based method that penalizes large amplitudes. When the second

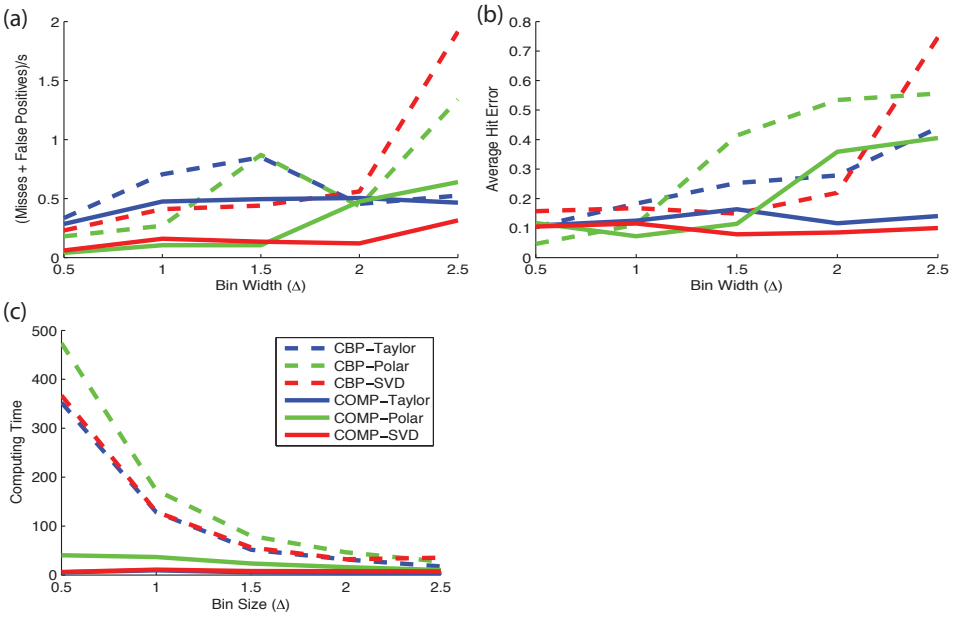

Figure 3.3: (a) Misses plus false positives, divided by the total number of events present, $s = s_1 + s_2$ over different values of bin width Δ. (b) Average distance between the true and estimated spike for each hit for each recovery method. (c) Run time for COMP (solid) and CBP (dashed) for each basis.

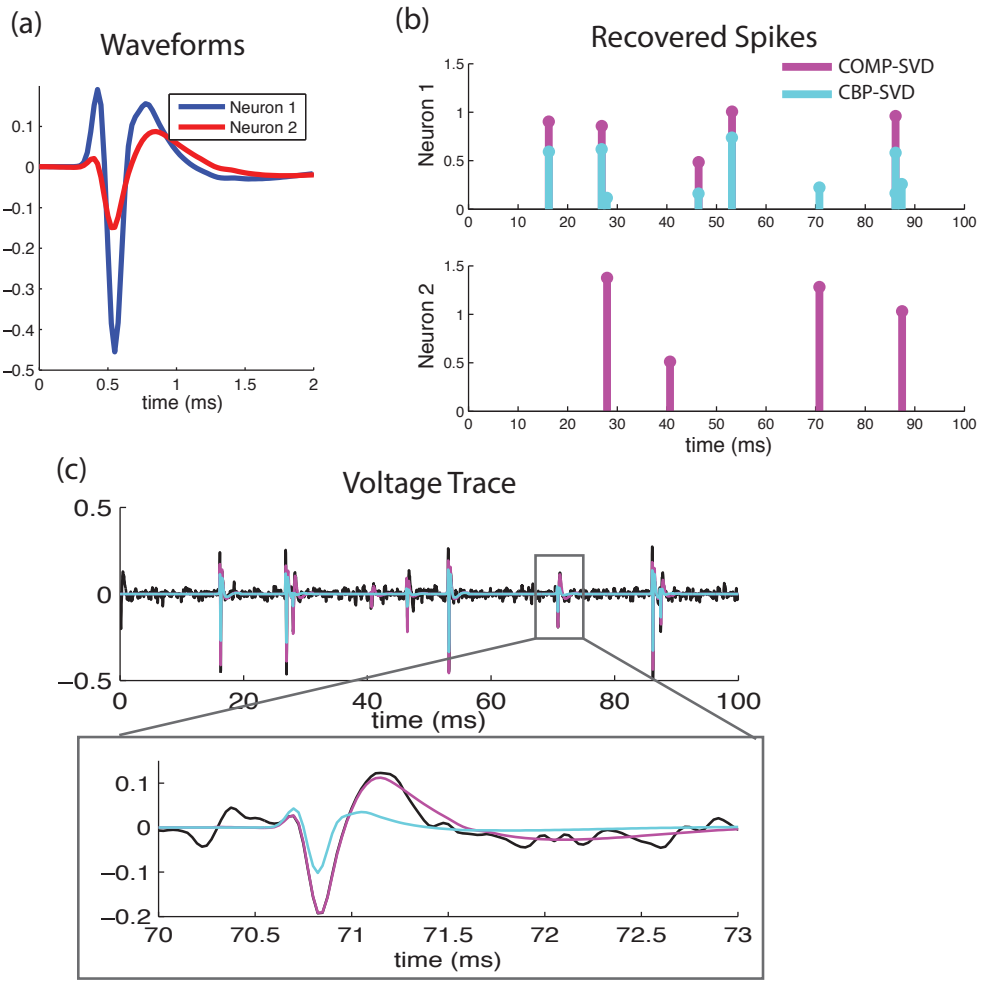

Figure 3.4: (a) Two neural waveforms; each is close to as scaled copy of the other (b) Recovery of spikes via COMP (magenta) and CBP (cyan) using the SVD basis. CBP tends to recover small-amplitude instances of waveform one where COMP recovers large amplitude instances of waveform two (c) Top: recovered traces. Lower panel: zooming in on an area of disagreement between COMP and CBP. The large-ampltude copy of waveform two more closely matches the trace.

waveform is present with an amplitude of one, for example, CBP is likely to incorrectly add a lower-amplitude copy of the first waveform to reduce the penalty on amplitudes, instead of correctly choosing the larger copy of the second waveform; the amplitude penalty for choosing the correct waveform can outweigh the higher ℓ_2 error caused by including the incorrect waveform. The limitation of CBP in the face of waveforms that are close to scaled copies of each other has been observed in [28].

In fact, incorrect assignment by CBP is exactly what we observe, at least for relatively high values of λ (Figure 3.4). We see that CBP tends to report small-amplitude copies of the first waveform where COMP reports large-amplitude copies of the second waveform. Although there is no ground truth here, the closer match of the recovered signal to the data on the waveforms for which the two methods conflict in their identification (Figure 3.4c) indicates that the waveform identities and amplitudes identified via COMP better explain the observed signal.

We note that CBP's mistaken recovery of spurious small-amplitude copies of the first neuron instead of copies of the second neuron depends on the weight of the amplitude penalty, determined by λ. The phenomenon will be observed with relatively high λ values (here $\lambda = .3$), but decreasing λ allows a better explanation of residual to more easily outweigh the choice of higher amplitude. On the other hand, lowering λ can lead to the recovery of many spurious smaller amplitude copies of spikes, which must be trimmed via an appropriate threshold during post-processing. (See [30] for a discussion of

choosing such a threshold for use with CBP).

