UNIVERSITY OF CALIFORNIA SAN DIEGO

Characterizing Biological Neural Systems Using Variational Annealing and Applications to Machine Learning Tasks

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Physics

by

Anna Miller

Committee in charge:

Professor Henry Abarbanel, Chair
Professor George Fuller
Professor Melvin Leok
Professor Mark Paddock
Professor Miller Puckette

2021

DEDICATION

To Jonathan who stuck by me.

To my friends and mentors who encouraged me.

To the LBLs who helped me find myself.

And to my very best friend. Your love and support got me through to the end.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Chapter 4 was adapted from work being prepared for publication of the material as it may appear in A. Miller, D. Li, J. Platt, A. Daou, D. Margoliash, and H.D.I. Abarbanel, Statistical Data Assimilation: Formulation and Examples From Neurobiology, with consent of the authors. The dissertation author was the primary investigator and author of this paper.

Chapter 5 was adapted from work being prepared for publication of the material as J. Platt, A. Miller, L.Fuller, H.D.I. Abarbanel, Machine Learning Classification Informed by a Functional Biophysical System, with consent of the authors. The dissertation author was one of the primary investigators and authors of this paper.

VITA

2014	B. S. in Physics *cum laude*, University of Illinois Urbana-Champaign,
2015-2020	Graduate Teaching Assistant, University of California San Diego
2021	Ph. D. in Physics, University of California San Diego

PUBLICATIONS

A. Miller, D. Li, J. Platt, A. Daou, D. Margoliash, and H.D.I. Abarbanel, “Statistical Data Assimilation: Formulation and Examples From Neurobiology”, *Frontiers in Applied Mathematics and Statistics,* 4:53 (2018).

J. Platt, A. Miller, L.Fuller, H.D.I. Abarbanel, “Machine Learning Classification Informed by a Functional Biophysical System”, *arXiv preprint arXiv:1911.08589 (2019).*

ABSTRACT OF THE DISSERTATION

Characterizing Biological Neural Systems Using Variational Annealing and Applications to Machine Learning Tasks

by

Anna Miller

Doctor of Philosophy in Physics

University of California San Diego, 2021

Professor Henry Abarbanel, Chair

Characterizing the many physical properties of biological neurons has been historically difficult. The use of conductance models describing how properties of these systems change with time can be combined with measured voltage traces to help characterize immeasurable properties of a neuron. This is accomplished using data assimilation, which formulates the inference of these properties as probability maximization using nonlinear optimization. Because measurement noise and model error are inevitable in the study of complex systems, the methods used in this dissertation are designed to cope with unknown processes.

This dissertation starts with an overview of data assimilation by formulating the problem data assimilation attempts to solve. I then introduce two methods, nudging and variational annealing, which I use to characterize the properties of a neuron in the Zebra Finch songbird system, $\mathrm{HVC_X}$. A key result from this experiment is that there are statistical differences in estimated model parameters for neurons from different birds.

Current artificial neural network models are unmanageably large (billions of parameters) and need massive amounts data and computational power to train. The insect olfactory system is a biological network which is capable of learning and identifying new odors quickly in the presence of interference. These properties make it attractive model to explore as a classifier for machine learning tasks.

I give an overview of the insect olfactory system by describing its key properties. I use these properties to create a simplified classification system using a winnerless competition network as a pre-processor to a support vector machine. I demonstrate this networks classification capabilities using classes designed to resemble the stimulus an insect receives in the presence of odors. One key result from this experiment is that our network is capable of identifying a mixtures of previously seen odors.

Chapter 1

Introduction

Prior to originating as its own field in the 1960s, the field of neuroscience was spread across many disciplines ranging from neuroanatomy and biology all the way to psychology and linguistics. This has created the field of neuroscience today, which encompasses many disciplines and approaches to the study of the nervous system. But why study it?

Understanding how the brain works can help identify when there are problems and shed light on potential solutions to those problems. One example is the study of Alzheimer's. While research is still ongoing, studying how the disease progresses in the brain has allowed for scientists to develop drugs that slow down the progression of the disease, improving patient quality of life.

Additionally, the brain has been used as inspiration for the artificial neural networks used in machine learning today. Their origin dates back to the 1940s in which Warren McCulloch and Walter Pitts built a model of neurons in the brain using simple neuron circuits. Frank Rosenblatt built upon this work by conceiving the first idea specifically designed to make machines learn, the perceptron. This idea differed from McCulloch-Pitts in that it included the ability for the network to learn through changes in the synapses between neurons. Since then, artificial neural networks have become more complex by vastly expanding the number of neurons in a network and using clever architecture meant to emulate processes in the brain (e.g., convolutional neural networks recurrent neural networks).

