Learning Structure in Time Series for Neuroscience and Beyond

David Pfau

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

under the Executive Committee

of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2015

©2014

David Pfau

All Rights Reserved

ABSTRACT

Learning Structure in Time Series for Neuroscience and Beyond

David Pfau

Advances in neuroscience are producing data at an astounding rate - data which are fiendishly complex both to process and to interpret. Biological neural networks are high-dimensional, nonlinear, noisy, heterogeneous, and in nearly every way defy the simplifying assumptions of standard statistical methods. In this dissertation we address a number of issues with understanding the structure of neural populations, from the abstract level of how to uncover structure in generic time series, to the practical matter of finding relevant biological structure in state-of-the-art experimental techniques. To learn the structure of generic time series, we develop a new statistical model, which we dub the probabilistic deterministic infinite automata (PDIA), which uses tools from nonparametric Bayesian inference to learn a very general class of sequence models. We show that the models learned by the PDIA often offer better predictive performance and faster inference than Hidden Markov Models, while being significantly more compact than models that simply memorize contexts. For large populations of neurons, models like the PDIA become unwieldy, and we instead investigate ways to robustly reduce the dimensionality of the data. In particular, we adapt the generalized linear model (GLM) framework for regression to the case of matrix completion, which we call the low-dimensional GLM. We show that subspaces and dynamics of neural activity can be accurately recovered from model data, and with only minimal assumptions about the structure of the dynamics can still lead to good predictive performance on real data. Finally, to bridge the gap between recording technology and analysis, particularly as recordings from ever-larger populations of neurons becomes the norm, automated methods for extracting activity from raw recordings become a necessity. We present a number of methods for automatically segmenting biological units from optical imaging data, with applications to light sheet recording of genetically encoded calcium indicator fluorescence in the larval zebrafish, and optical electrophysiology using genetically encoded voltage indicators in culture. Together, these methods are a powerful set of tools for addressing the diverse challenges of modern neuroscience.

Contents

List of Figures **iv**

Acknowledgments **xii**

Chapter 1 Introduction **1**

1.1 Information Theory 3

1.2 Nonparametric Bayes 10

1.3 Dimensionality Reduction 14

Chapter 2 Probabilistic Deterministic Infinite Automata **19**

2.1 Introduction 20

2.2 Probabilistic Deterministic Finite Automata 22

2.3 Bayesian PDFA Inference 23

2.3.1 A PDFA Prior 23

2.3.2 PDFA Inference 25

2.3.3 The Probabilistic Deterministic Infinite Automaton 29

2.4 Posterior Inference over PDFAs 30

2.5 Theory and Related Work 31

2.5.1 Mixtures of PDFA 31

2.5.2 State Merging Algorithms 32
2.5.3 State Splitting Algorithms 33
2.5.4 Spectral Methods 34
2.5.5 Applications to Neuroscience 34
2.6 Experiments and Results 35
2.6.1 Synthetic Grammars 35
2.6.2 Natural Language and DNA 36
2.6.3 Neural Data 40
2.7 Discussion 47

Chapter 3 Robust Learning of Low-Dimensional Dynamics From Large Neural Ensembles 49
3.1 Introduction 50
3.2 Low dimensional generalized linear models 52
3.3 Learning 54
3.3.1 Nuclear norm minimization 54
3.3.2 Stable principal component pursuit 55
3.3.3 Evaluation through Bregman divergence decomposition 56
3.4 Algorithms 57
3.4.1 Alternating Direction Method of Multipliers 58
3.4.2 Nuclear norm minimization 59
3.4.3 Stable principal component pursuit 61
3.4.4 Fitting Linear Dynamical Systems 64
3.5 Experiments 67
3.5.1 Recovering Subspaces 68

3.5.2 Learning Linear Dynamical Systems 68
3.5.3 Optimizing Smoothing Parameters 72
3.5.4 Predicting Dynamics of Real Neural Populations 74
3.6 Discussion 76

Chapter 4 Separating Biologically Relevant Signals From Optical Recording Data 78

4.1 Introduction 80
4.2 Online Region of Interest Detection for Whole Brain Recordings 85
4.2.1 Online Greedy Clustering 86
4.2.2 Applying Online Greedy Clustering to Whole Brain Activity 90
4.2.3 Visualizing Spontaneous Activity with Nonlinear Dimensionality Reduction 95
4.3 Convolutional Group Lasso 102
4.3.1 Finding ROI Locations with Group Lasso 104
4.3.2 Automatically Tuning the Sparsity Parameter 109
4.3.3 Finding ROI Shapes with Alternating SVD 111
4.3.4 Results 114

Chapter 5 Conclusions 125

Bibliography 128

List of Figures

2.1 Two PNFAs outside the class of PDFAs. (a) can be represented by a mixture of two PDFAs, one following the right branch from state 0, the other following the left branch. (b), in contrast, cannot be represented by any finite mixture of PDFAs. 32

2.2 Three synthetic PDFAs: (a) even process (Shalizi and Shalizi, 2004), (b) Reber grammar (Reber, 1967), (c) Feldman grammar (Feldman and Hanna, 1966). (d) posterior mean and standard deviation of number of states discovered during PDIA inference for varying amounts of data generated by each of the synthetic PDFAs. PDIA inference discovers PDFAs with the correct number of states 36

2.3 Subsampled PDIA sampler trace for Alice in Wonderland. The top trace is the joint log likelihood of the model and training data, the bottom trace is the number of states. 37

2.4 Response of a cultured neuron to periodic stimulation on long time scales, from Gal et al. (2010). The segments of data show in Fig. 2.5 and Fig. 2.6 are outlined in red. 41

2.5 Early response of a cultured neuron to periodic stimulation. The neuron spikes intermittently but with consistent probability. 42

2.6 Late response of a cultured neuron to periodic stimulation. Not only does the neuron spike intermittently, but it also frequently flips into a quiescent state. 42

2.7 Data sampled from a PDIA trained on 24 hours of the data in 2.4. Left: long time scale patterns in the data. Right: medium time scale patterns in the data. The PDIA is able to learn patterns on the order of hundreds to thousands of time bins, or several minutes of recorded data. 44

2.8 Data sampled from Bernoulli GLM trained via maximum likelihood on 24 hours of the data in 2.4. Left: long time scale patterns in the data. Right: medium time scale patterns in the data. 45

2.9 Data generated from a PDIA reproduces long time scale correlations better than data generated from generalized linear models. (a) Power spectral density for real and model data. Up to less than 10^{-2} Hz, the PDIA spectral density closely matches the ground truth. (b) The distribution of lengths of consecutive bins without a spike. The PDIA most closely matches the ground truth, only deviating noticeably at the bottom right (rare, long silences).(b) The distribution of lengths of consecutive bins with a spike. Again the PDIA most closely matches the ground truth. 46

3.1 Recovering low-dimensional subspaces from nonstationary model data. While the subspace remains the same, the dynamics switch between 5 different linear systems. Left top: one dimension of the latent trajectory, switching from one set of dynamics to another (red line). Left middle: firing rates of a subset of neurons during the same switch. Left bottom: covariance between spike counts for different neurons during each epoch of linear dynamics. Right top: Angle between the true subspace and top principal components directly from spike data, from natural rates recovered by nuclear norm minimization, and from the true natural rates. Right bottom: fraction of Bregman divergence explained by the top 1, 5 or 10 dimensions from nuclear norm minimization. Dotted lines are variance explained by the same number of principal components. For $\lambda < 0.1$ the divergence explained by a given number of dimensions exceeds the variance explained by the same number of PCs. 69

3.2 Recovered eigenvalues for the transition matrix of a linear dynamical system from model neural data. Black: true eigenvalues. Red: recovered eigenvalues. (a) Eigenvalues recovered from the true natural rates. (b) Eigenvalues recovered from subspace identification directly on spike counts. (c) Eigenvalues recovered from subspace identification on the natural rates estimated by nuclear norm minimization. 70

3.3 A comparison of the eigenvalues of the transition matrix A recovered by different methods of fitting linear dynamical systems to model data. True eigenvalues in black, estimates over different trials in red. 200 neurons, 10000 time bins, 8 latent dimensions, 10 trials. Top row: Results of estimating A by past-future regression (Eq. 3.27). Bottom row: Results of estimating the transition matrix by the MOESP subspace identification method. Left column: Nuclear norm minimization directly on the matrix Y. Right column: Nuclear norm minimization on the mean-centered matrix $\mathcal{A}(Y)$. In both cases the mean of Y was subtracted before estimating the transition matrix. Note that both mean-centering and subspace identification are necessary to arrive at unbiased estimates of the transition matrix. 71

3.4 Connectivity matrices recovered by SPCP on model data. Left: r^2 between true and recovered synaptic weights across a range of parameters. The position in parameter space of the data to the right is highlighted by the stars. Axes are on a log scale. Right: scatter plot of true versus recovered synaptic weights, illustrating the effect of the nuclear norm term. 72

3.5 Effect of varying the smoothing parameter λ on the recovered transition matrix eigenvalues. True values in black, recovered in red. The nuclear norm term helps reduce the variance of the estimates, but the results quickly degenerate when λ is too large. Note that the results are robust across a wide range of values of λ, from roughly 0.001 to 0.03. Model data, 200 neurons, 1000 time bins, 8 latent dimensions, 5 trials. 73

3.6 A simultaneous comparison of the eigenvalues of A and singular values of Y for various values of λ on the same data as Fig. 3.5. Note that with no smoothing, the top singular values of the recovered Y closely match the true values, but the noise leads to biases in the recovered A. At the other extreme, the quality of the recovered A degrades rapidly if the smoothing forces the rank of Y to be smaller than the true rank. 74

3.7 Log likelihood of held out motor cortex data versus number of latent dimensions for different latent linear dynamical systems. Prediction improves as λ increases, until it is comparable to EM. 75

4.1 Comparison of human and automated ROI detection. ROI centers found by online greedy clustering are in red, human annotation is in green. The ROI shapes are plotted underneath, where the hue is chosen randomly for each ROI, the mean of each pixel μ_k is given by the saturation and the precision of each pixel $1/\eta_k$ is given by the value. That is, in bright regions we are highly confident about the shape of that ROI, and in colored regions we believe that there is a high amount of calcium-dependent fluorescent activity. 91

4.2 Comparison of human and automated ROI detection for one z-slice of 3D data with "penciling" artifact. The pencil artifact goes vertically through the data about one third of the way from the left side of the frame. The color scheme is the same as for Fig. 4.1 92

4.3 Location of regions of interest found by online greedy clustering superimposed over mean fluorescence image of the larval zebrafish. The blue boxes give the location of the details in Fig. 4.4. 94

4.4 94

4.5 Shape and fluorescence time course of a random sampling of ROI. Some ROI both match the anatomy and show complex fluorescence dynamics (e.g. top middle, bottom left), while others show complex dynamics but do not match anything in the mean fluorescence (e.g. top right), and yet others are likely artifact (e.g. bottom right). 95

4.6 Top 3 principal components of spontaneous activity in regions of interest in the larval zebrafish brain. Color corresponds to the position of each region in the fish (Red: mediolateral axis. Green: rostrocaudal axis. Blue: dorsoventral axis.) 98

4.7 (a) Regions of interest in the larval zebrafish brain, colored by location of spontaneous activity projected into principal component space. The coordinate of the 3D embedding for each ROI is given by the red, green and blue values for each point. The hindbrain oscillator (A) and hindbrain-spinal circuit (B) can be distinguished by eye but other anatomical regions are less clear. (b) Sagittal view. (c) Horizontal view. 99

4.8 t-SNE applied to spontaneous activity in regions of interest in the larval zebrafish brain. Color corresponds to the position of each region in the fish (Red: mediolateral axis. Green: rostrocaudal axis. Blue: dorsoventral axis.) 100

4.9 (a) Regions of interest in the larval zebrafish brain, colored by location of spontaneous activity in t-SNE embedding space. The same embedding color scheme is used as in Fig. 4.7. (b) Sagittal view. (c) Horizontal view. 101

4.10 (a) Supervoxels in the larval zebrafish brain, colored by location of spontaneous activity in t-SNE embedding space. The embedding color scheme is the same as that used in Fig. 4.7. (b) Sagittal view. (c) Horizontal view. . . 103

4.11 Top: 90th percentile image of fluorescence activity in the zebrafish forebrain. Middle: contrast normalized image. Bottom: Regions of interest found by convolutional group lasso superimposed on contrast normalized image. Color indicates magnitude of fluorescence activity, blue is high activity and red is low activity. 117

4.12 Shapes and fluorescence activity of regions of interest arranged by magnitude of activity. ROIs 101-120 (a),(b), 1001-1020 (c),(d), 10001-10020 (e),(f), 31001-31020 (g),(h). Only near the very end does the quality of ROI degrade significantly. 118

4.13 Diversity of neural responses in the larval zebrafish. (a) Time course of drifting grating stimulus (blue) and two measures of tail deflection (red and green). (b) Fluorescence activity of the 100 most active ROIs, showing the mixture of behavior-correlated and uncorrelated ROIs. (c) Histogram of correlations between fluorescence and behavior, with gaussian mixture model fit on top. The data is clearly trimodal. (d) histogram of correlation with stimulus. The lack of strongly correlated ROIs is likely due to the filtered nature of the responses. 119

4.14 A selection of neural activity. Location of ROI on the left, fluorescence (green) superimposed over behavior or stimulus (blue). (a),(b) The ROI most correlated with behavior. (c),(d) The ROI most negatively correlated with behavior. Note that activity is suppressed even after the end of behavior. (e),(f) An ROI not significantly correlated with behavior. A large fraction of these are located in the forebrain. (g),(h) An ROI in the optic tectum correlated with the stimulus. Note the response is heavily temporally filtered. 120

4.15 Transverse view of the ROIs in the larval zebrafish brain found by convolutional group lasso. ROIs not significantly correlated with behavior (p>1e-6) are in green, while those significantly correlated with behavior are colored according to the instantaneous correlation coefficient between behavior and fluorescence activity. Anatomical regions negatively correlated with behavior are circled. 121
4.16 PCA applied to fluorescence activity of ROIs. Left: projection of ROIs into principal component space, colored by anatomical position (red: anterior-posterior. green: medial-lateral. blue: dorsal-ventral). Right: ROIs colored by position in principal component space. 122
4.17 t-SNE applied to fluorescence activity of ROIs. Left: projection of ROIs into principal component space, colored by anatomical position (red: anterior-posterior. green: medial-lateral. blue: dorsal-ventral). Right: ROIs colored by position in embedding space. 123
4.18 Fluorescence traces from ROI in the medial midbrain patch in Fig. 4.17 . . . 124

Acknowledgments

This thesis would not have been possible without the support of a huge number of people. Naturally, I owe a huge debt to my advisors, Drs. Liam Paninski and Frank Wood. Frank's mentorship gave me the confidence to develop into the researcher I am today, and his energy and way of viewing problems continues to influence me. Liam's insight, focus and depth of knowledge helped me mature in ways I could not have imagined. To the rest of my commitee: Drs. Larry Abbott, Ken Miller, Mark Churchland and Misha Ahrens, thank you for the time you devoted to my thesis, the advice and direction you gave me in my research, and Misha: thanks for the banana. I also have to thank Larry and Ken for their role as co-directors of the Center for Theoretical Neuroscience, which taught me a style of scientific thinking beyond the world of machine learning.

Many thanks to all those in the Department of Statistics and Center for Theoretical Neuroscience who made the last six years an intellectual adventure: Greg Wayne, Saul Kato, Yashar Ahmadian, Xaq Pitkow, Sean Escola, Brian DePasquale, Maxim Nikitchenko, Michael Vidne, Alex Ramirez, Carl Smith, Gustavo Lacerda, Dan Rubin, David Sussillo, Ann Kennedy, Josh Merel, Claudia Clopath, Kamiar Rahnama Rad, Mattia Rigotti, Ana Calabrese, Ari Pakman, Chris Cueva, Kolia Sadeghi...the list could go on much longer. And my fellow students in the Neurobiology program - that list would go on even longer, but a special thanks to Armen Enikolopov, Mariano Gabitto and Rebecca Brachman for being there when the going got tough.

I'd like to acknowledge the many other researchers I've had the chance to work with over the years. Thanks to my collaborators - Nicholas Bartlett, Finale Doshi-Velez, Alejandro Ramirez, Joel Zylberberg, Eftychios Pnevmatikakis, Daniel Soudry, Scott Linderman, Roy Fox. Some of those projects took off and some didn't, but it was always good to have someone else along for the ride. Thanks to Mike DeWeese for taking a chance on me and for his boundless enthusiasm. Thanks to Eric Jonas for giving me an audience. And thanks to Maneesh Sahani at University College London and Jeremy Freeman and Misha Ahrens at Janelia Farm for supporting me during my (brief) visits there.

And lastly, I would not have even started on this path if it hadn't been for the unwavering support of my family and the wonderful math and science teachers I had through my youth: Ms. Edelstein (my first coauthor!) and Mr. Chafee (I even snuck a *Flatland* quote in), David Newburg, Rob Sherry, Luke Miratrix and Farhad Riahi. You all challenged me at a level far beyond my years, and I carry your lessons with me to this day.

Dedicated to the memory of Prof. Farhad Riahi, Cornelia Sarason and Gladys Pfau

Chapter 1

Introduction

The White Rabbit put on his spectacles.

"Where shall I begin, please your Majesty?" he asked.

"Begin at the beginning," the King said gravely,

"and go on till you come to the end: then stop."

Lewis Carroll, *Alice's Adventures in Wonderland*

If one could look in to a brain with perfect electrical knowledge, one would see a constant stream of spikes. Somehow from this pattern of spiking, the full repertoire of perception, cognition and behavior emerges. Yet given just this stream of information, without prior knowledge of the biological structure behind it, it is extremely difficult to glean any understanding of the underlying computation. This thesis addresses a number of issues in how to learn from time series data when one does not know the structure of the process that generates it. In the introduction, we provide an overview of many of the mathematical ideas that will be used in later sections, primarily how to quantify randomness and complexity in time series. We then show how we can do unsupervised learning from generic time series using these tools, and can learn very compact models from natural data which still predict very well. These generic methods do not scale well to large populations of neurons, and so we instead move within the framework of generalized linear models (GLMs), which is well suited to the particularities of neural data, and show how dimensionality reduction techniques can be unified with the GLM framework to robustly learn subspaces and dynamics from populations of neurons. Finally we address some of the practical issues in experimental neuroscience today around making the dream of recording the activity of every neuron in a living brain a reality - namely, how to segment the activity of different neurons automatically at the scale now becoming possible in experimental neuroscience.

The title of this thesis might seem grandly ambitious: “structure” is quite a broad term. What we mean by that will depend on the context, but a working definition might be that *to learn structure is to find the least complex model in a given class that fits the data well.* Of course, without a a definition of complexity this is an empty statement. One working definition comes from information theory: a time series can be thought as “complex” if a large amount of information about the past must be retained to optimally predict the future. We use this definition and some simple consequences of it to motivate the choice of model class considered in Chapter 2. In the context of dimensionality reduction “complexity” is rather easier to define, as more complex data requires a higher number of dimensions to model it. Because material on information theory and its application to neuroscience and time series does not fit in naturally into any of the other chapters, we present a brief review here. Relevant background material particular to the topics in later chapters will be presented there. Following this review, the thesis is organized thusly: in Chapter 2, we discuss a nonparametric Bayesian method for learning approximately minimal models of stationary time series called the Probabilistic Deterministic Infinite Automata, and present applications to natural language and neural data with long time scale correlations. In Chapter 3 we discuss a robust approach to integrating dimensionality reduction into the generalized linear model (GLM) framework for modeling neural data, along with extensions to learning dynamics and synaptic connectivity. In Chapter 4 we present methods for automatically segmenting fluorescent recordings of spontaneous and driven activity in nearly the whole brain of the larval zebrafish, along with dimensionality reduction analyses not possible by other means. Finally in Chapter 5 we discuss unifying ideas and future directions.

1.1 Information Theory

The information-theoretic school of neuroscience dates back nearly to the foundation of information theory itself (Shannon and Weaver, 1949), and in the context of neural coding can be said to start with Barlow (1961), who hypothesized that the role of sensory coding may be to reduce the redundancy of external stimuli, defining redundancy in terms of entropy. While a full review of the application of information theoretic methods in neuroscience is well beyond the scope of this thesis (see Rieke (1999) for a review up to that point), we discuss some highlights here. Seminal studies in the fly visual system (Laughlin et al., 1981) showed that the contrast-response function of early visual neurons matched the statistics of natural scenes, suggesting that single neurons optimally encode the full range of stimuli they are exposed to. Foundational work on decoding in the fly visual system (Bialek et al., 1991) estimated the entropy rate of spike trains emerging from the H1 cell under the assumption of linear coding and Gaussian noise. Later work in the early mammalian visual system (Atick and Redlich, 1990; Atick, 1992; Atick and Redlich, 1992) extended Barlow's redundancy reduction hypothesis to the case of noisy transmission and found that many response properties of the early visual system are well matched to theoretical predictions.

Most of these studies were based on analytically tractable models that assumed linear tuning and Gaussian noise, and analyzed neural responses using standard signal processing methods like power spectral density. Later work relaxed these assumptions and tried to estimate information theoretic quantities in as close to a model-free manner as possible (Strong et al., 1998). For windows of size T and time bins of size Δt, the number of possible neural "words" is $2^{T/\Delta t}$, so these methods are only reliable when recordings are long enough that most words can be observed. This is very difficult if $T/\Delta t > \mathcal{O}(20)$.

A great deal of effort has gone into understanding and improving entropy estimation in the undersampled limit. Classic work included lower bounds based on coincidence probability (Ma, 1981), prefix tree constructions (Grassberger, 1989) and correction methods based on subtracting analytic calculations of the bias (Panzeri and Treves, 1995). Theoretical results due to Paninski (2003) put a damper on hopes for fully general entropy estimation. He gave a minimax analysis of entropy estimation and showed that standard methods for computing confidence intervals typically underestimated the error in the estimates, and that in certain cases (when the number of bins grows at the same rate as the amount of data), there are in fact no consistent estimators. For the more tractable case of undersampled data but fixed bin size, there are still no unbiased estimators. He introduced the best upper bound (BUB) estimator, which is somewhat conservative but still better behaved than the maximum likelihood estimator and simple extensions.

While the worst case for entropy estimation can be quite bad, empirical evidence suggests that regularized estimators, including Bayesian estimators, can still work well in practice. Nemenman et al. (2002) introduced a Bayesian estimator that places a mixture of Dirichlet distribution prior over the space of neural words, which leads to a nearly flat prior on the entropy of the resulting distribution. This work was extended to (potentially) infinite neural vocabularies in Archer et al. (2012) using Dirichlet and Pitman-Yor processes

(processes which we will use for a different application in Chapter 2), and further extended to use base distributions that more accurately account for the statistics of neural responses (Archer et al., 2013). Other work outside the Bayesian framework includes coverage-adjusted estimators (Chao and Shen, 2003; Vu et al., 2007), James-Stein estimators (Hausser and Strimmer, 2009) and the "unseen" estimator (Valiant and Valiant, 2013).

Most of this work assumed little about the structure of the underlying neural code words, that is, if observations were binary vectors of length N, the problem of estimation treated the different 2^N possible words as undifferentiated symbols in an alphabet (though see Archer et al. (2013)). These N binary observations could be a single neuron across several time bins, or a population of N neurons at the same time. In the latter context, a large amount of work has gone into incorporating more structure into models, making the estimation of relevant quantities more tractable so long as inference in these models is tractable. Notable examples include the pairwise maximum entropy models, inspired by the Ising model in statistical physics (Ising, 1925), which fit the maximum entropy model that captures the empirical pairwise correlations between neurons:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \frac{1}{\mathcal{Z}} e^{\frac{1}{2}\sum_{ij} s_{ij}\mathbf{x}_i\mathbf{x}_j + \sum_i h_i \mathbf{x}_i} \tag{1.1}$$

which can be quite difficult to fit in practice, due to the intractability of computing the partition function $\mathcal{Z}$. Despite this, it has been fruitfully applied to modeling the instantaneous correlations between neurons in the salamander retina (Schneidman et al., 2006) and primate retina (Shlens et al., 2006), and more efficient fitting of these models is an active research area (Sohl-Dickstein et al., 2011). Further research on larger neural populations showed that pairwise maximum entropy models do not accurately capture higher order correlations, while a proposed "reliable interaction" model fits these quite well (Ganmor et al., 2011). Unfortunately this reliable interaction model is not a principled statistical model, and in fact can potentially assign infinite probability to certain (admittedly unlikely) configurations of spikes.

Even with the introduction of some assumptions on the structure between neurons, most of these models do little to model the *dynamics* of neural activity. While estimating the joint distribution across $T/\Delta t$ time steps does give us an order $T/\Delta t - 1$ Markov model for free, as we have discussed, estimating this distribution becomes quite intractable when $T/\Delta t$ is large without imposing additional assumptions. Meanwhile maximum entropy models usually assume a neural ensemble resembles a system in statistical equilibrium. A number of models incorporating dynamics will be considered in the body of this thesis. Many of these models are motivated more by computational tractability or prior biophysical knowledge than by more universal considerations. Before diving into these models, we discuss some relevant connections between information theory and structure in stochastic time series, and use these ideas to motivate some of the modeling choices we make in later chapters.

In the context of time series, the typical application of information theory is to estimating the entropy rate, which quantifies the amount of randomness in a time series. However, this quantity says nothing about the amount of *structure* in a time series: a constant time series has zero entropy rate, while a time series with iid uniform samples has the highest possible entropy rate for a given set of symbols, yet both would be said to have almost no “structure”. Is there a definition of structure in time series that is as universal as the entropy rate is for defining randomness? And can this motivate useful learning algorithms? We argue that there is, and it can. Our argument closely follows Shalizi and Crutchfield (2001) and Bialek et al. (2001), and we use it as motivation for the choice of model class we consider in Chapter 2.

Consider a time series $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_t, \ldots$, where $\mathbf{X}_t$ is a random variable from some set Σ, and let $\mathbf{X}_{t_1:t_2}$ be shorthand for the sequence $\{\mathbf{X}_{t_1}, \mathbf{X}_{t_1+1}, \ldots, \mathbf{X}_{t_2}\}$. As usual, here capital letters denote random variables and lowercase letters denote observations of those random variables. We will restrict our attention to stationary time series, that is those with the property that the probability of observing any particular sequence is invariant with respect to time shifts[1]. Formally, for all subsets of times $\{t_1, \ldots, t_n\}$ and shift k:

$$\Pr[\mathbf{X}_{t_1} = \mathbf{x}_1, \ldots, \mathbf{X}_{t_n} = \mathbf{x}_n] = \Pr[\mathbf{X}_{t_{1+k}} = \mathbf{x}_1, \ldots, \mathbf{X}_{t_{n+k}} = \mathbf{x}_n] \tag{1.2}$$

Then the entropy rate is given by:

$$\begin{aligned} h[\mathbf{X}_t] &= \lim_{T\to\infty} \frac{1}{T} H[\mathbf{X}_{1:T}] && (1.3) \\ &= \lim_{T\to\infty} H[\mathbf{X}_T|\mathbf{X}_{1:T-1}] && (1.4) \\ &= \lim_{T\to\infty} \frac{1}{T} \log p(\mathbf{x}_{1:T}) && (1.5) \end{aligned}$$

where the three definitions are equivalent for stationary time series (Cover and Thomas, 1991). While this naturally quantifies the amount of randomness per unit time in the time series, the amount of structure is far harder to pin down. One way of defining structure might be that it is that part of a time series whose knowledge can help predict other parts. To make this precise, consider the mutual information between past and future sections of a time series, called the *predictive information* (Bialek et al., 2001):

[1]Of course in practice most data is not stationary. Spike trains, in particular, can change their properties quite dramatically over time. Indeed, for learning to be possible it's absolutely necessary that their response properties adapt with experience. There are two ways that this can be dealt with: either by training models on smaller chunks of data that are approximately stationary, or by treating the data as stationary but with extremely long correlation lengths. The latter is is the approach we take in Chapter 2 when learning from long experiments on cultured neurons.

$$\mathcal{I}_{pred}(T, T') = I[\mathbf{X}_{1:T}; \mathbf{X}_{-T':0}] \tag{1.6}$$

Since mutual information is a difference of entropies, and the time series is stationary, this can equivalently be written as $\mathcal{I}_{pred}(T, T') = H_{T+T'} - H_T - H_{T'}$ where H_T is the entropy of a length T block. Taking the limit as either T or T' goes to infinity, it is easy to show that the predictive information converges to the sub-extensive component of the block entropy. This is intuitively appealing, as it means that the block entropy of a stationary time series can be decomposed into a "random" component that grows linearly and a "structured" component that grows sublinearly. Of course, as discussed above, estimating block entropies from data is extremely challenging, so quantifying predictive information is rarely done in practice.

To account for all of the structure in a time series, ideally we would work with the predictive information between an infinite future and infinite past: $I[\mathbf{X}_{-\infty:0}; \mathbf{X}_{1:\infty}]$[2]. It would make analysis much simpler if we could replace the full history with a summary statistic. In particular, as noted by Shalizi and Crutchfield (2001), stationary sequences admit a simple description in terms of minimal sufficient statistics of the past for predicting the future. Consider an arbitrary mapping $T(\mathbf{X}_{-\infty:0})$ of histories into some space. By the data processing inequality, the mutual information between this statistic and the future is less than or equal to the predictive information:

$$I[T(\mathbf{X}_{-\infty:0}); \mathbf{X}_{1:\infty}] \leq I[\mathbf{X}_{-\infty:0}; \mathbf{X}_{1:\infty}] \tag{1.7}$$

Those mappings T^* for which Eq. 1.7 is an equality are by definition *sufficient statistics* of $\mathbf{X}_{-\infty:0}$. Just because a statistic is sufficient does not mean it is useful: the identity is

[2]As noted in Bialek et al. (2001), for many interesting processes this predictive information is actually infinite. Since we will always be interested in structures that can be derived from finite amounts of information, this will not be a practical issue, and we present things here in the infinite history limit for the sake of generality.

trivially a sufficient statistic but does not help model the data at all. What we really want are the *minimal* sufficient statistics of the past for predicting the future. That is, we want some function S such that $I[S(\mathbf{X}_{-\infty:0}); \mathbf{X}_{1:\infty}] = I[\mathbf{X}_{-\infty:0}; \mathbf{X}_{1:\infty}]$ and if T^* is also a sufficient statistic then there exists a function $f_{T^* \to S}$ such that $f_{T^* \to S} \circ T^* = S$. In Shalizi and Crutchfield (2001) they refer to these minimal sufficient statistics of the past for predicting the future as "causal states" of the system. While we make no claim as to whether these states really say anything about causality, we will adopt that terminology as it is less of a mouthful than "minimal sufficient statistics of the past for predicting the future".

It turns out that describing a stationary process in terms of its causal states leads to a natural representation of its generative model. Suppose we have two histories $\mathbf{x}_{-\infty:0}$ and $\mathbf{x'}_{-\infty:0}$ that are mapped to the same causal state: $S(\mathbf{x}_{-\infty:0}) = S(\mathbf{x'}_{-\infty:0})$, and consider one possible future $\mathbf{x}_{1:\infty}$. Given the two histories map to the same state, they must give equal probability to this future sequence: $p(\mathbf{x}_{1:\infty}|\mathbf{x}_{-\infty:0}) = p(\mathbf{x}_{1:\infty}|\mathbf{x}'_{-\infty:0})$, otherwise S would not be a sufficient statistic. Dividing by $p(\mathbf{x}_1|\mathbf{x}_{-\infty:0})$ and $p(\mathbf{x}_1|\mathbf{x}'_{-\infty:0})$, it is clear that the conditional probabilities one step into the future must be equal as well: $p(\mathbf{x}_{2:\infty}|\mathbf{x}_{-\infty:0}, \mathbf{x}_1) = p(\mathbf{x}_{2:\infty}|\mathbf{x}'_{-\infty:0}, \mathbf{x}_1)$, which means $S(\mathbf{x}_{-\infty:0}, \mathbf{x}_1) = S(\mathbf{x}'_{-\infty:0}, \mathbf{x}_1)$, or S would not be minimal. Therefore, to fully describe the generative process for this sequence we need two things: the conditional probability of a symbol given a causal state $\pi(\mathbf{x}|S)$, and a *deterministic* transition function $\delta(\mathbf{x}, S) = S'$ that describes how one state transitions to the next given a single observed symbol. If the space of causal states is finite, this is a kind of probabilistic model known as a *probabilistic deterministic finite automaton* (PDFA) (Rabin, 1963). A PDFA where the states correspond to causal states is referred to as a causal state machine, or sometimes an ϵ-machine[3].

[3]One more technical distinction between causal state machines and general PDFAs is that causal state machines, like other state space models, generally lack a terminating state, because they are meant to model sequences that can go on for an arbitrary length of time.

PDFA are just one kind of model within the very general class of state space models, however they have a number of attractive features that lead us to consider them here. One is that there is only a single possible state sequence that can generate a given observation. In fact, one way of defining PDFAs is as the set of Hidden Markov Models for which the posterior distribution over states always concentrates on a single state sequence. This makes forward inference very fast. It also makes it possible to compute a convenient upper bound on the predictive information. As entropy is an upper bound on mutual information, and sufficient statistics preserve mutual information, it's clear that the entropy of the random variable $\mathbf{S} \triangleq S(\mathbf{X}_{-\infty:0})$ is an upper bound on the predictive information. If $\mathbf{S}$ were not a sufficient statistic, as is the case with general state space models, there would be no particular relationship between $H[\mathbf{S}]$ and the predictive information, and if $\mathbf{S}$ were any other sufficient statistic, the bound would be less tight than for a minimal sufficient statistic. Finally, it may be the case for certain kinds of data that PDFA are more learnable, in a practical sense, than other state space models. We will see examples in Chapter 2, notably on natural language data, where learning PDFAs leads to superior predictive models than standard methods for training Hidden Markov Models.

1.2 Nonparametric Bayes

In Chapter 2, we employ the toolkit of nonparametric Bayesian inference to learn generic PDFAs. We provide a brief review here of some of the central concepts in nonparametric Bayesian inference. Bayesian nonparametrics is particularly appropriate for problems where the space over which inference is being performed is countable but infinite, for instance clustering models with an infinite number of latent clusters, or topic models with an infinite number of latent topics. Since the number of states in a PDFA that generates data is not known a priori, a nonparametric prior is important. A central object of study in Bayesian nonparametrics is the *Dirichlet process*, which can be seen as an extension of the Dirichlet distribution to infinite and even uncountable base spaces. The basic Dirichlet distribution is a distribution over discrete distributions with K categories, defined by a vector $(\alpha_1, \ldots, \alpha_K)$, $\alpha_k \geq 0$ of parameters that specifies how concentrated draws from that Dirichlet distribution are around that particular category. Thus if $\vec{\pi}$ is a multinomial distribution $\vec{\pi} = (\pi_1, \ldots, \pi_K)$, $\sum_{k=1}^{K} \pi_k = 1$, the probability of drawing $\vec{\pi}$ from a Dirichlet distribution with parameters $\vec{\alpha} = (\alpha_1, \ldots, \alpha_K)$ is:

$$p(\vec{\pi}|\vec{\alpha}) = \frac{\Gamma(\sum_{k=1}^{K} \alpha_k)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} \pi_k^{\alpha-1} \tag{1.8}$$

We can split up the parameter vector $\vec{\alpha}$ into a scalar $\alpha = \sum_{k=1}^{K} \alpha_k$ and a normalized vector $H = (\alpha_1/\alpha, \ldots, \alpha_K/\alpha)$. H is a base distribution that gives the mean sample from a Dirichlet distribution with these parameters, while α is a concentration parameter that controls how close samples are to H.

A Dirichlet process is also a distribution over distributions, parameterized by a concentration α and base distribution H. Dirichlet processes generalize Dirichlet distributions such that the base distribution H can be *any* probability distribution over *any* measurable space, not just multinomial distributions. Formally, let H be a distribution over a base space Θ, for instance a Gaussian on the real line, let $\mathcal{G} \sim DP(\alpha, H)$ be a sample from a Dirichlet process with concentration α and base distribution H (and therefore $\mathcal{G}$ is a distribution over Θ), and let $(\Theta_1, \Theta_2, \ldots, \Theta_K)$ be a finite partition of Θ: $\Theta_i \cap \Theta_j = \emptyset \; \forall i \neq j$, $\bigcup_{k=1}^{K} \Theta_k = \Theta$. A multinomial distribution can be constructed from any distribution on Θ by adding up the total probability mass inside one bin of the partition Θ_k, for instance $(\mathcal{G}(\Theta_1), \ldots, \mathcal{G}(\Theta_K))$ is a multinomial distribution. We call this the restriction of $\mathcal{G}$ to the partition. A distribution over distributions $DP(\alpha, H)$ is a Dirichlet process if, for all finite partitions of Θ, samples from $DP(\alpha, H)$ restricted to that partition are Dirichlet-distributed with concentration α and base distribution given by the restriction of H to that partition.

While mathematically precise, this definition is rather abstract and hard to work with constructively. There is an alternative, algorithmic way of describing a Dirichlet process known as the *stick-breaking construction.* In this construction, a draw $\mathcal{G} \sim DP(\alpha, H)$ is built up from a countable number of atoms:

$$\mathcal{G} = \sum_{k=1}^{\infty} \pi_k \delta(\theta_k) \tag{1.9}$$

It's always true that a draw from a Dirichlet process has this form of an infinite sum of delta functions. The stick breaking process gives a way of constructing the stick lengths π_k that define how much probability is assigned to any point, and stick locations $\theta_k \in \Theta$. We start with a "stick" of length one, and break off a section $\pi_1 \sim \text{Beta}(1, \alpha)$. The remaining stick clearly has length $(1 - \pi_1)$. We repeat this iteratively, sampling a fraction $\pi'_k \sim \text{Beta}(1, \alpha)$ of the stick to break off, then assigning $\pi_k = \pi'_k \prod_{i=1}^{k-1}(1 - \pi'_i)$. This gives us a countable number of stick lengths that clearly sum to one. The locations of the sticks θ_k are simply sampled iid from the base distribution H. This construction is useful if, for instance, one wishes to generate samples from $\mathcal{G}$ without representing the entire distribution, as the stick never needs to be broken to completion, but only up to some appropriate length.