As noted in Chapter 1, a typical approach to spike sorting would be to first pick out candidate spikes from the data, then sort them by applying some dimensionality reduction and then a clustering algorithm to the observed waveforms. However, as discussed in [56] [28], this clustering does not deal effectively with superimposed spikes. When faced with a waveform that is in fact the the sum of spikes from two different neurons, it typically either dismisses the event as an outlier, or includes it as an instance of some nonexistent third neuron. Proceedures like COMP, CBP, or Binary Pursuit [56] that explicitly model the signal as in (3.1) allow for a principled treatment of superimposed spikes (quite important if correlations between spike times are of interest, for example, as is frequently the case). COMP combines a principled approach to the signal of interest with computational speed and the ability to accurately recover continuous valued times of waveforms, even in the presence of similarly-shaped waveforms. Our preliminary applications to data indicate that COMP could be very useful in practical implementations of spike-sorting.

Chapter 4

Spike-train Entropy Rate Estimation Using Hierarchical Dirichlet Process Priors

4.1 Outline and Contributions of Chapter

[1] The neural and simulated data considered in this chapter will be binary sequences that in the neural context represent the spike train of a neuron, where each symbol represents either the presence of a spike in a bin (1) or the absence of a spike (0). (In the case of neural data, we assume that the bins are sufficiently small so that there is not more than one spike in a bin.) We view the data as a sample path from an underlying stochastic process. From the finite binary sequence, we seek to compute the *entropy rate* of the process, which is an information theoretic quantity that quantifies average uncertainty of each new symbol (0 or 1) given the previous symbols - or the amount of new information per symbol.

Information theory is useful in analyzing neural data in part because it provides quantifications of the amount of information transmitted by a signal without reference to assumptions about how the information is represented or used. The entropy rate is of interest as a measure of uncertainty per unit time,

[1] Material in this chapter is based on published work by the author and Jonathan Pillow in [43]

an upper bound on the rate of information transmission, and an intermediate step in computing mutual information rate between stimulus and neural response. However, accurate entropy rate estimation is difficult, and estimates from limited amounts of data are often severely biased. We present a Bayesian method for estimating entropy rates from binary data that uses hierarchical Dirichlet process priors (HDP) to reduce this bias.

Our method proceeds by modeling the source of the data as a Markov chain, and then using the fact that the entropy rate of a Markov chain is a deterministic function of its transition probabilities. We thus obtain both an approximation of the underlying stochastic process as a Markov chain, and an estimate of the entropy rate of the process. Fitting the model yields parameters relevant to two different of interest about the statistical properties of a spiking neuron: (1) what kind of time dependencies are present? and (2) how much information is the neuron transmitting?

We give a Bayes least squares and empirical Bayesian entropy rate estimators for binary spike trains using hierarchical Dirichlet process (HDP) priors. Our estimator leverages the fact that the entropy rate of an ergodic Markov Chain with known transition probabilities can be calculated analytically, and many stochastic processes that are non-Markovian can still be well approximated by Markov processes of sufficient depth. Choosing the appropriate depth of Markov model presents challenges due to possibly long time dependencies and short data sequences: a deeper model can better account for long time dependencies, but is more difficult to infer from limited data.

Our approach mitigates this difficulty by using a hierarchical prior to share statistical power across Markov chains of different depths. For binary data, the HDP reduces to a hierarchy of beta priors, where the prior distribution over each parameter representing the probability of the next symbol given a long history, is a beta distribution centered on the probability of that symbol given a truncated, one-symbol-shorter, history. The posterior over symbols given a certain history is thus "smoothed" by the probability over symbols given a shorter history. This smoothing is a key feature of the model.

We demonstrate the performance of both the fully Bayesian and empirical Bayes entropy rate estimator based on this model, and demonstrate their performance on simulated and real neural spike train data. The first method has the benefit of increased computational efficiency; the second has the advantage that it provides posterior samples of entropy rate rather than a point estimate, so we can evaluate Bayesian confidence intervals, for example.

The chapter is structured as follows. We begin in Section 4.2 by discussing several fundamental information theoretic quantities (including entropy rate, which is our focus here) as well as the relevance of information theory to neural coding. We discuss the challenges of entropy rate estimation and existing entropy rate estimators in Section 4.3. In Section 4.4, we review Markov models in the context of their relationship to entropy rate, and in Section 4.5 we discuss the hierarchical Dirichlet process prior. In Sections 4.6 and 4.7, we present two Bayesian estimates of entropy rate using the HDP prior, one involving a direct calculation of the posterior mean transition probabili-

ties of a Markov model, the other using Markov Chain Monte Carlo methods to sample from the posterior distribution of the entropy rate. In Section 4.8 we compare the HDP entropy rate estimators to existing entropy rate estimators including the context tree weighting entropy rate estimator from [42], the string-parsing method from [49] and [1], and finite-length block entropy rate estimators that makes use of the entropy estimators of [55] and [50]. We evaluate the results for simulated and real neural data.

4.2 Background

Information theory was introduced in 1948 by Shannon [60] and answers important questions in communication theory about data compression and the ability of communication channels to transmit information. Its general framework and quantification of the limits of communication are relevant to a broad range of subjects, including statistical mechanics, probability theory, communication theory, algorithmic complexity in computer science, and neuroscience [20]. A fundamental quantity in information theory is the *(Shannon) entropy* of a random variable.

Definition 13. *The entropy $H(X)$ of a discrete random variable X with values in $\mathcal{X}$ and probability mass function p is:*

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log_2 p(x)$$

Elements x with zero probability are permitted, but with the convention that in this case $p(x) \log p(x)$ is interpreted as $\lim_{\epsilon \to 0^+} \epsilon \log \epsilon = 0$. We use a

base two logarithm throughout this chapter so that entropy is given in bits. The natural logarithm can also be used, in which case the entropy would be in *nats*.

The entropy quantifies the uncertainty of a random variable. It is the average length of shortest description of a random variable. If $L^* = \sum_{x\in\mathcal{X}} l(x)p(x)$ is the expected length of an optimal binary code for random variable X, where $l(x)$ is the length of the binary codeword for x, then $H(X) \leq L^* \leq H(X) + 1$ (See, e.g. [20] Theorem 5.4.1).

With more than one random variable, it becomes useful to quantify how much information one random variable provides about the other. The *mutual information* $I(X, Y)$ for discrete random variables X and Y with probability mass functions given by $p(x)$ and $p(y)$ is

$$I(X,Y) = \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

While entropy characterizes the uncertainty present in a single distribution, entropy rate quantifies the disorder in a stochastic process. For a stochastic process $\{X_i\}_{i=1}^{\infty}$ the entropy of the random vector $(X_1, ..., X_k)$ grows with k; we are interested in how it grows. Define the *block entropy* H_k to be the entropy of the distribution of length-k sequences of symbols, $H_k = H(X_{i+1}, ...X_{i+k})$ Then we define the *entropy rate* as follows:

Definition 14. *The entropy rate of stochastic process* $\{X_i\}_{i=1}^{\infty}$ *is given by*

$$h = \lim_{k\to\infty} \frac{1}{k} H_k. \tag{4.1}$$

There are two other definitions for entropy rate, which are equivalent to the first for stationary processes:

$$h = \lim_{k\to\infty} H_{k+1} - H_k \tag{4.2}$$

$$h = \lim_{k\to\infty} H(X_{i+1}|X_i, X_{i-1}, ...X_{i-k}) \tag{4.3}$$

In computational neuroscience, these quantities in information theory have been used in several ways in the context of neural coding (that is, how a neuron or population of neurons encodes stimuli). See, for example, [23], [64], [12], [6],[3] or see [9] for a review. It is useful to quantify how much information a stimulus or response can provide (to the experimenter, to the organism itself, or both). Entropy gives a measure of neural response variability, and of the uncertainty of the stimulus. Mutual information provides a measure of the response variability that is due to a stimulus, and the reliability of stimulus-response functions. The mutual information of two different neurons has also been used as a measure of the redundancy in responses of two neurons or populations of neurons.