These two applications are just a small part of what the field has to offer. Because of the many disciplines which study neuroscience, there are also many approaches used within the study. One such approach is the use of animal models.

Animal models have the benefit of being much smaller and simpler than their human counterpart, making them more manageable study. They also allow for more invasive procedures, which can yield information that would otherwise be impossible to obtain. This dissertation focuses on the study of two such systems: the song system in Zebra Finch, and olfactory system of insects.

The song bird system is an interesting system of study because song is a learned vocal behavior controlled by discrete neural circuits. The song is learned within clear phases of development, which is reflected in the development of the underlying neural circuits [21]. These properties make the songbird system one which is relatively easy to study. Additionally, it has the potential to inform how humans develop speech because the songbird system contains regions analogous to regions in the human speech system. A deep understanding of the song bird system can lead to information about speech disorders in humans [22, 26]. Within this dissertation, I focus on a single neuron present in the song bird system, and use an inference technique in order to estimate the physical parameters of the neuron.

The second system of study is the olfactory system in insects. This system is an important system of study because it answers fundamental question of how animals detect, encode, and process sensory information. Of the properties of interest, this network learns quickly, can identify odors in the presence of strong noise, and is relatively well studied. These are attractive properties within the context of machine learning because current deep network models are becoming unmanageably large and data hungry, requiring multiple presentations of items within an extremely large data set. Within this dissertation, I use the insect olfactory system as an inspiration for a machine learning classifier.

The study of both these systems share a common theme: I use physics and computation tools to broadly understand the learning systems of neurons and neural networks, with the hope that insights into these systems may help inform our greater understanding about learning.

1.1 Overview of the Dissertation

This dissertation contains 5 chapters. The first chapter is an introduction to the systems of study and why they are interesting.

Chapter 2 sets up the idea of an inference problem one solution to that problem: data assimilation. Here I explain the overall approach of data assimilation as well as a probabilistic point of view and its relation to path integrals. I then present two methods data assimilation methods: one which incorporates the use of nudging, and an annealing method developed within the research group. Once these are established I give a brief mention of new methods being developed.

Chapter 3 provides the background knowledge on the models used for single celled neurons needed for the rest of the dissertation.

Chapter 4 starts with an overview of the bird song system in Zebra Finch. I then describe, in detail, the model for $\mathrm{HVC_X}$ neurons and briefly touch on the experimental data can be obtained. The results of the data assimilation methods described in chapter 2 is presented

Chapter 5 starts with an introduction for how the olfactory system in insects can be formulated as a winnerless competition network + support vector machine classifier. I then present the results of this network's ability to classify spatial signals independently and a mixture of learned spatial signals.

Chapter 2

Data Assimilation

The study of physics involves asking fundamental questions about the world around us and using experiments and observations to create models reflecting the behavior of the system and underlying physical properties. For simple systems, it's entirely possible to develop these models *ab initio*, or from first principles. However, for more complex systems, computational tools must be used to develop quantitative models of these systems. This chapter details one such approach, with a focus neuron models.

2.1 The Inverse Problem

We wish to model dynamical systems such that we are capable of accurately forecasting the behavior of that system (prediction), and learn about quantities/relationships within that system (inference). In practice, this is difficult as the physical system likely includes processes that the modeler does not know about. Another difficulty comes from the noise is real data. Any measurement tool is subject to noise, which limits the amount of knowledge one can infer from the data. Due to these limitations, the appropriate approach is a probabilistic one. Given the observations/measurements taken of the physical system, what is the most likely model to yield these measurements. We wish to calculate the model and set of parameters that gave rise to the observable data. This is called an inverse problem.

2.1.1 The Data and The Model

We start with a observation window in time $[t_0, t_F]$ within which we make a set of measurements at times $t = \{\tau_1, \tau_2, ..., \tau_k, \tau_F\}$; $t_0 \leq \tau_k \leq t_F$. At each of these measurement times, we observe L quantities $\mathbf{y}(\tau_k) = \{y_1(\tau_k), y_2(\tau_k), ..., y_L(\tau_k)\}$. The number L of observations at each measurement time τ_k is typically less, often much less, than the number of degrees of freedom D in the observed system; $D \gg L$.

These measurements are a window into the dynamical processes of a system we wish to characterize. The quantitative characterization is through a model we choose. The model describes the interactions among the states of the observed system. If we are observing the time course of a neuron, for example, we might measure the membrane voltage $y_1(\tau_k) = V_m(\tau_k)$ and the intracellular Ca^{2+} concentration $y_2(\tau_k) = [Ca^{2+}](\tau_k)$. From these data we want to estimate the unmeasured states of the model as a function of time as well as estimate biophysical parameters in the model.