There is one more construction relevant to Dirichlet processes that is particularly useful for inference, called the *Chinese Restaurant Process.* This deals directly with samples from $\mathcal{G}$ while integrating out $\mathcal{G}$ itself. It is most clearly constructed by starting in the limit of finite base space Θ, in which case we reduce back to the case of Dirichlet distributions. Suppose one has samples $x_1, \ldots, x_N$ from some multinomial distribution $\vec{\pi}$, which is itself sampled from a Dirichlet distribution $\mathrm{Dir}(\alpha, H)$. Then, integrating out $\vec{\pi}$, the conditional probability of the next observation given the observations that came before it is given by

$$\begin{aligned} p(x_{N+1} = k | x_1, \ldots, x_N, \alpha, H) &= \int d\vec{\pi} p(x_{N+1} = k|\vec{\pi}) p(\vec{\pi}|x_1, \ldots, x_N, \alpha, H) \\ &= \frac{n_k + \alpha_k}{N + \alpha} \end{aligned} \tag{1.10}$$

If we take all α_k to be uniform, so $H = (\alpha/K, \ldots, \alpha/K)$, then in the limit as $K \to \infty$, keeping N and α fixed, $p(x_{N+1} = k|x_1, \ldots, x_N, \alpha, H) \to n_k/(N+\alpha)$ if $n_k \neq 0$. If $n_k = 0$ the probability vanishes for any *particular* k, however the total probability that x_{N+1} is some k with zero observations does not vanish! Instead, it converges to $\alpha/(N+\alpha)$. This process is known as the Chinese Restaurant Process with concentration α, or $CRP(\alpha)$, and defines a distribution over possible ways of partitioning the set $\{x_1, \ldots, x_N\}$.

The Chinese Restaurant Process can be arrived at through Dirichlet Processes by the following construction. Suppose our observations $x_1, \ldots, x_N$ are sampled from $\mathcal{G} \sim DP(\alpha, H)$. From Eqn. 1.9, it is clear that the values of $x_1, \ldots, x_N$ will cluster together at the position of the sticks θ_k, and that the next observation x_{N+1} may be from some stick that has not yet been sampled, but as N grows it becomes more likely that it will correspond to some θ_k already observed in the data. If we marginalize out $\mathcal{G}$, then the distribution over partitions of the data is exactly given by $CRP(\alpha)$, while the value of the data in a particular position is sampled iid from H. Thus the Chinese Restaurant Process allows us to work with samples from $\mathcal{G}$ without having to represent $\mathcal{G}$, at the cost of introducing dependence between the data, which were iid conditioned on $\mathcal{G}$.

One last construction should be mentioned to give adequate background for the material in Chapter 2. Generative models can easily be made hierarchical, by using a sample from one level as the parameters for another level. Dirichlet processes are no exception, and the Hierarchical Dirichlet Process (HDP) has become an extremely useful tool in nonparametric Bayesian modeling in the last decade. The generative model is very easy to describe - data are assumed to come from different *contexts* which we index by j, and the data in each context are assumed to be sampled from $\mathcal{G}_j$, which are samples from the same Dirichlet process. What makes it a Hierarchical Dirichlet process is that the base distribution for this Dirichlet process is *itself* a sample $\mathcal{G}$ from a Dirichlet process. What this means is that there is a sharing of sticks between the different contexts - all $\mathcal{G}_j$ will have the same set of stick locations θ_{kj} but a different set of stick lengths. This creates a sharing of statistical strength between different contexts - in effect it says the data should be somewhat similar between contexts, but far more similar within contexts. Inference with HDPs can require very involved bookkeeping, which we will not go into here, but direct the curious reader to Teh et al. (2006a). There is even a Chinese Restaurant Process equivalent for HDPs that integrates out both levels of draws from an HDP, called the Chinese Restaurant Franchise, which is the representation we use for inference in the PDIA model in Chapter 2.

1.3 Dimensionality Reduction

Dimensionality reduction is a vast field - arguably any kind of data processing that outputs fewer parameters than the dimensionality of the input is a kind of dimensionality reduction. Even nonparametric models can be viewed as a kind of dimensionality reduction if the model is properly regularized so that in practice the number of parameters inferred is small. That is not to say that all useful models do dimensionality reduction - kernel methods, for instance, work by projecting data into a high-dimensional feature space, at least implicitly - but whenever data is very high dimensional, it is usually necessary to reduce the complexity somehow to apply any useful analysis. There are a number of different perspectives from which common dimensionality reduction techniques can be derived. Here we will use Principal Component Analysis (PCA) as a starting point to show how the dimensionality reduction techniques used in this thesis can be derived from a single starting point.

Principal Component Analysis is the grandmother of dimensionality reduction. PCA finds the orthogonal basis for a given dataset $X \in \mathbb{R}^{N \times T}$ that maximizes the variance of the data projected onto that basis. This has a number of mathematical properties that makes it appealing and easy to work with. Despite being a nonconvex problem:

$$\max_{U \in \mathbb{R}^{N \times k} \,\mathrm{st}\, U^T U = I} ||U^T X||_F^2 \tag{1.11}$$

PCA admits an exact solution - one of the few nonconvex problems for which this is possible (Eckart and Young, 1936). Moreover it can be computed using standard linear algebra methods - the top k principal vectors of the dataset X is given by the top k eigenvectors of the covariance matrix XX^T/N, while the corresponding eigenvalues give the amount of variance explained by that particular principal component. The principal vectors can also be computed along with the projection of the data onto those principal components without having to first compute the covariance by taking the singular value decomposition (SVD) of the data. This is a consequence of the close connection between the singular value decomposition of X and eigenvector decomposition of XX^T: the left singular vectors of the former are the eigenvectors of the latter, and the singular values of the former are the square-root of the eigenvalues of the latter.

There are a number of alternative interpretations of PCA that illustrate the close connection between different perspectives on dimensionality reduction. One way of generalizing PCA is from the point of view of low-rank matrix approximation. If U, Σ and V

give the SVD of X such that $U\Sigma V^T = X$, then the truncated SVD $X_k = U_k \Sigma_k V_k^T$ that only keeps the top k singular vectors is the solution to the optimization problem

$$\min_{X_k \,\mathrm{st}\, \mathrm{rank}(X_k)=k} ||X_k - X||_F^2 \tag{1.12}$$

That is, for the Frobenius norm, reconstructing the data from the top k principal components is equivalent to finding the minimum error rank-k approximation to the data matrix. Optimizing over rank-k matrices or minimizing the rank of a matrix under other constraints is generally a hard non-convex problem, but can be solved exactly for the Frobenius norm, as well as a larger class called unitarily invariant norms (Yu and Schuurmans, 2012). For more general optimization problems with low-rank matrices, approximations must be made.

There are other problems in matrix analysis that surprisingly admit exact solutions. For the case of matrix *completion*, where only m entries from X are observed, then if X actually is rank k then X can be recovered *exactly* with high probability if $m \geq Cn^{1.2}k\log n$, $n = \max(N, T)$ for some value of C (Candès and Recht, 2009). Algorithmically, this is achieved by finding the full matrix with entries matching the observed entries of X that minimizes the sum of singular values or *nuclear norm*.

$$\min_{\hat{X} \,\mathrm{st}\, \Omega(\hat{X})=\Omega(X)} ||\hat{X}||_* \tag{1.13}$$

where $\Omega(X)$ projects X onto the m observed values. This is a convex heuristic that plays precisely the same role for matrix completion that the ℓ_1 norm does for compressed sensing. Other applications to dimensionality reduction are possible with the nuclear norm: if we have full observation of a matrix X that is the sum of a rank-k component L and sparse noise S, then in certain limits we can exactly separate the two by solving a similar convex problem:

$$\min_L ||L||_* + \lambda ||X - L||_1 \tag{1.14}$$

which trades off between the dimensionality of L and the sparsity of the residual. Even if these exact conditions are not satisfied, nuclear norm minimization makes it possible to approximately solve dimensionality reduction problems that would not be possible exactly.

Another perspective on PCA is as maximum likelihood inference in a probabilistic model (Tipping and Bishop, 1999). In this view, each datum $\mathbf{x}_t \in \mathbb{R}^N$ is assumed to be generated from a linear Gaussian model:

$$\begin{aligned} \mathbf{x}_t &= W\mathbf{z}_t + \mu + \epsilon_t & (1.15) \\ \mathbf{z}_t &\sim \mathcal{N}(0, \mathbf{I}_k) & (1.16) \\ \epsilon_t &\sim \mathcal{N}(0, \sigma^2 \mathbf{I}_N) & (1.17) \end{aligned}$$

Here the low-dimensional latent factors $\mathbf{z}_t$ that generated the data are latent variables, and the maximum likelihood solution for W spans exactly the same subspace as the top k principal components. The advantage of this perspective is that the full toolkit of probabilistic inference can be brought to bear in the case of missing data, or generalizations and extensions to more complex models. While the matrix factorization community has generalized PCA in a few ways that still lead to theoretical guarantees of performance, the probabilistic point of view makes it possible to extend PCA to an almost endless variety of compositional models with a small number of latent variables, so long as one is willing to accept that inference will usually be approximate and possibly prone to local minima.

Finally, PCA can be viewed as finding a projection that not just maximizes the variance but the mutual information between data in the original high-dimensional space and the low dimensional space. This is similar to the probabilistic point of view in that we are assuming a linear Gaussian model behind the data. In this case the data and projection into lower-dimensional space are jointly Gaussian and the mutual information has a closed form, which can easily be shown to be maximized when the projection spans the same space as the top eigenvectors of the covariance. As discussed, nonparametric estimation of entropy and mutual information is a hard problem, and so this perspective has not lead to as many developments in the dimensionality reduction field, but it does provide an illuminating general perspective on dimensionality reduction techniques that find a parametric projection. The full scope of dimensionality reduction extends far beyond the techniques discussed here, and relevant methods are introduced throughout the text where appropriate.

Chapter 2

Probabilistic Deterministic Infinite Automata[1]

I once would have inquir'd cooly, what an Automaton might know
of Life, but now I only sat silent...

Thomas Pynchon, *Mason & Dixon*

We propose a novel Bayesian nonparametric approach to learning with probabilistic deterministic finite automata (PDFA). We define and develop a sampler for a PDFA with an infinite number of states which we call the probabilistic deterministic infinite automata (PDIA). Posterior predictive inference in this model, given a finite training sequence, can be interpreted as averaging over multiple PDFAs of varying structure, where each PDFA is biased towards having few states. We suggest that our method for averaging over PDFAs is a novel approach to predictive distribution smoothing. We test PDIA inference both on PDFA structure learning and on both natural language and DNA data prediction tasks. When applied to neural data, the PDIA is able to reproduce long-time scale correlations

[1]This work has, in part, been published (Pfau, Bartlett and Wood. *Advances in Neural Information Processing Systems*,2010)

better than a generalized linear model. The results suggest that the PDIA presents an attractive compromise between the computational cost of hidden Markov models and the storage requirements of hierarchically smoothed Markov models, particularly for data with long-range temporal correlations.

2.1 Introduction

The focus of this chapter is a novel Bayesian framework for learning with probabilistic deterministic finite automata (PDFA) (Rabin, 1963). A PDFA is a generative model for sequential data (PDFAs are reviewed in Section 2.2). Intuitively a PDFA is similar to a hidden Markov model (HMM) (Rabiner, 1989) in that it consists of a set of states, each of which when visited emits a symbol according to an emission probability distribution. It differs from an HMM in how state-to-state transitions occur; transitions are deterministic in a PDFA and nondeterministic in an HMM.

In our framework for learning with PDFAs we specify a prior over the parameters of a single large PDFA that encourages state reuse. The inductive bias introduced by the PDFA prior provides a soft constraint on the number of states used to generate the data. We take the limit as the number of states becomes infinite, yielding a model we call the probabilistic deterministic infinite automata (PDIA).

Given a finite training sequence, the PDIA posterior distribution is an infinite mixture of PDFAs. Samples from this distribution form a finite sample approximation to this infinite mixture, and can be drawn via Markov chain Monte Carlo (MCMC) (Gelman et al., 1995). Using such a mixture we can average over our uncertainty about the model parameters (including state cardinality) in a Bayesian way during prediction and other inference tasks. We find that averaging over a finite number of PDFAs trained on naturalistic data leads to better predictive performance than using a single “best” PDFA.

We chose to investigate learning with PDFAs because they are intermediate in expressive power between HMMs and finite-order Markov models, and thus strike a good balance between generalization performance and computational efficiency. A single PDFA is known to have relatively limited expressivity. We show in 2.5 that a finite mixture of PDFAs has greater expressivity than that of a single PDFA but is not as expressive as a probabilistic nondeterministic finite automata (PNFA)[2] . A PDIA is clearly highly expressive; an infinite mixture over the same is even more so. Even though ours is a Bayesian approach to PDIA learning, in practice we only ever deal with a finite approximation to the full posterior and thus limit our discussion to finite mixtures of PDFAs.

While model expressivity is a concern, computational considerations often dominate model choice. We show that prediction in a trained mixture of PDFAs can have lower asymptotic cost than forward prediction in the PNFA/HMM class of models. We also present evidence that averaging over PDFAs gives predictive performance superior to HMMs trained with standard methods on naturalistic data. We find that PDIA predictive performance is competitive with that of fixed-order, smoothed Markov models with the same number of states. While sequence learning approaches such as the HMM and smoothed Markov models are well known and now highly optimized, our PDIA approach to learning is novel and is amenable to future improvement.

Section 2.2 reviews PDFAs, Section 2.3 introduces Bayesian PDFA inference, Section 2.5 discusses related work on PDFA induction and the theoretical expressive power of mixtures of PDFAs, and Section 2.6 presents experimental results on DNA, natural language and neural recordings. In Section 2.7 we discuss ways in which PDIA predictive performance might be improved in future research.

[2]PNFAs with no final probability are equivalent to hidden Markov models (Dupont et al., 2005)

2.2 Probabilistic Deterministic Finite Automata

A PDFA is formally defined as a 5-tuple $M = (Q, \Sigma, \delta, \pi, q_0)$, where Q is a finite set of states, Σ is a finite alphabet of observable symbols, $\delta : Q \times \Sigma \to Q$ is the transition function from a state/symbol pair to the next state, $\pi : Q \times \Sigma \to [0, 1]$ is the probability of the next symbol given a state and q_0 is the initial state.[3] Throughout this chapter we will use i to index elements of Q, j to index elements of Σ, and t to index elements of an observed string. For example, δ_{ij} is shorthand for $\delta(q_i, \sigma_j)$, where $q_i \in Q$ and $\sigma_j \in \Sigma$.

Given a state q_i, the probability that the next symbol takes the value σ_j is given by $\pi(q_i, \sigma_j)$. We use the shorthand $\boldsymbol{\pi}_{q_i}$ for the state-specific discrete distribution over symbols for state q_i. We can also write $\sigma|q_i \sim \boldsymbol{\pi}_{q_i}$ where σ is a random variable that takes values in Σ. Given a state q_i and a symbol σ_j, however, the next state $q_{i'}$ is *deterministic*: $q_{i'} = \delta(q_i, \sigma_j)$. Generating from a PDFA involves first generating a symbol stochastically given the state the process is in: $x_t|\xi_t \sim \boldsymbol{\pi}_{\xi_t}$ where $\xi_t \in Q$ is the state at time t. Next, given ξ_t and x_t transitioning deterministically to the next state: $\xi_{t+1} = \delta(\xi_t, x_t)$. This is the reason for the confusing "probabilistic deterministic" name for these models. Turning this around, given data, q_0, and δ, there is no uncertainty about the path through the states. This is a primary source of computational savings relative to HMMs.

PDFAs are more general than nth-order Markov models (i.e. m-gram models, $m = n + 1$), but less expressive than hidden Markov models (HMMs)(Dupont et al., 2005). For the case of nth-order Markov models, we can construct a PDFA with one state per suffix $x_1 x_2 \ldots x_n$. Given a state and a symbol x_{n+1}, the unique next state is the one corresponding to the suffix $x_2 \ldots x_{n+1}$. Thus nth-order Markov models are a subclass of PDFAs with $\mathcal{O}(|\Sigma|^n)$ states. For an HMM, given data and an initial distribution over states, there is a

[3]In general q_0 may be replaced by a distribution over initial states.

posterior probability for every path through the state space. PDFAs are those HMMs for which, given a unique start state, the posterior probability over paths is degenerate at a single path. As we explain in Section 2.5, mixtures of PDFAs are strictly more expressive than single PDFAs, but still less expressive than PNFAs.

2.3 Bayesian PDFA Inference

We start our description of Bayesian PDFA inference by defining a prior distribution over the parameters of a finite PDFA. We then show how to analytically marginalize nuisance parameters out of the model and derive a Metropolis-Hastings sampler for posterior inference using the resulting collapsed representation. We discuss the limit of our model as the number of states in the PDFA goes to infinity. We call this limit the probabilistic deterministic infinite automaton (PDIA). We develop a PDIA sampler that carries over from the finite case in a natural way.

2.3.1 A PDFA Prior

We assume that the set of states Q, set of symbols Σ, and initial state q_0 of a PDFA are known but that the transition and emission functions are unknown. The PDFA prior then consists of a prior over both the transition function δ and the emission probability function π. In the finite case δ and π are representable as finite matrices, with one column per element of Σ and one row per element of Q. For each column j (j co-indexes columns and set elements) of the transition matrix δ, our prior stipulates that the elements of that column are i.i.d. draws from a discrete distribution $\boldsymbol{\phi}_j$ over Q, that is, $\delta_{ij} \sim [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{|\Sigma|}]$, $0 \leq i \leq |Q| - 1$. The $\boldsymbol{\phi}_j$ represent transition tendencies given a symbol, if the ith element of $\boldsymbol{\phi}_j$ is large then state q_i is likely to be transitioned to anytime the last symbol was σ_j. The $\boldsymbol{\phi}_j$'s are themselves given a shared Dirichlet prior with parameters $\alpha\boldsymbol{\mu}$, where α is a concentration and $\boldsymbol{\mu}$ is a template transition probability vector. If the ith element of $\boldsymbol{\mu}$ is large then the ith state is likely to be transitioned to regardless of the emitted symbol. We place a uniform Dirichlet prior on $\boldsymbol{\mu}$ itself, with γ total mass and average over $\boldsymbol{\mu}$ during inference. This hierarchical Dirichlet construction encourages both general and context specific state reuse. We also place a uniform Dirichlet prior over the per-state emission probabilities $\boldsymbol{\pi}_{q_i}$ with β total mass which smooths emission distribution estimates. Formally:

$$\begin{aligned}
\boldsymbol{\mu}|\gamma, |Q| &\sim \mathrm{Dir}\left(\gamma/|Q|, \ldots, \gamma/|Q|\right) \quad (2.1)\\
\boldsymbol{\phi}_j|\alpha, \boldsymbol{\mu} &\sim \mathrm{Dir}(\alpha\boldsymbol{\mu}) \quad (2.2)\\
\boldsymbol{\pi}_{q_i}|\beta, |\Sigma| &\sim \mathrm{Dir}(\beta/|\Sigma|, \ldots, \beta/|\Sigma|)\\
\delta_{ij} &\sim \boldsymbol{\phi}_j
\end{aligned}$$

where $0 \leq i \leq |Q| - 1$ and $1 \leq j \leq |\Sigma|$. Given a sample from this model we can run the PDFA to generate a sequence of T symbols. Using ξ_t to denote the state of the PDFA at position t in the sequence:

$$\xi_0 = q_0, \qquad x_0 \sim \boldsymbol{\pi}_{q_0}, \qquad \xi_t = \delta(\xi_{t-1}, x_{t-1}), \qquad x_t \sim \boldsymbol{\pi}_{\xi_t}$$

We choose this particular inductive bias, with transitions tied together within a column of δ, because we wanted the most recent symbol emission to be informative about what the next state is. If we instead had a single Dirichlet prior over all elements of δ, transitions to a few states would be highly likely no matter the context and those states would dominate the behavior of the automata. If we tied together rows of δ instead of columns, being in a particular state would tell us more about the sequence of states we came from than the symbols that got us there.

Note that this prior stipulates a fully connected PDFA in which all states may transition to all others and all symbols may be emitted from each state. This is slightly different that the canonical finite state machine literature where sparse connectivity is usually the norm.

2.3.2 PDFA Inference

Given observational data, we are interested in learning a posterior distribution over PDFAs. We do this by Gibbs sampling the transition matrix δ with $\boldsymbol{\pi}$ and $\boldsymbol{\phi}_j$ integrated out. To start inference we need the likelihood function for a fixed PDFA; it is given by

$$p(x_{0:T}|\pi, \delta) = \pi(\xi_0, x_0) \prod_{t=1}^{T} \pi(\xi_t, x_t).$$

Remember that $\xi_t|\xi_{t-1}, x_{t-1}$ is deterministic given the transition function δ. We can marginalize π out of this expression and express the likelihood of the data in a form that depends only on the counts of symbols emitted from each state. Define the count matrix c for the sequence $x_{0:T}$ and transition matrix δ as $c_{ij} = \sum_{t=0}^{T} I_{ij}(\xi_t, x_t)$, where $I_{ij}(\xi_t, x_t)$ is an indicator function for the automaton being in state q_i when it generates x_t, i.e. $\xi_t = q_i$ and $x_t = \sigma_j$. This matrix $c = [c_{ij}]$ gives the number of times each symbol is emitted from each state. Due to multinomial-Dirichlet conjugacy we can express the probability of a sequence given the transition function δ, the count matrix c and β:

$$p(x_{0:T}|\delta, c, \beta) = \int p(x_{0:T}|\pi, \delta)p(\pi|\beta)d\pi = \prod_{i=0}^{|Q|-1} \frac{\Gamma(\beta)}{\Gamma(\frac{\beta}{|\Sigma|})^{|\Sigma|}} \frac{\prod_{j=1}^{|\Sigma|} \Gamma(\frac{\beta}{|\Sigma|} + c_{ij})}{\Gamma(\beta + \sum_{j=1}^{|\Sigma|} c_{ij})} \tag{2.3}$$

If the transition matrix δ is observed we have a closed-form expression for its likelihood given $\boldsymbol{\mu}$ with all $\boldsymbol{\phi}_j$'s marginalized out. Let v_{ij} be the number of times state q_i is transitioned to given that σ_j was the last symbol emitted, i.e. v_{ij} is the number of times $\delta_{i'j} = q_i$ for all states i' in the column j. The marginal likelihood of δ in terms of $\boldsymbol{\mu}$ is then:

$$p(\delta|\boldsymbol{\mu}, \alpha) \;=\; \int p(\delta|\phi)p(\phi|\boldsymbol{\mu}, \alpha)d\phi = \prod_{j=1}^{|\Sigma|} \frac{\Gamma(\alpha)}{\prod_{i=0}^{|Q|-1}\Gamma(\alpha\mu_i)} \frac{\prod_{i=0}^{|Q|-1}\Gamma(\alpha\mu_i + v_{ij})}{\Gamma(\alpha + |Q|)} \tag{2.4}$$

We perform posterior inference in the finite model by sampling elements of δ and the vector $\boldsymbol{\mu}$. One can sample δ_{ij} given the rest of the matrix δ_{-ij} using

$$p(\delta_{ij}|\delta_{-ij}, x_{0:T}, \boldsymbol{\mu}, \alpha) \propto p(x_{0:T}|\delta_{ij}, \delta_{-ij})p(\delta_{ij}|\delta_{-ij}, \boldsymbol{\mu}, \alpha) \tag{2.5}$$

Both terms on the right hand side of this equation have closed-form expressions, the first given in (2.3). The second can be found from (2.4) and is

$$P(\delta_{ij} = q_{i'}|\delta_{-ij}, \alpha, \boldsymbol{\mu}) = \frac{\alpha\mu_{i'} + v_{i'j}}{\alpha + |Q| - 1} \tag{2.6}$$

where $v_{i'j}$ is the number of elements in column j equal to $q_{i'}$ excluding δ_{ij}. As $|Q|$ is finite, we compute (2.5) for all values of δ_{ij} and normalize to produce the required conditional probability distribution.

Note that in (2.3), the count matrix c may be profoundly impacted by changing even a single element of δ. The values in c depend on the specific sequence of states the automata used to generate x. Changing the value of a single element of δ affects the state trajectory the PDFA must follow to generate $x_{0:T}$. Among other things this means that some elements of c that were nonzero may become zero, and vice versa.

We can reduce the computational cost of inference by deleting transitions δ_{ij} for which the corresponding counts c_{ij} become 0. In practical sampler implementations this means that one need not even represent transitions corresponding to zero counts. The likelihood of the data (2.3) does not depend on the value of δ_{ij} if symbol σ_j is never emitted while the machine is in state q_i. In this case sampling from (2.5) is the same as sampling without conditioning on the data at all. Thus, if while sampling we change some transition that renders $c_{ij} = 0$ for some values for each of i and j, we can delete δ_{ij} until another transition is changed such that c_{ij} becomes nonzero again, when we sample δ_{ij} anew. Under the marginal joint distribution of a column of δ the row entries in that column are exchangeable, and so deleting an entry of δ has the same effect as marginalizing it out. When all δ_{ij} for some state q_i are marginalized out, we can say the state itself is marginalized out. When we delete an element from a column of δ, we replace the $|Q| - 1$ in the denominator of (2.6) with $D_j^+ = \sum_{i=0}^{|Q|-1} I(v_{ij} \neq 0)$, the number of entries in the jth column of δ that are *not* marginalized out yielding

$$P(\delta_{ij} = q_{i'} | \delta_{-ij}, \alpha, \boldsymbol{\mu}) = \frac{\alpha \mu_{i'} + v_{i'j}}{\alpha + D_j^+}. \tag{2.7}$$

If when sampling δ_{ij} it is assigned it a state $q_{i'}$ such that some $c_{i'j'}$ which was zero is now nonzero, we simply reinstantiate $\delta_{i'j'}$ by drawing from (2.7) and update $D_{j'}^+$. When sampling a single δ_{ij} there can be many such transitions as the path through the machine dictated by $x_{0:T}$ may use many transitions in δ that were deleted. In this case we update incrementally, increasing D_j^+ and v_{ij} as we go.

While it is possible to construct a Gibbs sampler using (2.5) in this collapsed representation, such a sampler requires a Monte Carlo integration over a potentially large subset of the marginalized-out transitions in δ, which may be costly. A simpler strategy is to pretend that all entries of δ exist but are sampled in a "just-in-time" manner. This gives rise to a Metropolis Hastings (MH) sampler for δ where the proposed value for δ_{ij} is either one of the instantiated states or any one of the equivalent marginalized out states. Any time any marginalized out element of δ is required we can pretend as if we had just sampled its value, and we know that because its value had no effect on the likelihood of the data, we know that it would have been sampled directly from (2.7). It is in this sense that all marginalized out states are equivalent – we known nothing more about their connectivity structure than that given by the prior in (2.7).

For the MH sampler, denote the set of non-marginalized out δ entries $\delta^+ = \{\delta_{ij} : c_{ij} > 0\}$. We propose a new value q_{i^*} for one $\delta_{ij} \in \delta^+$ according to (2.7). The conditional posterior probability of this proposal is proportional to $p(x_{0:T}|\delta_{ij} = q_{i^*}, \delta^+_{-ij})P(\delta_{ij} = q_{i^*}|\delta^+_{-ij})$. The Hastings correction exactly cancels out the proposal probability in the accept/reject ratio leaving an MH accept probability for the δ_{ij} being set to q_{i^*} given that its previous value was $q_{i'}$ of

$$\alpha(\delta_{ij} = q_{i^*}|\delta_{ij} = q_{i'}) = \min\left(1, \frac{p(x_{0:T}|\delta_{ij} = q_{i^*}, \delta^+_{-ij})}{p(x_{0:T}|\delta_{ij} = q_{i'}, \delta^+_{-ij})}\right). \tag{2.8}$$

Whether q_{i^*} is marginalized out or not, evaluating $p(x_{0:T}|\delta_{ij} = q_{i^*}, \delta^+_{-ij})$ may require reinstantiating marginalized out elements of δ. As before, these values are sampled from (2.7) on a just-in-time schedule. If the new value is accepted, all $\delta_{ij} \in \delta^+$ for which $c_{ij} = 0$ are removed, and then move to the next transition in δ to sample.

In the finite case, one can sample $\boldsymbol{\mu}$ by Metropolis-Hastings or use a MAP estimate as in (MacKay and Peto, 1995). Hyperparameters α, β and γ can be sampled via Metropolis-Hastings updates. In our experiments we use Gamma(1,1) hyperpriors.

2.3.3 The Probabilistic Deterministic Infinite Automaton

We would like to avoid placing a strict upper bound on the number of states so that model complexity can grow with the amount of training data. To see how to do this, consider what happens when $|Q| \to \infty$. In this case, the right hand side of equations (2.1) and (2.2) must be replaced by infinite dimensional alternatives

$$\begin{aligned}
\boldsymbol{\mu} &\sim \mathrm{PY}(\gamma, d_0, H) \\
\boldsymbol{\phi}_j &\sim \mathrm{PY}(\alpha, d, \boldsymbol{\mu}) \\
\delta_{ij} &\sim \boldsymbol{\phi}_j
\end{aligned}$$

where PY stands for Pitman Yor process and H in our case is a geometric distribution over the integers with parameter λ. The resulting hierarchical model becomes the hierarchical Pitman-Yor process (HPYP) over a discrete alphabet (Teh, 2006). The discount parameters d_0 and d are particular to the infinite case, and when both are zero the HPYP becomes the well known hierarchical Dirichlet process (HDP), which is the infinite dimensional limit of (2.1) and (2.2) (Teh et al., 2006b). Given a finite amount of data, there can only be nonzero counts for a finite number of state/symbol pairs, so our marginalization procedure from the finite case will yield a δ with at most T elements. Denote these non-marginalized out entries by δ^+. We can sample the elements of δ^+ as before using (2.8) provided that we can propose from the HPYP. In many HPYP sampler representations this is easy to do. We use the Chinese restaurant franchise representation (Teh et al., 2006b) in which the posterior predictive distribution of δ_{ij} given δ^+_{-ij} can be expressed with $\boldsymbol{\phi}_j$ and $\boldsymbol{\mu}$ integrated out as

$$P(\delta_{ij} = q_{i'}|\delta^+_{-ij}, \alpha, \gamma) = \mathbb{E}\left[\frac{v_{i'j} - k_{i'j}d}{\alpha + D^+_j} + \frac{\alpha + k_{\cdot j}d}{\alpha + D^+_j}\left(\frac{w_{i'} - \kappa_{i'}d_0}{\gamma + w_{\cdot}} + \frac{\gamma + \kappa_{\cdot}d_0}{\gamma + w_{\cdot}}H(q_{i'})\right)\right] \quad (2.9)$$

where $w_{i'}$, $k_{i'j}$, $\kappa_{i'}$, $w_\cdot = \sum_i w_i$, $k_{\cdot j} = \sum_i k_{ij}$, and $\kappa_\cdot = \sum_i \kappa_i$ are stochastic bookkeeping counts required by the Chinese Restaurant franchise sampler. These counts must themselves be sampled (Teh et al., 2006b). The discount hyperparameters can also be sampled by Metropolis-Hastings.

2.4 Posterior Inference over PDFAs

We perform posterior inference by sampling assignments for δ_{ij} individually. Rather than ordinary Gibbs sampling, we use a mixed Gibbs/Metropolis-Hastings update. To see why, consider the following case: δ_{ij} is the only sampled element of δ that is assigned to the state $q_{i'}$. When we removed δ_{ij} from the counts to sample it, the probability of assigning it back to $q_{i'}$ becomes zero, and even if it is assigned to some new state $q_{i''}$, the probability that $\delta_{i'j} = \delta_{i''j}$ for all j visited by the data is low. We do not want to forget a good sample, so instead we propose a new δ_{ij} and accept or reject according to the usual Metropolis-Hastings ratio.

Samples from the CRF are exchangeable, so we can remove δ_{ij} and propose a sample δij^* according to the CRF given δ^T_{-ij}, the elements of δ visited by $x_{0:T}$ excluding δ_{ij}. This is the prior probability excluding the data, which cancels with the equivalent term in the posterior, meaning that the accept probability $\alpha(\delta_{ij}, \delta^*_{ij})$ is given by the ratio of the likelihood of the data

$$\alpha(\delta_{ij}, \delta^*_{ij}) = \min\left(1, \frac{p(x_{0:T}|\delta^*_{ij}, \delta^T_{-ij})}{p(x_{0:T}|\delta_{ij}, \delta^T_{-ij})}\right)$$

In general the numerator cannot be evaluated because changing δ_{ij} means changing the entire sequence of states visited by the data after δ_{ij} is first visited. In practice we estimate the numerator by Monte Carlo approximation, sampling elements of δ according to the CRF as they are first visited by the data. Changing one element of δ means many already-sampled state/symbol pairs may never be visited by the data, and have no effect on the likelihood. The posterior probability of any value for these elements of δ is the same as the prior, and therefore we may remove them from δ^T after accepting a new δ_{ij}.

2.5 Theory and Related Work

Despite a wealth of research on both the theory and practice of learning PDFAs, the work presented here is, to our knowledge, the first algorithm to generate samples from a posterior over automata rather than returning a deterministic estimate. Prior work has focused on greedy algorithms, which work by either merging or splitting elements of Q according to some statistical test. Theoretical work has shown that PDFAs are both identifiable in the limit and, with a few restrictions on the model class[4], are also PAC-learnable[5] using KL divergence between automata as a measure of accuracy.

2.5.1 Mixtures of PDFA

The PDIA posterior distribution takes the form of an infinite mixture of PDFAs. In practice, we run a sampler for some number of iterations and approximate the posterior with a finite mixture of PDFAs. For this reason, we now consider the expressive power of finite mixtures of PDFAs. Mixtures of PDFAs are strictly more expressive than PDFAs, but strictly less expressive than hidden Markov models. Probabilistic *non*-deterministic finite automata (PNFA) are a strictly larger model class than PDFAs. For example, the PNFA in 2.1(a)

[4]A polynomial number of states in the true automata, a minimum divergence between the distribution over strings that follow two states, and a polynomial bound on the probability of generating strings above a certain length

[5]A model class is said to be *PAC-learnable* if there is an algorithm that will return, in time polynomial in $\frac{1}{\delta}, \frac{1}{\epsilon}$ and $|D|$, an estimate within an accuracy ϵ of the true model from $|D|$ examples with probability $1 - \delta$.

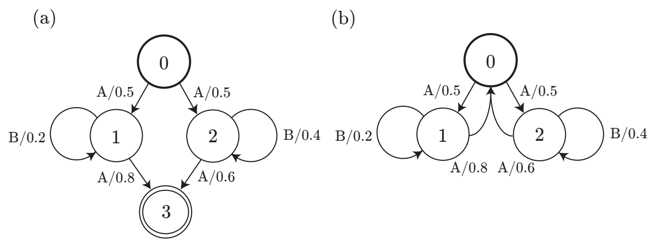


Figure 2.1: Two PNFAs outside the class of PDFAs. (a) can be represented by a mixture of two PDFAs, one following the right branch from state 0, the other following the left branch. (b), in contrast, cannot be represented by any finite mixture of PDFAs.

cannot be expressed as a PDFA (Dupont et al., 2005). However, it can be expressed as a mixture of two PDFAs, one with $Q = \{q_0, q_1, q_3\}$ and the other with $Q = \{q_0, q_2, q_3\}$. Thus mixtures of PDFAs are a strictly larger model class than PDFAs. In general, any PNFA where the nondeterministic transitions can only be visited once can be expressed as a mixture of PDFAs. However, if we replace transitions to q_3 with transitions to q_0, as in 2.1(b), there is no longer any equivalent finite mixture of PDFAs, since the nondeterministic branch from q_0 can be visited an arbitrary number of times.

2.5.2 State Merging Algorithms

A variety of algorithms work by starting with the trivial automata built from the prefix tree of the data, and generalizes by merging states that pass a similarity test. Merging two states is not trivial: if $\delta(q_1, s_j) \neq \delta(q_2, s_j)$, then merging q_1 and q_2 will produce a state with nondeterministic transitions. This is avoided by recursively merging the states δ_{1j} and δ_{2j} until the resulting automata is deterministic. The result is a *quotient automata* of the original. One of the earliest algorithms to use this method is ALERGIA (Carrasco and Oncina, 1994), which uses a test based on the Hoeffding bound to decide whether to merge states, and was proven to converge to the true automata in the limit of infinite data.

Later algorithms have mostly focused on improving the state merging test. MDI uses a test based on the KL divergence between automata, and penalizes automata with many states. Thus it can be interpreted as a greedy maximum posterior estimator with a minimum description length prior. Empirically, it has better predictive performance than ALERGIA on natural language data.

(Clark and Thollard, 2004) presented a state-merging algorithm for cyclic PDFAs that is PAC-learnable given very few restrictions on the model class. While the emphasis was on theoretical rather than empirical performance, it was also shown to work well from small data. Further work has improved upon these results (Castro and Gavaldà, 2008), making the theoretical bounds tighter, using measures of similarity other than KL divergence, and also showing superior results from small samples.

2.5.3 State Splitting Algorithms

State splitting algorithms, by contrast, start with the most general single-state automata and become more selective by adding more states. (Ron et al., 1996) learned a variable-order Markov model using a state-splitting algorithm, where a state corresponding to a string suffix was split into states corresponding to longer suffixes according to some test. Splitting might occasionally produce nondeterministic transitions. For example, after splitting the context 01 into 001 and 101, the context 0 and symbol 1 might transition to either one, unless the context 0 is also split into 10 and 00, but these probabilistic suffix trees could be mapped onto PDFAs after learning.