In addition to simply quantifying the amount of information conveyed, information theoretic analyses have been used to address questions of optimality and efficiency of neural coding. One can compute the responses that provide maximum mutual information between a given stimulus distribution and its response, and then compare these 'optimal' responses to the true ones (e.g. [48]). One can also vary the stimulus statistics (e.g. from broadband noise to naturalistic stimulus statistics), and examine the corresponding vari-

ation in the amount of information conveyed by neural response (e.g. [59]). Information theory has also found a use in determining the *limiting spike timing precision* of neural encoding– the shortest time scale over which differences in timing might convey information [54].

While entropy and mutual information are useful for examining the distribution of responses (in terms of average spike rate of a neuron, for example) to stimuli, entropy and mutual information *rates* have been necessary in many of these applications in order to look at the transfer of information *per unit time* in spike trains. This chapter focuses on entropy rate. For spiking neurons, the entropy rate places an upper bound on the rate at which the spike train can convey stimulus information, and a large literature has focused on the problem of estimating entropy rate from spike train data.

4.3 Existing Methods of Entropy Rate Estimation

We briefly survey previous methods of entropy rate estimation for binary (or spike train) sequences, and some of the principal challenges in accurately estimating entropy rates.

4.3.1 Block Entropy Rate Estimators

One class of entropy rate estimators consists of those that we call *block entropy rate estimators*, which make use of the existing body of work surrounding accurate estimation entropy from data. Equations 4.1 and 4.2 suggest a simple entropy rate estimator, in which one first selects a block size k and

then a suitable entropy estimator with which to estimate H_k. The simples block entropy rate estimator is the *plugin* entropy estimator, for which one estimates the probability of each length-k block using the frequency of its appearance in the data. That is, one estimates the probability of each length-k block $(x_1, ..., x_k)$ with the proportion of all observed length-k blocks are equal to $(x_1, ..., x_k)$. For binary data there are 2^k possible length-k blocks. When N denotes the data length and c_i the number of observations of each block in the data, the plugin estimator is:

$$\hat{H}_{\text{plugin}} = \sum_{i=1}^{2^k} -\frac{c_i}{N} \log \frac{c_i}{N} \tag{4.4}$$

from which we can immediately estimate the entropy rate with $h_{plugin,k} = \hat{H}_{\text{plugin}}/k$, for some appropriately chosen k (the subject of "appropriate choice" will be taken up in more detail later).

We would expect that using better block entropy estimators would yield better corresponding entropy rate estimators, and so we also consider two other block based entropy rate estimators. The first uses the Bayesian entropy estimator H_{NSB}, named for Nemenman, Shafee and Bialek [55], which gives a Bayesian least squares estimate for entropy given a mixture-of-Dirichlet prior. The second uses the Miller and Madow estimator [50], which gives a first-order correction to the (often significantly biased) plugin entropy estimator of Equation 4.4:

$$\hat{H}_{MM} = \sum_{i=1}^{2^k} -\frac{c_i}{N} \log \frac{c_i}{N} + \frac{A-1}{2N} \log(e) \tag{4.5}$$

where A is the size of the alphabet of symbols ($A = 2$ for the binary data sequences presently considered). For a given k, we obtain entropy rate estimators $h_{NSB,k} = \hat{H}_{NSB}/k$ and $h_{MM,k} = \hat{H}_{MM}/k$ by applying the entropy estimators from [55] and [50] respectively to the empirical distribution of the length-k blocks.

While we can improve the accuracy of these block entropy rate estimates by choosing a better entropy estimator (e.g. choosing the NSB estimator instead of the plugin estimator), selecting a block size k remains a challenge. If we choose k to be small, we miss long time dependencies in the data and tend to overestimate the entropy; intuitively, the time series will seem more unpredictable than it actually is, because we are ignoring long-time dependencies. On the other hand, as we consider larger k, limited data leads to underestimates of the entropy rate. See the plots of h_{plugin}, h_{NSB}, and h_{MM} in Figure 4.3d for an instance of this effect of block size on entropy rate estimates. We might hope that in between the overestimates of entropy rate for short blocks and the the underestimates for longer blocks, there is some "plateau" region where the entropy rate stays relatively constant with respect to block size, which we could use as a heuristic to select the proper block length [42]. Unfortunately, the block entropy rate at this plateau may still be biased, and for data sequences that are short with respect to their time dependencies, there may be no discernible plateau at all (e.g. [42], Figure 1).

4.3.2 Other Entropy Rate Estimators

Not all existing techniques for entropy rate estimation involve an explicit choice of block length. The Lempel-Ziv complexity [49], for example, can be used to estimate entropy rate [1]. One proceeds by parsing the full string of symbols in the data by starting from the first symbol, and sequentially removing and counting as a "phrase" the shortest substring that has not yet appeared. When M is the number of distinct phrases counted in this way (the Lempel-Ziv complexity), the entropy rate can be estimated as: $h_{LZ} = \frac{M}{N} \log N$, free from any explicit block length parameters.

Another method, proposed in [61], is to extrapolate the linear trend (if any) in the plot of the inverse block length $1/k$ versus the plug-in block entropy rate. The extrapolation to $1/k = 0$ is then used as the estimate for entropy rate.

A fixed block length model like the ones described in the previous section uses the entropy of the distribution of all the blocks of a some length - e.g. all the blocks in the terminal nodes of a context tree like the one in Figure 4.2a. In the context tree weighting (CTW) framework of [42], one instead uses a minimum descriptive length criterion to weight different tree topologies, which have within the same tree terminal nodes corresponding to blocks of different lengths. This weighting is then used to generate Monte Carlo samples and approximate the integral $\int h(\theta) p(\theta|\mathrm{T},\, \mathrm{data}) p(\mathrm{T}|\mathrm{data})\, d\theta\, d\mathrm{T}$, in which T represents the tree topology, and θ represents transition probabilities associated with the terminal nodes of the tree.

In our approach, the HDP prior combined with a Markov model of our data will be a key tool in overcoming some of the difficulties of choosing a block-length appropriately for entropy rate estimation. It will allow us to choose a block length that is large enough to capture possibly important long time dependencies, while easing the difficulty of estimating the properties of these long time dependencies from short data.

4.4 Markov Models

The usefulness of approximating our data source with a Markov model comes from (1) the flexibility of Markov models including their ability to well approximate even many processes that are not truly Markovian, and (2) the fact that for a Markov chain with known transition probabilities the entropy rate need not be estimated but is in fact a deterministic function of the transition probabilities.

A Markov chain is a sequence of random variables that has the property that the probability of the next state depends only on the present state, and not on any previous states. That is, $P(X_{i+1}|X_i, ..., X_1) = P(X_{i+1}|X_i)$. We consider the state variables to be strings of symbols of length k rather than individual 0s and 1s, Thus we will discuss *depth-k* Markov models, where the probability of the next state depends only previous k symbols, or what we will call the length-k *context* of the symbol. With a binary alphabet, we have 2^k states the depth-k chain can take, and from each state s, transitions are possible only to two other states. (So that for, example, 1**10** can transition to

state **10**1 or state **10**0, but not to any other state). Because we allow only two transitions from each state, the transition probability distribution from each s is completely specified by only one parameter, which we denote g_s, the probability of observing a 1 given the context s. A description of the entire Markov Chain model thus requires 2^s parameters: $\{g_s | s \text{ a length-}k \text{ binary word}\}$.