The processes characterizing the state of the system (e.g. a neuron) we call $x_a(t)$; $a = 1, 2, ..., D \geq L$, and they are selected by the user to describe the dynamical behavior of the observations through a set of differential equations in continuous time

$$\frac{dx_a(t)}{dt} = F_a(\mathbf{x}(t), \mathbf{q}). \tag{2.1}$$

Most physical systems can be expressed in this manner, as a set of first-order differential equations. Given that we cannot take measurements in continuous time, the use of a discretized model of equation 2.1 is necessary. In discrete time, where $t_n = t_0 + n\Delta t$; $n = 0, 1, ..., N$; $t_N = t_F$, equation 2.1 becomes

$$x_a(t_{n+1}) = x_a(n+1) = f_a(\mathbf{x}(t_n), \mathbf{q}) = f_a(\mathbf{x}(n), \mathbf{q}), \tag{2.2}$$

where $\mathbf{q}$ is a set of fixed parameters associated with the model. $\mathbf{f}(\mathbf{x}(n), \mathbf{q})$ is related to $\mathbf{F}(\mathbf{x}(t), \mathbf{q})$ through the choice the user makes for solving the continuous time flow for $\mathbf{x}(t)$ through a numerical solution method of choice [81].

Thinking of neuron activity, equation (2.1) could be coupled Hodgkin-Huxley (HH) equations [54, 98] for voltage, ion channel gating variables, constituent concentrations, and other ingredients. Typical parameters might be maximal conductances of the ion channels, reversal potentials, and other time-independent numbers describing the kinetics of the gating variables. Specifics of this model are detailed in chapter 3. In many experiments L might be only 1, namely, the voltage across the cell membrane, while D the number of state variable may be 10's; that is, many, and $D \gg L$.

As we proceed from the initiation of the observation window at t_0 we must move our model equation variables $\mathbf{x}(0)$, equation (2.2), from t_0 to τ_1 where the first measurement is made. Then using the model dynamics we move along to τ_2 and so forth until we reach the last measurement time τ_F and finally move the model from $\mathbf{x}(\tau_F)$ to $\mathbf{x}(t_F)$. In each stepping of the model equations, we may make many steps of Δt in time to achieve accuracy in the representation of the model dynamics. The full set of times t_n at which we evaluate the model $\mathbf{x}(t_n)$ we collect into the path of the state of the model through D-dimensional space: $\mathbf{X} = \{\mathbf{x}(0), \mathbf{x}(1), ..., \mathbf{x}(n), ..., \mathbf{x}(N) = \mathbf{x}(F)\}$. The dimension of the path is $(N+1)D + N_q$, where N_q is the number of parameters $\mathbf{q}$ in our model.

It is worth a pause here to note that we have now collected two of the needed three ingredients to effect our transfer of the information in the collection of all measurements $\mathbf{Y} = \{\mathbf{y}(\tau_1), \mathbf{y}(\tau_2), ..., \mathbf{y}(\tau_F)\}$ to the model $\mathbf{f}(\mathbf{x}(n), \mathbf{q})$ along the path $\mathbf{X}$ through the observation window