CSSR (Shalizi and Shalizi, 2004) took a similar approach, but with a model class that contains all PDFAs. Their philosophical motivation, similar to ours, is to learn the minimal sufficient statistics for predicting the future given the past. Given a stationary sequence with infinitely long past and future, those statistics form a PDFA, which they call a *causal state machine.* Each state is a set of suffixes, rather than a single suffix, which means the model class includes all PDFAs. If the predictive distribution for a suffix passes a Kolmogorov-Smirnov test, that state is split in two and suffixes in the original state are divided. Nondeterministic transitions are removed recursively by backing up to the states preceding a split state and splitting in the natural way, much like the reverse of how states are merged when forming quotient automata.

2.5.4 Spectral Methods

In recent years spectral methods have also been applied to the problem of automata learning, paralleling a broad resurgence in interest in the machine learning community. Much of this interest is likely attributable to their speed and ease of implementation, and theoretical results guaranteeing consistency. The downside of these methods is that they generally require a much larger sample size than those that directly fit a generative model. Inspired by classic work in system identification on spectral methods for learning linear Gaussian systems (which we discuss in Chapter 3) Hsu et al. (2012) introduced a simple algorithm for learning hidden Markov models directly from the SVD of matrices of 3rd-order statistics. Further developments have produced spectral methods for a large class of sequence models such as weighted automata (Bailly, 2011; Balle and Mohri, 2012; Balle et al., 2013), finite state transducers (Balle et al., 2011) and even context free grammars (Cohen et al., 2012).

2.5.5 Applications to Neuroscience

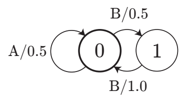
Others have looked at learning PDFA models of neural activity, most notably (Haslinger et al., 2010), who applied the Causal State Splitting Reconstruction (CSSR) algorithm (Shalizi and Shalizi, 2004) to evoked spike trains from the rat barrel cortex, and were able to automatically uncover simple features of spiking neurons, such as refractory periods. They also decompose the total entropy of the generative process (states and observations) over a finite window of time into a complexity term (the entropy of the latent state), an internal entropy rate (the entropy *rate* of the latent state) and a "residual randomness" (the entropy of the observation given the state). The complexity term is a constant while the latter two terms grow linearly with the length of the data, thus the complexity term captures the *subextensive* entropy of the generative process. One downside of CSSR is that, as it is based on frequentist statistical methods, it can be prone to overfitting without some kind of regularization. Our method is, to our knowledge, the first that uses fully Bayesian methods to learn PDFAs.

While it is interesting that methods like CSSR can automatically discover refractory periods, there are many other statistical models that can capture this particular feature of neural activity. As is well known, adding a linear spike history filter to a linear-nonlinear-Poisson model is sufficient to model a number of history-dependent effects of neural activity, including refractory periods (Paninski et al., 2004). In our analysis of neural data we are more interested in capturing long-range temporal correlations, which is far more difficult to do with spike history terms.

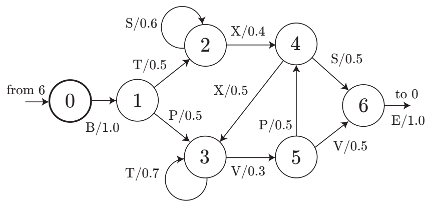
2.6 Experiments and Results

2.6.1 Synthetic Grammars

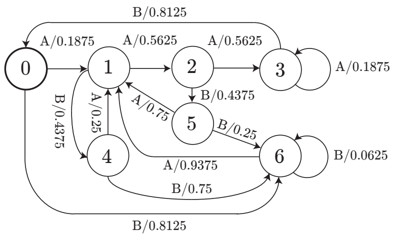
To test if we can learn the generative mechanism given our inductive bias, we trained the PDIA on data from three synthetic grammars: the even process (Shalizi and Shalizi, 2004), the Reber grammar (Reber, 1967) and the Feldman grammar (Feldman and Hanna, 1966), which have up to 7 states and 7 symbols in the alphabet. In each case the mean number of states discovered by the model approached the correct number as more data was used in training. Results are presented in Figure 2.2. Furthermore, the predictive performance of the PDIA was nearly equivalent to the actual data generating mechanism.



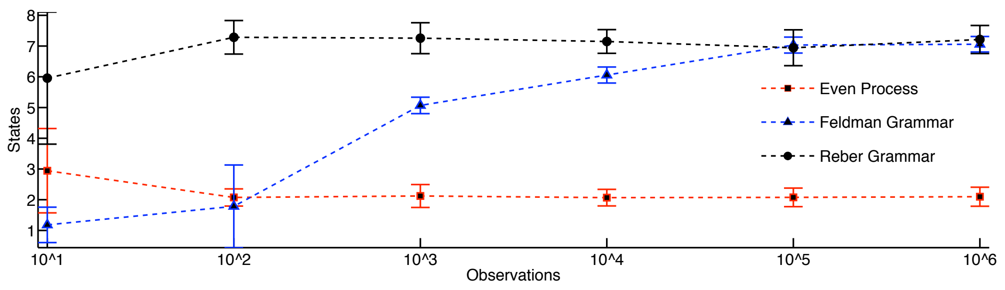
(a) Even



(b) Reber



(c) Feldman



(d) Posterior marginal PDIA state cardinality distribution

Figure 2.2: Three synthetic PDFAs: (a) even process (Shalizi and Shalizi, 2004), (b) Reber grammar (Reber, 1967), (c) Feldman grammar (Feldman and Hanna, 1966). (d) posterior mean and standard deviation of number of states discovered during PDIA inference for varying amounts of data generated by each of the synthetic PDFAs. PDIA inference discovers PDFAs with the correct number of states

2.6.2 Natural Language and DNA

To test our PDIA inference approach we evaluated it on discrete natural sequence prediction and compared its performance to HMMs and smoothed n-gram models. We trained the models on two datasets: a character sequence from *Alice in Wonderland* (Carroll, 1865) and a short sequence of mouse DNA. The *Alice in Wonderland* (AIW) dataset was preprocessed to remove all characters but letters and spaces, shift all letters from upper to lower case,

	PDIA	PDIA-MAP	HMM-EM	bigram	trigram	4-gram	5-gram	6-gram	SSM
AIW	5.13	5.46	7.89	9.71	6.45	5.13	4.80	4.69	4.78
	365.6	379	52	28	382	2,023	5,592	10,838	19,358
DNA	3.72	3.72	3.76	3.77	3.75	3.74	3.73	3.72	3.56
	64.7	54	19	5	21	85	341	1,365	314,166

Table 2.1: PDIA inference performance relative to HMM and fixed order Markov models. Top rows: perplexity. Bottom rows: number of states in each model. For the PDIA this is an average number.

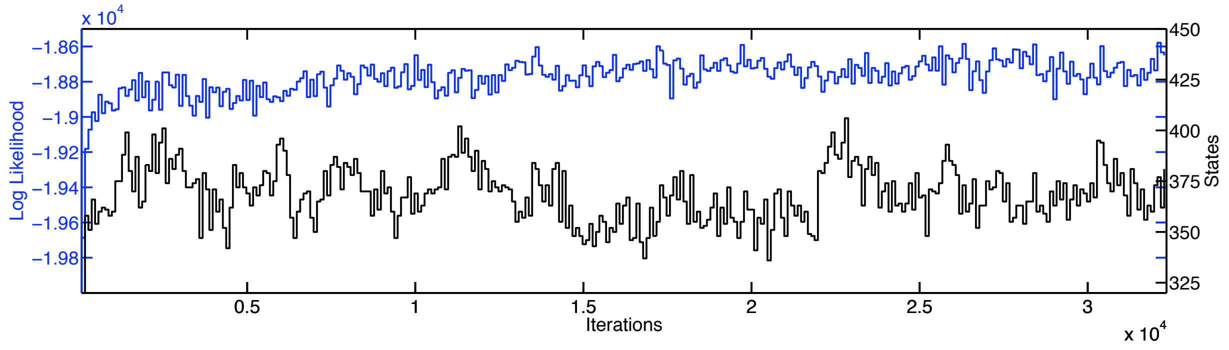


Figure 2.3: Subsampled PDIA sampler trace for Alice in Wonderland. The top trace is the joint log likelihood of the model and training data, the bottom trace is the number of states.

and split along sentence dividers to yield a 27-character alphabet (a-z and space). We trained on 100 random sentences (9,986 characters) and tested on 50 random sentences (3,891 characters). The mouse DNA dataset consisted of a fragment of chromosome 2 with 194,173 base pairs, which we treated as a single unbroken string. We used the first 150,000 base pairs for training and the rest for testing. For AIW, the state of the PDIA model was always set to q_0 at the start of each sentence. For DNA, the state of the PDIA model at the start of the test data was set to the last state of the model after accepting the training data. We placed Gamma(1,1) priors over α, β and γ, set $\lambda = .001$, and used uniform priors for d_0 and d.

We evaluated the performance of the learned models by calculating the average per character predictive perplexity of the test data. For training data $x_{1:T}$ and test data $y_{1:T'}$ this is given by $2^{-\frac{1}{T'}\log_2 P(y_{1:T'}|x_{1:T})}$. It is a measure of the average uncertainty the model has about what character comes next given the sequence up to that point, and is at most $|\Sigma|$. We evaluated the probability of the test data incrementally, integrating the test data into the model in the standard Bayesian way.

Test perplexity results are shown in Table 2.1 on the first line of each subtable. Each sample passed through every instantiated transition. Every fifth sample for AIW and every tenth sample for DNA after burn-in was used for prediction. For AIW, we ran 15,000 burn-in samples and used 3,500 samples for predictive inference. Subsampled sampler diagnostic plots are shown in Figure 2.3 that demonstrate the convergence properties of our sampler. When modeling the DNA dataset we burn-in for 1,000 samples and use 900 samples for inference. For the smoothed n-gram models, we report thousand-sample average perplexity results for hierarchical Pitman-Yor process (HPYP) (Teh, 2006) models of varying Markov order (1 through 5 notated as bigram through 6-gram) after burning each model in for one hundred samples. We also show the performance of the single particle incremental variant of the sequence memoizer (SM) (Gasthaus et al., 2010), the SM being the limit of an n-gram model as $n \to \infty$. We also show results for a hidden Markov model (HMM) (Murphy, 2005) trained using expectation-maximization (EM). We determined the best number of hidden states by cross-validation on the test data (a procedure used here to produce optimistic HMM performance for comparison purposes only).

The performance of the PDIA exceeds that of the HMM and is approximately equal to that of a smoothed 4-gram model, though it does not outperform very deep, smoothed Markov models. This is in contrast to (Thollard, 2001), which found that PDFAs trained on natural language data were able to predict as well as *unsmoothed* trigrams, but were significantly worse than smoothed trigrams, even when averaging over multiple learned

PDFAs. As can be seen in the second line of each subtable in Table 2.1, the MAP number of states learned by the PDIA is significantly lower than that of the n-gram model with equal predictive performance.

For natural language, we can also assess the quality of model fit by looking at data generated from learned PDFA. Some example synthetic data:

> what a mushroom very softly have the the way either little about a deal she what to kept i to when b...
>
> what you her and took when pim bill alice himself ignvy conversationer after treat eye going very to...
>
> seside must upon to the a othering in for the the i of i him of hrisall a is either mock turtle and...
>
> however nor rats come perself for everywheelsome something ll in hoor of but her said you heople was...
>
> whistled inqueer hersonate it doing daiek the ll the she be away the the queen than of that miss pea...

Data sampled from the model are a mixture of recognizable words, entire phrases memorized from the dataset (e.g. "mock turtle") and nonsense words that still resemble English at the level of syllables (e.g. "everywheelsome"). This is not quite at the level of more recent advances in using recurrent neural networks to model natural language (Sutskever et al., 2011; Hermans and Schrauwen, 2013), where sampled data typically reproduce entire grammatically-sensible phrases (if not full sentences), but those models are trained on three to four orders of magnitude more data, and the state space is significantly larger. We find it remarkable that this much information about the statistics of natural language can be captured by a character-level model with only a few hundred states trained on only 10^5

characters.

Unlike the HMM, the computational complexity of PDFA prediction does not depend on the number of states in the model because only a single path through the states is followed. This means that the asymptotic cost of prediction for the PDIA is $\mathcal{O}(LT')$, where L is the number of posterior samples and T' is the length of the test sequence. For any single HMM it is $\mathcal{O}(KT')$, where K is the number of states in the HMM. This is because all possible paths must be followed to achieve the given HMM predictive performance (although a subset of possible paths could be followed if doing approximate inference). In PDIA inference we too can choose the number of samples used for prediction, but here even a single sample has empirical prediction performance superior to averaging over all paths in an HMM. The computational complexity of smoothing n-gram inference is equivalent to PDIA inference, however, the storage cost for the large n-gram models is significantly higher than that of the estimated PDIA for the same predictive performance.

2.6.3 Neural Data

We assess how effective the PDIA is at capturing complex temporal dynamics in single neurons, which are well suited for modeling by PDFAs because of the small space of observations (binary, if one is only measuring the presence or absence of a spike) but possible long-time scale correlations.

We run our analysis on data from (Gal et al., 2010). In that study rat cortical neurons were grown in culture, synaptically blocked and stimulated extracellularly for up to 55 hours. They found that, if stimulated at high frequency, neurons will enter an intermittent state where spike responses become unreliable and chaotic, possibly because there is insufficient time for ionic gradients to recover. On long time scales, the response properties of neurons change dramatically, switching between long silent periods and periods of variable responsiveness. The statistics of these long time series show power law correlations by a number of measures. Here we investigate whether the PDIA can learn models that reproduce these power laws statistics.

We train a PDIA on data from a single neuron that was stimulated at 20 Hz for 24 hours. Each observation in the time series records whether or not the recorded neuron spiked after stimulation. For the first several minutes the neuron reliably spiked after every stimulation, but eventually settled into an intermittent state where it fired apparently at random following stimulation. After several hours of stimulation the neuron would then slip into long quiescent periods of no firing at all. This can be seen in Figs. 2.4, 2.5 and 2.6. The full training set consisted of 1727707 time bins, the full 24 hours of recording. We ran MCMC for 1000 samples and generated data from the last sample.

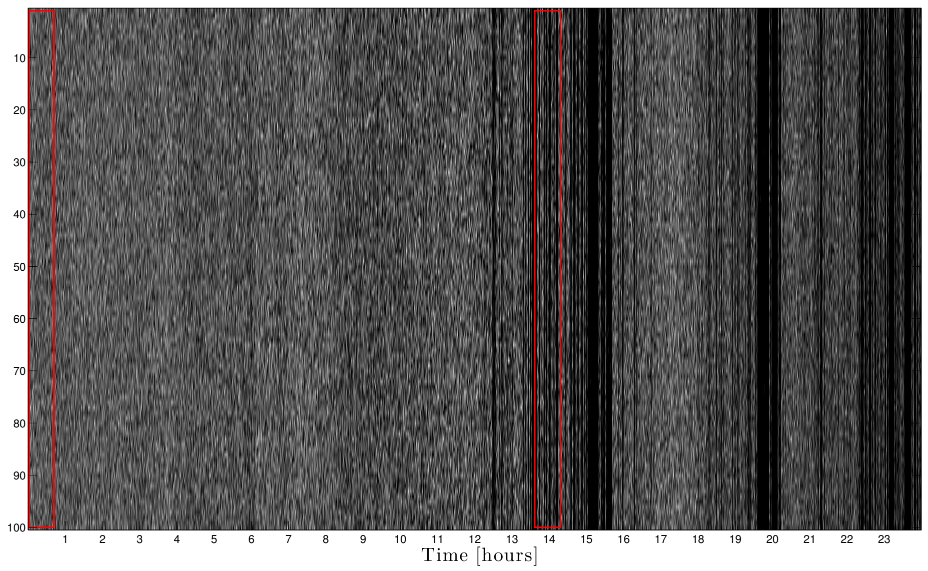


Figure 2.4: Response of a cultured neuron to periodic stimulation on long time scales, from Gal et al. (2010). The segments of data show in Fig. 2.5 and Fig. 2.6 are outlined in red.

As a control, we also fit a generalized linear model with logistic link function and

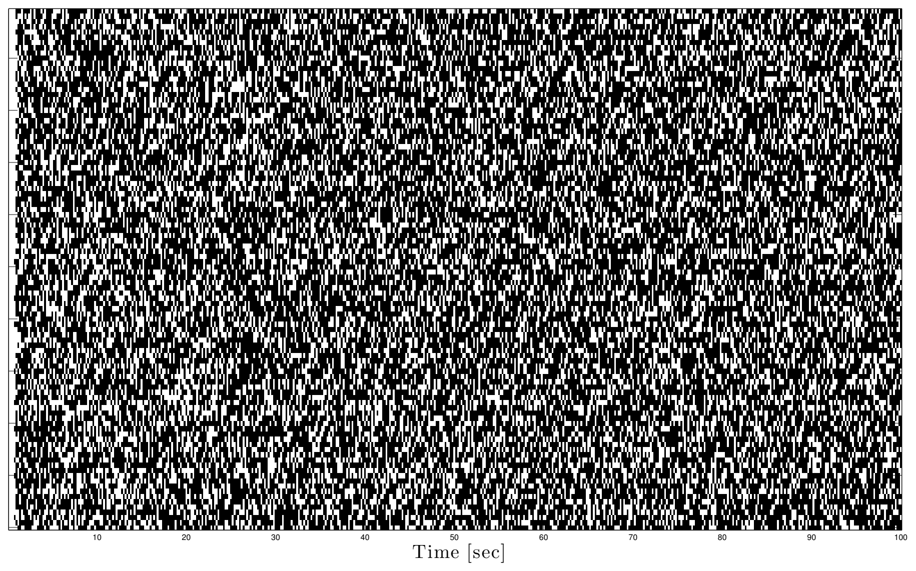


Figure 2.5: Early response of a cultured neuron to periodic stimulation. The neuron spikes intermittently but with consistent probability.

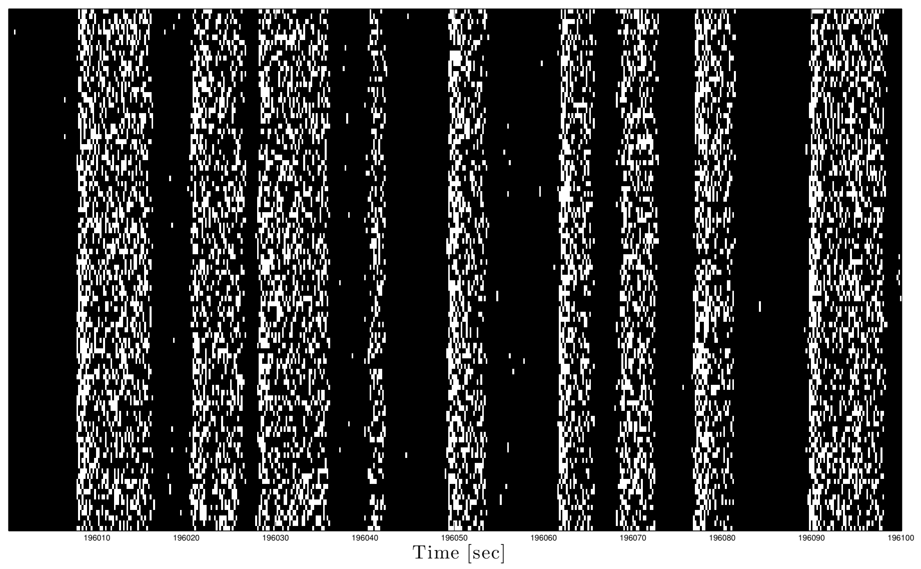


Figure 2.6: Late response of a cultured neuron to periodic stimulation. Not only does the neuron spike intermittently, but it also frequently flips into a quiescent state.

Bernoulli noise model:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1},\ldots,\mathbf{x}_{t-k}) = \left[\frac{1}{e^{-\sum_{\tau=1}^{k}\mathbf{x}_{t-\tau}\theta_\tau - b}+1}\right]^{\mathbf{x}_t}\left[\frac{1}{e^{\sum_{\tau=1}^{k}\mathbf{x}_{t-\tau}\theta_\tau + b}+1}\right]^{1-\mathbf{x}_t} \tag{2.10}$$

where the parameters $\{\theta_\tau\}_{\tau=1}^{k}$ and b are fit by maximum likelihood. We refer to this model with history length k as GLM(k). We trained the Bernoulli GLM with history lengths of 20, 50 and 100 time bins. Beyond this length the qualitative behavior of the data did not change.

Qualitatively, data generated from a trained PDIA closely resembled the original data on medium-length time scales on the order of minutes (Fig. 2.7), though it seemed to lack some of the very-long-time-scale features of the original data. The data generated by the GLM was high variable depending on the history length. Shorter histories matched the average firing rate but did not find medium-time-scale structures, while histories on the order of 50 time bins alternated between active and quiescent periods much like the real data, but the active periods were saturated well above actual values, while very long history dependencies led to the firing rates saturating uniformly.

To quantify this observation, we reproduced the spectral analysis of Gal et al. (2010) on model data (Fig. 2.9(a)). Consistent with our qualitative observations, the power spectral density of model data from the PDIA most closely tracked the true PSD across a range of frequencies. Only below 10^{-2} Hz did the two diverge noticeably. Data from GLM(50), which exhibited the most interesting long-time-scale correlations, actually *exceeded* the ground truth in power at very low frequencies. We also looked at the distribution of lengths of sequences of all spikes or all silences. Due to the presence of very long quiescent periods, there are a small number of sequences that are extremely long, and plotted on a log-log plot the ground truth looks roughly like a power law (though we did not quantify this). The PDIA data closely matched this distribution for all but the very longest sequences, while none of the GLM data reproduced the spread of long but rare quiescent periods (Fig. 2.9(b)). Sequences of contiguous spikes more closely followed an exponential distribution, and again were most closely matched to the PDIA (Fig. 2.9(c)). Overall this shows that the PDIA is able to capture structure on time scales up to nearly 1000 time steps, and by almost all measures outperforms models with a simple linear dependence on the past. A similar modeling study was performed by Soudry and Meir (2014), based on a biophysical model with slow and fast variables, showing that correlations on long time scales (days) can be produced by neurons that only integrate information on a much shorter time scale (minutes). The novelty of our result is in the ability to learn the correlation structure on long time scales without any prior biophysical knowledge, as there is absolutely nothing in our learning algorithm that incorporates domain knowledge about ion channels or other neuronal biology. This could be useful in neural engineering applications where the biophysical details of a new system might not be known a priori and would have to be learned after a device is implanted.

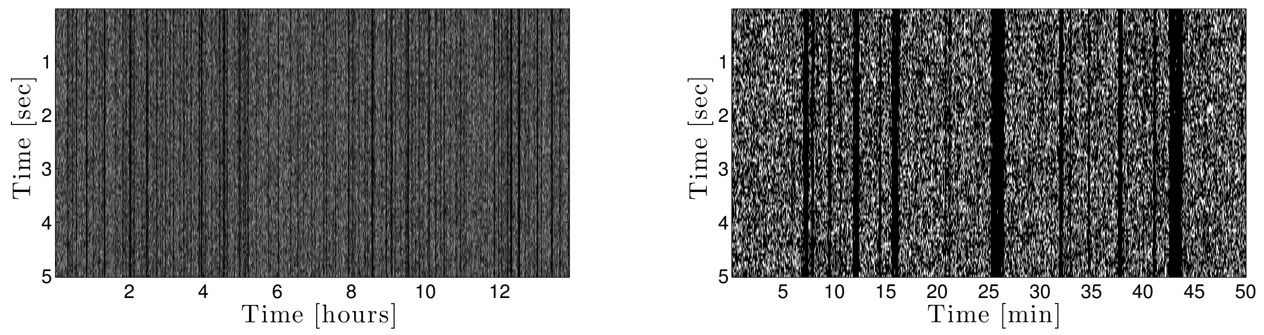


Figure 2.7: Data sampled from a PDIA trained on 24 hours of the data in 2.4. Left: long time scale patterns in the data. Right: medium time scale patterns in the data. The PDIA is able to learn patterns on the order of hundreds to thousands of time bins, or several minutes of recorded data.

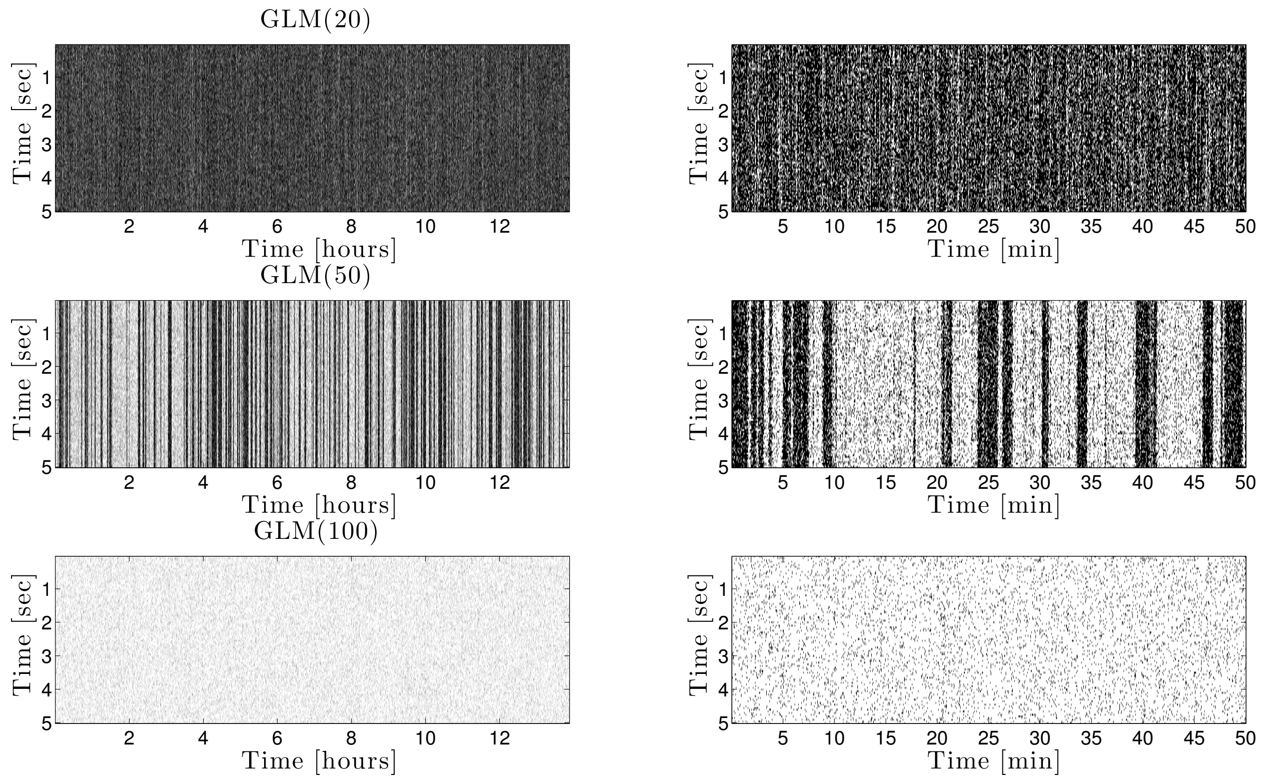
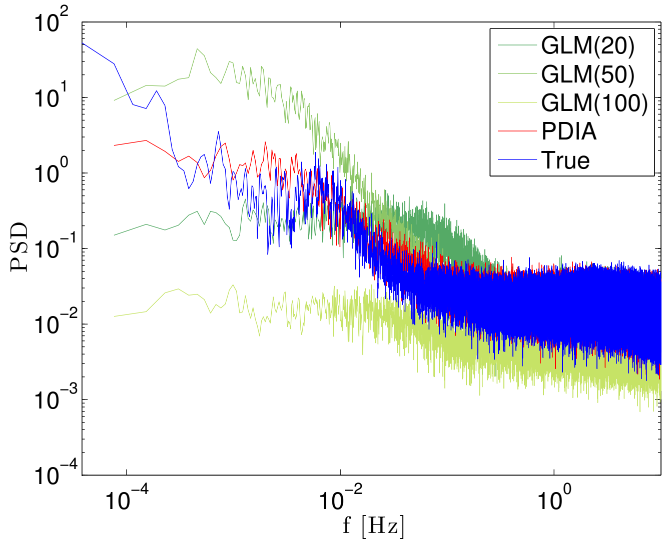
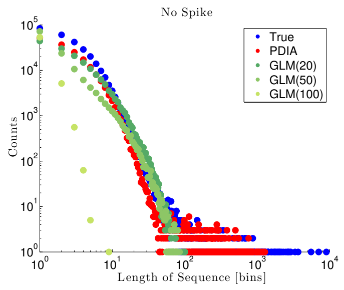


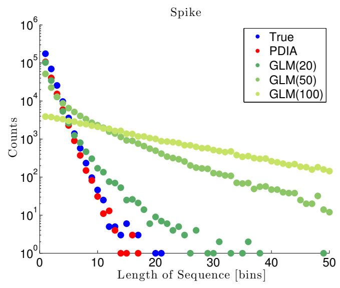
Figure 2.8: Data sampled from Bernoulli GLM trained via maximum likelihood on 24 hours of the data in 2.4. Left: long time scale patterns in the data. Right: medium time scale patterns in the data.



(a)



(b)



(c)

Figure 2.9: Data generated from a PDIA reproduces long time scale correlations better than data generated from generalized linear models. (a) Power spectral density for real and model data. Up to less than 10^{-2} Hz, the PDIA spectral density closely matches the ground truth. (b) The distribution of lengths of consecutive bins without a spike. The PDIA most closely matches the ground truth, only deviating noticeably at the bottom right (rare, long silences).(b) The distribution of lengths of consecutive bins with a spike. Again the PDIA most closely matches the ground truth.

2.7 Discussion

Our Bayesian approach to PDIA inference can be interpreted as a stochastic search procedure for PDFA structure learning where the number of states is unknown. In Section 2.5 we presented evidence that PDFA samples from our PDIA inference algorithm have the same characteristics as the true generative process. This in and of itself may be of interest to the PDFA induction community.

We ourselves are more interested in establishing new ways to produce smoothed predictive conditional distributions. Inference in the PDIA presents a completely new approach to smoothing, smoothing by averaging over PDFA model structure rather than hierarchically smoothing related emission distribution estimates. Our PDIA approach gives us an attractive ability to trade-off between model simplicity in terms of number of states, computational complexity in terms of asymptotic cost of prediction, and predictive perplexity. While our PDIA approach may not yet outperform the best smoothing Markov model approaches in terms of predictive perplexity alone, it does outperform them in terms of model complexity required to achieve the same predictive perplexity, and outperforms HMMs in terms of asymptotic time complexity of prediction. This suggests that a future combination of smoothing over model structure *and* smoothing over emission distributions could produce excellent results. PDIA inference gives researchers another tool to choose from when building models. If very fast prediction is desirable and the predictive perplexity difference between the PDIA and, for instance, the most competitive n-gram is insignificant from an application perspective, then doing finite sample inference in the PDIA offers a significant computational advantage in terms of memory.

We indeed believe the most promising approach to improving PDIA predictive performance is to construct a smoothing hierarchy over the state specific emission distributions, as is done in the smoothing n-gram models. For an n-gram, where every state corresponds to a suffix of the sequence, the predictive distributions for a suffix is smoothed by the predictive distribution for a shorter suffix, for which there are more observations. This makes it possible to increase the size of the model indefinitely without generalization performance suffering (Wood et al., 2009). In the PDIA, by contrast, the predictive probabilities for states are not tied together. Since states of the PDIA are not uniquely identified by suffixes, it is no longer clear what the natural smoothing hierarchy is. It is somewhat surprising that PDIA learning works nearly as well as n-gram modeling even without a smoothing hierarchy for its emission distributions. Imposing a hierarchical smoothing of the PDIA emission distributions remains an open problem.

Chapter 3

Robust Learning of Low-Dimensional Dynamics From Large Neural Ensembles[1]

And even as we, who are now in Space, look down on Flatland and see the inside of all things, so of a certainty there is yet above us some higher, purer region...some yet more spacious Space, some more dimensionable Dimensionality, from the vantage-ground of which we shall look down together upon the revealed insides of solid things

Edwin Abbott Abbott, *Flatland*

Recordings from large populations of neurons make it possible to search for hypothesized low-dimensional dynamics. Finding these dynamics requires models that take into account biophysical constraints and can be fit efficiently and robustly. Here, we present an approach to dimensionality reduction for neural data that is convex, does not make strong

[1]This work has, in part, been published (Pfau, Pnevmatikakis and Paninski. *Advances in Neural Information Processing Systems*,2013)

assumptions about dynamics, does not require averaging over many trials and is extensible to more complex statistical models that combine local and global influences. The results can be combined with spectral methods to learn dynamical systems models. The basic method extends PCA to the exponential family using nuclear norm minimization. We evaluate the effectiveness of this method using an exact decomposition of the Bregman divergence that is analogous to variance explained for PCA. We show on model data that the parameters of latent linear dynamical systems can be recovered, and that even if the dynamics are not stationary we can still recover the true latent subspace. We also demonstrate an extension of nuclear norm minimization that can separate sparse local connections from global latent dynamics. Finally, we demonstrate improved prediction on real neural data from monkey motor cortex compared to fitting linear dynamical models without nuclear norm smoothing.

3.1 Introduction

Progress in neural recording technology has made it possible to record spikes from ever larger populations of neurons (Stevenson and Kording, 2011). Analysis of these large populations suggests that much of the activity can be explained by simple population-level dynamics (Okun et al., 2012). Typically, this low-dimensional activity is extracted by principal component analysis (PCA) (Briggman et al., 2005; Machens et al., 2010; Stopfer et al., 2003), but in recent years a number of extensions have been introduced in the neuroscience literature, including jPCA (Churchland et al., 2012) and demixed principal component analysis (dPCA) (Brendel et al., 2011). A downside of these methods is that they do not treat either the discrete nature of spike data or the positivity of firing rates in a statistically principled way. Standard practice smooths the data substantially or averages it over many trials, losing information about fine temporal structure and inter-trial variability.

One alternative is to fit a more complex statistical model directly from spike data, where temporal dependencies are attributed to latent low dimensional dynamics (Paninski et al., 2010; Yu et al., 2009). Such models can account for the discreteness of spikes by using point-process models for the observations, and can incorporate temporal dependencies into the latent state model. State space models can include complex interactions such as switching linear dynamics (Petreska et al., 2011) and direct coupling between neurons (Kulkarni and Paninski, 2007). These methods have drawbacks too: they are typically fit by approximate EM (Smith and Brown, 2003) or other methods that are prone to local minima, the number of latent dimensions is typically chosen ahead of time, and a certain class of possible dynamics must be chosen before doing dimensionality reduction.

In this chapter we attempt to combine the computational tractability of PCA and related methods with the statistical richness of state space models. Our approach is convex and based on recent advances in system identification using *nuclear norm minimization* (Fazel et al., 2001; Liu and Vandenberghe, 2009; Liu et al., 2013), a convex relaxation of matrix rank minimization. Compared to recent work on spectral methods for fitting state space models (Buesing et al., 2012), our method more easily generalizes to handle different nonlinearities, non-Gaussian, non-linear, and non-stationary latent dynamics, and direct connections between observed neurons. When applied to model data, we find that: (1) low-dimensional subspaces can be accurately recovered, even when the dynamics are unknown and nonstationary (2) standard spectral methods can robustly recover the parameters of state space models when applied to data projected into the recovered subspace (3) the confounding effects of common input for inferring sparse synaptic connectivity can be ameliorated by accounting for low-dimensional dynamics. In applications to real data we find comparable performance to models trained by EM with less computational overhead, particularly as the number of latent dimensions grows.

This chapter is organized as follows. In Sec. 3.2 we introduce the class of models we aim to fit, which we call low-dimensional generalized linear models (LD-GLM). In Sec. 3.3 we present a convex formulation of the parameter learning problem for these models, as well as a generalization of variance explained to LD-GLMs used for evaluating results. In Sec. 3.4 we show how to fit these models using the *alternating direction method of multipliers* (ADMM). In Sec. 3.5 we present results on real and artificial neural datasets. We discuss the results and future directions in Sec. 3.6.

3.2 Low dimensional generalized linear models

Our model is closely related to the generalized linear model (GLM) framework for neural data (Paninski et al., 2004). Unlike the standard GLM, where the inputs driving the neurons are observed, we assume that the driving activity is unobserved, but lies on some low dimensional subspace. This can be a useful way of capturing spontaneous activity, or accounting for strong correlations in large populations of neurons. Thus, instead of fitting a linear receptive field, the goal of learning in low-dimensional GLMs is to accurately recover the latent subspace of activity.

Let $x_t \in \mathbb{R}^m$ be the value of the dynamics at time t. To turn this into spiking activity, we project this into the space of neurons: $y_t = Cx_t + b$ is a vector in $\mathbb{R}^n$, $n \gg m$, where each dimension of y_t corresponds to one neuron. $C \in \mathbb{R}^{n\times m}$ denotes the subspace of the neural population and $b \in \mathbb{R}^n$ the bias vector for all the neurons. As y_t can take on negative values, we cannot use this directly as a firing rate, and so we pass each element of y_t through some convex and log-concave increasing point-wise nonlinearity $f : \mathbb{R} \to \mathbb{R}_+$. Popular choices for nonlinearities include $f(x) = \exp(x)$ and $f(x) = \log(1+\exp(x))$. To account for biophysical effects such as refractory periods, bursting, and direct synaptic connections, we include a linear dependence on spike history before the nonlinearity. The firing rate $f(y_t)$ is used as the rate for some point process ξ such as a Poisson process to generate a vector of spike counts s_t for all neurons at that time:

$$\begin{aligned} y_t &= Cx_t + \sum_{\tau=1}^{k} D_\tau s_{t-\tau} + b && (3.1) \\ s_t &\sim \xi(f(y_t)) && (3.2) \end{aligned}$$

Much of this chapter is focused on estimating y_t, which is the natural parameter for the Poisson distribution in the case $f(\cdot) = \exp(\cdot)$, and so we refer to y_t as the *natural* rate to avoid confusion with the *actual* rate $f(y_t)$. We will see that our approach works with any point process with a log-concave likelihood, not only Poisson processes.