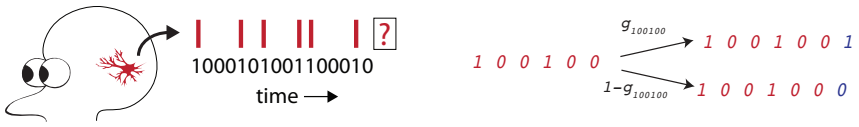

Figure 4.1: A neural spike train as a depth-6 Markov chain

The entropy rate of an ergodic Markov chain with finite state set A is given by:

$$h = \sum_{s \in A} p(s) H(x|s), \tag{4.6}$$

where $p(s)$ is the stationary probability associated with state s, and $H(x|s)$ is the entropy of the distribution of possible transitions from state s. The vector of stationary state probabilities $p(\mathbf{s})$ for all s is computed as a left eigenvector of the transition matrix $\mathbf{T}$:

$$p(\mathbf{s})\mathbf{T} = p(\mathbf{s}) \ , \ \sum_{s} p(s) = 1 \tag{4.7}$$

Since each row of the transition matrix $\mathbf{T}$ contains only two non-zero entries, g_s, and $1 - g_s$, $p(\mathbf{s})$ can be calculated relatively quickly for modest values of k. With equations 4.6 and 4.7, h can be calculated analytically from the vector of all 2^k transition probabilities $\{g_s\}$. A Bayesian estimator of entropy rate

based on a Markov model of order k is given by

$$\hat{h}_{Bayes} = \int h(\mathbf{g})p(\mathbf{g}|\text{data})d\mathbf{g} \tag{4.8}$$

where $\mathbf{g} = \{g_s : |s| = k\}$, h is the deterministic function of $\mathbf{g}$ given by Equations 4.6 and 4.7, and $p(\mathbf{g}|\text{data}) \propto p(\text{data}|\mathbf{g})p(\mathbf{g})$ given some appropriate prior over $\mathbf{g}$.

Modeling a time series as a Markov chain requires a choice of the depth of that chain, so we have not avoided the depth selection problem yet. What will actually mitigate the difficulty here is the use of hierarchical Dirichlet process priors.

4.5 Hierarchical Dirichlet Process Priors

We describe a hierarchical beta prior, a special case of the hierarchical Dirichlet process (HDP), which was presented in [63] and applied to problems of natural language processing in [62] and [70].

The true entropy rate $h = \lim_{k\to\infty} H_k/k$ captures time dependencies of infinite depth. Therefore to calculate the estimate $\hat{h}_{Bayes}$ in Equation 4.8 we would like to choose some large k. However, it is difficult to estimate transition probabilities for long blocks with short data sequences, so choosing large k may lead to inaccurate posterior estimates for the transition probabilities $\mathbf{g}$. In particular, shorter data sequences may not even have observations of all possible symbol sequences of a given length.

This motivates our use of hierarchical priors as follows. Suppose we

have a data sequence in which the subsequence 0011 is never observed. Then we would not expect to have a very good estimate for g_{0011}; however, we could improve this by using the assumption that, a priori, g_{0011} should be similar to g_{011}. That is, the probability of observing a 1 after the context sequence 0011 should be similar to that of seeing a 1 after 011, since it might be reasonable to assume that context symbols from the more distant past matter less. Thus we choose for our prior:

$$g_s|g_{s'} \sim \text{Beta}(\alpha_{|s|}g_{s'}, \alpha_{|s|}(1 - g_{s'})) \tag{4.9}$$

where s' denotes the context s with the earliest symbol removed. This choice gives the prior distribution of g_s mean $g_{s'}$, as desired. The $\alpha_{|s|}$ is a concentration parameter that governs how sharply peaked around $g_{s'}$ is the prior distribution of g_s. We continue constructing the prior with

$$g_{s''}|g_{s'} \sim \text{Beta}(\alpha_{|s'|}g_{s''}, \alpha_{|s'|}(1 - g_{s''}))$$

and so on until $g_{[]} \sim \text{Beta}(\alpha_0 p_\emptyset, \alpha_0(1 - p_\emptyset))$ where $g_{[]}$ is the probability of a spike given no context information and $p_\emptyset$ is a hyperparameter reflecting our prior belief about the probability of a spike. This hierarchy gives our prior the tree structure as shown in in Figure 4.2. A priori, the distribution of each transition probability is centered around the transition probability from a one-symbol-shorter block of symbols. As long as the assumption that more distant contextual symbols matter less actually holds (at least to some degree), this structure allows the sharing of statistical information across different contextual depths. We can obtain reasonable estimates for the transition probabilities

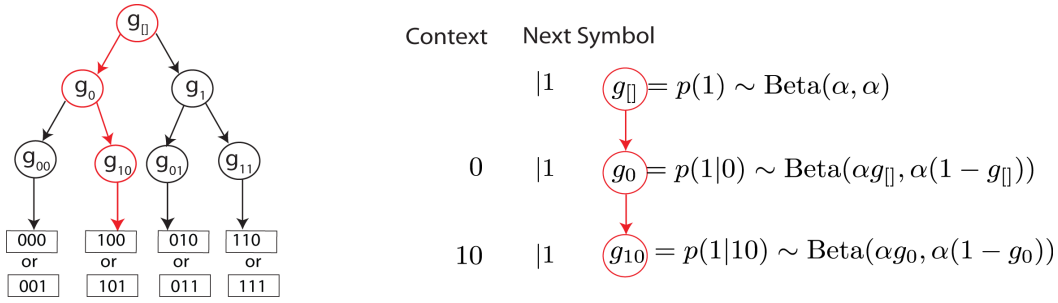

Figure 4.2: A depth-3 hierarchical Dirichlet prior for binary data

from long blocks of symbols, even from data that is so short that we may have few (or no) observations of each of these long blocks of symbols.

We could use any number of distributions with mean $g_{s'}$ to center the prior distribution of g_s at $g_{s'}$; we use beta distributions because they are conjugate to the likelihood. The $\alpha_{|s|}$, which control how concentrated the distribution is about its mean, can also be estimated from the data. We assume that there is one value of α for each level in the hierarchy, but one could also fix alpha to be constant throughout all levels, or let it vary within each level.

This hierarchy of beta distributions is a special case of the hierarchical Dirichlet process. A Dirichlet process (DP) is a stochastic process whose sample paths are each probability distributions. Formally, if G is a finite measure on a set S, then $X \sim DP(\alpha, G)$ if for any finite measurable partition of the sample space $(A_1, ...A_n)$ we have that $X(A_1), ...X(A_n) \sim$ Dirichlet$(\alpha G(A_1), ..., \alpha G(A_n))$. Thus for a partition into only two sets, the Dirichlet process reduces to a beta distribution, which is why when we specialize the HDP to binary data, we obtain a hierarchical beta distribution. In

[63] the authors present a hierarchy of DPs where the base measure for each DP is again a DP. In our case, for example, we have $G_{011} = \{g_{011}, 1 - g_{011}\} \sim DP(\alpha_3, G_{11})$, or more generally, $G_s \sim DP(\alpha_{|s|}, G_{s'})$.