We can extend this simple model by adding dynamics to the low-dimensional latent state, including input-driven dynamics. In this case the model is closely related to the common input model used in neuroscience (Kulkarni and Paninski, 2007), the difference being that the observed input is added to x_t rather than being directly mapped to y_t. The case without history terms and with linear Gaussian dynamics is a well-studied state space model for neural data, usually fit by EM (Chornoboy et al., 1988; Smith and Brown, 2003; Macke et al., 2011), though a consistent spectral method has been derived (Buesing et al., 2012) for the case $f(\cdot) = \exp(\cdot)$. Unlike these methods, our approach largely decouples the problem of dimensionality reduction and learning dynamics: even in the case of nonstationary, non-Gaussian dynamics where A, B and $\mathrm{Cov}[\epsilon]$ change over time, we can still robustly recover the latent subspace spanned by x_t.

3.3 Learning

3.3.1 Nuclear norm minimization

In the case that the spike history terms $D_{1:k}$ are zero, the natural rate at time t is $y_t = Cx_t + b$, so all y_t are elements of some m-dimensional affine space given by the span of the columns of C offset by b. Ideally, our estimate of $y_{1:T}$ would trade off between making the dimension of this affine space as low as possible and the likelihood of $y_{1:T}$ as high as possible. Let $Y = [y_1, \ldots, y_T]$ be the $n \times T$ matrix of natural rates and let $\mathcal{A}(\cdot)$ be the row mean centering operator $\mathcal{A}(Y) = Y - \frac{1}{T} Y \mathbf{1}_T \mathbf{1}_T^T$. Then $\text{rank}(\mathcal{A}(Y)) = m$. Ideally we would minimize $\lambda n T \text{rank}(\mathcal{A}(Y)) - \sum_{t=1}^T \log p(s_t | y_t)$, where λ controls how much we trade off between a simple solution and the likelihood of the data, however general rank minimization is a hard non convex problem. Instead we replace the matrix rank with its convex envelope: the sum of singular values or *nuclear norm* $\| \cdot \|_*$ (Fazel et al., 2001), which can be seen as the analogue of the ℓ_1 norm for vector sparsity. Our problem then becomes:

$$\min_Y \lambda \sqrt{nT} ||\mathcal{A}(Y)||_* - \sum_{t=1}^T \log p(s_t | y_t) \tag{3.3}$$

Since the log likelihood scales linearly with the size of the data, and the singular values scale with the square root of the size, we also add a factor of $\sqrt{nT}$ in front of the nuclear norm term. In the examples in this chapter, we assume spikes are drawn from a Poisson distribution:

$$\log p(s_t | y_t) = \sum_{i=1}^N s_{it} \log f(y_{it}) - f(y_{it}) - \log s_{it}! \tag{3.4}$$

However, this method can be used with any point process with a log-concave likelihood. This can be viewed as a convex formulation of exponential family PCA (Collins et al., 2001; Solo and Pasha, 2013) which does not fix the number of principal components ahead of time.

3.3.2 Stable principal component pursuit

The model above is appropriate for cases where the spike history terms D_τ are zero, that is the observed data can entirely be described by some low-dimensional global dynamics. In real data neurons exhibit history-dependent behavior like bursting and refractory periods. Moreover if the recorded neurons are close to each other some may have direct synaptic connections. In this case D_τ may have full column rank, so from Eq. 3.1 it is clear that y_t is no longer restricted to a low-dimensional affine space. In most practical cases we expect D_τ to be sparse, since most neurons are not connected to one another. In this case the natural rates matrix combines a low-rank term and a sparse term, and we can minimize a convex function that trades off between the rank of one term via the nuclear norm, the sparsity of another via the ℓ_1 norm, and the data log likelihood:

$$\begin{aligned} &\min_{Y, D_{1:k}, L} \lambda\sqrt{nT}||\mathcal{A}(L)||_* + \gamma\frac{T}{n}\sum_{\tau=1}^{k}||D_\tau||_1 - \sum_{t=1}^{T}\log p(s_t|y_t) \\ &\text{s.t.}\, Y = L + \sum_{\tau=1}^{k} D_\tau S_\tau, \text{ with } S_\tau = [0_{n,\tau}, s_1, \ldots, s_{T-\tau}], \end{aligned} \tag{3.5}$$

where $0_{n,\tau}$ is a matrix of zeros of size $n \times \tau$, used to account for boundary effects. This is an extension of *stable principal component pursuit* (Zhou et al., 2010), which separates sparse and low-rank components of a noise-corrupted matrix. Again to ensure that every term in the objective function of Eq. 3.5 has roughly the same scaling $\mathcal{O}(nT)$ we have multiplied each ℓ_1 norm with T/n. One can also consider the use of a group sparsity penalty where each group collects a specific synaptic weight across all the k time lags.

3.3.3 Evaluation through Bregman divergence decomposition

We need a way to evaluate the model on held out data, without assuming a particular form for the dynamics. As we recover a subspace spanned by the columns of Y rather than a single parameter, this presents a challenge. One option is to compute the marginal likelihood of the data integrated over the entire subspace, but this is computationally difficult. For the case of PCA, we can project the held out data onto a subspace spanned by principal components and compute what fraction of total variance is explained by this subspace. We extend this approach beyond the linear Gaussian case by use of a generalized Pythagorean theorem.

For any exponential family with natural parameters θ, link function g, function F such that $\nabla F = g^{-1}$ and sufficient statistic T, the log likelihood can be written as $D_F[\theta||g(T(x))] - h(x)$, where $D_\cdot[\cdot||\cdot]$ is a Bregman divergence (Bregman, 1967; Banerjee et al., 2005): $D_F[x||y] = F(x) - F(y) - (x-y)^T \nabla F(y)$. Intuitively, the Bregman divergence between x and y is the difference between the value of $F(x)$ and the value of the best linear approximation around y. Bregman divergences obey a generalization of the Pythagorean theorem: for any affine set Ω and points $x \notin \Omega$ and $y \in \Omega$, it follows that $D_F[x||y] = D_F[x||\Pi_\Omega(x)] + D_F[\Pi_\Omega(x)||y]$ where $\Pi_\Omega(x) = \arg\min_{\omega\in\Omega} D_F[x||\omega]$ is the projection of x onto Ω. In the case of squared error this is just a linear projection, and for the case of GLM log likelihoods this is equivalent to maximum likelihood estimation when the natural parameters are restricted to Ω.

Given a matrix of natural rates recovered from training data, we compute the *fraction of Bregman divergence explained* by a sequence of subspaces as follows. Let u_i be the ith singular vector of the recovered natural rates. Let b be the mean natural rate, and let $y_t^{(q)}$

be the maximum likelihood natural rates restricted to the space spanned by $u_1, \ldots, u_q$:

$$\begin{aligned} y_t^{(q)} &= \sum_{i=1}^{q} u_i v_{it}^{(q)} + \sum_{\tau=1}^{k} D_\tau s_{t-\tau} + b \\ v_t^{(q)} &= \arg\max_v \log p\left(s_t \middle| \sum_{i=1}^{q} u_i v_{it} + \sum_{\tau=1}^{k} D_\tau s_{t-\tau} + b\right) \end{aligned} \tag{3.6}$$

Here $v_t^{(q)}$ is the projection of $y_t^{(q)}$ onto the singular vectors. Then the divergence from the mean explained by the qth dimension is given by

$$\frac{\sum_t D_F\left[y_t^{(q-1)} \middle|\middle| y_t^{(q)}\right]}{\sum_t D_F\left[y_t^{(0)} \middle|\middle| g(s_t)\right]} \tag{3.7}$$

where $y_t^{(0)}$ is the bias b plus the spike history terms. The sum of divergences explained over all q is equal to one by virtue of the generalized Pythagorean theorem. For Gaussian noise $g(x) = x$ and $F(x) = \frac{1}{2}||x||^2$ and this is exactly the variance explained by each principal component, while for Poisson noise $g(x) = \log(x)$ and $F(x) = \sum_i \exp(x_i)$. This decomposition is only exact if $f = g^{-1}$ in Eq. 3.4, that is, if the nonlinearity is exponential. However, for other nonlinearities this may still be a useful approximation, and gives us a principled way of evaluating the goodness of fit of a learned subspace.

3.4 Algorithms

Minimizing Eq. 3.3 and Eq. 3.5 is difficult, because the nuclear and ℓ_1 norm are not differentiable everywhere. By using the *alternating direction method of multipliers* (ADMM), we can turn these problems into a sequence of tractable subproblems (Boyd et al., 2011). While not always the fastest method for solving a particular problem (in particular, for large-scale problems online methods are often preferable), we use it for its simplicity and generality. Here X^i denotes the ith row of the matrix X.

3.4.1 Alternating Direction Method of Multipliers

ADMM is a method for solving problems of the form:

$$\min_{Y} f(Y) + g(Y) \tag{3.8}$$

where $f(\cdot)$ and $g(\cdot)$ are both convex, but not necessarily differentiable everywhere. We introduce an auxiliary variable Z, a Lagrange multiplier Λ, and an augmented term that depends on a learning rate ρ to form the augmented Lagrangian:

$$\mathcal{L}_\rho(Y, Z, \Lambda) \triangleq f(Y) + g(Z) + \langle \Lambda, Y - Z \rangle + \frac{\rho}{2}||Y - Z||_F^2 \tag{3.9}$$

If we minimize $\mathcal{L}_\rho$ with respect to Y and Z the result is a concave function of Λ (the convex conjugate or Legendre-Fenchel transform), and the value of Y at the solution to $\max_\Lambda \inf_{Y,Z} \mathcal{L}_\rho$ is also the solution to Eq. 3.8. At this solution the augmented term $\frac{\rho}{2}||Y - Z||_F^2$ vanishes; it is there to guarantee that $\inf_{Y,Z} \mathcal{L}_\rho$ is well behaved before convergence.

ADMM does not directly maximize $\inf_{Y,Z} \mathcal{L}_\rho$. Instead, it alternates between minimizing Y, minimizing Z, and gradient ascent on Λ:

$$\begin{aligned} Y_{k+1} &= \arg\min_{Y} \mathcal{L}_\rho(Y, Z_k, \Lambda_k) && (3.10) \\ Z_{k+1} &= \arg\min_{Z} \mathcal{L}_\rho(Y_{k+1}, Z, \Lambda_k) && (3.11) \\ \Lambda_{k+1} &= \Lambda_k + \rho(Y_{k+1} - Z_{k+1}) && (3.12) \end{aligned}$$

This is guaranteed to converge to the global solution of Eq. 3.8.

3.4.2 Nuclear norm minimization

To find the optimal Y we alternate between minimizing an augmented Lagrangian with respect to Y, minimizing with respect to an auxiliary variable Z, and performing gradient ascent on a Lagrange multiplier Λ. The augmented Lagrangian is

$$\mathcal{L}_\rho(Y, Z, \Lambda) = \lambda\sqrt{nT}||Z||_* - \sum_t \log p(s_t|y_t) + \langle \Lambda, \mathcal{A}(Y) - Z\rangle + \frac{\rho}{2}||\mathcal{A}(Y) - Z||_F^2 \qquad (3.13)$$

which is a smooth function of Y and can be minimized by Newton's method. The gradient with respect to Y at iteration k is

$$\nabla_Y \mathcal{L}_\rho \;=\; -\nabla_Y \sum_t \log p(s_t|y_t) + \rho\mathcal{A}(Y) - \mathcal{A}^T(\rho Z_k - \Lambda_k) \qquad (3.14)$$

while the Hessian of Eq. 3.13 with respect to Y is given by

$$\nabla_Y^2 \mathcal{L}_\rho = -\nabla_Y^2 \sum_t \log p(s_t|y_t) + \rho\mathcal{A}^T\mathcal{A} \qquad (3.15)$$

where $\mathcal{A}^T(\cdot)$ is the transpose of the operator $\mathcal{A}(\cdot)$ and $\mathcal{A}^T\mathcal{A}$ is the product of the operator and its transpose written in matrix form. The Newton search direction $-(\nabla_Y^2\mathcal{L}_\rho)^{-1}\nabla_Y\mathcal{L}_\rho$ can be computed efficiently by exploiting the structure of the Hessian. The Hessian of the log likelihood term is diagonal, since the likelihood of the data s_{it} for neuron i at time t only depends on y_{it}. Moreover, if vec$(\cdot)$ denotes the vectorizing operator then the mean centering operator $\mathcal{A}(\cdot)$ can be expressed as

$$\mathrm{vec}(\mathcal{A}(Y)) = \left(I_{nT} - \frac{1}{T}(\mathbf{1}_T \otimes I_n)(\mathbf{1}_T \otimes I_n)^T\right)\mathrm{vec}(Y), \qquad (3.16)$$

It follows that $\mathcal{A}$ is self-adjoint and idempotent, and the Hessian simplifies to

$$\nabla^2_Y \mathcal{L}_\rho = -\nabla^2_Y \sum_t \log p(s_t|y_t) + \rho I_{nT} - \rho \frac{1}{T}(\mathbf{1}_T \otimes I_n)(\mathbf{1}_T \otimes I_n)^T \tag{3.17}$$

which is the sum of a diagonal term, $-\nabla^2_Y \sum_t \log p(s_t|y_t) + \rho I_{nT}$, and the term, $-\rho\frac{1}{T}(\mathbf{1}_T \otimes I_n)(\mathbf{1}_T \otimes I_n)^T$, which is only rank n rather than nT. Let D be the diagonal part of the Hessian

$$D = -\nabla^2_Y \sum_t \mathrm{log} p(s_t|y_t) + \rho I_{nT}. \tag{3.18}$$

Using the Woodbury lemma we have

$$(\nabla^2_Y \mathcal{L}_\rho)^{-1} = D^{-1} + D^{-1}(\mathbf{1}_T \otimes I_n)((T/\rho)I_n - (\mathbf{1}_T \otimes I_n)^T D^{-1}(\mathbf{1}_T \otimes I_n))^{-1}(\mathbf{1}_T \otimes I_n)^T D^{-1}. \tag{3.19}$$

Now let $\boldsymbol{d} = \mathrm{diag}\{D^{-1}\}$ and Δ the $n \times T$ matrix such that $\mathrm{vec}(\Delta) = \boldsymbol{d}$. A quick calculation shows that the matrix $(\mathbf{1}_T \otimes I_n)^T D^{-1}(\mathbf{1}_T \otimes I_n)$ is diagonal, with its diagonal equal to $\Delta\mathbf{1}_T$. It follows that the Newton direction $-(\nabla^2_Y \mathcal{L}_\rho)^{-1}\nabla_Y \mathcal{L}_\rho$ can be computed efficiently in $\mathcal{O}(nT)$ time and with $\mathcal{O}(nT)$ memory requirements, without having to explicitly construct the Hessian.

The minimum of Eq. 3.9 with respect to Z is given exactly by singular value thresholding:

$$Z_{k+1} = U\mathcal{S}_{\lambda\sqrt{nT}/\rho}(\Sigma)V^T, \tag{3.20}$$

where $U\Sigma V^T$ is the singular value decomposition of $\mathcal{A}(Y_{k+1}) + \Lambda_k/\rho$, and $\mathcal{S}_t(\cdot)$ is the (pointwise) soft thresholding operator $\mathcal{S}_t(x) = \mathrm{sgn}(x)\mathrm{max}(0, |x| - t)$. Finally, the update to Λ is a simple gradient ascent step: $\Lambda_{k+1} = \Lambda_k + \rho(\mathcal{A}(Y_{k+1}) - Z_{k+1})$ where ρ is a step size that can be chosen.

Algorithm 1 Alternating Direction Method of Multipliers for Nuclear Norm minimization without connectivity (Eq. 3.3)

```
input Matrix of spike counts $S$, learning rate $\rho$, parameter $\lambda$
$Y \leftarrow \log(S+1), Z \leftarrow 0, \Lambda \leftarrow 0$
while $r_p > \epsilon_p$ and $r_d > \epsilon_d$ do
    while $(\nabla_Y \mathcal{L}_\rho)^T (\nabla_Y^2 \mathcal{L}_\rho)^{-1} (\nabla_Y \mathcal{L}_\rho) > \epsilon$ do
        $\nabla_Y \mathcal{L}_\rho \triangleq -\nabla_Y \sum_t \log p(s_t|y_t) + \rho \mathcal{A}(Y) - \mathcal{A}^T(\rho Z - \Lambda)$
        $\nabla_Y^2 \mathcal{L}_\rho \triangleq -\nabla_Y^2 \sum_t \log p(s_t|y_t) + \rho I_{nT} - \rho \frac{1}{T}(\mathbf{1}_T \otimes I_n)(\mathbf{1}_T \otimes I_n)^T$
        $Y \leftarrow Y - (\nabla_Y^2 \mathcal{L}_\rho)^{-1} \nabla_Y \mathcal{L}_\rho$
    end while
    $U \Sigma V^T \triangleq \mathrm{SVD}(\mathcal{A}(Y) + \Lambda/\rho)$
    $Z' \leftarrow U \mathcal{S}_{\lambda\sqrt{nT}/\rho}(\Sigma) V^T$
    $\Lambda \leftarrow \Lambda + \rho(\mathcal{A}(Y) - Z')$
    $r_p \leftarrow ||\mathcal{A}(Y) - Z'||_F$
    $r_d \leftarrow \rho ||\mathcal{A}^T(Z - Z')||_F$
    $\epsilon_p \leftarrow \sqrt{nT}\epsilon_{abs} + \epsilon_{rel} \max(||\mathcal{A}(Y)||_F, ||Z'||_F)$
    $\epsilon_d \leftarrow \sqrt{nT}\epsilon_{abs} + \epsilon_{rel} ||\mathcal{A}^T(\Lambda)||_F$
    $Z \leftarrow Z'$
end while
return Y
```

3.4.3 Stable principal component pursuit

To extend ADMM to the problem in Eq. 3.5 we only need to add one extra step, taking the minimum over the connectivity matrices with the other parameters held fixed. To simplify the notation, we group the connectivity matrices into a single matrix $D = (D_1, \ldots, D_k)$, and stack the different time-shifted matrices of spike histories on top of one another to form a single spike history matrix H. The objective then becomes

$$\min_{Y,D} \lambda\sqrt{nT}||\mathcal{A}(Y - DH)||_* + \gamma\frac{T}{n}||D||_1 - \sum_t \log p(s_t|y_t) \tag{3.21}$$

where we have substituted $Y - DH$ for the variable L, and the augmented Lagrangian is

$$\begin{aligned} \mathcal{L}_\rho(Y, Z, D, \Lambda) &= \lambda\sqrt{nT}||Z||_* + \gamma\frac{T}{n}||D||_1 - \sum_t \log p(s_t|y_t) \\ &\quad +\langle\Lambda, \mathcal{A}(Y - DH) - Z\rangle + \frac{\rho}{2}||\mathcal{A}(Y - DH) - Z||_F^2 \end{aligned} \tag{3.22}$$

The updates for Λ and Z are almost unchanged, except that $\mathcal{A}(Y)$ becomes $\mathcal{A}(Y - DH)$. Likewise for Y the only change is one additional term in the gradient:

$$\nabla_Y \mathcal{L}_\rho = -\nabla_Y \sum_t \log p(s_t|y_t) + \rho\mathcal{A}(Y) - \mathcal{A}^T\left(\rho Z + \rho\mathcal{A}(DH) - \Lambda\right) \tag{3.23}$$

Minimizing D requires solving:

$$\arg\min_D \gamma\frac{T}{n}||D||_1 + \frac{\rho}{2}||\mathcal{A}(DH) + Z - \mathcal{A}(Y) - \Lambda/\rho||_F^2 \tag{3.24}$$

This objective has the same form as LASSO regression. We solve this using ADMM as well, but any method for LASSO regression can be substituted.

Algorithm 2 Alternating Direction Method of Multipliers for Nuclear Norm minimization with connectivity (Stable Principal Component Pursuit) (Eq. 3.21)

```
input Matrix of spike counts $S$ and spike histories $H$, learning rate $\rho$, parameters $\lambda$, $\gamma$
$Y \leftarrow \log(S+1)$, $Z \leftarrow 0$, $D \leftarrow 0$, $\Lambda \leftarrow 0$
while $r_p > \epsilon_p$ and $r_d > \epsilon_d$ do
  while $(\nabla_Y \mathcal{L}_\rho)^T (\nabla^2_Y \mathcal{L}_\rho)^{-1} (\nabla_Y \mathcal{L}_\rho) > \epsilon$ do
    $\nabla_Y \mathcal{L}_\rho \triangleq -\nabla_Y \sum_t \log p(s_t|y_t) + \rho\mathcal{A}(Y) - \mathcal{A}^T(\rho Z + \rho\mathcal{A}(DH) - \Lambda)$
    $\nabla^2_Y \mathcal{L}_\rho \triangleq -\nabla^2_Y \sum_t \log p(s_t|y_t) + \rho I_{nT} - \rho\frac{1}{T}(\mathbf{1}_T \otimes I_n)(\mathbf{1}_T \otimes I_n)^T$
    $Y \leftarrow Y - (\nabla^2_Y \mathcal{L}_\rho)^{-1} \nabla_Y \mathcal{L}_\rho$
  end while
  $D \leftarrow \arg\min_D \gamma\frac{T}{n}||D||_1 + \frac{\rho}{2}||\mathcal{A}(DH) + Z - \mathcal{A}(Y) - \Lambda/\rho||^2_F$ (See Alg. 3)
  $U\Sigma V^T \triangleq \mathrm{SVD}(\mathcal{A}(Y - DH) + \Lambda/\rho)$
  $Z' \leftarrow U\mathcal{S}_{\lambda\sqrt{nT}/\rho}(\Sigma)V^T$
  $\Lambda \leftarrow \Lambda + \rho(\mathcal{A}(Y) - Z')$
  $r_p \leftarrow ||\mathcal{A}(Y - DH) - Z'||_F$
  $r_d \leftarrow \rho||\mathcal{A}^T(Z - Z')||_F$
  $\epsilon_p \leftarrow \sqrt{nT}\epsilon_{abs} + \epsilon_{rel}\max(||\mathcal{A}(Y - DH)||_F, ||Z'||_F)$
  $\epsilon_d \leftarrow \sqrt{nT}\epsilon_{abs} + \epsilon_{rel}||\mathcal{A}^T(\Lambda)||_F$
  $Z \leftarrow Z'$
end while
return Y
```

Algorithm 3 Alternating Direction Method of Multipliers for Updating D (Eq. 3.24)

```
**input** Variables from Alg. 2, learning rate $\alpha$
$E \leftarrow D$, $\Gamma \leftarrow 0$
**while** $r_p > \epsilon_p$ **and** $r_d > \epsilon_d$ **do**
  **for** $i = 1 \to n$ **do**
    $D^i \leftarrow (\mathcal{A}^T(\mathcal{A}(Y^i) - Z^i + \Lambda^i/\rho)H^T + \alpha E_i - \Gamma_i)(\mathcal{A}(H)\mathcal{A}(H)^T + \alpha I_{nk})^{-1}$
  **end for**
  $E' \leftarrow \mathcal{S}_{\gamma T/n\rho\alpha}(D + \Gamma/\alpha)$
  $\Gamma \leftarrow \Gamma + \alpha(D - E')$
  $r_p \leftarrow ||D - E'||_F$
  $r_d \leftarrow \alpha||E - E'||_F$
  $\epsilon_p \leftarrow \sqrt{n^2k}\epsilon_{abs} + \epsilon_{rel}\max(||D||_F, ||E'||_F)$
  $\epsilon_d \leftarrow \sqrt{n^2k}\epsilon_{abs} + \epsilon_{rel}||\Gamma||_F$
  $E \leftarrow E'$
**end while**
**return** D
```

3.4.4 Fitting Linear Dynamical Systems

It is not immediately obvious that we should fit linear dynamical systems by the particular subspace method used in this chapter, or that we should minimize the nuclear norm of $\mathcal{A}(Y)$ instead of Y. Here we show empirical results on model data with two methods for fitting linear dynamical systems, and minimizing $||\mathcal{A}(Y)||_*$ versus $||Y||_*$, for a total of 4 combinations. The first method for fitting linear dynamical systems is perhaps the easiest. Let $X = (x_1, \ldots, x_T)$ be the matrix of latent states, just as $Y = (y_1, \ldots, y_T)$ is the matrix of natural rates, and suppose x_t is generated by a linear dynamical system:

$$\begin{aligned} x_{t+1} &= Ax_t + \epsilon_t \\ \mathbb{E}[\epsilon_t] &= 0 \end{aligned} \tag{3.25}$$

First we take the singular value decomposition of $\mathcal{A}(Y)$, so that $U\Sigma V^T = \mathcal{A}(Y)$. Since $\mathcal{A}(Y) = CX$, the left and right singular vectors should be equal to C and X up to some arbitrary rotation M:

$$\begin{aligned} U\sqrt{\Sigma} &= CM \\ \sqrt{\Sigma}V^T &= M^{-1}X \end{aligned} \tag{3.26}$$

where $\sqrt{\Sigma}$ takes the element-wise square root of the diagonal matrix of singular values.

It is clear that $X_{2:T} = (x_2, \ldots, x_T) = AX_{1:T-1} + E = A(x_1, \ldots, x_{T-1}) + (\epsilon_1, \ldots, \epsilon_{T-1})$, that is each column of X is a noisy linear mapping of the column to the left of it. That suggests we could estimate A by doing regression between past and future columns of X, or by proxy, $\sqrt{\Sigma}V^T$:

$$\hat{A} = \left(V^{2:T}\sqrt{\Sigma^T}\right)\left(V^{1:T-1}\sqrt{\Sigma^T}\right)^\dagger \tag{3.27}$$

here $X^{i:j}$ denotes the ith to jth *rows* of X, while $X_{i:j}$ denotes the ith to jth columns. We refer to this method of fitting linear dynamical systems as *past-future regression*. While this estimate of A is off by a change of coordinate, the eigenvalues should on average be the same as the true A if our estimator is unbiased. We find that this is not the case.

Alternately, we use a variant of the Multivariable Output Error State sPace (MOESP) method for fitting linear dynamical systems, a type of subspace identification (Van Overschee and De Moor, 1996). Our implementation of MOESP works as follows: take the covariance between Y one and two time steps into the past and one and two time steps into the future:

$$\Gamma = \begin{pmatrix} Y_{3:T-1} \\ Y_{4:T} \end{pmatrix} \begin{pmatrix} Y_{1:T-3} \\ Y_{2:T-2} \end{pmatrix}^T \tag{3.28}$$

where the matrix on the left is known as the block-Hankel matrix of future outputs and the matrix on the right is the block-Hankel matrix of past outputs in the terminology of subspace identification. The number of block-rows can be greater than 2, but as long as the number of latent dimensions is less than the number of observed dimensions, only two are needed.

From Eqs. 3.1 and 3.25 we can expand out y_t as $CA^k x_{t-k} + \sum_{\tau=1}^{k} CA^{k-\tau}\epsilon_{t-\tau}$ and plug this into the true past-future covariance to find:

$$\mathrm{Cov}\left[\begin{pmatrix} y_t \\ y_{t+1} \end{pmatrix}, \begin{pmatrix} y_{t-2} \\ y_{t-1} \end{pmatrix}\right] = \begin{pmatrix} C \\ CA \end{pmatrix} \begin{pmatrix} C\mathrm{Cov}[x_t]A^{2T} \\ CA\mathrm{Cov}[x_t]A^{2T} + C\mathrm{Cov}[\epsilon_t]A^T \end{pmatrix}^T \tag{3.29}$$

of which $\Gamma/(T-2)$ is the maximum likelihood estimate. We can then say the left singular values of Γ should asymptotically be equal to $\begin{pmatrix} C \\ CA \end{pmatrix}$ up to a rotation, and we estimate $\hat{A}$ by doing least squares regression between the top n rows and bottom n rows of the left singular vectors of Γ. As with the past-future regression method described above, we find it useful to scale the left singular vectors by the square root of the singular values.

In the experiments on model data, we know the true dimensionality m, and truncated Σ so that all singular values after the mth are set to 0. On real data we would use some heuristic, such as truncating everything below the geometric mean of the first and last singular value. If we had access to known input as well, we would also include a projection onto the orthogonal subspace of the input, and include past inputs in the right-hand matrix in Eq. 3.28.

3.5 Experiments

We demonstrate our method on a number of artificial datasets and one real dataset. First, we show in the absence of spike history terms that the true low dimensional subspace can be recovered in the limit of large data, even when the dynamics are nonstationary. Second, we show that spectral methods can accurately recover the transition matrix when dynamics are linear. Third, we show that local connectivity can be separated from low-dimensional common input. Lastly, we show that nuclear-norm penalized subspace recovery leads to improved prediction on real neural data recorded from macaque motor cortex.

Model data was generated with 8 latent dimension and 200 neurons, without any external input. For linear dynamical systems, the transition matrix was sampled from a Gaussian distribution, and the eigenvalues rescaled so the magnitude fell between .9 and .99 and the angle between $\pm\frac{\pi}{10}$, yielding slow and stable dynamics. The linear projection C was a random Gaussian matrix with standard deviation 1/3, and the biases b_i were sampled from $\mathcal{N}(-4, 1)$, which we found gave reasonable firing rates with nonlinearity $f(x) = \log(1+\exp(x))$. To investigate the variance of our estimates, we generated multiple trials of data with the same parameters but different innovations.

3.5.1 Recovering Subspaces

We first sought to show that we could accurately recover the subspace in which the dynamics take place even when those dynamics are not stationary. We split each trial into 5 epochs and in each epoch resampled the transition matrix A and set the covariance of innovations ϵ_t to QQ^T where Q is a random Gaussian matrix. We performed nuclear norm minimization on data generated from this model, varying the smoothing parameter λ from 10^{-3} to 10, and compared the subspace angle between the top 8 principal components and the true matrix C. We repeated this over 10 trials to compute the variance of our estimator. We found that when smoothing was optimized the recovered subspace was significantly closer to the true subspace than the top principal components taken directly from spike data. Increasing the amount of data from 1000 to 10000 time bins significantly reduced the average subspace angle at the optimal λ. The top PCs of the true natural rates Y, while not spanning exactly the same space as C due to differences between the mean column and true bias b, was still closer to the true subspace than the result of nuclear norm minimization.

We also computed the fraction of Bregman divergence explained by the sequence of spaces spanned by successive principal components, solving Eq. 3.6 by Newton's method. We did not find a clear drop at the true dimensionality of the subspace, but we did find that a larger share of the divergence could be explained by the top dimensions than by PCA directly on spikes. Results are presented in Fig. 3.1.

3.5.2 Learning Linear Dynamical Systems

To show that the parameters of a latent dynamical system can be recovered, we investigated the performance of spectral methods on model data with linear Gaussian latent dynamics. As the model is a linear dynamical system with GLM output, we call this a GLM-LDS

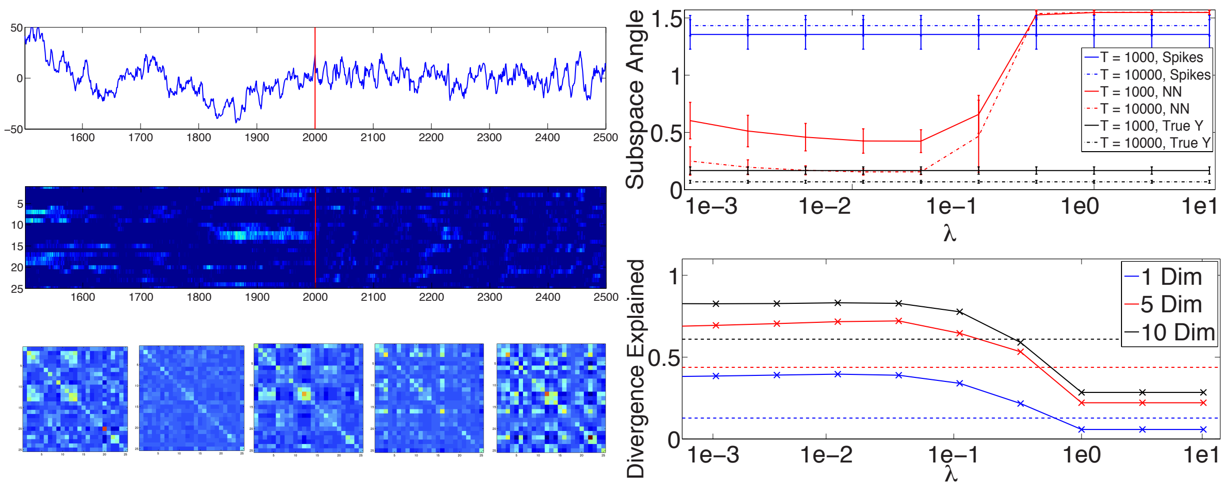


Figure 3.1: Recovering low-dimensional subspaces from nonstationary model data. While the subspace remains the same, the dynamics switch between 5 different linear systems. Left top: one dimension of the latent trajectory, switching from one set of dynamics to another (red line). Left middle: firing rates of a subset of neurons during the same switch. Left bottom: covariance between spike counts for different neurons during each epoch of linear dynamics. Right top: Angle between the true subspace and top principal components directly from spike data, from natural rates recovered by nuclear norm minimization, and from the true natural rates. Right bottom: fraction of Bregman divergence explained by the top 1, 5 or 10 dimensions from nuclear norm minimization. Dotted lines are variance explained by the same number of principal components. For $\lambda < 0.1$ the divergence explained by a given number of dimensions exceeds the variance explained by the same number of PCs.

model. After estimating natural rates by nuclear norm minimization with $\lambda = 0.01$ on 10 trials of 10000 time bins with unit-variance innovations ϵ_t, we fit the transition matrix A by subspace identification (SSID) (Van Overschee and De Moor, 1996). The transition matrix is only identifiable up to a change of coordinates, so we evaluated our fit by comparing the eigenvalues of the true and estimated A. Results are presented in Fig. 3.2. As expected, SSID directly on spikes led to biased estimates of the transition. By contrast, SSID on the output of nuclear norm minimization had little bias, and seemed to perform almost as well as SSID directly on the true natural rates. We found that other methods for fitting linear dynamical systems from the estimated natural rates were biased, as was SSID on the result of nuclear norm minimization *without* mean-centering (Fig. 3.3).

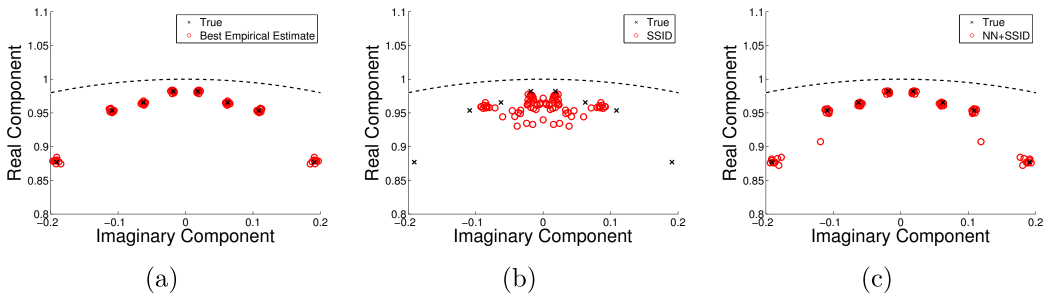


Figure 3.2: Recovered eigenvalues for the transition matrix of a linear dynamical system from model neural data. Black: true eigenvalues. Red: recovered eigenvalues. (a) Eigenvalues recovered from the true natural rates. (b) Eigenvalues recovered from subspace identification directly on spike counts. (c) Eigenvalues recovered from subspace identification on the natural rates estimated by nuclear norm minimization.

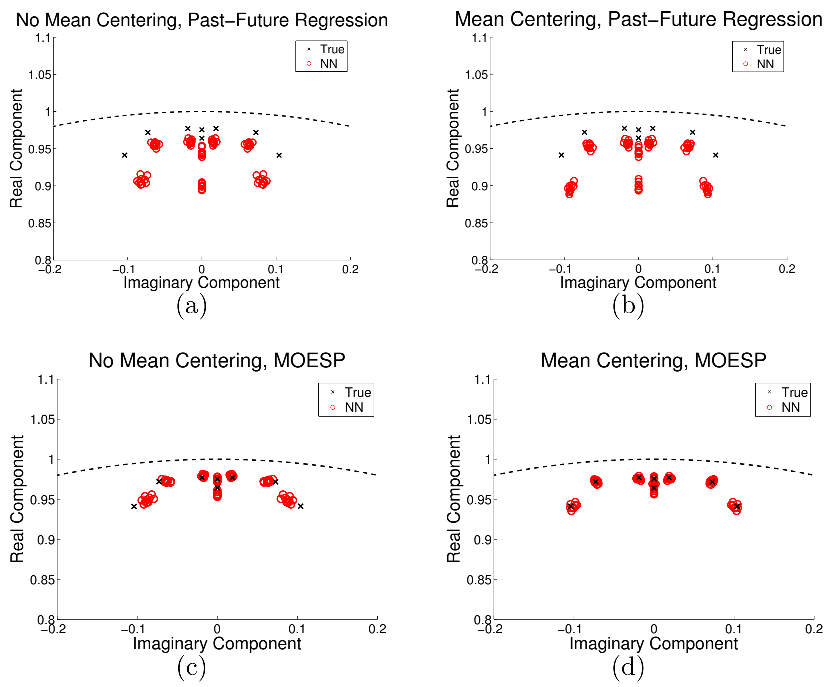


Figure 3.3: A comparison of the eigenvalues of the transition matrix A recovered by different methods of fitting linear dynamical systems to model data. True eigenvalues in black, estimates over different trials in red. 200 neurons, 10000 time bins, 8 latent dimensions, 10 trials. Top row: Results of estimating A by past-future regression (Eq. 3.27). Bottom row: Results of estimating the transition matrix by the MOESP subspace identification method. Left column: Nuclear norm minimization directly on the matrix Y. Right column: Nuclear norm minimization on the mean-centered matrix $\mathcal{A}(Y)$. In both cases the mean of Y was subtracted before estimating the transition matrix. Note that both mean-centering and subspace identification are necessary to arrive at unbiased estimates of the transition matrix.

We incorporated spike history terms into our model data to see whether local connectivity and global dynamics could be separated. Our model network consisted of 50 neurons, randomly connected with 95% sparsity, and synaptic weights sampled from a unit variance

Gaussian. Data were sampled from 10000 time bins. The parameters λ and γ were both varied from 10^{-10} to 10^4. We found that we could recover synaptic weights with an r^2 up to .4 on this data by combining both a nuclear norm and ℓ_1 penalty, compared to at most .25 for an ℓ_1 penalty alone, or 0.33 for a nuclear norm penalty alone. Somewhat surprisingly, at the extreme of either no nuclear norm penalty or a dominant nuclear norm penalty, increasing the ℓ_1 penalty never improved estimation. This suggests that in a regime with strong common inputs, some kind of correction is necessary not only for sparse penalties to achieve optimal performance, but to achieve any improvement over maximum likelihood. It is also of interest that the peak in r^2 is near a sharp transition to total sparsity.

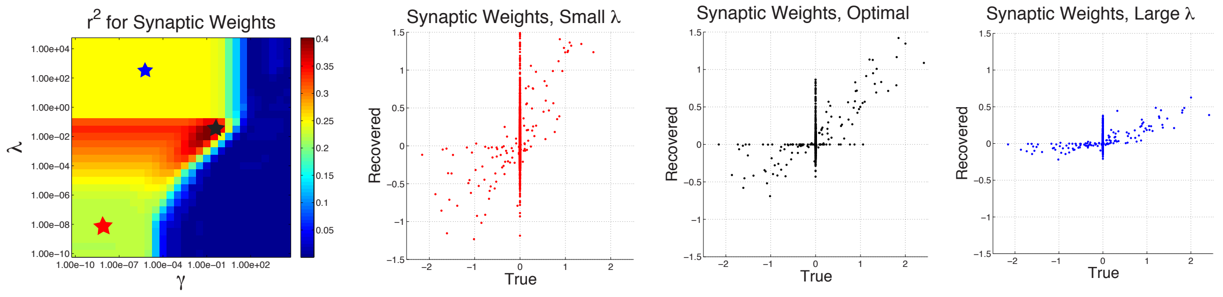


Figure 3.4: Connectivity matrices recovered by SPCP on model data. Left: r^2 between true and recovered synaptic weights across a range of parameters. The position in parameter space of the data to the right is highlighted by the stars. Axes are on a log scale. Right: scatter plot of true versus recovered synaptic weights, illustrating the effect of the nuclear norm term.

3.5.3 Optimizing Smoothing Parameters

The only free parameter in our minimization is λ, which controls the tradeoff between the data likelihood and nuclear norm penalty. As $\lambda \to \infty$, the natural rates are forced to a low-rank solution, and eventually to the mean of the data (after being passed through the inverse nonlinearity) if the mean-centering operator is included, or zero if not. At the other extreme, the nuclear norm penalty vanishes as $\lambda \to 0$ and the natural rates go to the maximum likelihood solution. We demonstrate the effect of varying the nuclear norm penalty on the spectra of the natural rates and the eigenvalues of the recovered transition matrix and show that the nuclear norm penalty leads to less biased estimates of the transitions. As λ increases the quality of the estimates improves, until the rank of the natural rates is forced to below the actual number of latent dimensions. In a nutshell, the solution should be as low rank as possible, but no lower.

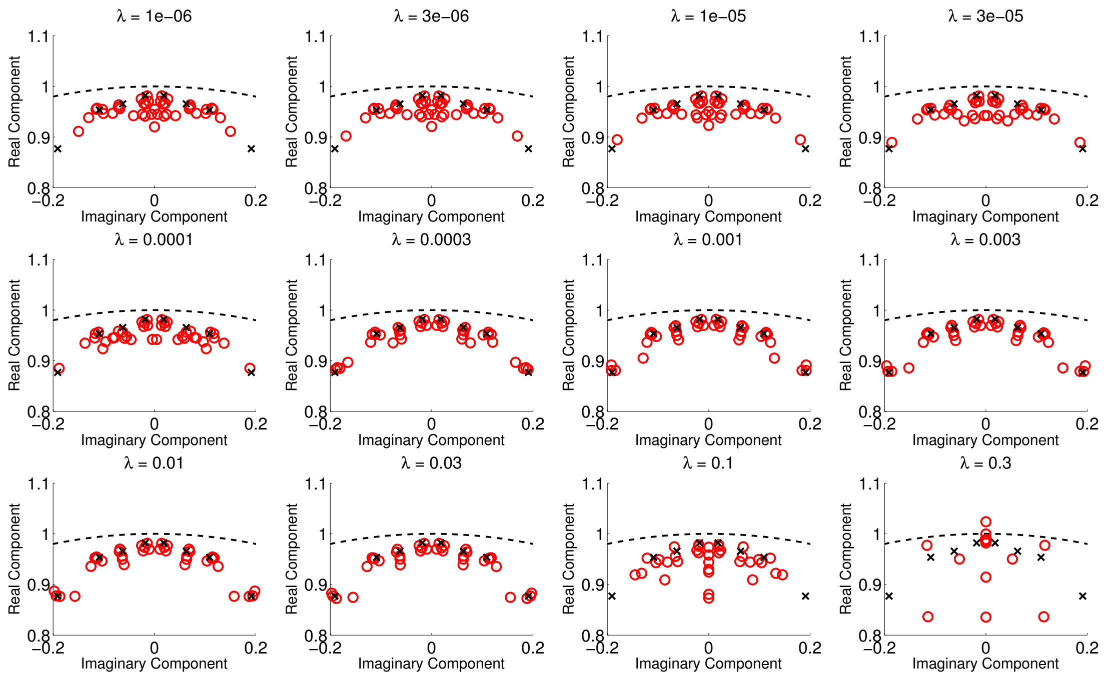


Figure 3.5: Effect of varying the smoothing parameter λ on the recovered transition matrix eigenvalues. True values in black, recovered in red. The nuclear norm term helps reduce the variance of the estimates, but the results quickly degenerate when λ is too large. Note that the results are robust across a wide range of values of λ, from roughly 0.001 to 0.03. Model data, 200 neurons, 1000 time bins, 8 latent dimensions, 5 trials.

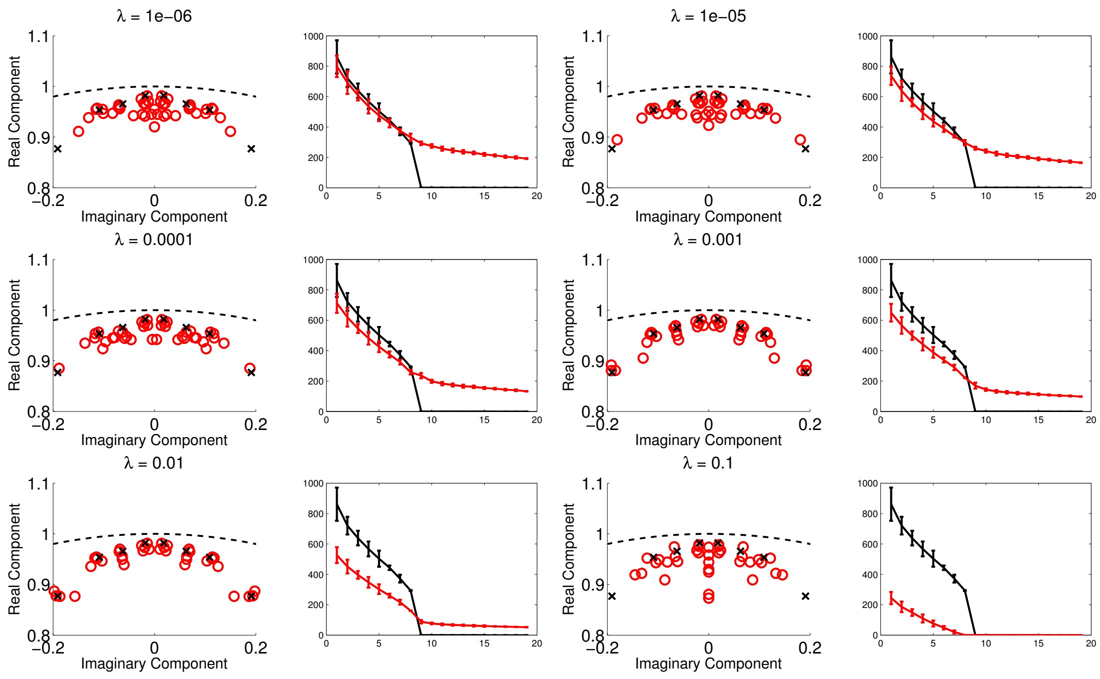


Figure 3.6: A simultaneous comparison of the eigenvalues of A and singular values of Y for various values of λ on the same data as Fig. 3.5. Note that with no smoothing, the top singular values of the recovered Y closely match the true values, but the noise leads to biases in the recovered A. At the other extreme, the quality of the recovered A degrades rapidly if the smoothing forces the rank of Y to be smaller than the true rank.

3.5.4 Predicting Dynamics of Real Neural Populations

Finally, we demonstrated the utility of our method on real recordings from a large population of neurons. The data consists of 125 well-isolated units from a multi-electrode recording in macaque motor cortex while the animal was performing a pinball task in two dimensions. Previous studies on this data (Lawhern et al., 2010) have shown that information about arm velocity can be reliably decoded. As the electrodes are spaced far apart, we do not expect any direct connections between the units, and so leave out the ℓ_1 penalty term from the objective. We used 800 seconds of data binned every 100 ms for training and 200 seconds for testing. We fit linear dynamical systems by subspace identification as in Fig. 3.2, but as we did not have access to a "true" linear dynamical system for comparison, we evaluated our model fits by approximating the held out log likelihood by Laplace-Gaussian filtering (Koyama et al., 2010). We also fit the GLM-LDS model by running randomly initialized

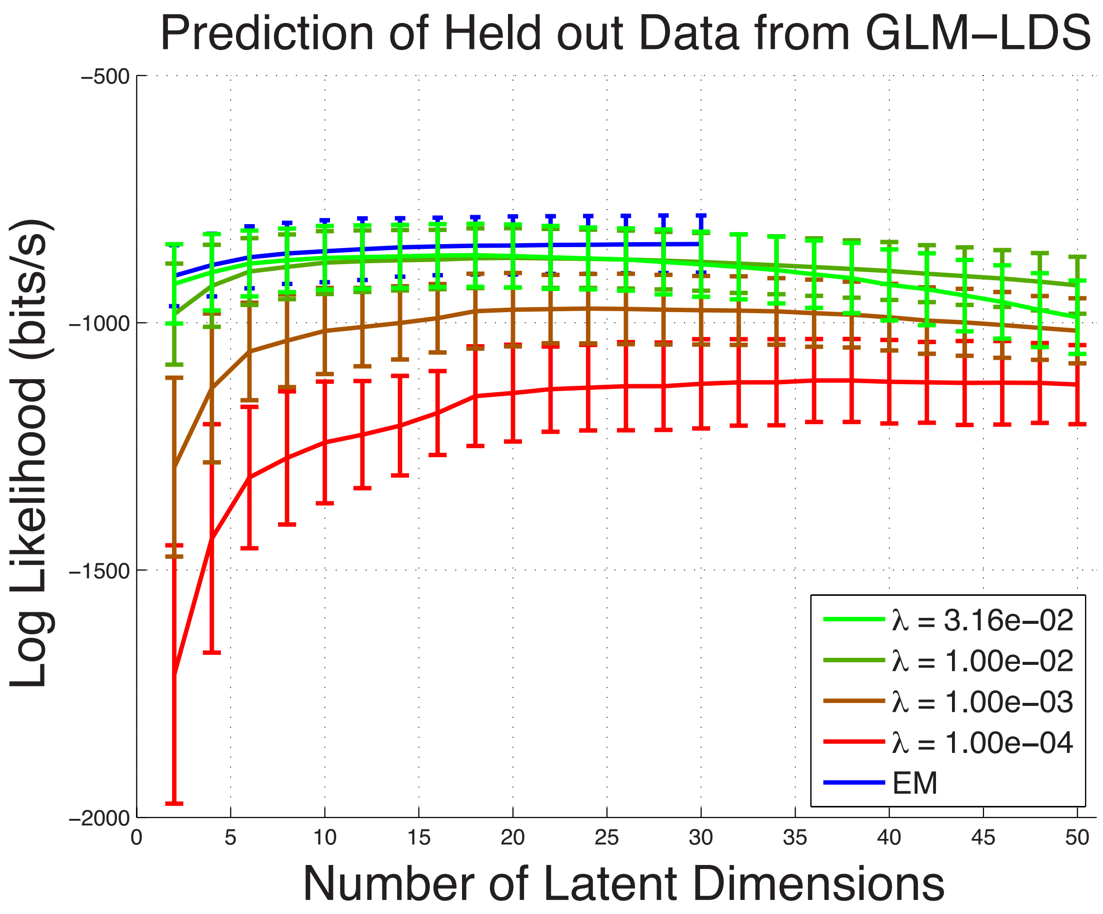


Figure 3.7: Log likelihood of held out motor cortex data versus number of latent dimensions for different latent linear dynamical systems. Prediction improves as λ increases, until it is comparable to EM.

EM for 50 iterations for models with up to 30 latent dimensions (beyond which training was prohibitively slow). We found that a strong nuclear norm penalty improved prediction by several hundred bits per second, and that fewer dimensions were needed for optimal prediction as the nuclear norm penalty was increased. The best fit models predicted held out data nearly as well as models trained via EM, even though nuclear norm minimization is not directly maximizing the likelihood of a linear dynamical system.

3.6 Discussion

The method presented here has a number of straightforward extensions. If the dimensionality of the latent state is *greater* than the dimensionality of the data, for instance when there are long-range history dependencies in a small population of neurons, we would extend the natural rate matrix Y so that each column contains multiple time steps of data. Y is then a *block-Hankel matrix*. Constructing the block-Hankel matrix is also a linear operation, so the objective is still convex and can be efficiently minimized (Liu et al., 2013). If there are also observed inputs u_t then the term inside the nuclear norm should also include a projection orthogonal to the row space of the inputs. This could enable joint learning of dynamics and receptive fields for small populations of neurons with high dimensional inputs.

Our model data results on connectivity inference have important implications for practitioners working with highly correlated data. GLM models with sparsity penalties have been used to infer connectivity in real neural networks (Pillow et al., 2008), and in most cases these networks are only partially observed and have large amounts of common input. We offer one promising route to removing the confounding influence of unobserved correlated inputs, which explicitly models the common input rather than conditioning on it (Harrison, 2012).

It remains an open question what kinds of dynamics can be learned from the recovered natural parameters. In this chapter we have focused on linear systems, but nuclear norm minimization could just as easily be combined with spectral methods for switching linear systems and general nonlinear systems. We believe that the techniques presented here offer a powerful, extensible and robust framework for extracting structure from neural activity.

Chapter 4

Separating Biologically Relevant Signals From Optical Imaging Data[1]

Dear friend, all theory is grey,

And green is the golden tree of life.

Johann Wolfgang von Goethe, *Faust*

That's all well and good in practice...

...but how does it work in theory?

Anonymous

The previous chapters focused on statistical methods derived from theoretical considerations first. When dealing with practical data, other considerations must often be

[1]This work represents preliminary work in collaboration with Daniel Soudry, Yuanjun Gao, Jeremy Freeman, Yu Mu and Misha Ahrens

taken into account before these methods can be usefully applied. A number of technological avenues are being developed that can help bridge the divide between theory and practice. Optical recording technologies in particular are an essential part of the push towards recording from very large populations of neurons. Converting raw recording data to neurons and spike times typically involves a laborious manual process of annotating regions of interest (ROIs), along with ad hoc heuristics for extracting time series from those ROIs. For very large populations of neurons, it becomes impractical to hand-label all neurons, and automated methods are highly desirable. Here we describe a number of approaches to automatically extracting regions of interest from the cutting-edge of optical imaging technology. First we describe an online clustering algorithm for region of interest detection for whole-brain calcium recordings from larval zebrafish that can run on a single machine. We present analyses on the time series extracted from these regions of interest that cannot be performed on a pixel-by-pixel basis. We then describe a method for analyzing regions of interest that is based on a more principled generative model of imaging data and can be solved efficiently by convex optimization, which we call convolutional group lasso. For large datasets such as those generated by light-sheet imaging, these methods can be applied on individual patches. Finally we switch from large-scale calcium imaging to voltage imaging at in culture, where the challenges are instead due to low signal-to-noise ratio and the multiple scales of imaging. We show that independent component based methods can be effective when proper pre- and post-processing is applied, and can be used as an important step in a fully automated pipeline for all-optical electrophysiology.

4.1 Introduction

It is a long-standing goal of neuroscience to be able to record every spike from every neuron in a large population or even entire animal. Optical imaging has the potential to make this a reality, but there are many challenges in translating raw data into a raster of spikes and neurons[2]. Much like in the last chapter, this can be viewed as a dimensionality reduction problem, as there are far fewer neurons than pixels in most recordings. However, in this case there is a very particular structure imposed by prior knowledge of the shape of neural structures and dynamics of neural and fluorescent indicator activity. These two kinds of structure lead to two orthogonal problems: that of *calcium deconvolution*, which aims to recover spike times from noisy calcium indicator data, and *region of interest detection*, which separates the imaging field into biologically relevant units (neurons, neuropil, etc). In this chapter we are concerned strictly with the latter problem, as the data we are dealing with lacks the temporal resolution to resolve single spikes.

While optical imaging is a broad field, nearly all neuroscience applications work by loading an indicator whose fluorescence depends on some biophysical parameter into neurons, exciting the fluorescent indicators by laser light and recording from a large field of view (Yuste and Konnerth, 2004). The indicators may be synthetic dyes (Baker et al., 2005) or engineered proteins, and sensitive to intracellular calcium concentration or transmembrane voltage. At present, great strides have been made in improving both the speed and signal-to-noise ratio of genetically encoded calcium indicators (GECIs) such that they are now often preferred to synthetic dyes (Sun et al., 2013). While not as mature a technology a GECIs, the field of voltage-sensitive fluorescent proteins is also advancing quickly (Barnett et al., 2012; Kralj et al., 2012; Akemann et al., 2013; Cao et al., 2013; Gong et al., 2013;

[2]Though note, in many cases, we may be interested in functional units smaller than a neuron, such as dendritic spines (Yuste and Denk, 1995)

Storace et al., 2013).

A wide variety of microscopy techniques can be used to image this fluorescent activity, most commonly two photon microscopy (Denk et al., 1990; Svoboda and Yasuda, 2006), but also selective plane illumination or “light-sheet” microscopy (Holekamp et al., 2008) and light field microscopy (Levoy et al., 2006; Broxton et al., 2013), both of which can image volumes of tissue with single-photon optics.

The volume of data being generated by these methods is often staggering. For instance, in (Ahrens et al., 2013) the authors image an 800x600x200 μm^3 volume at 0.8 Hz with 41 image planes and an effective lateral resolution of 0.65 μm, meaning that an hour of continuous recording amounts to nearly 1 TB of data. Clearly this presents a challenge for analysis. One possibility is to take advantage of distributed computing (Attiya and Welch, 2004), applying operations in parallel to each time series and collecting the results. In particular, the MapReduce framework (Dean and Ghemawat, 2008) has become a standard in the last decade. One downside of the MapReduce framework is that only a single function can be applied (mapped) before the output is collected (reduced). For machine learning and optimization applications, where operations have to be applied iteratively, this can be quite slow. To address this issue, the Spark framework (Zaharia et al., 2010) was developed, in which distributed data can be acted on multiple times before results are collected. Already a number of standard tools in neural data analysis have been ported to Spark under the Thunder library (Freeman et al., 2014). Alteratively, one could take advantage of online methods, that operate on manageable chunks of data sequentially, updating parameters of a model as they go along. This has the advantage that it can be executed on a single machine, rather than requiring a distributed cluster. However the speed of reading data sequentially is still significantly slower than reading in parallel, so distributed methods are preferable when available.

Ultimately what one would like to read out from this data is the spike time from every visible neuron. For Ca^{2+} imaging data, a significant amount of effort has been applied both to the problem of separating spatial signals (region of interest detection) and inverting the filtering effects of both Ca^{2+} transients and Ca^{2+} indicator dynamics (spike deconvolution). Calcium deconvolution in particular has matured significantly in recent years. Early work on calcium imaging showed that individual spikes could be detected so long as they were well separated in time (Smetters et al., 1999), but for practical experiments new methods had to be developed. A wide variety of techniques have been developed, sometimes suited to the quirks of the data in that lab, sometimes built around more principled biophysical considerations. Kerr et al. (2005); Greenberg et al. (2008) used a template matching approach that approximates a complicated nonconvex optimization to find calcium transients, while Holekamp et al. (2008) and Yaksi and Friedrich (2006) used linear methods to deconvolve calcium signals, and Sato et al. (2007) took a clustering approach based on the magnitude of response when combined with a template. Sasaki et al. (2008) used a supervised learning approach combining PCA for feature extraction with a linear SVM. In Grewe et al. (2010) they develop a “peeling” approach to detecting spikes at rates up to 20 Hz, which closely resembles matching pursuit (Mallat and Zhang, 1993), and Oñativia et al. (2013) uses tools from finite rate of innovation theory (Vetterli et al., 2002). Other work has focused on fitting more principled generative models that take into account biophysical knowledge (Vogelstein et al., 2009) and algorithmic advances that make inference faster (Vogelstein et al., 2009), or integrate information across many pixels to jointly deconvolve spikes and demix regions of interest (Pnevmatikakis et al., 2013).

Recording technology has now advanced to the point where it is possible to record from nearly the whole brain of small animals (Ahrens et al., 2013; Prevedel et al., 2014). Even in larger animals, recording from thousands of neurons simultaneously is becoming more common. At this scale it is no longer practical to manually identify regions of interest, and automated techniques become very valuable. Optical recording is now common enough that a comprehensive review of the literature is outside the scope of this thesis, but most prominent studies that established optical imaging of large populations used manual (Göbel et al., 2007; Dombeck et al., 2007; Kerr et al., 2005; Niell and Smith, 2005) or semiautomated (Ohki et al., 2005; Ozden et al., 2008; Junek et al., 2009; Dorostkar et al., 2010) methods for identification.

A variety of automated ROI detection methods have been developed, similar to the situation in calcium deconvolution, where there is a mix of principled statistical methods and effective heuristics. One prominent approach to automated ROI detection is the CellSort ICA algorithm (Mukamel et al., 2009), which applies spatiotemporal independent component analysis (Stone et al., 2002) to principal components of videos of calcium activity, followed by a postprocessing step that identifies morphologically distinct regions. To work well, this approach depends critically on the number of principal components being near the number of distinct functional components, as well as overlapping neurons having distinct activity (though not necessarily independent activity). Many other approaches also fit into the matrix factorization framework. A number of groups have had success using extensions of nonnegative matrix factorization that include some prior structure (Soelter et al., 2014; Maruyama et al., 2014) as well as sparse structured principal component analysis (Diego et al., 2013) and hierarchical extensions (Diego and Hamprecht, 2013). Other groups have taken advantage of machine learning techniques outside the matrix factorization framework: Valmianski et al. (2010) use a mix of supervised learning techniques to train a classifier for neuronal morphology. Rather than relying entirely on the fluorescence signal, Miri et al. (2011) regresses the fluorescence signal against stimulus information to find relevant pixels, then uses custom morphological techniques to segment the resulting maps. Lastly, purely morphological approaches that throw away the majority of neural activity can be effective, so long as regions of interest are well separated in the signal of choice (usually mean or max fluorescence). For instance, convolutional sparse coding methods that learn a dictionary of ROI shapes have been shown to be effective on some calcium imaging and Nissl-stained data (Pachitariu et al., 2013), even when the indicator is expressed cytosolically, leading to densely packed "donut"-like shapes that have to be separated.

Most of these methods have been applied to what might be called "medium-scale" recordings - in the range of dozens to a thousand regions of interest. For whole-brain recordings in larval zebrafish, where nearly 100,000 neurons are being imaged simultaneously, the complexity of data analysis has limited the applicability of some of these region of interest detection methods. Most analyses have been applied to individual voxels or "supervoxels" that are soma-sized cubes of data (Ahrens et al., 2013). In Portugues et al. (2014) they apply a local correlation based region of interest detection method to find anatomical regions correlated with optokinetic response. While more likely to separate out individual neurons than supervoxels, the method still misclassifies overlapping regions, which may confound analysis of neurons that are strongly selective with very different response properties from their neighbors.

This chapter is organized as follows. We present two algorithms for separating single units from calcium fluorescence data that can be applied to very large scale recordings. In 4.2.1 we present an online greedy clustering algorithm tailored to the specific statistics of fluorescent imaging noise. In 4.2.3 we apply this method to whole-brain recordings of spontaneous activity in the larval zebrafish brain (Ahrens et al., 2013) and show results of analyses that could not be performed without first reducing the dimensionality of the data. This algorithm contains a number of heuristics, and we present in 4.3 a more elegant algorithm based on an explicit convex optimization problem, which requires fewer heuristics than the online greedy approach, and can easily be adapted to a distributed computing implementation, making it possible to run analyses in minutes instead of hours.

4.2 Online Region of Interest Detection for Whole Brain Recordings

One strategy for handling data at this scale is to process each frame sequentially. Since a single frame of data can fit in memory on a single machine even when the full dataset cannot, this is an effective strategy when distributed computing resources are not readily available. In addition, if the signal-to-noise ratio is high enough, regions of interest can be picked out from a single frame. ROI detection is then divided into the problem of processing single frames, and clustering the results across many frames, since neurons may drop in and out across various frames. We present an online algorithm that processes single frames based on standard image processing methods and a fast online clustering method to decide which ROIs are matched across frames. Due to the scale of the problem, whenever possible we choose to optimize for speed over statistical correctness. We applied this method to recordings of spontaneous activity in a larval zebrafish expressing GCaMP5 and were able to separate several thousand regions of interest, the majority of which agreed with human identification of neuronal cell bodies. Applying nonlinear dimensionality reduction techniques to this data sorts the ROIs into distinct but overlapping populations, while neither PCA nor nonlinear methods applied to downsampled data were able to reveal these populations.

4.2.1 Online Greedy Clustering

Our approach to online ROI detection splits the problem into two parts: first, identify all the regions of interest in a single frame, second, decide which ROIs from the current frame should be merged with already-identified ROIs from previous frames. Since we don't update assignments after the fact, this is a greedy method, so we will refer to it as online greedy clustering throughout this chapter, in contrast to other ROI detection methods presented later. Let us introduce some notation: throughout this chapter, a frame of data shaped as a 2D or 3D array will be denoted in uppercase, and the same data as a vector will be denoted in lowercase, that is $\text{vec}(\mathbf{X}) = \mathbf{x}$. Let $\mathcal{R}$ denote the range of indices, D the number of dimensions in one frame (2 or 3 in all real data), K be the total number of ROIs, and let $\mathbf{Y}^t$ be the observed calcium fluorescence at time t, so that $\mathbf{y}^t$ is a vector in $\mathbb{R}^{|\mathcal{R}|}$. We assume a very simple generative model for the data, so that learning is fast. Let $\mathbf{X}^k$ be the shape of the kth region of interest, $\mathbf{X} = \left(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^K\right)$ be the full matrix of ROI shapes, and let $\mathbf{r}^t \in \mathbb{R}^K$ be the level of fluorescence signal across ROIs at time t. Then we model one frame of data as

$$\mathbf{y}^t|\mathbf{X} \sim \mathcal{N}\left(\mathbf{X}\mathbf{r}^{\mathbf{t}}, \sigma^2\mathbf{I} + \rho\text{diag}\left(\mathbb{I}(\mathbf{X})\mathbf{r}^t\right)\right) \tag{4.1}$$

where σ is a baseline variance, $\mathbb{I}$ is an indicator function $\mathbb{I}(0) = 0$, $\mathbb{I}(x) = 1$ otherwise, applied elementwise, and ρ is a factor that determines the linear dependence of the variance on the level of calcium activity. We have found that this linear dependence of variance on fluorescence level is necessary to make the ROI merging step robust, which likely reflects the fact that the largest source of noise in calcium imaging is photon shot noise, which is Poisson distributed. In later sections we will drop this linear dependence of variance on rate, as it has not been necessary for other methods to work well. We assume a maximum size of $\mathcal{S}_d$ along dimension d for any ROI, and at each step maintain an estimate of the location of the ROIs, along with the posterior mean $\mu_k^t \in \mathbb{R}^{\mathcal{S}_1 \times \ldots \times \mathcal{S}_D}$ and variance $\eta_k^t \in \mathbb{R}^{\mathcal{S}_1 \times \ldots \times \mathcal{S}_D}$ of each voxel in that ROI:

$$\mathbf{x}^k | \mathbf{y}^1, \ldots, \mathbf{y}^t \sim \mathcal{N}\left(\mu_k^t, \mathrm{diag}(\eta_k^t)\right) \tag{4.2}$$

While there is definitely significant covariance between voxels in a single ROI (otherwise, neurons would have no recognizable shape), we find that the signal to noise ratio is usually high enough in the data we are looking at that the shape of neurons can be accurately estimated without assuming any covariance between pixels, and learning is significantly faster because the number of parameters to track is lower and updates to the posterior do not require a full matrix inversion.

To identify regions of interest in a single frame, we use a simple, standard image processing pipeline. A soma can very roughly be approximated as a Gaussian bump[3], so finding regions of an image likely to contain a neuron can be approximated as finding parts of the image that match a Gaussian bump template. We convolve each frame with a Gaussian bump and keep anything that crosses a threshold as a potential region of interest. This is equivalent to keeping anything that has a high enough correlation with a Gaussian template. We then divide the areas above threshold into separate watershed regions, and identify each watershed region as one ROI. We also took advantage of GPU-optimized routines in MATLAB to accelerate this step. The Gaussian template size and threshold are parameters that must be chosen.

We decide whether or not to merge an ROI from one frame with ROIs from previous

[3]For fluorescent indicators localized in the cytosol, soma often appear more "donut" shaped, but we still find a Gaussian template works well in practice. Learning an optimal template or template basis for activity-based methods remains an open problem.

frames by seeing if the residual within a watershed region exceeds a threshold once the influence of already-detected ROIs has been subtracted off. To compute this residual, we have to estimate the fluorescence activity $\mathbf{r}^t$ given our posterior probability of ROI shape $p(\mathbf{x}^k|\mathbf{y}^1,\ldots,\mathbf{y}^{t-1}) = \mathcal{N}(\mathbf{x}^k|\mu_k^t, \mathrm{diag}(\eta_k^t)^2)$. From Eqn. 4.1 and Eqn. 4.2, we have that the marginal distribution of the data at time t given the past is given by

$$\mathbf{y}^t|\mathbf{y}^1,\ldots,\mathbf{y}^{t-1},\mathbf{r}^t \sim \mathcal{N}\left(\sum_{k=1}^{K} r_k^t\mu_k^t, \sigma^2\mathbf{I} + \sum_{k=1}^{K}\rho\mathrm{diag}\left(\mathbb{I}(\mathbf{x}^k)r_k^t\right) + \mathrm{diag}\left(r_k^t\eta_k^t\right)^2\right) \tag{4.3}$$

To estimate $\mathbf{r}^t$ we simplify considerably by dropping the dependence of the variance on $\mathbf{r}^t$, reducing the problem to least squares regression.

Once we have estimated $\hat{\mathbf{r}}^t = \arg\min_{\mathbf{r}^t} ||\mathbf{y}^t - \mu^t\mathbf{r}^t||_2^2$, the residual $\mathbf{y}^t - \mu^t\hat{\mathbf{r}}^t$ should be a sample from a zero-mean Gaussian with variance given above. Let Ω_j^t be the support of the jth watershed region in frame t, and let $\mathcal{P}_{\Omega_j^t}$ be the projection onto that watershed region. Then the squared scaled residual inside the watershed should be χ^2 distributed with $|\Omega_j^t|$ degrees of freedom:

$$\left\|\mathcal{P}_{\Omega_t^j}\left(\left(\sigma^2\mathbf{I} + \sum_{k=1}^{K}\rho\mathrm{diag}\left(\mathbb{I}(\mathbf{x}^k)r_k^t\right) + \mathrm{diag}\left(r_k^t\eta_k^t\right)^2\right)^{-1/2}(\mathbf{y}^t - \mu^t\hat{\mathbf{r}}^t)\right)\right\|_2^2 \sim \chi^2(|\Omega_j^t|) \tag{4.4}$$

If the scaled residual error inside a watershed is highly unlikely under this distribution, that means that the residual is far above what we'd expect if we'd detected all the ROIs already. Therefore this watershed region likely corresponds to a new region of interest, and we create a new ROI accordingly, with μ_{K+1}^t set to the value of the residual within the watershed and zero outside, and η_{K+1}^t set to the background variance. Generally we set this threshold very high, in practice we found a cumulative probability on the order of $1 - 10^{-14}$ worked well. If the residual is consistent with background noise, then that particular watershed region is assigned to the the ROI with maximum power in that region: $\arg\max_k ||\mathcal{P}_{\Omega_j^t}(\mu_k^t r_k^t)||_2$, and the posterior distribution is updated. For simplicity, and to keep the number of parameters manageable, we ignore cross-terms between ROIs in the posterior update. This could be considered a variational approximation to the true posterior, albeit a trivial one. In practice this does not strongly affect the inferred ROI shapes. For the sake of clarity, let $\Sigma = \sigma^2\mathbf{I} + \rho\mathrm{diag}(\mathbb{I}(\mathbf{X})\mathbf{r}^t)$, then the posterior update takes the form:

$$\mathrm{diag}(\eta_k^t) = ((r_k^t)^2\Sigma^{-1} + \mathrm{diag}(\eta_k^{t-1})^{-1})^{-1} \tag{4.5}$$

$$\mu_k^t = \mathrm{diag}(\eta_k^t)(r_k^t\Sigma^{-1}\mathcal{P}_{\Omega_j^t}(y) + \mathrm{diag}(\eta_k^{t-1})\mu_k^{t-1}) \tag{4.6}$$

We tested this model on a small sample of data from the optic tectum to validate the approach. We chose the threshold and template width by inspection on the first few frames, while σ and ρ were fit automatically: since a large number of ROI were constant across frames, if the regional maxima between two frames was within 3 pixels the ROI were automatically merged, and by comparing the residual variance between the two frames, σ and ρ could be fit by linear regression. We compared the results on this data sample to human annotators (Fig. 4.1) and found that false positives and negatives were on the order of 5-10%, comparable to the difference in judgements between annotators. We also looked at a sample of data across z-planes that includes structured background artifacts due to refraction of the laser pencil (Fig. 4.2). This leads to a streaky moving shadow in a line across the data, which can be misinterpreted as an ROI. While there do seem to be slightly more false positives in the section of the data crossed by this artifact, ROI detection is mostly robust.

It is because we simply choose the ROI with the highest power that we call this online greedy clustering. Alternatively, one could do something like online inference in a Dirichlet process mixture model. However, the Dirichlet process prior leads to a rich-gets-richer effect regardless of where the data is located, making it uniformly less likely for a new ROI to be inferred no matter where in the image one looks. This nonlocal effect is not desirable. One could also use a prior on ROI locations such as a determinantal point process (Borodin, 2009), however efficient inference with DPPs is still an ongoing area of research, and while the greedy clustering approach is somewhat heuristic it generally worked well in practice. In the next section we will present a model with a simple but principled sparsity prior on ROI locations that also works well.

4.2.2 Applying Online Greedy Clustering to Whole Brain Activity

We applied online greedy clustering to a set of data from the larval zebrafish brain. GCaMP5 was expressed cytosolically and 1000 frames of spontaneous activity from 80% of the brain were recorded by light-sheet microscopy at 0.8 Hz while the animal's head was suspended in agarose gel. More details on the experimental preparation and recording can be found in Ahrens et al. (2013). Applying the Gaussian convolution to one frame of data took 30s on a desktop CPU, but only 1-2 seconds on a Tesla K20 GPU. Both the size of the Gaussian template and the threshold were chosen by hand by looking at 2-3 frames of data. The slowest step at each iteration was the regression to compute the fluorescence activity of existing ROIs: $\hat{\mathbf{r}}^t = \arg\min_{\mathbf{r}^t} ||\mathbf{y}^t - \mu^t \mathbf{r}^t||_2^2$, which took on the order of dozens of seconds, even with custom code that takes into account the fact that μ_t is zero outside a small region.