4.6 Empirical Bayesian Estimator

One can generate a sequence from an HDP by drawing each subsequent symbol from the transition probability distribution associated with its context, which is given recursively by [62]:

$$p(1|s) = \begin{cases} \frac{c_{s1}}{\alpha_{|s|}+c_s} + \frac{\alpha_{|s|}}{\alpha_{|s|}+c_s} p(1|s') & \text{if } s \neq \emptyset \\ \frac{c_1}{\alpha_0+N} + \frac{\alpha_0}{\alpha_0+N} p_\emptyset & \text{if } s = \emptyset \end{cases} \tag{4.10}$$

where N is the length of the data string, $p_\emptyset$ is a hyperparameter representing the a prior probability of observing a 1 given no contextual information, c_{s1} is the number of times the symbol sequence s followed by a 1 was observed, and c_s is the number of times the symbol sequence s was observed.

We can calculate the posterior predictive distribution $\hat{\mathbf{g}}_{pr}$ which is specified by the 2^k values $\{g_s = p(1|s) : |s| = k\}$ by using counts c from the data and performing the above recursive calculation to estimate g_s for each of the 2^k states s. Given the estimated Markov transition probabilities $\hat{\mathbf{g}}_{pr}$ we then have an empirical Bayesian entropy rate estimate via Equations 4.6 and 4.7. We denote this estimator h_{empHDP}. Note that while $\hat{\mathbf{g}}_{pr}$ is the posterior mean of the transition probabilities, the entropy rate estimator h_{empHDP} is no longer a fully Bayesian estimate, and is not equivalent to the $\hat{h}_{Bayes}$ of equation 4.8. We thus lose some clarity and access to the full posterior distribution of the

entropy rate (which allows us the ability to easily compute Bayesian confidence intervals, for example). However, we gain a good deal of computational efficiency because calculating h_{empHDP} from $\hat{\mathbf{g}}_{pr}$ involves only one eigenvector computation, instead of the many needed for the Monte Carlo approximation to the integral in Equation 4.8. We present a fully Bayesian estimate next.

4.7 Fully Bayesian Estimator

Here we return to the Bayes least squares estimator $\hat{h}_{Bayes}$ of Equation 4.8. The integral is not analytically tractable, but we can approximate it using Markov Chain Monte Carlo techniques. We use Gibbs sampling [34],[33] to simulate N_{MC} samples $\mathbf{g}^{(i)} \sim \mathbf{g}|data$ from the posterior distribution and then calculate $h^{(i)}$ from each $\mathbf{g}^{(i)}$ via Equations 4.6 and 4.7 to obtain the Bayesian estimate:

$$h_{HDP} = \frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} h^{(i)} \tag{4.11}$$

To perform the Gibbs sampling, we need the posterior conditional probabilities of each g_s. Because the parameters of the model have the structure of a tree, each g_s for $|s| < k$ is conditionally independent from all but its immediate ancestor in the tree, $g_{s'}$, and its two descendants, g_{0s} and g_{1s}. We have:

$$\begin{aligned} p(g_s|g_{s'}, g_{0s}, g_{1s}.\alpha_{|s|}, \alpha_{|s|=1}) \propto & \text{Beta}(g_s; \alpha_{|s|}g_{s'}, \alpha_{|s|}(1-g_{s'})) \\ & \text{Beta}(g_{0s}; \alpha_{|s|+1}g_s, \alpha_{|s|+1}(1-g_s)) \\ & \text{Beta}(g_{1s}; \alpha_{|s|+1}g_s, \alpha_{|s|+1}(1-g_s)) \end{aligned} \tag{4.12}$$

and we can compute these probabilities on a discrete grid since they are each one dimensional, then sample the posterior g_s via this grid. We used a uniform grid of 100 points on the interval [0,1] for our computation in Section 4.8. For the transition probabilities from the bottom level of the tree $\{g_s : |s| = k\}$, the conjugacy of the beta distributions with binomial likelihood function gives the posterior conditional of g_s a recognizable form: $p(g_s|g_{s'}, \text{data}) = \text{Beta}(\alpha_k g_{s'} + c_{s1}, \alpha_k(1 - g_{s'}) + c_{s0})$.

In the HDP model we may treat each α as a fixed hyperparameter, but it is also straightforward to set a prior over each α and then sample α along with the other model parameters with each pass of the Gibbs sampler. The full posterior conditional for α_i with a uniform prior is (from Bayes' theorem):

$$p(\alpha_i|g_s, g_{s0}, g_{s1} : |s| = i-1) \propto \prod_{\{|s|=i-1\}} \frac{(g_{s1}g_{s0})^{\alpha_i g_s - 1}((1 - g_{s1})(1 - g_{s0}))^{\alpha_i(1-g_s)-1}}{\text{Beta}(\alpha_i g_s, \alpha_i(1 - g_s))^2} \tag{4.13}$$

For Section 4.8 we sampled α by computing the probabilities above on a grid of values spanning the range $[1, 2000]$. This upper bound on α is rather arbitrary, but we verified that increasing the range for α had little effect on the entropy rate estimate, at least for the ranges and block sizes considered.

In some applications, the Markov transition probabilities $\mathbf{g}$, and not the entropy rate alone, may be of interest as a description of the time dependencies present in the data. The Gibbs sampler above yields samples from the distribution $\mathbf{g}|\text{data}$, and averaging these N_{MC} samples yields a Bayes least squares estimator of transition probabilities, $\hat{\mathbf{g}}_{gibbsHDP}$. Note that this esti-

mate is closely related to the estimate $\hat{\mathbf{g}}_{pr}$ from the previous section; with more MC samples, $\hat{\mathbf{g}}_{gibbsHDP}$ converges to the posterior mean $\hat{\mathbf{g}}_{pr}$ (when the α are fixed rather than sampled, to match the fixed α per level used in Equation 4.10).

4.8 Applications to Data

We applied the model to both simulated data with a known entropy rate and to neural data, where the entropy rate is unknown. We examine the accuracy of the fully Bayesian and empirical Bayesian entropy rate estimators h_{HDP} and h_{empHDP}, and compare the entropy rate estimators h_{plugin}, h_{NSB} [55], h_{MM}[50], h_{LZ} [49] [1], and h_{CTW} [42], which are described in Section 4.3. We also consider estimates of the Markov transition probabilities $\mathbf{g}$ produced by both inference methods.

4.8.1 Simulation

We considered data simulated from a Markov model with transition probabilities set so that transition probabilities from states with similar suffixes are similar (i.e. the process actually does have the property that more distant context symbols matter less than more recent ones in determining transitions). We used a depth-5 Markov model, whose true transition probabilities are shown in black in Figure 4.3a , where each of the 32 points on the x axis represents the probability that the next symbol is a 1 given the specified 5-symbol context.