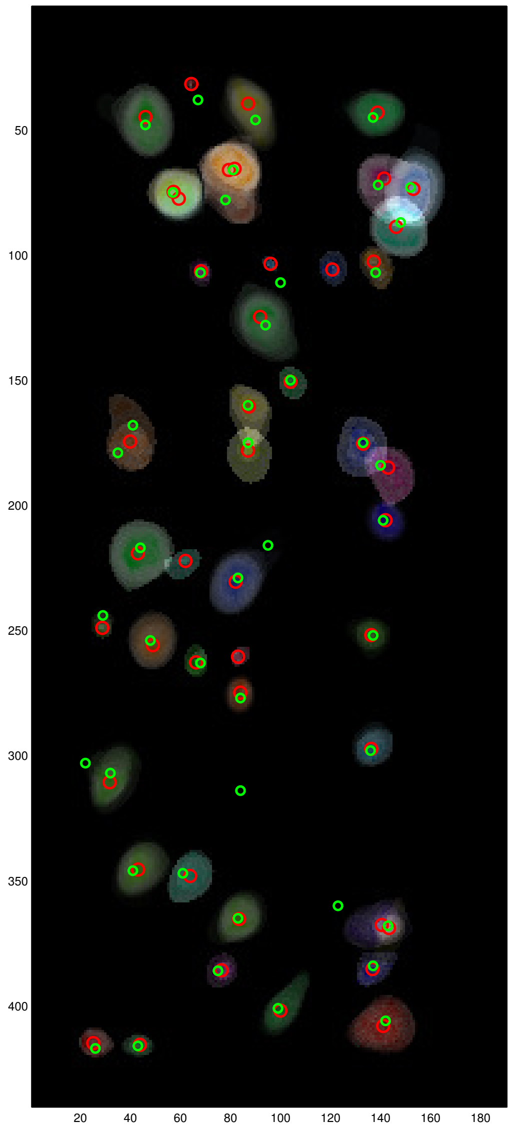


Figure 4.1: Comparison of human and automated ROI detection. ROI centers found by online greedy clustering are in red, human annotation is in green. The ROI shapes are plotted underneath, where the hue is chosen randomly for each ROI, the mean of each pixel μ_k is given by the saturation and the precision of each pixel $1/\eta_k$ is given by the value. That is, in bright regions we are highly confident about the shape of that ROI, and in colored regions we believe that there is a high amount of calcium-dependent fluorescent activity.

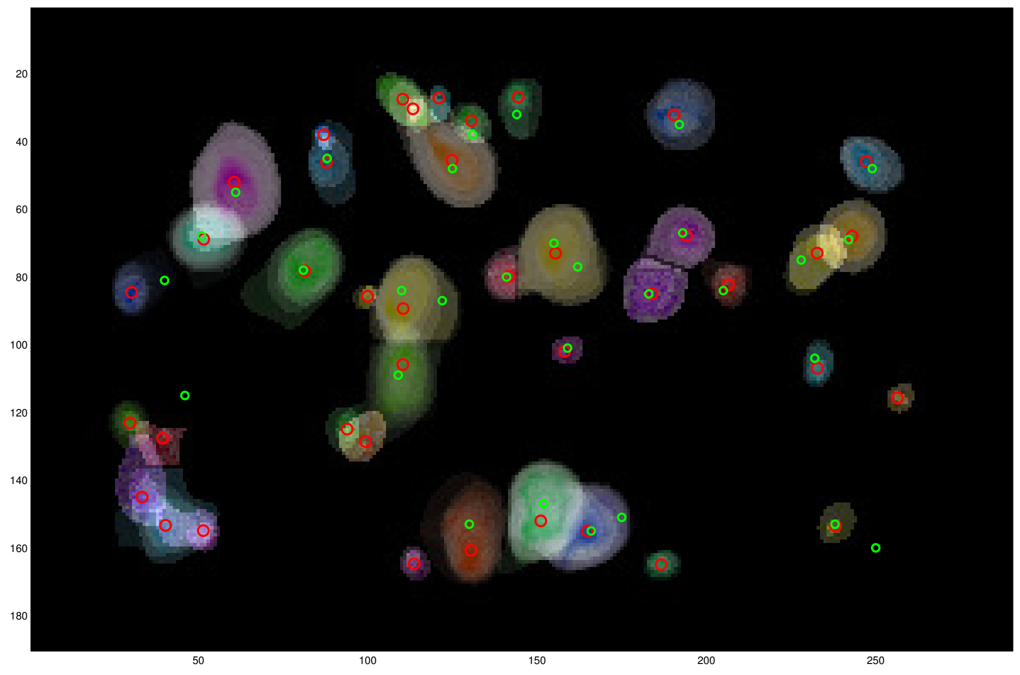


Figure 4.2: Comparison of human and automated ROI detection for one z-slice of 3D data with "penciling" artifact. The pencil artifact goes vertically through the data about one third of the way from the left side of the frame. The color scheme is the same as for Fig. 4.1

Overall, depending on where the threshold was set, one complete run of the algorithm on an HP SL250 Gen8 node (64 GB) with dual Intel E5-2650L Processors (1.8 GHz) took between 8 hours and a day, with the overall number of ROIs found varying between 2,300 and 6,000. At the high end of where the threshold was set, visual inspection shows that some neurons are clearly being missed, while at the low end, a noticeable fraction appear to be false positives. The rest of the analysis in this section is with the threshold at its lowest reasonable value, yielding 2,331 regions of interest.

The distribution of ROIs can be seen in Fig. 4.3. The number is dramatically smaller than the total number of neurons in the field of view, estimated to be closer to 80,000. There are two likely factors contributing to this: when the animal is not behaving, many neurons are never active and will not be detected, and the threshold is higher than necessary, to avoid false positives (for instance, visual inspection of the optic tectum shows some very sparsely active neurons that are missed). However, we believe that we detect almost all of the easily visible neurons.

Focusing in more closely on a small number of regions reveals that the detected ROIs are mostly consistent with what can be seen by eye, despite the diversity of neural morphology in different regions. In Fig. 4.4 we show detail from a number of different brain regions, with ROI in color overlaid on the mean fluorescence image. In the right habenula (Fig. 4.4(a)) the neurons are very densely packed and have diverse (and diffuse) shapes. ROI detection mostly separates these neurons, though two neurons are occasionally erroneously merged together. In Fig. 4.4(b) we look at the optic tectum, where neurons are much more sparsely distributed, and find ROI are well-isolated. There is also a large amount of active neuropil in the data, and we identify that as ROIs as well, as in Fig. 4.4(c), which divides an axon tract heading to the spinal cord into many ROI.

Looking at the time course of fluorescence activity also helps to evaluate the per-

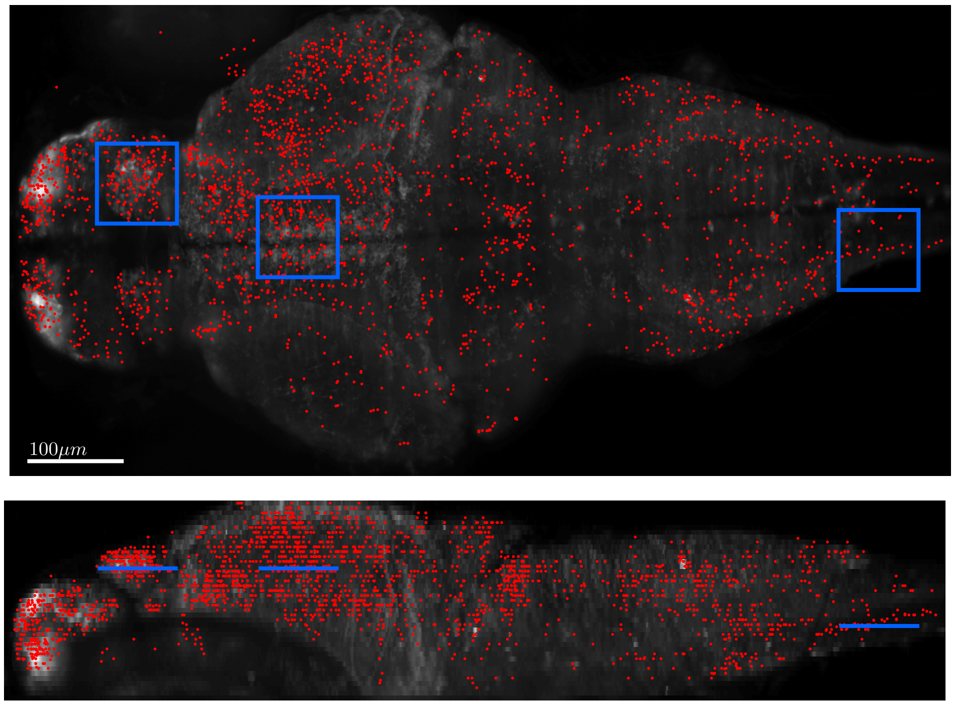
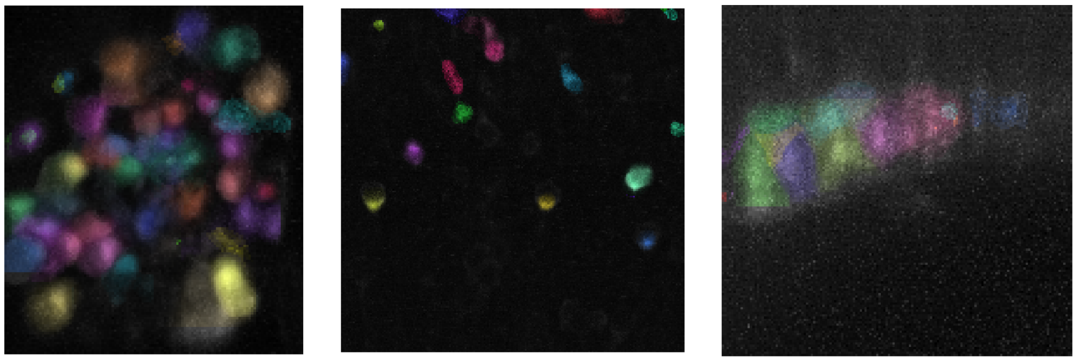


Figure 4.3: Location of regions of interest found by online greedy clustering superimposed over mean fluorescence image of the larval zebrafish. The blue boxes give the location of the details in Fig. 4.4.



(a) Habenula (b) Optic Tectum (c) Hindbrain

Figure 4.4:

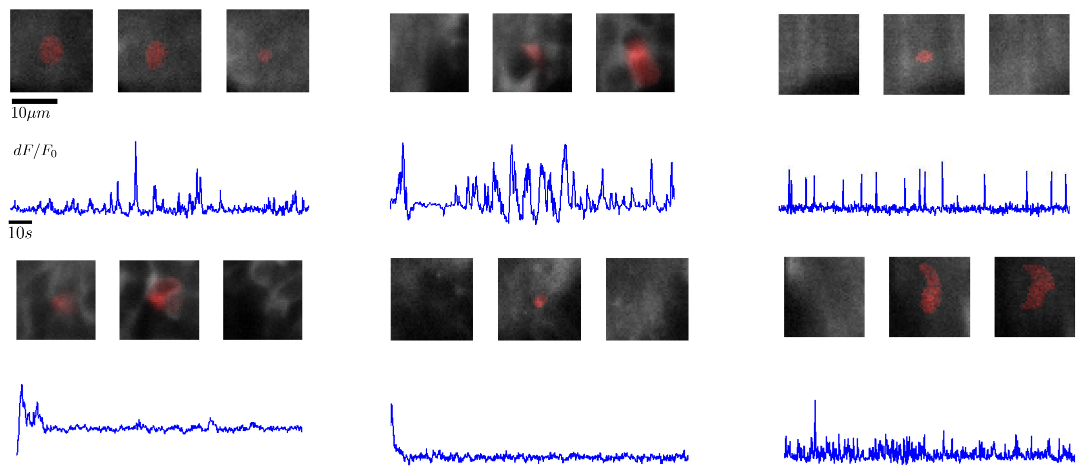


Figure 4.5: Shape and fluorescence time course of a random sampling of ROI. Some ROI both match the anatomy and show complex fluorescence dynamics (e.g. top middle, bottom left), while others show complex dynamics but do not match anything in the mean fluorescence (e.g. top right), and yet others are likely artifact (e.g. bottom right).

formance of ROI detection, and gives a sense of how diverse neural activity is across the zebrafish brain. In Fig. 4.2.2 we look at both the shape and time course of a number of individual ROIs. Some closely match the mean fluorescence image (where neurons can be distinguished by dark spots, since GCaMP is not expressed in the nucleus of these animals), while others are so sparsely active that they are not visible at all in the mean image, though the activity is clearly not random. Others are likely artifact, judging by the fact that they only cross a threshold for detection once, which is a useful heuristic for being discarded.

4.2.3 Visualizing Spontaneous Activity with Nonlinear Dimensionality Reduction

Once we have reduced the data from raw voxels to activity in regions of interest, the range of possible analyses becomes much wider. In particular, nonlinear dimensionality reduction techniques can be practically applied to data at this scale which could not be applied to raw voxels, even with the use of distributed clusters. Nonlinear dimensionality reduction encompasses a wide range of techniques in machine learning, including classic algorithms such as multi-dimensional scaling (Kruskal and Wish, 1978) and modern algorithms like Isomap (Tenenbaum et al., 2000), Locally Linear Embedding (LLE) (Roweis and Saul, 2000), Laplacian Eigenmaps (Belkin and Niyogi, 2003), Hessian Eigenmaps (Donoho and Grimes, 2003), Semidefinite Embedding (SDE) (Weinberger and Saul, 2006b), and Maximum Variance Unfolding (MVU) (Weinberger and Saul, 2006a) among others. For a review of the nonlinear dimensionality reduction literature, we direct the reader to (Lee and Verleysen, 2007).

Nonlinear dimensionality reduction techniques mostly require the construction of a pairwise similarity matrix, and for this reason scale at least as $\mathcal{O}(N^2)$ unless approximations are used, making them difficult to implement in distributed systems. However, for medium-scale problems they present an attractive approach to exploratory data analysis, as they can produce visualizations that much more cleanly separate interesting structure than linear methods like PCA. For instance, in (Stopfer et al., 2003), applying LLE to multiunit recordings from projection neurons in the locust antennal lobe clearly reveals distinct trajectories of population activity in response to different odorants, and a change in the structure of these trajectories in response to changes in intensity, thus showing that both odor intensity and identity are encoded in a separable way at the population level.

We chose to apply t-distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and Hinton, 2008) to the extracted calcium activity time courses. While many standard nonlinear dimensionality reduction techniques are not robust to noise and work well when the data comes from a single manifold, t-SNE is particularly well adapted to the case when the data contains groups or clusters that exist on many separate manifolds, for instance multiple handwritten digits, or facial expressions from many individuals. In the case of whole-brain calcium recordings, different brain regions contain qualitatively different kinds of neurons and patterns of activity, while the activity of neurons within those regions are likely to be similar but still fall on some low-dimensional manifold. Conceptually, t-SNE works by defining, for every pair of data points, a conditional probability that one point is the neighbor of another point, and then finding points in a low dimensional space such that the probability distributions are well matched. In the original paper on Stochastic Neighbor Embedding (Hinton and Roweis, 2002), the probability of having a neighbor at a certain distance was taken to be a Gaussian in both the high and low dimensional space. However, this led to the problem of "crowding", where most data points were mapped into tight clumps in the center of the low dimensional space. By replacing the Gaussian distribution with a Student-t distribution for the low dimensional mapping, the low dimensional embeddings can "spread out" more, in practice leading to improved visualization.

We looked at t-SNE and PCA applied to 2331 regions of interest recovered by online greedy clustering. When mapped into the space of principal components, neurons were largely clumped together and difficult to distinguish, with the exception of two large lobes that roughly correspond to the hindbrain oscillator and hindbrain-spinal circuit (Fig. 4.8). This can be more clearly seen if the regions of interest are plotted by position in the zebrafish, and colored based on position in principal component space, so that ROIs further from the center have brighter hue (Fig. 4.7). Again, the hindbrain oscillator and hindbrain-spinal circuit stand out but other anatomical regions are not clearly distinguished. By contrast, t-SNE projected the regions of interest into a number of groups. The left and right parts of the hindbrain-spinal circuit were cleanly separated from the rest of the regions of interest, and functionally distinct regions of the zebrafish brain were far easier to distinguish. This shows that nonlinear dimensionality reduction techniques, which cannot be applied on a voxel-by-voxel basis even using distributed algorithms, can be a powerful tool for exploratory data analysis in whole brain recordings.

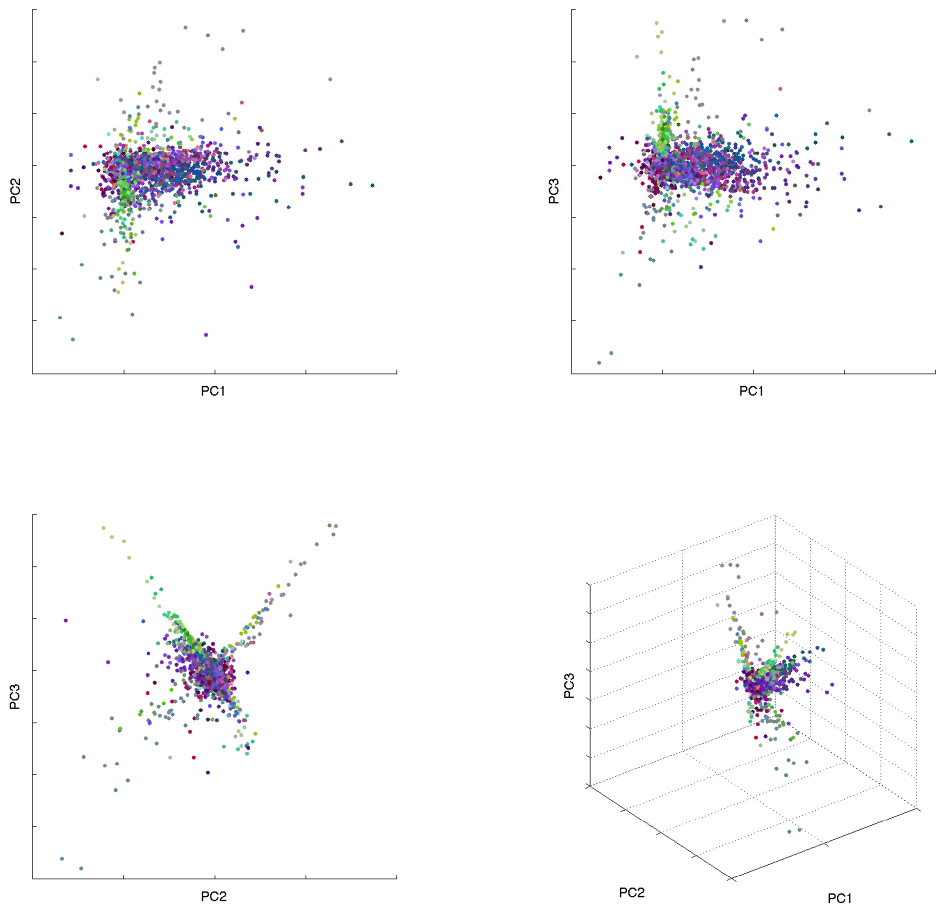
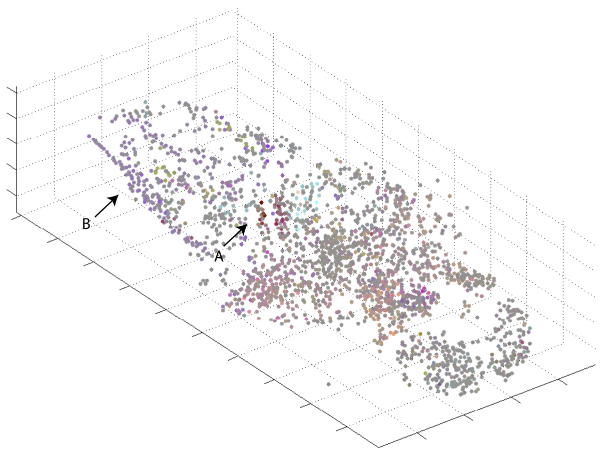
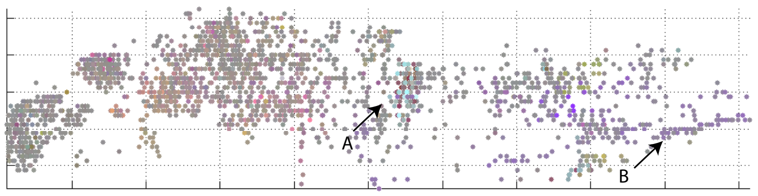


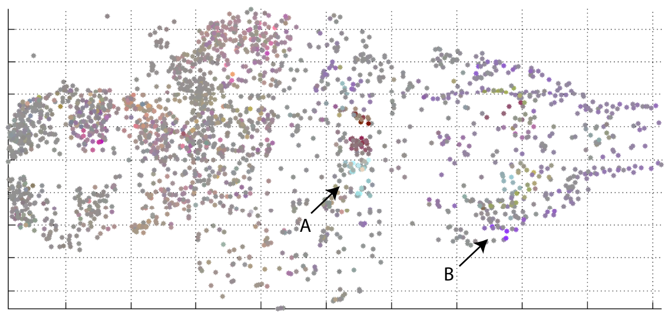
Figure 4.6: Top 3 principal components of spontaneous activity in regions of interest in the larval zebrafish brain. Color corresponds to the position of each region in the fish (Red: mediolateral axis. Green: rostrocaudal axis. Blue: dorsoventral axis.)



(a)



(b)



(c)

Figure 4.7: (a) Regions of interest in the larval zebrafish brain, colored by location of spontaneous activity projected into principal component space. The coordinate of the 3D embedding for each ROI is given by the red, green and blue values for each point. The hindbrain oscillator (A) and hindbrain-spinal circuit (B) can be distinguished by eye but other anatomical regions are less clear. (b) Sagittal view. (c) Horizontal view.

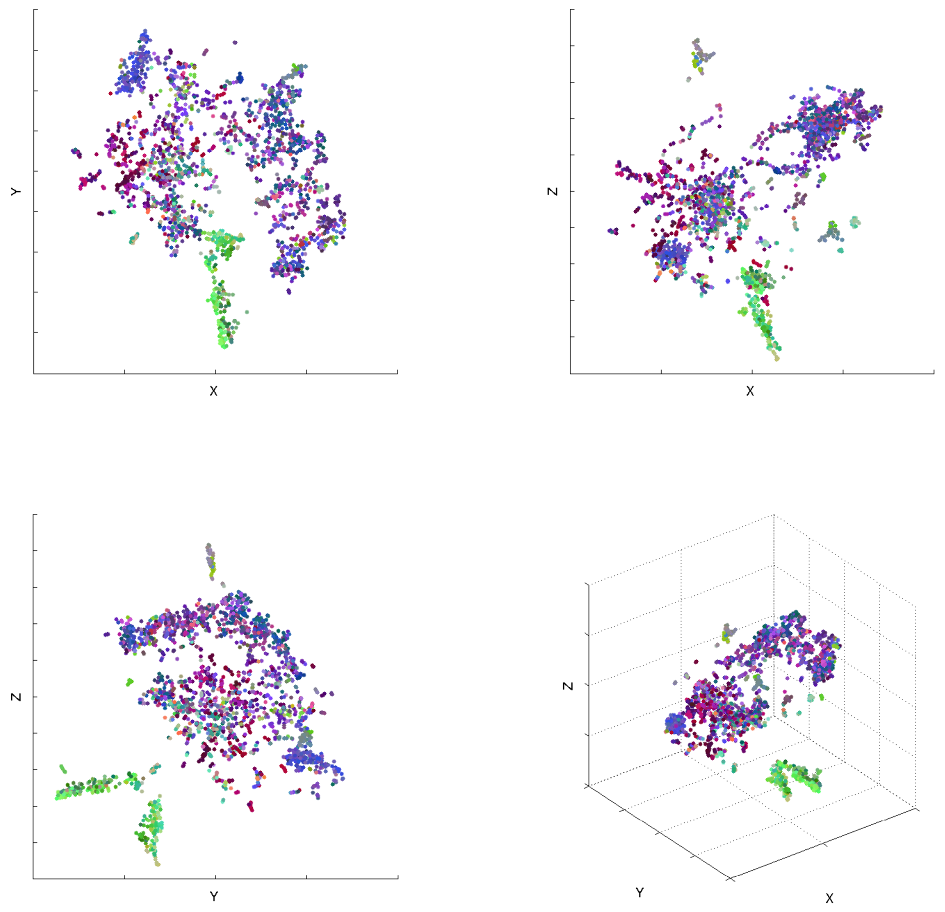
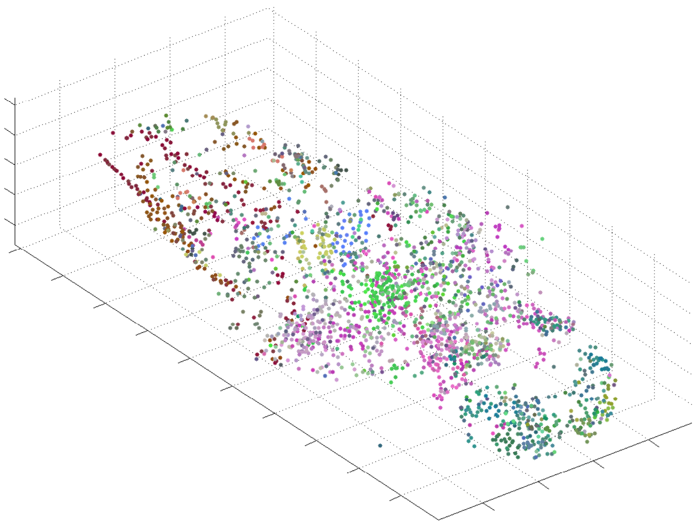
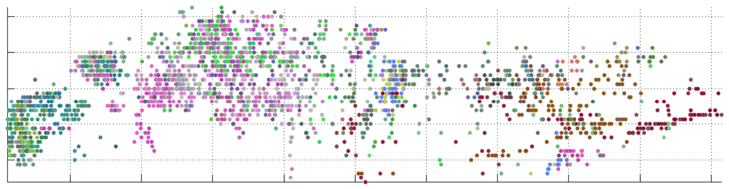


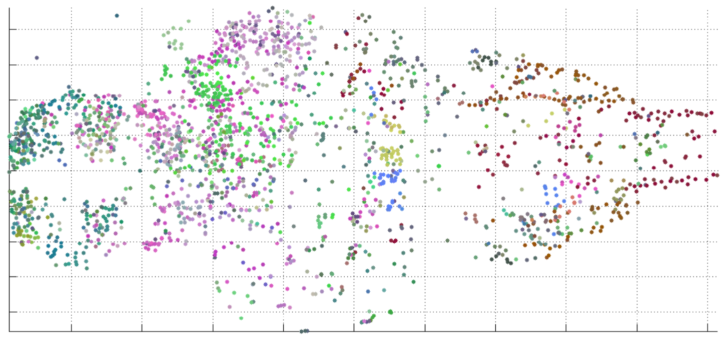
Figure 4.8: t-SNE applied to spontaneous activity in regions of interest in the larval zebrafish brain. Color corresponds to the position of each region in the fish (Red: mediolateral axis. Green: rostrocaudal axis. Blue: dorsoventral axis.)



(a)



(b)



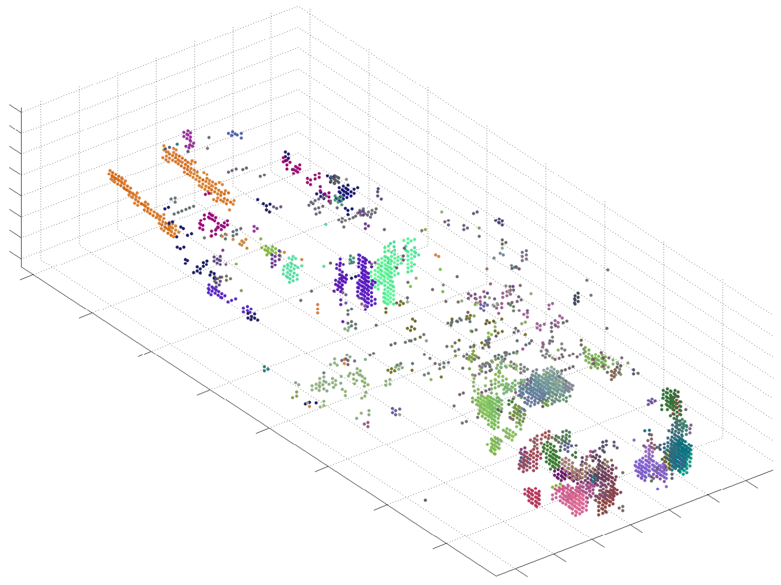
(c)

Figure 4.9: (a) Regions of interest in the larval zebrafish brain, colored by location of spontaneous activity in t-SNE embedding space. The same embedding color scheme is used as in Fig. 4.7. (b) Sagittal view. (c) Horizontal view.

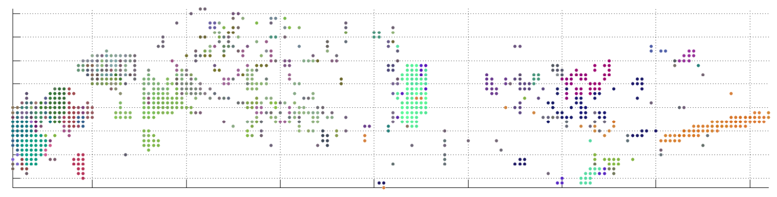
It is possible that similar results using t-SNE could be achieved with much simpler region of interest detection. To see if online greedy clustering really does find a more interpretable, meaningful representation of the data, we applied the same analysis to supervoxels generated by downsampling the data in the x and y direction. Voxels are separated by 406.25 nm in the x and y direction and roughly 5 μm in the z direction, and were downsampled by a factor of 12.5 in x and y, so that each supervoxel is roughly 5 μm on a side. We then discarded any supervoxel where the variance of fluorescence activity did not cross a threshold so that the number of supervoxels was of the same order of magnitude as the number of regions of interest (3079 total). Applying t-SNE to each supervoxel time series clustered the data into nearly uniform anatomical group (Fig. 4.10), as opposed to the embedding found on the results of online greedy clustering, which finds widely spatially distributed ROIs with correlated activity. This reflects the fact that downsampling the data mixes the signal between adjacent neurons together, so that neighboring supervoxels will almost always be correlated and be far more likely to fall into the same cluster. Fine details of the circuitry are lost, for instance a single neuron that is highly correlated with a distant region of the brain will have its activity mixed together with neighboring neurons, making it far more difficult to distinguish.Thus naive methods for reducing the dimensionality of the data can throw away useful information, making the next stage of analysis far less informative.

4.3 Convolutional Group Lasso

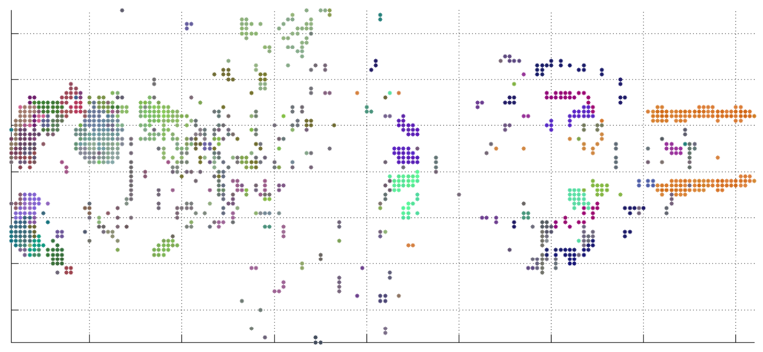
The online greedy clustering method presented in the last section is effective at finding a large fraction of active neurons in whole-brain recordings, but has a number of flaws. It has several free parameters that need to be tuned, relies on a number of heuristics that make implementation complicated, and still takes a large fraction of a day to run even with



(a)



(b)



(c)

Figure 4.10: (a) Supervoxels in the larval zebrafish brain, colored by location of spontaneous activity in t-SNE embedding space. The embedding color scheme is the same as that used in Fig. 4.7. (b) Sagittal view. (c) Horizontal view.

state-of-the-art GPU technology (though it could likely be optimized further). If we pursue a distributed rather than online algorithm, this frees us to pursue a much wider range of models, some of which could be considerably easier to implement. In this section we describe one such model that is simple but still effective, which we refer to as *convolutional group lasso*. The basic idea behind the convolutional group lasso is that most regions of interest can be approximated by a simple template, only a small number of locations in the field of view will be at the center of a region of interest, and that while the intensity of that template may change over the course of recording, the location will not (so long as the data are well aligned). Once the location of the ROIs are found, the shape of the ROIs can be approximated by fitting a constrained least squares problem that is easily implemented by alternately taking the first principal component in an ROI, conditioned on the influence of any overlapping ROIs having already been subtracted off.

4.3.1 Finding ROI Locations with Group Lasso

Let the range of indices in one frame of data be denotes by the set $\mathcal{R}$, the template for the shape of an ROI be denoted $\psi \in \mathbb{R}^{|\mathcal{R}|}$, where $|\mathcal{R}|$ is the number of voxels in one frame of the video, let $\mathcal{Z} = \{\mathbf{z}_k | k = 1, \ldots, K\} \subset \mathcal{R}$ denote the set of region of interest locations, and let r_{kt} denote the intensity of fluorescence activity from ROI k at time t, and as a convention let $\mathbf{x}^i$ denote the ith column of a matrix $\mathbf{X}$, while $\mathbf{x}_j$ denotes the jth row. Finally let a column of a matrix reshaped as a 2D or 3D array in the shape of one frame of video data be denoted by a capital letter with a superscript, so $\mathbf{X}^t$ is just $\mathbf{x}^t$ reshaped. Then a single frame of the calcium fluorescence video at time t can be modeled as

$$\begin{aligned}
\mathbf{Y}^t &= \left(\psi * \mathbf{x}^t\right) + \mathbf{E}^t && (4.7)\\
\mathbf{X}^t_{z_k} &= r_{kt}\,\forall k = 1, \ldots, K && (4.8)\\
\mathbf{X}^t_z &= 0\,\text{if}\ z \notin \mathcal{Z} && (4.9)\\
\mathbf{E}^t &\sim \mathcal{N}(0, \sigma^2\mathbf{I}) && (4.10)
\end{aligned}$$

This differs from the model used in online greedy clustering. Instead of $\mathbf{x}$ denoting the shape of an ROI, the full array $\mathbf{X}$ stores the location of ROIs and intensity of calcium fluorescence as a large sparse array. If we can recover the matrix $\mathbf{X} = \left(\mathbf{x}^1, \ldots, \mathbf{x}^T\right)$ we will have full knowledge of the location and activity rate of all ROIs (though not detailed knowledge of their shape). For the rest of this chapter however we will work in vectorized notation wherever possible. Since convolution is a linear operation we can represent it as a matrix: let $\mathbf{A}_\psi \in \mathbb{M}^{|\mathcal{R}|\times|\mathcal{R}|}$ be the square matrix that performs 2- or 3-dimensional convolution with the template ψ. There are two critical things to note about the above generative model: $\mathbf{x}^t$ is *extremely* sparse (for instance, for whole-brain zebrafish data one frame is on the order of 10^8 voxels but 10^4 regions of interest), and for an index $z \notin \mathcal{Z}$, $\mathbf{x}^t_z = 0$ for all times t. Thus to recover $\mathbf{X}$, we should penalize the rows of the matrix (corresponding to the same voxel in each frame over time) such that the matrix $\mathbf{X}$ is not only sparse, but if one entry in $\mathbf{X}$ is pushed to zero, then all entries in that row are pushed to zero. That is exactly the structure induced by using a group sparsity penalty (Huang and Zhang, 2010), which generalizes the ℓ_1 penalty to groups of entries in a vector that should be made sparse *together*. The objective we seek to minimize is given by:

$$\min_{\mathbf{X}} \left[\|\mathbf{A}_\psi \mathbf{X} - \mathbf{Y}\|^2 + \lambda \sum_z \|\mathbf{x}_z\|_2\right]. \qquad (4.11)$$

where the sum is taken over all rows z of the matrix $\mathbf{X}$, corresponding to the same voxel in each frame.

This is a group-sparse optimization problem with a convolutional basis. Exactly recovering sparse (or in this case group-sparse) solutions is often difficult with a convolutional basis, as the basis elements are highly correlated. Trying to recover a sparse vector from a convolutional basis by ℓ_1 minimization approaches will lead to the solution being "smeared out" due to correlations in the basis. If the regions of interest are well-separated, matching pursuit (Mallat and Zhang, 1993) can be an effective approximation. For sparse recovery of continuous-valued locations, continuous basis pursuit is another possibility (Ekanadham et al., 2011). In practice we find that, despite the smearing inherent in using a generalization of the ℓ_1 penalty for sparse recovery, the ROIs are generally well-separated enough that a simple threshold-and-regional-maximum step at the end finds good locations.

We solve Eqn. 4.11 by fast iterative soft-thresholding (FISTA) (Beck and Teboulle, 2009). FISTA is a variant of iterative soft-thresholding (ISTA), which is a very simple algorithm for minimizing convex functions that are the sum of a smooth and nonsmooth term. Because iterative soft thresholding algorithms only require the computation of the gradient of the smooth term and the proximal operator of the nonsmooth term, they can be applied to very large scale problems. However their convergence can be quite slow - sublinear in the worst case. FISTA extends ISTA to include a momentum term in the updates, and provably accelerates convergence from $\mathcal{O}(1/k)$ to $\mathcal{O}(1/k^2)$ in the number of iterations k. There are also extensions to ISTA that under certain conditions can lead to *exponential* convergence rates, namely approximate message passing (AMP) (Donoho et al., 2009), however those conditions are not met when the basis is convolutional, and in empirical experiments AMP did not fare well compared to FISTA.

FISTA solves the following minimization problem:

$$\mathbf{x}^* = \min_{\mathbf{x}\in\mathbb{R}^n} \left[f\left(\mathbf{x}\right) + \lambda g(\mathbf{x})\right] \,. \tag{4.12}$$

where $f(\mathbf{x})$ is a smooth function with a Lipschitz-continuous gradient: $\exists L$ s.t. $||\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})|| \leq L||\mathbf{x}-\mathbf{y}||\ \forall \mathbf{x},\mathbf{y} \in \mathbb{R}^{|\mathcal{R}|}$ and $g(\mathbf{x})$ is a convex function with a proximal operator $\mathcal{T}_t(\mathbf{x}) \triangleq \arg\min_{\mathbf{z}} g(\mathbf{z}) + \frac{1}{2t}||\mathbf{x}-\mathbf{z}||_2^2$, via the following iterative algorith:

Algorithm 4 Fast Iterative Soft Threshold (FISTA)

```
**Input** Smooth function $f(\mathbf{x})$, gradient $\nabla f(\mathbf{x})$, convex function $g(\mathbf{x})$, proximal operator $\mathcal{T}_t(\mathbf{x})$, Lipschitz constant $L$ of $\nabla f$, initial point $\mathbf{x}_0 \in \mathbb{R}^|\mathcal{R}|$
**Output** Solution $\mathbf{x}^*$ to Eqn. 4.12
$\mathbf{y}_1 = \mathbf{x}_0$, $t_1 = 1$, $\mu = \lambda/L$
**while** $||\mathbf{x}_k - \mathbf{x}_{k-1}|| > \epsilon$ **do**
    $\mathbf{x}_k = \mathcal{T}_\mu\left[\mathbf{y}_k - \frac{2}{L}\nabla f(\mathbf{y}_k)\right]$
    $t_{k+1} = \left(1+\sqrt{1+4t_k^2}\right)/2$
    $\mathbf{y}_{k+1} = \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}}\left(\mathbf{x}_k - \mathbf{x}_{k-1}\right)$
**end while**
$\mathbf{x}^* = \mathbf{x}_k$
```

For the convolutional group lasso problem, the proximal operator for the group sparsity penalty has a simple closed form:

$$[\mathcal{T}_t(\mathbf{X})]_z = \left(1 - \frac{t}{\|\mathbf{x}_z\|_2}\right)_+ \mathbf{x}_z \tag{4.13}$$

That is, an entire group is scaled by the same factor $1 - \frac{t}{\|\mathbf{x}_z\|_2}$ unless this factor is less than zero, in which case the group is set to zero. The gradient of the smooth part of the objective is also simple: $\nabla f(X) = \mathbf{A}_\psi^T(\mathbf{A}_\psi \mathbf{X} - \mathbf{Y})$, which has a Lipschitz constant (equal to the maximum eigenvalue of $\mathbf{A}_\psi^T\mathbf{A}_\psi$) of 2 if the template ψ is normalized, leading to the following algorithm:

Algorithm 5 FISTA for Convolutional Group Lasso

```
**Input** Convolutional template $\psi$, data $\mathbf{Y} \in \mathbb{M}^{|\mathcal{R}|\times T}$, initial point $\mathbf{X}_0 \in \mathbb{M}^{|\mathcal{R}|\times T}$
**Output** Solution $\mathbf{X}^*$ to Eqn. 4.11
$\mathbf{W}_1 = \mathbf{X}_0$, $t_1 = 1$, $\mu = \lambda/2$
**while** $||\mathbf{X}_k - \mathbf{X}_{k-1}|| > \epsilon$ **do**
    $\mathbf{X}_k = \mathcal{T}_\mu\left[\mathbf{W}_k - A_\psi^T\mathbf{A}_\psi\mathbf{W}_k + \mathbf{A}_\psi^T\mathbf{Y}\right]$
    $t_{k+1} = \left(1 + \sqrt{1+4t_k^2}\right)/2$
    $\mathbf{W}_{k+1} = \mathbf{X}_k + \left(\frac{t_k-1}{t_{k+1}}\right)(\mathbf{X}_k - \mathbf{X}_{k-1})$
**end while**
$\mathbf{X}^* = \mathbf{X}_k$
```

Note we replace $\mathbf{y}$ in Alg. 4 with $\mathbf{W}$ to avoid confusion with the data $\mathbf{Y}$. Since $\mathbf{A}_\psi$ is a convolution operator, $\mathbf{A}_\psi^T = \mathbf{A}_\psi$ and $\mathbf{A}_\psi^T\mathbf{A}_\psi = \mathbf{A}_{\psi*\psi}$. One natural choice of template is a Gaussian kernel with covariance Σ: $\psi(\mathbf{x}) = \frac{1}{\sqrt{2\pi|\Sigma|}}\exp(-\frac{1}{2}\mathbf{x}^T\Sigma^{-1}\mathbf{x})$. Then applying the convolution twice is the same as applying a convolution with a Gaussian kernel with covariance 2Σ. Moreover, the matrix $\mathbf{A}_\psi$ never has to be explicitly stored in memory, and multiplication can be implemented using a fast Fourier transform, meaning one iteration of FISTA can be run in $\mathcal{O}(|\mathcal{R}|\log|\mathcal{R}|)$ time. Moreover, for very large fields of view we can split the data into patches with a border around them, run FISTA on each patch and stitch the results back together, so if the data is split into K patches, each step is only $\mathcal{O}(\frac{|\mathcal{R}|}{K}\log|\mathcal{R}|)$. So long as the border is as wide as the support of the filter applied twice $\psi * \psi$, the results will not change. This is exactly the strategy for distributed optimization we will adopt.

There are various ways of extending this basic algorithm to be more robust and automated. For instance, we may also want to subtract off a constant background signal $\mathbf{u}$, in which case the objective becomes:

$$\min_{\mathbf{X},\mathbf{u}} \left[\left\| \mathbf{A}_\psi \mathbf{X} - \mathbf{Y} + \frac{1}{T}\mathbf{u}\mathbf{1}_T^T \right\|^2 + \lambda \sum_z \left\| \mathbf{x}_z \right\|_2 \right]. \tag{4.14}$$

the Lipschitz constant becomes $2(1+T^{-1})$ as the derivative depends on $\mathbf{u}$ as well, and FISTA can be extended to handle this case as follows:

Algorithm 6 FISTA for Convolutional Group Lasso with Bias

```
**Input** Convolutional template $\psi$, data $\mathbf{Y} \in \mathbb{M}^{|\mathcal{R}| \times T}$, initial points $\mathbf{X}_0 \in \mathbb{M}^{|\mathcal{R}| \times T}$, $\mathbf{u}_0 \in \mathbb{R}^{|\mathcal{R}|}$
**Output** Solution $\mathbf{X}^*$ to Eqn. 4.14
$\mathbf{W}_1 = \mathbf{X}_0$, $\mathbf{v}_1 = \mathbf{u}_0$, $t_1 = 1$, $\mu = \frac{\lambda}{2(1+T^{-1})}$
**while** $\max(||\mathbf{X}_k - \mathbf{X}_{k-1}||, ||\mathbf{u}_k - \mathbf{u}_{k-1}||) > \epsilon$ **do**
  $\mathbf{X}_k = \mathcal{T}_\mu \left[ \mathbf{W}_k - \frac{T}{1+T}\mathbf{A}_\psi^T \mathbf{A}_\psi \mathbf{W}_k - \frac{1}{1+T}\mathbf{A}_\psi^T \mathbf{v}_k \mathbf{1}_T^T + \frac{T}{1+T}\mathbf{A}_\psi^T \mathbf{Y} \right]$
  $\mathbf{u}_{k+1} = \mathbf{v}_k - \frac{1}{1+T}\left( \mathbf{A}_\psi^T \mathbf{W}_k + \mathbf{v}_k \mathbf{1}_T^T - Y \right) \mathbf{1}_T^T$
  $t_{k+1} = \left(1 + \sqrt{1 + 4t_k^2}\right)/2$
  $\mathbf{W}_{k+1} = \mathbf{X}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(\mathbf{X}_k - \mathbf{X}_{k-1})$
  $\mathbf{v}_{k+1} = \mathbf{u}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(\mathbf{u}_k - \mathbf{u}_{k-1})$
**end while**
$\mathbf{X}^* = \mathbf{X}_k$, $\mathbf{u}^* = \mathbf{u}_k$
```

4.3.2 Automatically Tuning the Sparsity Parameter

The only free parameters in Eqn. 4.11 are the template ψ and sparsity parameter λ. We have developed a method for automatically tuning λ by estimating the background noise and updating λ by gradient descent until the residual variance matches our estimate of σ

in Eqn. 4.10.

If we have found all regions of interest using group lasso, the the residual is just white noise:

$$\frac{1}{T}\left\|\mathbf{A}_{\psi}\mathbf{X}-\mathbf{Y}\right\|^{2}=\frac{1}{T}\left\|\mathbf{E}\right\|^{2}=\frac{1}{T}\sum_{t=1}^{T}\|\mathbf{E}^{t}\|^{2}\xrightarrow{a.s.}\mathbb{E}[\|\mathbf{E}^{t}\|^{2}]=|\mathcal{R}|\sigma^{2} \tag{4.15}$$

where the limit is taken as $T\rightarrow\infty$. Therefore, if we denote

$$\mathbf{X}^{*}\left(\lambda\right)=\mathrm{argmin}_{\mathbf{X}}\left[\left\|\mathbf{A}_{\psi}\mathbf{X}-\mathbf{Y}\right\|^{2}+\lambda\sum_{z}\left\|\mathbf{x}_{z}\right\|_{2}\right] \tag{4.16}$$

then we would like to choose λ according to $\lambda^{*}=\mathrm{argmin}_{\lambda}\left(\left\|\mathbf{A}_{\psi}\mathbf{X}^{*}\left(\lambda\right)-\mathbf{Y}\right\|^{2}-|\mathcal{R}|T\sigma^{2}\right)^{2}$.

Note that this estimate of λ would be good, if the data model (Eqn. 4.10) is correct. However, in order to do this we need to find $\left\|\mathbf{E}\right\|^{2}\approx|\mathcal{R}|T\sigma^{2}$. Denoting $\mathbf{r}$ and $\mathbf{k}$ to be 2D or 3D indices in $\mathcal{R}$ and $I_{\mathbf{Y}^{t}}\left(\mathbf{k}\right)$ to be periodogram (an estimator of the power spectral density) of $\mathbf{Y}^{t}$

$$\begin{aligned}
I_{\mathbf{Y}^{t}}\left(\mathbf{k}\right) &= \frac{1}{|\mathcal{R}|}\left|\sum_{\mathbf{r}\in\mathcal{R}}\left(\mathbf{Y}^{t}\right)_{\mathbf{r}}e^{2\pi i(\mathbf{r}\cdot\mathbf{k})}\right|^{2} && (4.17)\\
&= \frac{1}{|\mathcal{R}|}\left|\sum_{\mathbf{r}\in\mathcal{R}}\left(\mathbf{A}_{\psi}\mathbf{X}^{t}+\mathbf{E}^{t}\right)_{\mathbf{r}}e^{2\pi i(\mathbf{r}\cdot\mathbf{k})}\right|^{2} && (4.18)\\
&= \frac{1}{|\mathcal{R}|}\left|\sum_{\mathbf{r}\in\mathcal{R}}\left(\mathbf{A}_{\psi}\mathbf{X}^{t}\right)_{\mathbf{r}}e^{2\pi i(\mathbf{r}\cdot\mathbf{k})}\right|^{2}+\frac{1}{|\mathcal{R}|}\left|\sum_{\mathbf{r}\in\mathcal{R}}\left(\mathbf{E}^{t}\right)_{\mathbf{r}}e^{2\pi i(\mathbf{r}\cdot\mathbf{k})}\right|^{2} && (4.19)\\
&+ \frac{2}{|\mathcal{R}|}\left(\sum_{\mathbf{r}\in\mathcal{R}}\left(\mathbf{A}_{\psi}\mathbf{X}^{t}\right)_{\mathbf{r}}e^{2\pi i(\mathbf{r}\cdot\mathbf{k})}\right)\left(\sum_{\mathbf{u}\in\mathcal{R}}\left(\mathbf{E}^{t}\right)_{\mathbf{u}}e^{2\pi i(\mathbf{u}\cdot\mathbf{k})}\right) && (4.20)\\
&\approx \frac{1}{|\mathcal{R}|}\left|\sum_{\mathbf{r}\in\mathcal{R}}\left(\mathbf{A}_{\psi}\mathbf{X}^{t}\right)_{\mathbf{r}}e^{2\pi i(\mathbf{r}\cdot\mathbf{k})}\right|^{2}+\sigma^{2} && (4.21)
\end{aligned}$$

Where in the last line we used the fact that $\mathbf{E}^{t}$ is a white noise process, which is uncorrelated from the spiking process: therefore the second term can be approximated as the constant

$|\mathcal{R}|\sigma^2 \approx \|\mathbf{E}^t\|^2$, and the last term (the interaction term) can be neglected. The first term represent the power spectral density of the region of interest shapes scaled by calcium activity, which is small in high frequencies. Empirically, we notice that this high frequency range covers most of the indices in $\mathcal{R}$. Therefore, a simple estimator of $\|\mathbf{E}^t\|^2$ would be the median of $I_{\mathbf{Y}^t}(\mathbf{k})$. And an estimate of σ^2 would be $\hat{\sigma^2} = \frac{1}{T}\sum_t \text{median}\left[I_{\mathbf{Y}^t}(\mathbf{k})\right]$.

To summarize, we aim to reach $\lambda = \lambda^*$ (from Eqn. 4.16). If we solved Eqn. 4.16 by a homotopy or active set method such as LARS that constructs the full regularization path, we could simply run such a method until the desired residual variance is reached and stop. However, for the mixed ℓ_1/ℓ_2 norm, no exact homotopy method exists (though approximate homotopy methods do exist (Bach et al., 2004), as well as exact homotopy methods for the mixed ℓ_1/ℓ_∞ norm (Zhao et al., 2009)). Thankfully, in practice we find good performance by a much simpler method: we adapt λ during FISTA using a gradient descent rule interleaved with the FISTA updates: $\lambda_{k+1} = \lambda_k + \eta\left(\|\mathbf{A}_\psi \mathbf{X}_k(\lambda_k) - \mathbf{Y}\|^2 - |\mathcal{R}|T\hat{\sigma^2}\right)$ where η is some learning parameter, tuned so we obtain convergence. Note that this converge can occur only if the estimate $\hat{\sigma^2}$ is indeed the desired level of MSE. This could be wrong in two cases: (1) there is non-white noise (which is often the case, for instance noise due to scattering of the laser pencil as it passes through tissue in light-sheet microscopy) (2) $\mathbf{A}_\psi\mathbf{X}^t$ cannot actually match the shape of the signal at a reasonable sparsity level, for instance if ψ is a Gaussian filter which is too broad to be able to reconstruct neuronal shapes. Conversely, we would not want to make the template width too small, since the shape of a single neuron would then be fit to several disconnected components in $\mathbf{X}^t$.

4.3.3 Finding ROI Shapes with Alternating SVD

In the ideal case, the fluorescence activity of each neuron is just given by the appropriate row of $\mathbf{X}$. Many factors make this more complicated in practice. First, when the basis for the data is convolutional, sparse or group-sparse penalties do not give sharp solutions, meaning that the nonzero rows of $\mathbf{X}^*$ will be "smeared out" in relation to the true $\mathbf{X}$, even when the data is generated from the model. In addition, for real data neuron shapes are not exactly matched to the template. We can approximately find the center of a region of interest by taking the norm of each row of $\mathbf{X}^*$ to get an activity map, and taking the ROI centers to be located at regional maxima. The fluorescence activity of each ROI can then be taken to be the average of the rows of $\mathbf{X}^*$ within some distance of each peak. However, if we really want to accurately separate the signals from neighboring or overlapping neurons, we'd like a more detailed dictionary of neuron shapes, as we have with the greedy clustering method of Sec. 4.2. This may also be useful if we are interested in classifying different cell types based on morphology, among other things. We can use the ROI locations as a seed for a simple algorithm for finding the detailed shape of a neuron.

Suppose we are given a set of masks $\Omega_1, \ldots, \Omega_k$, where Ω_k is the support of the kth ROI. Then we would like to find a dictionary of neuron shapes $\mu^1, \ldots, \mu^K$ and fluorescence time courses $\mathbf{r}^1, \ldots, \mathbf{r}^K$ that capture as much variance of the data as possible given the constraint that μ^k is zero outside of Ω_k:

$$\min_{\mu^1,\ldots,\mu^K,\mathbf{r}^1,\ldots,\mathbf{r}^K} \left\| \mathbf{Y} - \sum_{k=1}^{K} \mu^k \mathbf{r}^k \right\|_F^2 \qquad \text{given} \qquad \mathcal{P}_{\Omega_k}^{\perp}(\mu^k) = 0 \tag{4.22}$$

where $\mathcal{P}_{\Omega}^{\perp}$ is the projection onto all indices not in Ω.

Starting from initial estimates $\mu^1, \ldots, \mu^K \in \mathbb{R}^{|\mathcal{R}|}$ and $\mathbf{r}^1, \ldots, \mathbf{r}^K$ we can alternately minimize the objective for each k keeping all other parameters fixed. Separating k from the rest of the indices in Eqn. 4.22 gives

$$\min_{\mu^k,\mathbf{r}^k} \left\| \left(\mathbf{Y} - \sum_{k' \neq k} \mu^{k'} \mathbf{r}^{k'} \right) - \mu^k \mathbf{r}^k \right\|_F^2 \qquad \text{given} \qquad \mathcal{P}_{\Omega^k}^{\perp}(\mu^k) = 0 \tag{4.23}$$

or equivalently

$$\min_{\mu^k, \mathbf{r}^k} \left\| \mathcal{P}_{\Omega_k} \left(\mathbf{Y} - \sum_{k' \neq k} \mu^{k'} \mathbf{r}^{k'} \right) - \mathcal{P}_{\Omega_k} \left(\mu^k \right) \mathbf{r}^k \right\|_F^2 \tag{4.24}$$

which can be solved exactly by setting $\mathcal{P}_{\Omega_k}(\mu^k)$ and $\mathbf{r}^k$ to the first left and right singular vector of $\mathcal{P}_{\Omega_k}\left(\mathbf{Y} - \sum_{k' \neq k} \mu^{k'} \mathbf{r}^{k'}\right)$, with the appropriate scaling. Thus we can locally descend the objective function Eqn. 4.22 by alternately taking the top singular vector or principal component of the residual within each ROI conditioned on all other ROI shapes. This is very similar in spirit to the K-SVD algorithm for dictionary learning (Aharon et al., 2006), which learns dictionary elements by alternately taking the top singular vector over all data that currently have that dictionary element as active. In our case, the "active" elements are voxels in the support of the kth ROI, and do not change after the group lasso step.

One further extension that has proven useful is to add a nonnegativity constraint to μ_k and $\mathbf{r}_k$, since both the firing rate of a neuron and level of GCaMP expression must be positive. This also helps to force regions with no GCaMP expression or times with zero firing rate to zero. In this case Eqn. 4.24 is no longer solved by the top singular vectors, but by alternately solving a least squares problem with soft threshold:

$$\mathcal{P}_{\Omega_k}(\mu_k) \leftarrow \left[\mathcal{P}_{\Omega_k} \left(\frac{Y \mathbf{r}_k^T}{\|\mathbf{r}_k\|^2} \right) \right]_+ \tag{4.25}$$

$$\mathbf{r}_k \leftarrow \left[\frac{\mu_k^T Y}{\|\mu_k\|^2} \right]_+ \tag{4.26}$$

repeating these two steps until a desired convergence is reached. While slower than SVD, the reconstructed ROIs and time series do look significantly cleaner using this technique.

4.3.4 Results

We applied convolutional group lasso to data from larval zebrafish expressing nuclear-localized GCaMP. At regular intervals the fish was shown a drifting grating moving upward from the fish's point of view. On some trials this triggers the optomotor response, a reflex in which the fish orients itself in parallel with the direction of visual motion. We chose this data over the spontaneous data in the previous section for two reasons. First, neurons expressing nuclear-localized GCaMP are much better approximated by a Gaussian template, and the lack of dendritic or axonal processes means a single template does a better job of capturing the range of morphologies. Secondly, the addition of stimulus and behavior means a much richer set of questions can be asked about the data once we have regions of interest in hand. We did not apply alternating SVD, as we generally found the regions of interest were well-separated enough to get a good estimate of the fluorescence activity for each cell body. In total we found 31345 regions of interest from this dataset. From visual inspection, a large fraction correspond to cell bodies from the most active neurons (Fig. 4.11), and the ROIs had a consistent shape for all but the thousand or so least active (Fig. 4.12).

The most prominent feature of the fluorescence activity from this data is that the majority of ROIs were significantly correlated with behavior. In this case the behavior consists of the total tail deflection, which was triggered on some presentations of the drifting grating but not others. Just taking the instantaneous correlation between behavior and fluorescence, 22112 of 31345 ROIs had significant ($p<1e\text{-}6$) correlation with behavior. The correlated ROIs were distributed almost uniformly through the brain, except in the forebrain, where correlation declined moving in the anterior direction, until the correlation was no longer significant (Fig. 4.15). Among ROIs correlated with behavior, a large number had *negative* correlation with behavior, and these ROIs were mostly located in two anatomical regions: two small regions bilaterally in the posterior hindbrain, and two large regions bilaterally at the lateral anterior hindbrain.

Looking at the fluorescence activity from different ROIs reveals a rich diversity of dynamics. The ROIs most strongly correlated with behavior were very well approximated by filtered versions of the behavior itself (Fig. 4.14(b)), so much so that over 90% of the variance in fluorescence is accounted for by a linear model driven by behavior. Behavioral activity was very sparse (Fig. 4.13(a)), and ROI negatively correlated with behavior were almost always silent during behavior. In particular the ROI in the medial posterior region of the hindbrain displayed "ramping" activity, where they were silent following a bout of activity, ramped up to a constant level of activity afterward and then immediately silenced at the start of behavior (Fig. 4.14(d)). ROIs uncorrelated with behavior naturally showed a variety of patterns, but the most active ROIs uncorrelated with behavior, particularly in the forebrain, generally showed long time scale persistent activity not clearly initiated by anything behavioral (Fig. 4.14(f)). There are also ROIs correlated with the stimulus, though the neural response is typically more filtered, so instantaneous correlation is not as good a measure of how "sensory" a neuron is (Fig. 4.14(h)).

Having reduced the data down from hundreds of millions of voxels to tens of thousand of ROIs makes it possible to try analyses that would not scale to raw data. We applied t-distributed stochastic neighbors embedding (t-SNE) Van der Maaten and Hinton (2008), a nonlinear dimensionality reduction technique, to the top 5000 ROIs. t-SNE, like many nonlinear dimensionality reduction techniques, finds a low-dimensional embedding of high-dimensional data that preserves pairwise distances. Because it builds off of pairwise information between time series, it scales at least as $\mathcal{O}(N^2)$ where N is the number of data, and thus can't be practically applied to raw voxels, at least without significant approximation. t-SNE is particularly well suited to data that is naturally structured into clusters, but still has nonlinear low-dimensional structure within the clusters.

Because so many ROI were strongly correlated with behavior, we deliberately subtracted off any correlations with behavior before doing dimensionality reduction. In particular, we projected the data orthogonally to many time-shifted versions of the data, as the fluorescence activity was often somewhat lagging behavior due to filtering of the calcium signal. As a control, we applied PCA to the same data (Fig. 4.16). Projecting ROIs onto the top 3 principal components, there was no apparent clustering of ROIs into different functional groups. To make comparison to anatomy easier, we also visualized ROIs by location in the fish and colored each ROI based on its position in embedding space. ROIs were roughly evenly distributed through the embedding space, with the most noticeable feature being some separation along the anterior-to-posterior axis, likely reflecting the fact that neurons in the forebrain have qualitatively different firing patterns than neurons in the hindbrain and midbrain.

t-SNE, by contrast, grouped ROIs into a number of clusters, making it significantly easier to distinguish functionally distinct regions (Fig. 4.17). ROIs clumped together throughout the embedding space, and when used to label ROI in anatomical space, several features are easy to spot. For instance, a large region of the left medial midbrain clearly stands out in orange. Looking at the fluorescence activity of a number of neurons in that group reveals they are highly correlated, and not closely locked to behavior (Fig. 4.18). Regions in the left and right lateral forebrain can clearly be seen to be highly correlated, as they both stand out in light blue. While drawing conclusions from a single dataset is difficult, and clearly these analyses would need to be repeated on data from other animals and experiments under similar conditions, this does show the feasibility of extracting interesting information using more complicated analyses that can only be applied to data at the scale of ROIs.

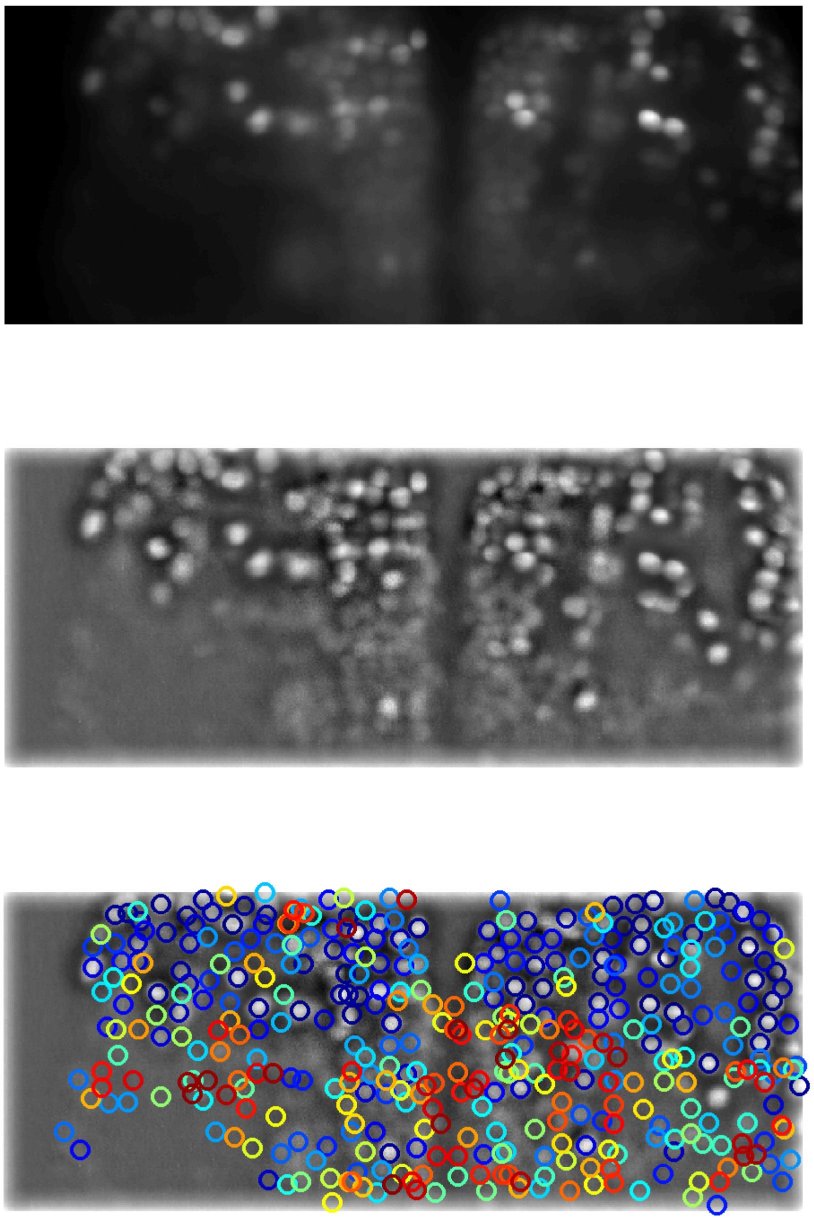


Figure 4.11: Top: 90th percentile image of fluorescence activity in the zebrafish forebrain. Middle: contrast normalized image. Bottom: Regions of interest found by convolutional group lasso superimposed on contrast normalized image. Color indicates magnitude of fluorescence activity, blue is high activity and red is low activity.

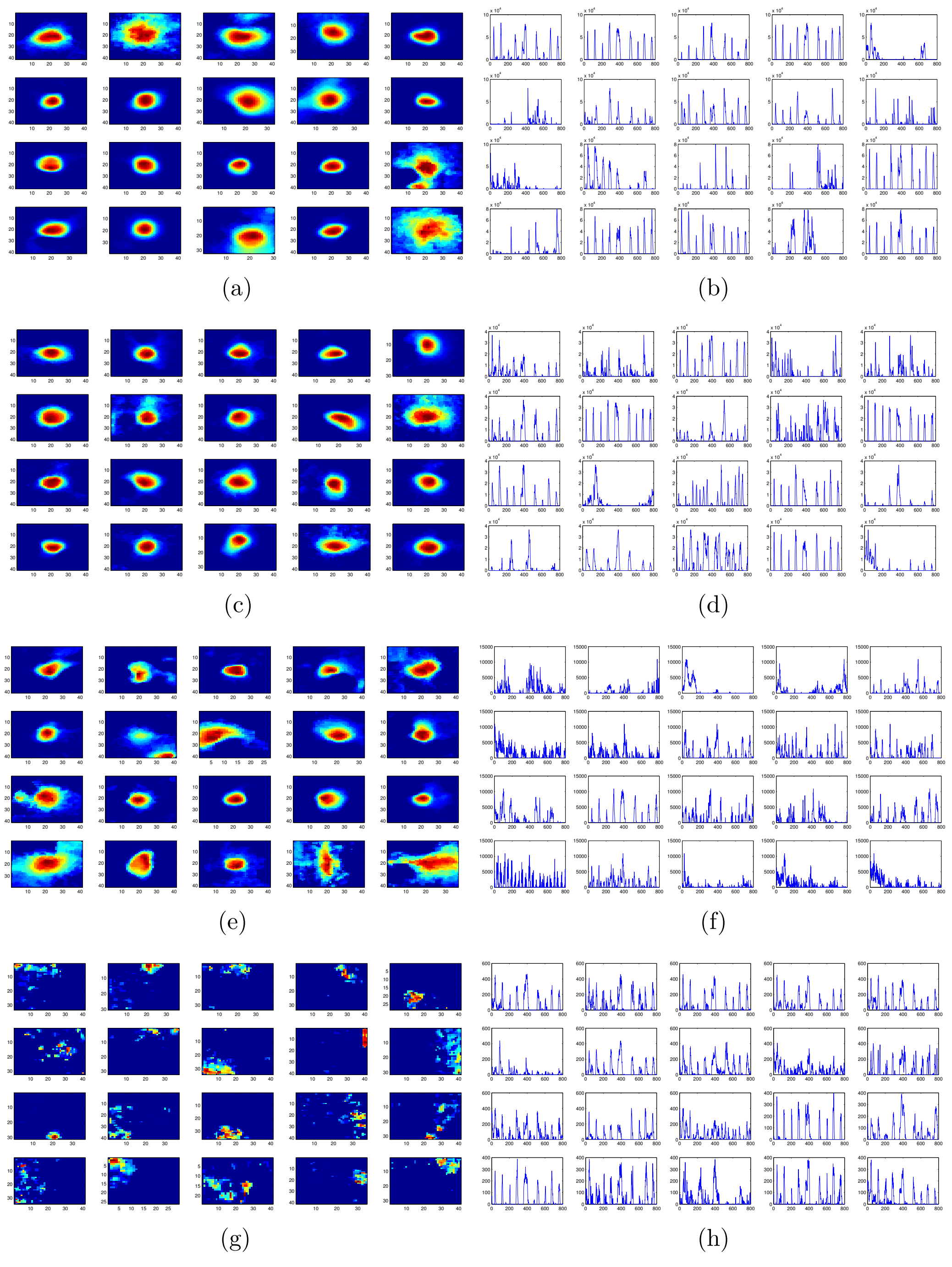
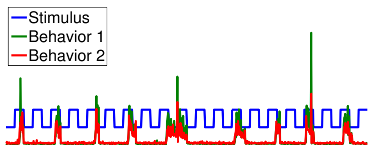
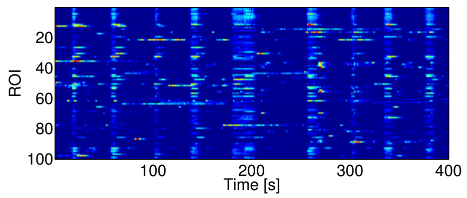


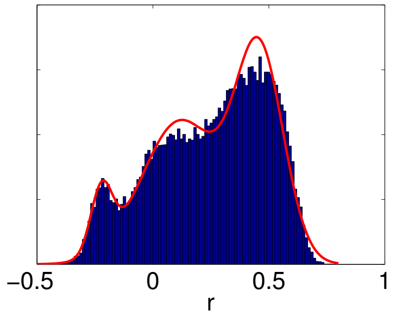
Figure 4.12: Shapes and fluorescence activity of regions of interest arranged by magnitude of activity. ROIs 101-120 (a),(b), 1001-1020 (c),(d), 10001-10020 (e),(f), 31001-31020 (g),(h). Only near the very end does the quality of ROI degrade significantly.



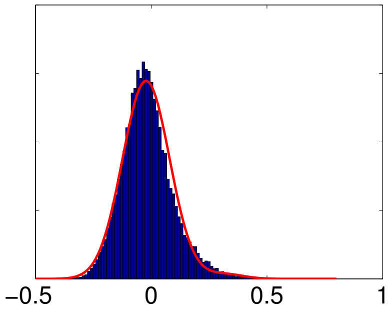
(a)



(b)

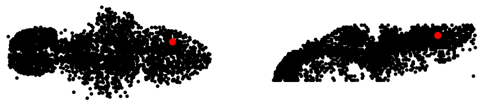


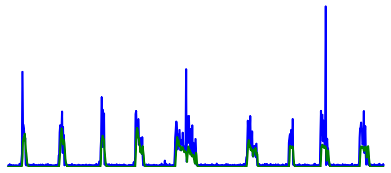
(c)

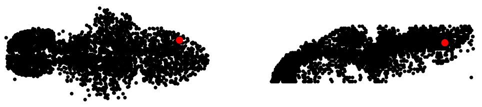


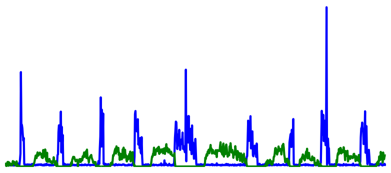
(d)

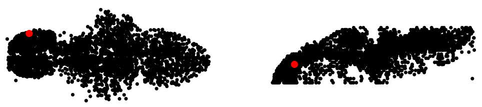
Figure 4.13: Diversity of neural responses in the larval zebrafish. (a) Time course of drifting grating stimulus (blue) and two measures of tail deflection (red and green). (b) Fluorescence activity of the 100 most active ROIs, showing the mixture of behavior-correlated and uncorrelated ROIs. (c) Histogram of correlations between fluorescence and behavior, with gaussian mixture model fit on top. The data is clearly trimodal. (d) histogram of correlation with stimulus. The lack of strongly correlated ROIs is likely due to the filtered nature of the responses.

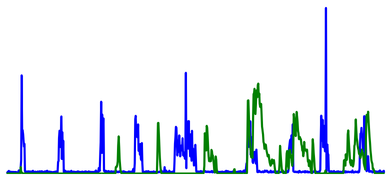

(a)

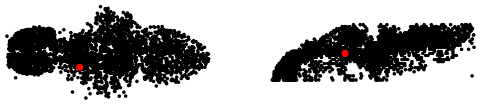

(b)


(c)


(d)


(e)


(f)


(g)

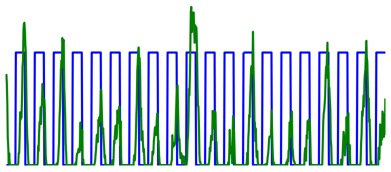

(h)

Figure 4.14: A selection of neural activity. Location of ROI on the left, fluorescence (green) superimposed over behavior or stimulus (blue). (a),(b) The ROI most correlated with behavior. (c),(d) The ROI most negatively correlated with behavior. Note that activity is suppressed even after the end of behavior. (e),(f) An ROI not significantly correlated with behavior. A large fraction of these are located in the forebrain. (g),(h) An ROI in the optic tectum correlated with the stimulus. Note the response is heavily temporally filtered.

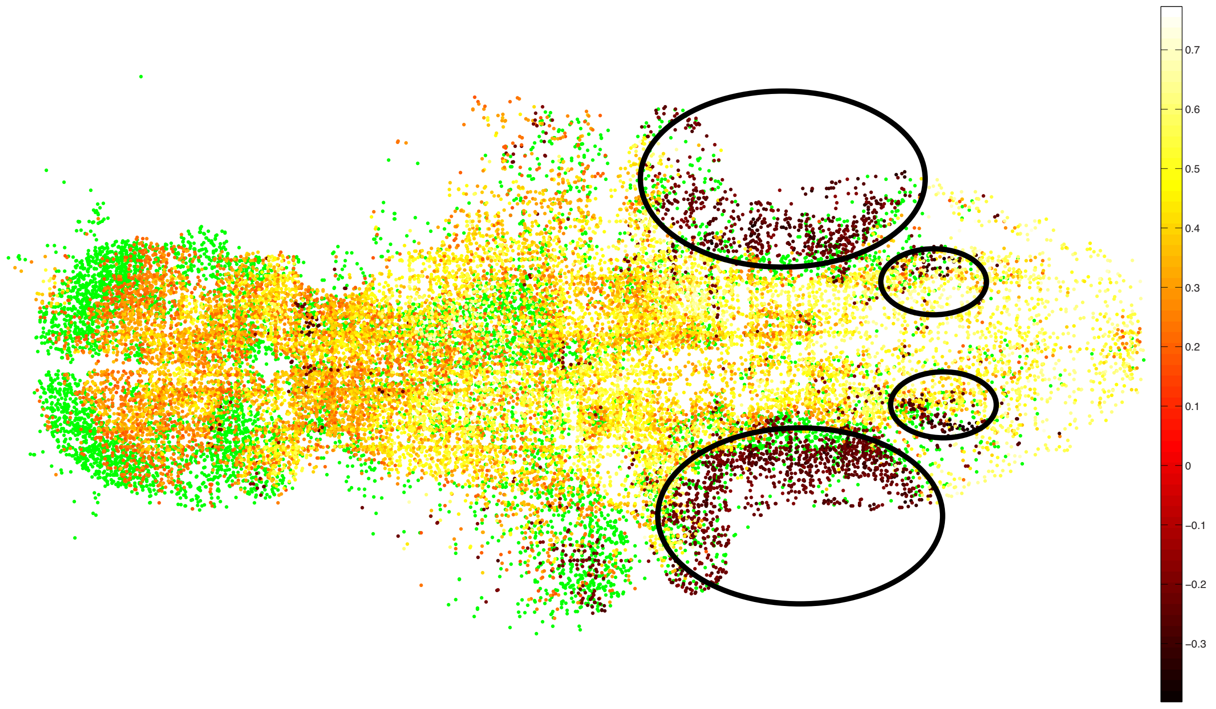


Figure 4.15: Transverse view of the ROIs in the larval zebrafish brain found by convolutional group lasso. ROIs not significantly correlated with behavior (p>1e-6) are in green, while those significantly correlated with behavior are colored according to the instantaneous correlation coefficient between behavior and fluorescence activity. Anatomical regions negatively correlated with behavior are circled.