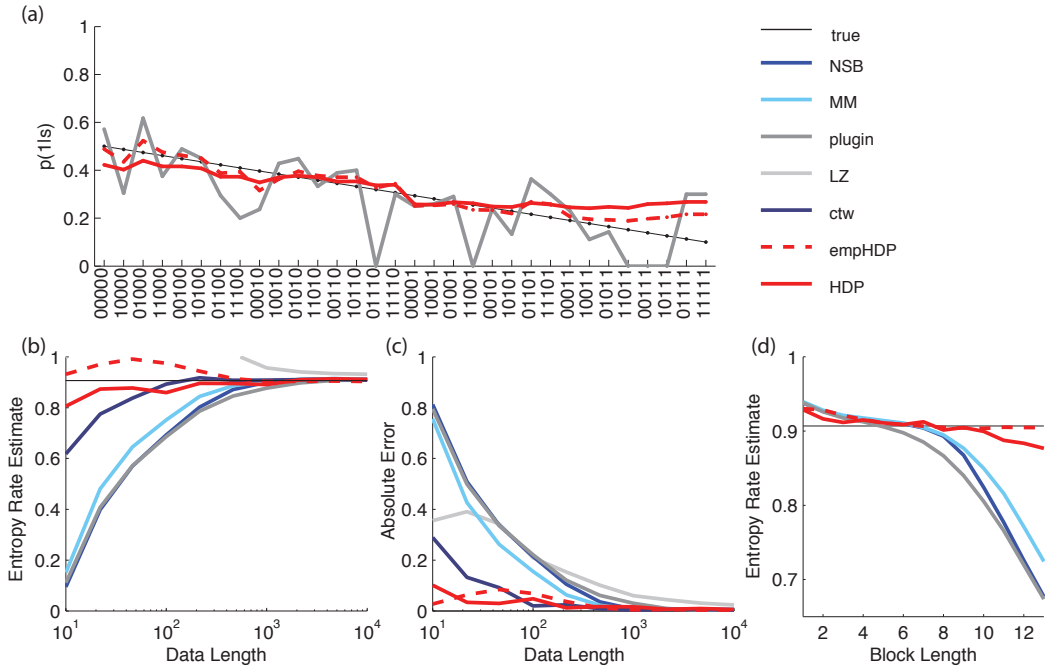

Figure 4.3: Comparison of estimated (a) transition probability and (b,c,d) entropy rate for data simulated from a Markov model of depth 5. In (a) and (d), data sets are 500 symbols long. The block-based and HDP estimators in (b) and (c) use block size $k = 8$. In (b,c,d) results were averaged over 5 data sequences, and (c) plots the average absolute value of the difference between true and estimated entropy rates.

In Figure 4.3a we compare HDP estimates of transition probabilities of this simulated data to the plugin estimator of transition probabilities $\hat{g}_s = \frac{c_{s1}}{c_s}$ calculated from a 500-symbol sequence. (The other estimators do not include calculating transition probabilities as an intermediate step, and so cannot be included here.) With a series of 500 symbols, we do not expect enough observations of each of possible transitions to adequately estimate the 2^k transition probabilities, even for rather modest depths such as $k = 5$. And

indeed, the "plugin" estimates of transition probabilities do not match the true transition probabilities well. On the other hand, the transition probabilities estimated using the HDP prior show the kind of "smoothing" the prior was meant to encourage, where states corresponding to contexts with same suffixes have similar estimated transition probabilities.

Lastly, we plot the convergence of the entropy rate estimators with increased length of the data sequence and the associated error in Figures 4.3b,c. If the true depth of the model is no larger than the depth k considered in the estimators, all the estimators considered should converge. We see in Figure 4.3c that the HDP-based entropy rate estimates converge quickly with increasing data, relative to other models.

The motivation of the hierarchical prior was to allow observations of transitions from shorter contexts to inform estimates of transitions from longer contexts. This, it was hoped, would mitigate the drop-off with larger block-size seen in block-entropy based entropy rate estimators. Figure 4.3d indicates that for simulated data that is indeed the case, although we do see some bias the fully Bayesian entropy rate estimator for large block lengths. The empirical Bayes and and fully Bayesian entropy rate estimators with HDP priors produce estimates that are close to the true entropy rate across a wider range of block-size.

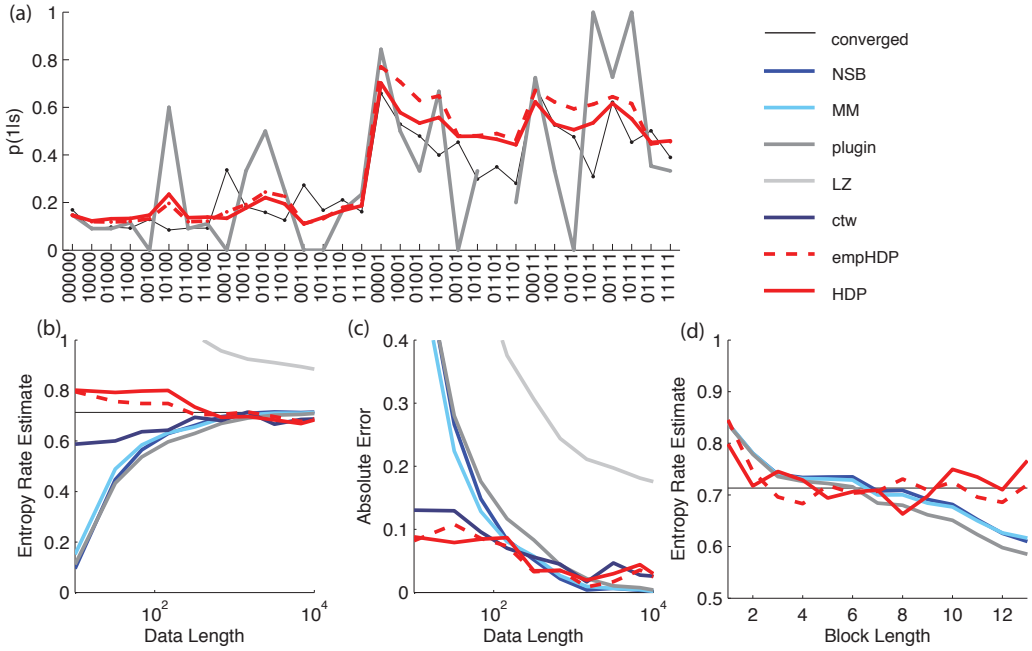

Figure 4.4: Comparison of estimated (a) transition probability and (b,c,d) entropy rate for neural data. The 'converged' estimates are calculated from 700s of data with 4ms bins (167,000 symbols). In (a) and (d), training data sequences are 500 symbols (2s) long. The block-based and HDP estimators in (b) and (c) use block size $k = 8$. In (b,c,d), results were averaged over 5 data sequences sampled randomly from the full dataset.

4.8.2 Neural Data

We applied the same analysis to neural spike train data collected from primate retinal ganglion cells stimulated with binary full-field movies refreshed at 100 Hz [68]. In this case, the true transition probabilities are unknown (and indeed the process may not be exactly Markovian). However, we calculate the plug-in transition probabilities from a longer data sequence (167,000 time bins) so that the estimates are approximately converged (black trace in Figure 4.4a), and note that transition probabilities from contexts with the same most-

recent context symbols do appear to be similar. Thus the estimated transition probabilities reflect the idea that more distant context cues matter less, and the smoothing of the HDP prior appears to be appropriate for this neural data.

The true entropy rate is also unknown, but we estimate it using the plugin estimator on a large data set. We again note the relatively fast convergence of h_{HDP} and h_{empHDP} in Figures 4.4b,c, and the long plateau of the estimators in Figure 4.4d indicating the relative stability of the HDP entropy rate estimators with respect to choice of model depth.

Both estimators show excellent performance on simulated and neural data in terms of their robustness to the choice of model depth, their accuracy on short data sequences, and their convergence with increased data. Both methods of entropy rate estimation also yield estimates of the transition probabilities when the data is modeled as a Markov chain, parameters which may be of interest in the own right as descriptive of the statistical structure and time dependencies in a spike train. Our results indicate that tools from modern Bayesian nonparametric statistics hold promise for revealing the structure of neural spike trains despite the challenges of limited data.