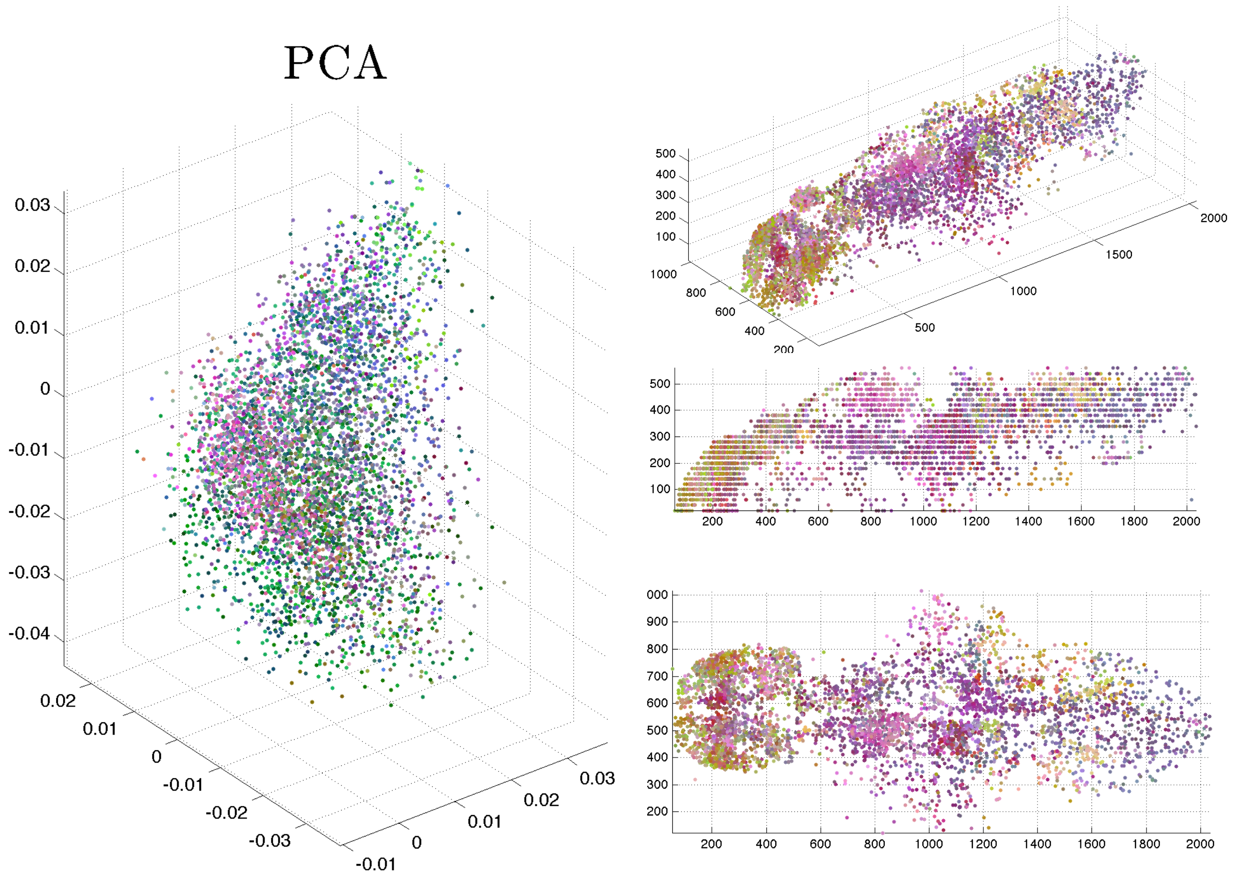


Figure 4.16: PCA applied to fluorescence activity of ROIs. Left: projection of ROIs into principal component space, colored by anatomical position (red: anterior-posterior. green: medial-lateral. blue: dorsal-ventral). Right: ROIs colored by position in principal component space.

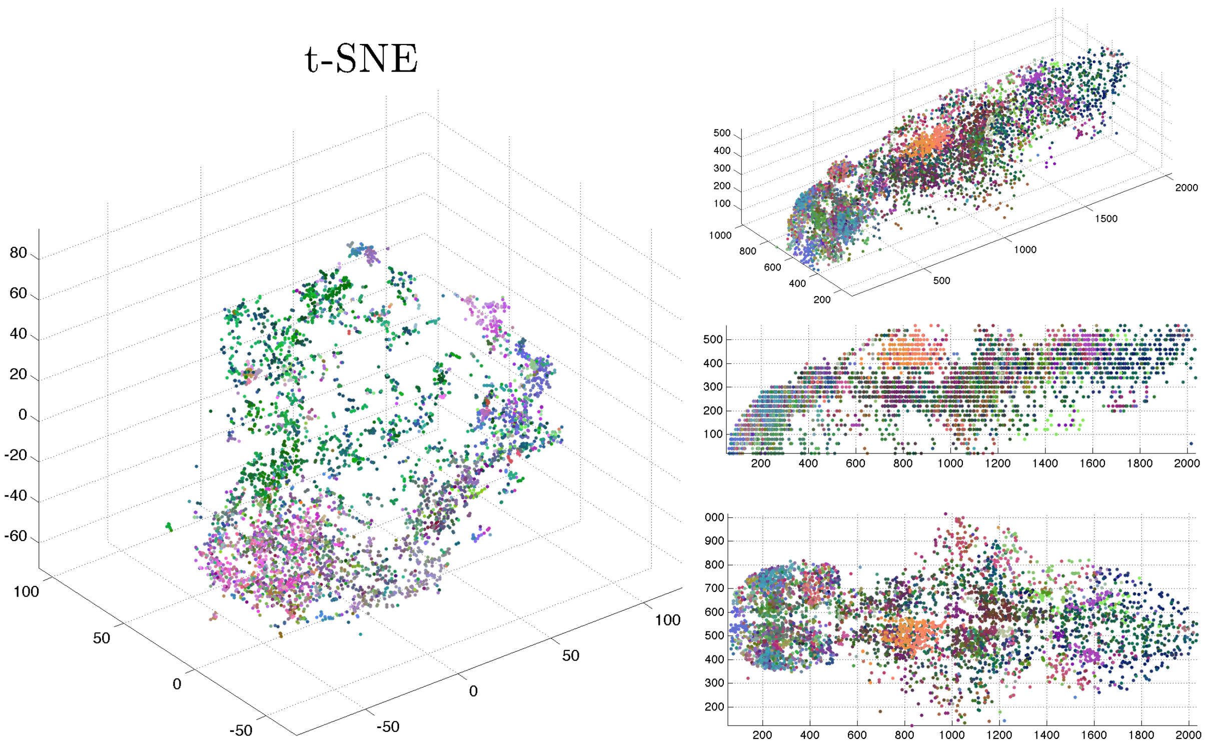


Figure 4.17: t-SNE applied to fluorescence activity of ROIs. Left: projection of ROIs into principal component space, colored by anatomical position (red: anterior-posterior. green: medial-lateral. blue: dorsal-ventral). Right: ROIs colored by position in embedding space.

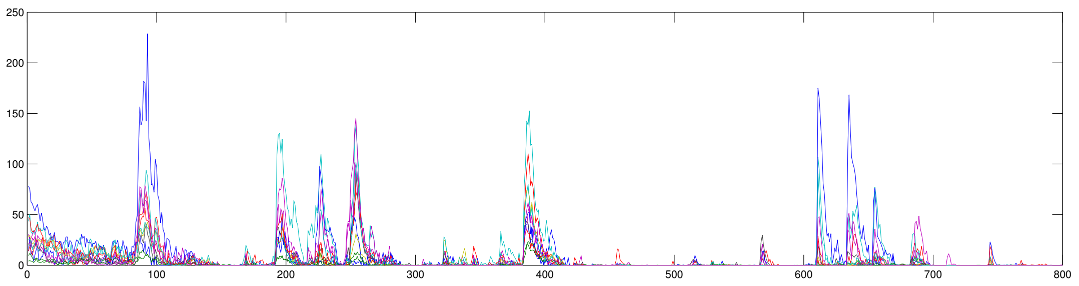


Figure 4.18: Fluorescence traces from ROI in the medial midbrain patch in Fig. 4.17

Chapter 5

Conclusions

If an eternal traveler should journey in any direction,

he would find after untold centuries that the same volumes

are repeated in the same disorder

– which, repeated, becomes order

Jorge Luis Borges, *The Library of Babel*

It is said that for every question answered in science, ten new questions are introduced. It certainly feels that way at the completion of this thesis. Is it really possible - or desirable - to create a truly “black box” model for sequential information? Can we make models that are scalable, robust and also fit nonlinear dynamics? Even if we can accurately predict the dynamics of a system, what have we actually learned about how it works? And how can we bridge the gap between rich, powerful statistical models of systems and the current best methods we have for measuring said systems?

Many of the questions raised by this thesis are paralleled in the ongoing debate happening now over “big data” outside the walls of academia. On many real-world machine learning challenges, state-of-the-art prediction can be achieved with models that have no domain knowledge built into them - in fact often models perform better without domain knowledge. This situation is not new in machine learning: Frederick Jelinek, a pioneer of natural language processing, once famously remarked that “every time a linguist leaves my group, the recognition rate goes up.” This scenario is in many ways deeply troubling - if we can learn models directly from data that are able to predict just as well as models with detailed biophysical knowledge, did we really need the detailed biophysical knowledge in the first place?

We are not quite so radical as to suggest that knowledge of mechanism is unnecessary. While we have shown that it is possible to learn good models of how a neuron in culture will respond to prolonged stimulation, nothing about that model tells you how it will change if one performs some novel manipulation, say, changing the concentration of some ion in the medium. Only an understanding of the underlying causes for some phenomenon can achieve that. And prediction alone is not the only use of a model. Inferring known latent structure is often just as important, as in most decoding applications, where domain knowledge does have to be built in. And to actually make a novel scientific discovery, one must already have a good grasp of what is or is not known about a system, so no black-box algorithm could say how unexpected some particular dynamics are. Actually extracting scientific insight from data will continue to be a job for the creative human mind long after most other analytic tasks have been automated.

There is little doubt that we are moving into an age of “big data” neuroscience. While many of the advances brought by this will be thanks to existing approaches scaled up - compute all the receptive fields! - there are surely qualitative changes in the kinds of insights we can glean from data at this scale. There remains much to be done to develop the technology to make these insights possible, and further refine these technologies to make heroically difficult tasks into everyday lab work. We may soon be in the envious position of being able to look in to a brain with perfect electrical knowledge. Then the real fun begins.

Bibliography

Aharon, M., M. Elad, and A. Bruckstein (2006). K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on 54*(11), 4311–4322.

Ahrens, M. B., M. B. Orger, D. N. Robson, J. M. Li, and P. J. Keller (2013). Whole-brain functional imaging at cellular resolution using light-sheet microscopy. *Nature methods*.

Akemann, W., M. Sasaki, H. Mutoh, T. Imamura, N. Honkura, and T. Knöpfel (2013). Two-photon voltage imaging using a genetically encoded voltage indicator. *Scientific reports 3*.

Archer, E., I. M. Park, and J. W. Pillow (2012). Bayesian estimation of discrete entropy with mixtures of stick-breaking priors. In *Advances in Neural Information Processing Systems*, pp. 2024–2032.

Archer, E. W., I. M. Park, and J. W. Pillow (2013). Bayesian entropy estimation for binary spike train data using parametric prior knowledge. In *Advances in Neural Information Processing Systems*, pp. 1700–1708.

Atick, J. J. (1992). Could information theory provide an ecological theory of sensory processing? *Network: Computation in neural systems 3*(2), 213–251.

Atick, J. J. and A. N. Redlich (1990). Towards a theory of early visual processing. *Neural Computation 2*(3), 308–320.

Atick, J. J. and A. N. Redlich (1992). What does the retina know about natural scenes? *Neural computation 4*(2), 196–210.

Attiya, H. and J. Welch (2004). *Distributed computing: fundamentals, simulations, and advanced topics*, Volume 19. John Wiley & Sons.

Bach, F. R., R. Thibaux, and M. I. Jordan (2004). Computing regularization paths for learning multiple kernels. In *NIPS*.

Bailly, R. (2011). Quadratic weighted automata: Spectral algorithm and likelihood maximization. *Journal of Machine Learning Research 20*, 147–162.

Baker, B. J., E. K. Kosmidis, D. Vucinic, C. X. Falk, L. B. Cohen, M. Djurisic, and D. Zecevic (2005). Imaging brain activity with voltage-and calcium-sensitive dyes. *Cellular and molecular neurobiology 25*(2), 245–282.

Balle, B., X. Carreras, F. M. Luque, and A. Quattoni (2013). Spectral learning of weighted automata. *Machine Learning*, 1–31.

Balle, B. and M. Mohri (2012). Spectral learning of general weighted automata via constrained matrix completion. In *NIPS*, pp. 2168–2176.

Balle, B., A. Quattoni, and X. Carreras (2011). A spectral learning algorithm for finite state transducers. In *Machine Learning and Knowledge Discovery in Databases*, pp. 156–171. Springer.

Banerjee, A., S. Merugu, I. S. Dhillon, and J. Ghosh (2005). Clustering with Bregman divergences. *The Journal of Machine Learning Research 6*, 1705–1749.

Barlow, H. B. (1961). Possible principles underlying the transformation of sensory messages. *Sensory communication*, 217–234.

Barnett, L., J. Platisa, M. Popovic, V. A. Pieribone, and T. Hughes (2012). A fluorescent, genetically-encoded voltage probe capable of resolving action potentials. *PloS one 7*(9), e43454.

Beck, A. and M. Teboulle (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences 2*(1), 183–202.

Belkin, M. and P. Niyogi (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation 15*(6), 1373–1396.

Bialek, W., I. Nemenman, and N. Tishby (2001). Predictability, complexity, and learning. *Neural computation 13*(11), 2409–2463.

Bialek, W., F. Rieke, R. d. R. Van Steveninck, and D. Warland (1991). Reading a neural code. *Science 252*(5014), 1854–1857.

Borodin, A. (2009). Determinantal point processes. *arXiv preprint arXiv:0911.1153*.

Boyd, S. P., N. Parikh, E. Chu, B. Peleato, and J. Eckstein (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning 3*(1), 1–122.

Bregman, L. M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics 7*(3), 200–217.

Brendel, W., R. Romo, and C. K. Machens (2011). Demixed principal component analysis. *Advances in Neural Information Processing Systems 24*, 1–9.

Briggman, K. L., H. D. I. Abarbanel, and W. B. Kristan (2005). Optical imaging of neuronal populations during decision-making. *Science 307*(5711), 896–901.

Broxton, M., L. Grosenick, S. Yang, N. Cohen, A. Andalman, K. Deisseroth, and M. Levoy (2013). Wave optics theory and 3-d deconvolution for the light field microscope. *Optics express 21*(21), 25418–25439.

Buesing, L., J. Macke, and M. Sahani (2012). Spectral learning of linear dynamics from generalised-linear observations with application to neural population data. *Advances in neural information processing systems 25*.

Candès, E. J. and B. Recht (2009). Exact matrix completion via convex optimization. *Foundations of Computational mathematics 9*(6), 717–772.

Cao, G., J. Platisa, V. A. Pieribone, D. Raccuglia, M. Kunst, and M. N. Nitabach (2013). Genetically targeted optical electrophysiology in intact neural circuits. *Cell 154*(4), 904–913.

Carrasco, R. and J. Oncina (1994). Learning stochastic regular grammars by means of a state merging method. *Grammatical Inference and Applications*, 139–152.

Carroll, L. (1865). *Alice's Adventures in Wonderland.* Macmillan.

Castro, J. and R. Gavaldà (2008). Towards feasible pac-learning of probabilistic deterministic finite automata. In *Grammatical Inference: Algorithms and Applications*, pp. 163–174. Springer.

Chao, A. and T.-J. Shen (2003). Nonparametric estimation of shannon's index of diversity when there are unseen species in sample. *Environmental and ecological statistics 10*(4), 429–443.

Chornoboy, E., L. Schramm, and A. Karr (1988). Maximum likelihood identification of neural point process systems. *Biological cybernetics 59*(4-5), 265–275.

Churchland, M. M., J. P. Cunningham, M. T. Kaufman, J. D. Foster, P. Nuyujukian, S. I. Ryu, and K. V. Shenoy (2012). Neural population dynamics during reaching. *Nature.*

Clark, A. and F. Thollard (2004). PAC-learnability of probabilistic deterministic finite state automata. *The Journal of Machine Learning Research 5*, 473–497.

Cohen, S. B., K. Stratos, M. Collins, D. P. Foster, and L. Ungar (2012). Spectral learning of latent-variable pcfgs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pp. 223–231. Association for Computational Linguistics.

Collins, M., S. Dasgupta, and R. E. Schapire (2001). A generalization of principal component analysis to the exponential family. *Advances in neural information processing systems 14*.

Cover, T. and J. Thomas (1991). *Elements of information theory*. New York: Wiley.

Dean, J. and S. Ghemawat (2008). Mapreduce: simplified data processing on large clusters. *Communications of the ACM 51*(1), 107–113.

Denk, W., J. H. Strickler, and W. W. Webb (1990). Two-photon laser scanning fluorescence microscopy. *Science 248*(4951), 73–76.

Diego, F. and F. A. Hamprecht (2013). Learning multi-level sparse representations. In *Advances in Neural Information Processing Systems*, pp. 818–826.

Diego, F., S. Reichinnek, M. Both, and F. A. Hamprecht (2013). Automated identification of neuronal activity from calcium imaging by sparse dictionary learning. In *Biomedical Imaging (ISBI), 2013 IEEE 10th International Symposium on*, pp. 1058–1061. IEEE.

Dombeck, D. A., A. N. Khabbaz, F. Collman, T. L. Adelman, and D. W. Tank (2007). Imaging large-scale neural activity with cellular resolution in awake, mobile mice. *Neuron 56*(1), 43–57.

Donoho, D. L. and C. Grimes (2003). Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences 100*(10), 5591–5596.

Donoho, D. L., A. Maleki, and A. Montanari (2009). Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences 106*(45), 18914–18919.

Dorostkar, M. M., E. Dreosti, B. Odermatt, and L. Lagnado (2010). Computational processing of optical measurements of neuronal and synaptic activity in networks. *Journal of neuroscience methods 188*(1), 141–150.

Dupont, P., F. Denis, and Y. Esposito (2005). Links between probabilistic automata and hidden Markov models: probability distributions, learning models and induction algorithms. *Pattern recognition 38*(9), 1349–1371.

Eckart, C. and G. Young (1936). The approximation of one matrix by another of lower rank. *Psychometrika 1*(3), 211–218.

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

Fazel, M., H. Hindi, and S. P. Boyd (2001). A rank minimization heuristic with application to minimum order system approximation. *Proceedings of the American Control Conference. 6*, 4734–4739.

Feldman, J. and J. Hanna (1966). The structure of responses to a sequence of binary events. *Journal of Mathematical Psychology 3*(2), 371–387.

Freeman, J., N. Vladimirov, T. Kawashima, Y. Mu, N. J. Sofroniew, D. V. Bennett, J. Rosen, C.-T. Yang, L. L. Looger, and M. B. Ahrens (2014). Mapping brain activity at scale with cluster computing. *Nature methods 11*(9), 941–950.

Gal, A., D. Eytan, A. Wallach, M. Sandler, J. Schiller, and S. Marom (2010). Dynamics of excitability over extended timescales in cultured cortical neurons. *The Journal of Neuroscience 30*(48), 16332–16342.

Ganmor, E., R. Segev, and E. Schneidman (2011). Sparse low-order interaction network underlies a highly correlated and learnable neural population code. *Proceedings of the National Academy of Sciences 108*(23), 9679–9684.

Gasthaus, J., F. Wood, and Y. W. Teh (2010). Lossless compression based on the Sequence Memoizer. In *Data Compression Conference 2010*, pp. 337–345.

Gelman, A., J. B. Carlin, H. S. Stern, and D. B. Rubin (1995). *Bayesian data analysis.* New York: Chapman & Hall.

Göbel, W., B. M. Kampa, and F. Helmchen (2007). Imaging cellular network dynamics in three dimensions using fast 3d laser scanning. *Nature methods 4*(1), 73–79.

Gong, Y., J. Z. Li, and M. J. Schnitzer (2013). Enhanced archaerhodopsin fluorescent protein voltage indicators. *PloS one 8*(6), e66959.

Grassberger, P. (1989). Estimating the information content of symbol sequences and efficient codes. *Information Theory, IEEE Transactions on 35*(3), 669–675.

Greenberg, D. S., A. R. Houweling, and J. N. Kerr (2008). Population imaging of ongoing neuronal activity in the visual cortex of awake rats. *Nature neuroscience 11*(7), 749–751.

Grewe, B. F., D. Langer, H. Kasper, B. M. Kampa, and F. Helmchen (2010). High-speed in vivo calcium imaging reveals neuronal network activity with near-millisecond precision. *Nature methods 7*(5), 399–405.

Harrison, M. (2012). Conditional inference for learning the network structure of cortical microcircuits. In *2012 Joint Statistical Meeting*, San Diego, CA.

Haslinger, R., K. L. Klinkner, and C. R. Shalizi (2010). The computational structure of spike trains. *Neural computation 22*(1), 121–157.

Hausser, J. and K. Strimmer (2009). Entropy inference and the james-stein estimator, with application to nonlinear gene association networks. *The Journal of Machine Learning Research 10*, 1469–1484.

Hermans, M. and B. Schrauwen (2013). Training and analysing deep recurrent neural networks. In *Advances in Neural Information Processing Systems*, pp. 190–198.

Hinton, G. and S. Roweis (2002). Stochastic neighbor embedding. In *NIPS*, Volume 2, pp. 833–840.

Holekamp, T. F., D. Turaga, and T. E. Holy (2008). Fast three-dimensional fluorescence imaging of activity in neural populations by objective-coupled planar illumination microscopy. *Neuron 57*(5), 661–672.

Hsu, D., S. M. Kakade, and T. Zhang (2012). A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences 78*(5), 1460–1480.

Huang, J. and T. Zhang (2010). The benefit of group sparsity. *The Annals of Statistics 38*(4), 1978–2004.

Ising, E. (1925). Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik A Hadrons and Nuclei 31*(1), 253–258.

Junek, S., T.-W. Chen, M. Alevra, and D. Schild (2009). Activity correlation imaging: visualizing function and structure of neuronal populations. *Biophysical journal 96*(9), 3801–3809.

Kerr, J. N., D. Greenberg, and F. Helmchen (2005). Imaging input and output of neocortical networks in vivo. *Proceedings of the National Academy of Sciences of the United States of America 102*(39), 14063–14068.

Koyama, S., L. Castellanos Pérez-Bolde, C. R. Shalizi, and R. E. Kass (2010). Approximate methods for state-space models. *Journal of the American Statistical Association 105*(489), 170–180.

Kralj, J. M., A. D. Douglass, D. R. Hochbaum, D. Maclaurin, and A. E. Cohen (2012). Optical recording of action potentials in mammalian neurons using a microbial rhodopsin. *Nature methods 9*(1), 90–95.

Kruskal, J. B. and M. Wish (1978). *Multidimensional scaling*, Volume 11. Sage.

Kulkarni, J. E. and L. Paninski (2007). Common-input models for multiple neural spike-train data. *Network: Computation in Neural Systems 18*(4), 375–407.

Laughlin, S. B. et al. (1981). A simple coding procedure enhances a neuron's information capacity. *Z. Naturforsch 36*(910-912), 51.

Lawhern, V., W. Wu, N. Hatsopoulos, and L. Paninski (2010). Population decoding of motor cortical activity using a generalized linear model with hidden states. *Journal of neuroscience methods 189*(2), 267–280.

Lee, J. A. and M. Verleysen (2007). *Nonlinear dimensionality reduction.* Springer.

Levoy, M., R. Ng, A. Adams, M. Footer, and M. Horowitz (2006). Light field microscopy. In *ACM Transactions on Graphics (TOG)*, Volume 25, pp. 924–934. ACM.

Liu, Z., A. Hansson, and L. Vandenberghe (2013). Nuclear norm system identification with missing inputs and outputs. *Systems & Control Letters 62*(8), 605–612.

Liu, Z. and L. Vandenberghe (2009). Interior-point method for nuclear norm approximation with application to system identification. *SIAM Journal on Matrix Analysis and Applications 31*, 1235–1256.

Ma, S.-k. (1981). Calculation of entropy from data of motion. *Journal of Statistical Physics 26*(2), 221–240.

Machens, C. K., R. Romo, and C. D. Brody (2010). Functional, but not anatomical, separation of "what" and "when" in prefrontal cortex. *The Journal of Neuroscience 30*(1), 350–360.

MacKay, D. J. C. and L. B. Peto (1995). A hierarchical Dirichlet language model. *Natural language engineering 1*(2), 289–307.

Macke, J., J. Cunningham, M. Byron, K. Shenoy, and M. Sahani (2011). Empirical models of spiking in neural populations. *Advances in neural information processing systems 24.*

Mallat, S. G. and Z. Zhang (1993). Matching pursuits with time-frequency dictionaries. *Signal Processing, IEEE Transactions on 41*(12), 3397–3415.

Maruyama, R., K. Maeda, H. Moroda, I. Kato, M. Inoue, H. Miyakawa, and T. Aonishi (2014). Detecting cells using non-negative matrix factorization on calcium imaging data. *Neural Networks 55*, 11–19.

Miri, A., K. Daie, R. D. Burdine, E. Aksay, and D. W. Tank (2011). Regression-based identification of behavior-encoding. *J Neurophysiol 105*, 964–980.

Mukamel, E. A., A. Nimmerjahn, and M. J. Schnitzer (2009). Automated analysis of cellular signals from large-scale calcium imaging data. *Neuron 63*(6), 747–760.

Murphy, K. (2005). Hidden Markov model (HMM) toolbox for Matlab.

Nemenman, I., F. Shafee, and W. Bialek (2002). Entropy and inference, revisited. *Advances in neural information processing systems 1*, 471–478.

Niell, C. M. and S. J. Smith (2005). Functional imaging reveals rapid development of visual response properties in the zebrafish tectum. *Neuron 45*(6), 941–951.

Ohki, K., S. Chung, Y. H. Ch'ng, P. Kara, and R. C. Reid (2005). Functional imaging with cellular resolution reveals precise micro-architecture in visual cortex. *Nature 433*(7026), 597–603.

Okun, M., P. Yger, S. L. Marguet, F. Gerard-Mercier, A. Benucci, S. Katzner, L. Busse, M. Carandini, and K. D. Harris (2012). Population rate dynamics and multineuron firing patterns in sensory cortex. *The Journal of Neuroscience 32*(48), 17108–17119.

Oñativia, J., S. R. Schultz, and P. L. Dragotti (2013). A finite rate of innovation algorithm for fast and accurate spike detection from two-photon calcium imaging. *Journal of neural engineering 10*(4), 046017.

Ozden, I., H. M. Lee, M. R. Sullivan, and S. S.-H. Wang (2008). Identification and clustering of event patterns from in vivo multiphoton optical recordings of neuronal ensembles. *Journal of neurophysiology 100*(1), 495.

Pachitariu, M., A. M. Packer, N. Pettit, H. Dalgleish, M. Hausser, and M. Sahani (2013). Extracting regions of interest from biological images with convolutional sparse block coding. In *Advances in Neural Information Processing Systems*, pp. 1745–1753.

Paninski, L. (2003). Estimation of entropy and mutual information. *Neural Computation 15*(6), 1191–1253.

Paninski, L., Y. Ahmadian, D. G. Ferreira, S. Koyama, K. R. Rad, M. Vidne, J. Vogelstein, and W. Wu (2010). A new look at state-space models for neural data. *Journal of Computational Neuroscience 29*(1-2), 107–126.

Paninski, L., J. Pillow, and E. Simoncelli (2004). Maximum likelihood estimation of a stochastic integrate-and-fire neural encoding model. *Neural computation 16*(12), 2533–2561.

Panzeri, S. and A. Treves (1995). Analytical estimates of limited sampling biases in different information measures.

Petreska, B., B. M. Yu, J. P. Cunningham, G. Santhanam, S. I. Ryu, K. V. Shenoy, and M. Sahani (2011). Dynamical segmentation of single trials from population neural data. *Advances in neural information processing systems 24*.

Pillow, J. W., J. Shlens, L. Paninski, A. Sher, A. M. Litke, E. Chichilnisky, and E. P. Simoncelli (2008). Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature 454*(7207), 995–999.

Pnevmatikakis, E., T. Machado, L. Grosenick, B. Poole, J. Vogelstein, and L. Paninski (2013). Rank-penalized nonnegative spatiotemporal deconvolution and demixing of calcium imaging data. In *Computational and Systems Neuroscience Meeting COSYNE*.

Portugues, R., C. E. Feierstein, F. Engert, and M. B. Orger (2014). Whole-brain activity maps reveal stereotyped, distributed networks for visuomotor behavior. *Neuron 81*(6), 1328–1343.

Prevedel, R., Y.-G. Yoon, M. Hoffmann, N. Pak, G. Wetzstein, S. Kato, T. Schrödel, R. Raskar, M. Zimmer, E. S. Boyden, et al. (2014). Simultaneous whole-animal 3d-imaging of neuronal activity using light field microscopy. *arXiv preprint arXiv:1401.5333*.

Rabin, M. (1963). Probabilistic automata. *Information and control 6*(3), 230–245.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE 77*, 257–286.

Reber, A. (1967). Implicit learning of artificial grammars. *Journal of verbal learning and verbal behavior 6*(6), 855–863.

Rieke, F. (1999). *Spikes: exploring the neural code*. MIT press.

Ron, D., Y. Singer, and N. Tishby (1996). The power of amnesia: Learning probabilistic automata with variable memory length. *Machine learning 25*(2), 117–149.

Roweis, S. T. and L. K. Saul (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science 290*(5500), 2323–2326.

Sasaki, T., N. Takahashi, N. Matsuki, and Y. Ikegaya (2008). Fast and accurate detection of action potentials from somatic calcium fluctuations. *Journal of neurophysiology 100*(3), 1668–1676.

Sato, T. R., N. W. Gray, Z. F. Mainen, and K. Svoboda (2007). The functional microarchitecture of the mouse barrel cortex. *PLoS biology 5*(7), e189.

Schneidman, E., M. J. Berry, R. Segev, and W. Bialek (2006). Weak pairwise correlations imply strongly correlated network states in a neural population. *Nature 440*(7087), 1007–1012.

Shalizi, C. and K. Shalizi (2004). Blind construction of optimal nonlinear recursive predictors for discrete sequences. In *Proceedings of the 20th conference on Uncertainty in Artificial Intelligence*, pp. 504–511. UAI Press.

Shalizi, C. R. and J. P. Crutchfield (2001). Computational mechanics: Pattern and prediction, structure and simplicity. *Journal of statistical physics 104*(3-4), 817–879.

Shannon, C. E. and W. Weaver (1949). The mathematical theory of information.

Shlens, J., G. D. Field, J. L. Gauthier, M. I. Grivich, D. Petrusca, A. Sher, A. M. Litke, and E. Chichilnisky (2006). The structure of multi-neuron firing patterns in primate retina. *The Journal of neuroscience 26*(32), 8254–8266.

Smetters, D., A. Majewska, and R. Yuste (1999). Detecting action potentials in neuronal populations with calcium imaging. *Methods 18*(2), 215–221.

Smith, A. and E. Brown (2003). Estimating a state-space model from point process observations. *Neural Computation 15*(5), 965–991.

Soelter, J., J. Schumacher, H. Spors, and M. Schmuker (2014). Automatic segmentation of odor maps in the mouse olfactory bulb using regularized non-negative matrix factorization. *NeuroImage*.

Sohl-Dickstein, J., P. B. Battaglino, and M. R. DeWeese (2011). New method for parameter estimation in probabilistic models: minimum probability flow. *Physical review letters 107*(22), 220601.

Solo, V. and S. A. Pasha (2013). Point-process principal components analysis via geometric optimization. *Neural Computation 25*(1), 101–122.

Soudry, D. and R. Meir (2014). The neuronal response at extended timescales: long-term correlations without long-term memory. *Frontiers in computational neuroscience 8*.

Stevenson, I. H. and K. P. Kording (2011). How advances in neural recording affect data analysis. *Nature neuroscience 14*(2), 139–142.

Stone, J., J. Porrill, N. Porter, and I. Wilkinson (2002). Spatiotemporal independent component analysis of event-related fmri data using skewed probability density functions. *NeuroImage 15*(2), 407–421.

Stopfer, M., V. Jayaraman, and G. Laurent (2003). Intensity versus identity coding in an olfactory system. *Neuron 39*(6), 991–1004.

Storace, D. A., U. Sung, J. Platisa, L. B. Cohen, and V. A. Pieribone (2013). In vivo imaging of odor-evoked responses in the olfactory bulb using arclight, a novel fp voltage probe. *Biophysical Journal 104*, 679.

Strong, S. P., R. Koberle, R. R. d. R. van Steveninck, and W. Bialek (1998). Entropy and information in neural spike trains. *Physical review letters 80*(1), 197.

Sun, X. R., A. Badura, D. A. Pacheco, L. A. Lynch, E. R. Schneider, M. P. Taylor, I. B. Hogue, L. W. Enquist, M. Murthy, and S. S.-H. Wang (2013). Fast gcamps for improved tracking of neuronal activity. *Nature communications 4*.

Sutskever, I., J. Martens, and G. E. Hinton (2011). Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 1017–1024.

Svoboda, K. and R. Yasuda (2006). Principles of two-photon excitation microscopy and its applications to neuroscience. *Neuron 50*(6), 823–839.

Teh, Y. W. (2006). A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the Association for Computational Linguistics*, pp. 985–992.

Teh, Y. W., M. I. Jordan, M. J. Beal, and D. M. Blei (2006a). Hierarchical Dirichlet Processes. *Journal of the American Statistical Association 101*(476), 1566–1581.

Teh, Y. W., M. I. Jordan, M. J. Beal, and D. M. Blei (2006b). Hierarchical Dirichlet processes. *Journal of the American Statistical Association 101*(476), 1566–1581.

Tenenbaum, J. B., V. De Silva, and J. C. Langford (2000). A global geometric framework for nonlinear dimensionality reduction. *Science 290*(5500), 2319–2323.

Thollard, F. (2001). Improving probabilistic grammatical inference core algorithms with post-processing techniques. In *Eighteenth International Conference on Machine Learning*, pp. 561–568.

Tipping, M. E. and C. M. Bishop (1999). Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 61*(3), 611–622.

Valiant, P. and G. Valiant (2013). Estimating the unseen: Improved estimators for entropy and other properties. In *Advances in Neural Information Processing Systems*, pp. 2157–2165.

Valmianski, I., A. Y. Shih, J. D. Driscoll, D. W. Matthews, Y. Freund, and D. Kleinfeld (2010). Automatic identification of fluorescently labeled brain cells for rapid functional imaging. *Journal of neurophysiology 104*(3), 1803.

Van der Maaten, L. and G. Hinton (2008). Visualizing data using t-sne. *Journal of Machine Learning Research 9*(11).

Van Overschee, P. and B. De Moor (1996). Subspace identification for linear systems: theory, implementation, applications.

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

Vogelstein, J. T., A. M. Packer, T. A. Machado, T. Sippy, B. Babadi, R. Yuste, and L. Paninski (2009). Fast non-negative deconvolution for spike train inference from population calcium imaging. *arXiv preprint arXiv:0912.1637*.

Vogelstein, J. T., B. O. Watson, A. M. Packer, R. Yuste, B. Jedynak, and L. Paninski (2009). Spike inference from calcium imaging using sequential monte carlo methods. *Biophysical journal 97*(2), 636–655.

Vu, V. Q., B. Yu, and R. E. Kass (2007). Coverage-adjusted entropy estimation. *Statistics in medicine 26*(21), 4039–4060.

Weinberger, K. Q. and L. K. Saul (2006a). An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *AAAI*, Volume 6, pp. 1683–1686.

Weinberger, K. Q. and L. K. Saul (2006b). Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision 70*(1), 77–90.

Wood, F., C. Archambeau, J. Gasthaus, L. James, and Y. W. Teh (2009). A stochastic memoizer for sequence data. In *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, pp. 1129–1136.

Yaksi, E. and R. W. Friedrich (2006). Reconstruction of firing rate changes across neuronal populations by temporally deconvolved ca2+ imaging. *Nature Methods 3*(5), 377–383.

Yu, B. M., J. P. Cunningham, G. Santhanam, S. I. Ryu, K. V. Shenoy, and M. Sahani (2009). Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Journal of neurophysiology 102*(1), 614–635.

Yu, Y.-L. and D. Schuurmans (2012). Rank/norm regularization with closed-form solutions: Application to subspace clustering. *arXiv preprint arXiv:1202.3772*.

Yuste, R. and W. Denk (1995). Dendritic spines as basic functional units of neuronal integration. *Nature 375*(6533), 682–684.

Yuste, R. and A. Konnerth (2004). *Imaging in neuroscience and development.* Cold Spring Harbor Lab. Press.

Zaharia, M., M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica (2010). Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, pp. 10–10.

Zhao, P., G. Rocha, and B. Yu (2009). The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, 3468–3497.

Zhou, Z., X. Li, J. Wright, E. Candes, and Y. Ma (2010). Stable principal component pursuit. *Proceedings of the IEEE International Symposium on Information Theory*, 1518–1522.