Chapter 5

Conclusions and Future Research

In Chapter 2, we showed that norm recovery, while impossible from one-bit measurements $\text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle)$, is indeed possible from one-bit measurements of the form $\text{sign}(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle + b_i)$ for known nonzero b_i and $\boldsymbol{a}_i$ with i.i.d. standard Gaussian entries. We presented two methods for norm recovery, the first of which also produces an estimate of $\boldsymbol{x}$ and uses randomly chosen b_i, and the second of which uses fixed, deterministic b_i and produces estimates of $\|\boldsymbol{x}\|_2$ only. In both cases, we gave uniform guarantees of accurate recovery (with high probability) given sufficient a number of measurements and provided we had some prior upper bound (or upper and lower bound) for the norm of $\boldsymbol{x}$.

We presented the augmented convex program method making use of the reconstruction algorithm of [58], but the resulting norm recovery method can easily adapted by substituting a different reconstruction method, as mentioned in Remark 6. Thus, improvements in guarantees for the recovery of signals assumed to have norm of one will correspond directly to improvements in our reconstruction guarantees for vectors of unknown (but within known bounds) norm. In particular, as mentioned in Remark 7, the rate of decay of the upper bound on reconstruction error is suboptimal with respect to the lower

bound of [39] (it goes roughly as $m^{-1/5}$ compared to m^{-1}). It would thus be interesting and useful to see if this upper bound on reconstruction error could be decreased. Numerical experiments show the rate of the decrease of error with increasing measurements to be faster than the theoretical bound, indicating that this increase in tightness may indeed be possible.

More generally, quantization to one-bit is just one of many nonlinearities that could be applied to compressive measurements and potentially used for reconstruction. Compressive sensing with nonlinear measurements has been recently addressed in [57] and [7], and the success of reconstruction from one-bit compressed sending suggests that this will be a fruitful avenue to pursue.

In Chapter 3, we presented a new greedy method (COMP) for identifying the timings and amplitudes for waveforms from a signal that has the form of a (noisy) sum of shifted and scaled copies of several known waveforms. We drew upon the method of Continuous Basis Pursuit, and then extended it in several ways. We leveraged the success of greedy methods in the realm of sparse recovery, used a different basis derived from a singular value decomposition, and moved to the frequency domain with the discrete Fourier transform in order to to fine-tune the recovered time shifts. Our SVD basis can also be used within the CBP framework and in our simulations it increased performance of CBP as compared to previously used bases. Our results for simulated and neural data indicated that COMP obtains increased accuracy as well as greatly increased speed over CBP across nearly all regimes tested.

These results suggest that greedy methods of the type introduced here may be quite promising for, among other applications, spike-sorting during the processing of neural data. An immediate extension would be to analyze extensions of other greedy algorithms, like CoSaMP, using the same basis choices and modified projection and greedy steps developed in COMP. Moreover, while a previous method considered a method based on basis pursuit, and we consider extensions of the greedy algorithm OMP, the use of thresholding-based methods such as iterative hard thresholding (IHT) and hard thresholding pursuit (HTP) in the same context have not been considered and may prove useful. It would be fairly straightforward to use the basis we present and adapt these thresholding methods to parallel COMP and CBP and recover continuous times and amplitudes of events.

Another direction of research would be to provide theoretical guarantees of recovery using COMP. Guarantees for OMP exist in terms of the restricted isometry constant or the coherence of the measurement matrix. While the constraint set on the coefficient make close parallels of these proofs difficult, it may still be possible to make guarantees.

In Chapter 4, we presented two estimators of the entropy rate of a spike train or arbitrary binary sequence. The true entropy rate of a stochastic process involves consideration of infinitely long time dependencies. To make entropy rate estimation tractable, one can try to fix a maximum depth of time dependencies to be considered, but it is difficult to choose an appropriate depth that is large enough to take into account long time dependencies and

small enough relative to the data at hand to avoid a severe downward bias of the estimate. We approached this problem by modeling the data as a Markov chain and estimating transition probabilities using a hierarchical prior that links transition probabilities from longer contexts to transition probabilities from shorter contexts. This allowed us to choose a large depth even in the presence of limited data, since the structure of the prior allowed observations of transitions from shorter contexts (of which we have many instances in the data) to inform estimates of transitions from longer contexts (of which we may have only a few instances).

An interesting next step would be to extend our entropy rate estimator to a mutual information rate estimator. This would be useful in examining redundancy of the spike trains of several neurons, or in quantifying the amount of information provided by a neuron about a signal. More broadly, we point out that our entropy rate estimator involved adapting hierarchical Dirichlet process priors that are used in nonparametric Bayesian methods. Nonparametric Bayesian methods show promise in the analysis of neural data and deserve further study in this context.

Bibliography

[1] J. Amigó, J. Szczepański, E. Wajnryb, and M. V. Sanchez-Vives. Estimating the entropy rate of spike trains via lempel-ziv complexity. *Neural Computation*, 16(4):717–736, 2004.

[2] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.

[3] M. J. Berry, D. K. Warland, and M. Meister. The structure and precision of retinal spike trains. *Proceedings of the National Academy of Sciences*, 94(10):5411–5416, 1997.

[4] M. Bethge and P. Berens. Near-maximum entropy models for binary neural representations of natural images. *Advances in Neural Information Processing Systems*, 20:97–104, 2008.

[5] W. Bialek, M DeWeese, F. Rieke, and D. Warland. Bits and brains: Information flow in the nervous system. *Physica A: Statistical Mechanics and its Applications*, 200(1):581–593, 1993.

[6] W. Bialek, F. Rieke, R. R. de Ruyter van Steveninck, and D. Warland. Reading a neural code. *Science*, 252(5014):1854–1857, 1991.

[7] T. Blumensath. Compressed sensing with nonlinear observations and related nonlinear optimization problems. *Information Theory, IEEE Transactions on*, 59(6):3466–3474, 2013.

[8] T. Blumensath and M. Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14(5-6):629–654, 2008.

[9] A. Borst and F. Theunissen. Information theory and neural coding. *Nature neuroscience*, 2(11):947–957, 1999.

[10] P. Boufounos. Greedy sparse signal reconstruction from sign measurements. In *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, pages 1305–1309. IEEE, 2009.

[11] P. Boufounos and R. Baraniuk. 1-bit compressive sensing. *42nd Annual Conference on Information Sciences and Systems (CISS)*, 2008.

[12] G. T. Buracas, A. M. Zador, M. R. DeWeese, and T. D. Albright. Efficient discrimination of temporal patterns by motion-sensitive neurons in primate visual cortex. *Neuron*, 20(5):959–969, 1998.

[13] E. Candès. Compressive sampling. In *Proceedings of the International Congress of Mathematicians: Madrid, August 22-30, 2006: invited lectures*, pages 1433–1452, 2006.

[14] E. Candès. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(9):589–592, 2008.

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

[16] E. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *Information Theory, IEEE Transactions on*, 52(12):5406–5425, 2006.

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

[18] V. Chandrasekaran, B. Recht, P. Parrilo, and A. Willsky. The convex geometry of linear inverse problems. *Foundations of Computational Mathematics*, 12(6):805–849, 2012.

[19] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

[20] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[21] W. Dai and O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *Information Theory, IEEE Transactions on*, 55(5):2230–2249, 2009.

[22] M. Davenport, Y. Plan, E. Berg, and M. Wootters. 1-bit matrix completion. *arXiv preprint arXiv:1209.3672*, 2012.

[23] R. De Ruyter van Steveninck and W. Bialek. Real-time performance of a movement-sensitive neuron in the blowfly visual system: coding and information transfer in short spike sequences. *Proceedings of the Royal society of London. Series B. Biological sciences*, 234(1277):379–414, 1988.

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

[25] D. L. Donoho, Y. Tsaig, I. Drori, and J. L. Starck. Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. *Information Theory, IEEE Transactions on*, 58(2):1094–1121, 2012.

[26] A. Dvoretzky, J. Kiefer, and J. Wolfowitz. Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *The Annals of Mathematical Statistics*, pages 642–669, 1956.

[27] C. Ekanadham. *Continuous basis pursuit and its applications*. PhD thesis, New York University, 2012.

[28] C. Ekanadham, D. Tranchina, and E. P. Simoncelli. A blind deconvolution method for neural spike identification. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS11)*, volume 23, 2011.

[29] C. Ekanadham, D. Tranchina, and E. P. Simoncelli. Recovery of sparse translation-invariant signals with continuous basis pursuit. *Signal Processing, IEEE Transactions on*, 59(10):4735–4744, 2011.

[30] D. Ekanadham, C.vand Tranchina and E. P. Simoncelli. A unified framework and method for automatic neural spike identification. *Journal of Neuroscience Methods*, 222:47–55, 2014.

[31] S. Foucart. Sparse recovery algorithms: sufficient conditions in terms of restricted isometry constants. In *Approximation Theory XIII: San Antonio 2010*, pages 65–77. Springer, 2010.

[32] S. Foucart and H. Rauhut. *A Mathematical Introduction to Compressive Sensing*. Springer, 2013.

[33] A. E. Gelfand and A. F.M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410):398–409, 1990.

[34] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):721–741, 1984.

[35] C. Gold, D. A. Henze, C. Koch, and G. Buzsáki. On the origin of the extracellular action potential waveform: a modeling study. *Journal of Neurophysiology*, 95(5):3113–3128, 2006.

[36] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[37] J. Haupt and R. Baraniuk. Robust support recovery using sparse compressive sensing matrices. In *Information Sciences and Systems (CISS), 2011 45th Annual Conference on*, pages 1–6. IEEE, 2011.

[38] CVX Research Inc. CVX: Matlab software for disciplined convex programming, version 2.0. `http://cvxr.com/cvx`, August 2012.

[39] L. Jacques, J. Laska, P. Boufounos, and R. Baraniuk. Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. *arXiv preprint arXiv:1104.3160*, 2011.

[40] L. Jacques, J. Laska, P. Boufounos, and R. Baraniuk. Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. *Information Theory, IEEE Transactions on*, 59(4):2082–2102, April 2013.

[41] U. Kamilov, A. Bourquard, A. Amini, and M. Unser. One-bit measurements with adaptive thresholds. *Signal Processing Letters, IEEE*,

19(10):607–610, 2012.

[42] M. B. Kennel, J. Shlens, H. D.I. Abarbanel, and E.J. Chichilnisky. Estimating entropy rates with bayesian confidence intervals. *Neural Computation*, 17(7):1531–1576, 2005.

[43] K. C. Knudson and J. W. Pillow. Spike train entropy-rate estimation using hierarchical dirichlet process priors. In *Advances in Neural Information Processing Systems*, pages 2076–2084, 2013.

[44] K. C. Knudson, R. Saab, and R. Ward. One-bit compressive sensing with norm estimation. *arXiv preprint arXiv:1404.6853*, 2014.

[45] K. C. Knudson, J. L. Yates, A. C. Huk, and J. W. Pillow. Inferring sparse representations of continuous signals with continuous orthogonal matching pursuit. In *Advances in Neural Information Processing Systems*, 2014 (accepted).

[46] J. Laska and R. Baraniuk. Regime change: Bit-depth versus measurement-rate in compressive sensing. *Signal Processing, IEEE Transactions on*, 60(7):3496–3505, 2012.

[47] J. N. Laska, Z. Wen, W. Yin, and R. G. Baraniuk. Trust, but verify: Fast and accurate signal recovery from 1-bit compressive measurements. *Signal Processing, IEEE Transactions on*, 59(11):5289–5301, 2011.

[48] S. B. Laughlin et al. A simple coding procedure enhances a neurons information capacity. *Z. Naturforsch*, 36(910-912):51, 1981.

[49] A. Lempel and J. Ziv. On the complexity of finite sequences. *Information Theory, IEEE Transactions on*, 22(1):75–81, 1976.

[50] G. A. Miller and W. G. Madow. *On the Maximum Likelihood Estimate of the Shannon-Weiner Measure of Information*. Operational Applications Laboratory, Air Force Cambridge Research Center, Air Research and Development Command, Bolling Air Force Base, 1954.

[51] A. Movahed, A. Panahi, and G. Durisi. A robust rfpi-based 1-bit compressive sensing reconstruction algorithm. In *Information Theory Workshop (ITW), 2012 IEEE*, pages 567–571. IEEE, 2012.

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

[53] D. Needell and R. Vershynin. Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. *Selected Topics in Signal Processing, IEEE Journal of*, 4(2):310–316, 2010.

[54] I. Nemenman, G. D. Lewen, W. Bialek, and R. R. de Ruyter van Steveninck. Neural coding of natural stimuli: information at sub-millisecond resolution. *PLoS computational biology*, 4(3):e1000025, 2008.

[55] I. Nemenman, F. Shafee, and W. Bialek. Entropy and inference, revisited. In *Advances in Neural Information Processing Systems*, pages 471–478, 2002.

[56] J. W Pillow, J. Shlens, E. J. Chichilnisky, and E. P. Simoncelli. A model-based spike sorting algorithm for removing correlation artifacts in multi-neuron recordings. *PloS one*, 8(5):e62123, 2013.

[57] Y. Plan and R. Vershynin. Robust 1-bit compressed sensing and sparse logistic regression: A convex programming approach. 2012.

[58] Y. Plan and R. Vershynin. One-bit compressed sensing by linear programming. *Communications on Pure and Applied Mathematics*, 2013.

[59] F. Rieke, D.A. Bodnar, and W. Bialek. Naturalistic stimuli increase the rate and efficiency of information transmission by primary auditory afferents. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 262(1365):259–265, 1995.

[60] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 623–656, 1948.

[61] S. P. Strong, R. Koberle, R. R. de Ruyter van Steveninck, and W. Bialek. Entropy and information in neural spike trains. *Phys. Rev. Lett.*, 80:197–200, Jan 1998.

[62] Y. W. Teh. A hierarchical bayesian language model based on pitman-yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 985–992. Association for Computational Linguistics, 2006.

[63] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476), 2006.

[64] F. Theunissen, J. C. Roddey, S. Stufflebeam, H. Clague, and J.P. Miller. Information theoretic analysis of dynamical encoding by four identified primary sensory interneurons in the cricket cercal system. *Journal of Neurophysiology*, 75(4):1345–1364, 1996.

[65] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

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

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

[68] V. J. Uzzell and E. J. Chichilnisky. Precision of spike trains in primate retinal ganglion cells. *Journal of Neurophysiology*, 92:780–789, 2004.

[69] M. Vetterli, P. Marziliano, and T. Blu. Sampling signals with finite rate of innovation. *Signal Processing, IEEE Transactions on*, 50(6):1417–1428, 2002.

[70] F. Wood, C. Archambeau, J. Gasthaus, L. James, and Y. W. Teh. A stochastic memoizer for sequence data. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1129–1136. ACM, 2009.

[71] M. Yan, Y. Yang, and S. Osher. Robust 1-bit compressive sensing using adaptive outlier pursuit. *Signal Processing, IEEE Transactions on*, 60(7):3868–3875, 2012.