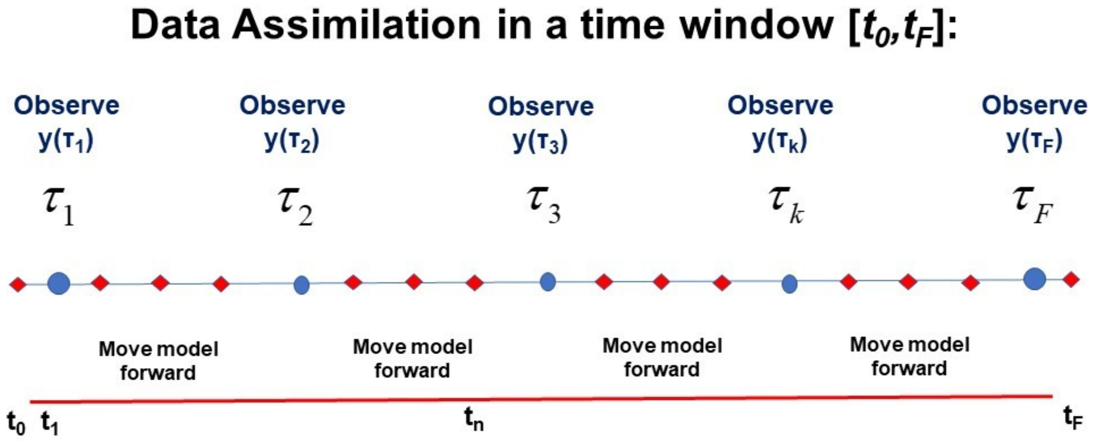

Figure 2.1: A visual representation of the window $t_0 \le t \le t_F$ in time during which L-dimensional observations $\mathbf{y}(\tau_k)$ are performed at observation times $t = \tau_k$; $k = 1,...,F$. This also shows times at which the D-dimensional model developed by the user $\mathbf{x}(n+1) = \mathbf{f}(\mathbf{x}(n),\mathbf{q})$ is used to move forward from time n to time $n+1$: $t_n = t_0 + n\Delta t$; $n = 0,1,...,N$. $D \ge L$. The path of the model $\mathbf{X} = \{\mathbf{x}(0),\mathbf{x}(1),...,\mathbf{x}(n),...,\mathbf{x}(N) = \mathbf{x}(F)\}$ and the collection $\mathbf{Y}$ of L-dimensional observations at each observation time τ_k, $\mathbf{Y} = \{\mathbf{y}(\tau_1),\mathbf{y}(\tau_2),...,\mathbf{y}(\tau_k),...,\mathbf{y}(\tau_F\}$ ($\mathbf{y} = \{y_1,y_2,...,y_L\}$) is also indicated.

$[t_0, t_F]$: (1) data $\mathbf{Y}$, hopefully well curated, and (2) a model of the processes in $\mathbf{Y}$, devised by our experience and knowledge of those processes. The notation and a visual presentation of this is found in Fig. (2.1). The third ingredient is the methods to generate the transfer from $\mathbf{Y}$ to properties of the model. The next section details the formalism for how we approach this information transfer.

2.1.2 Probabilistic Formulation of Data Assimilation

Inevitably, the data we collect is noisy, and equally, the model we select to describe the production of those data has errors. This means we must use a statistical description of the assimilation of information transfer from the data, $\mathbf{Y}$, to the model. This takes the form of a conditional probability distribution $P(\mathbf{X}|\mathbf{Y})$. Our goal is not the evaluation of the the probability distribution, $P(\mathbf{X}|\mathbf{Y})$, but rather the estimation of some physical quantities of the system, denoted $G(\mathbf{X})$. We seek the conditional expected value of the quantity $G(\mathbf{X})$ on the observed data. The expected value of $G(\mathbf{X})$ is

$$E[G(\mathbf{X})|\mathbf{Y}] = \langle G(\mathbf{X}) \rangle = \frac{\int d\mathbf{X}\, G(\mathbf{X}) P(\mathbf{X}|\mathbf{Y})}{\int d\mathbf{X}\, P(\mathbf{X}|\mathbf{Y})}. \tag{2.3}$$

A usual quantity of interest of $G(\mathbf{X})$ is the path $G(\mathbf{X}) = \mathbf{X}$. If one has an anticipated form for the distribution at large $\mathbf{X}$, then $G(\mathbf{X})$ may be chosen as a parametrized version of that form and those parameters determined near the maximum of $P(\mathbf{X}|\mathbf{Y})$.

2.1.3 Defining the Conditional Probability $P(\mathbf{X}|\mathbf{Y})$

In order to evaluate equation 2.3, we must first write down an expression for the probability distribution $P(\mathbf{X}|\mathbf{Y}) = P(\mathbf{x}(0), \mathbf{x}(1), ..., \mathbf{x}(N)|\mathbf{y}(\tau_1), \mathbf{y}(\tau_2), ..., \mathbf{y}(\tau_F))$. We start with defining the transfer probability of the system from state $\mathbf{x}(n-1)$ to state $\mathbf{x}(n)$, $P(\mathbf{x}(n)|\mathbf{x}(n-1))$.

We assume that the mapping of system from state $\mathbf{x}(n-1)$ to state $\mathbf{x}(n)$ is Markovian. That is, the probability of an event at time t_n only depends on the state at time t_{n-1}. Or, $P(\mathbf{x}(n)|\mathbf{x}(n-1),...,\mathbf{x}(0)) = P(\mathbf{x}(n)|\mathbf{x}(n-1))$. When we know our model entirely, this transfer probability is deterministic. However, with error, our discretized system becomes

$$\mathbf{x}(n) = \mathbf{f}(\mathbf{x}(n-1),\mathbf{q}) + \varepsilon \tag{2.4}$$

where $\varepsilon = \mathcal{N}(0,\mathbf{R}_f^{-1})$ and $\mathbf{R}_f^{-1}$ is a D-by-D covariance matrix. The transfer probability is

$$P(\mathbf{x}(n)|\mathbf{x}(n-1)) = P(\varepsilon) \propto e^{-\frac{1}{2}\varepsilon^T\cdot\mathbf{R}_f\cdot\varepsilon}. \tag{2.5}$$

Support Vector Machines (SVMs) are a class of learning algorithms that are used in classification tasks. By default, the SVM is only capable of binary classification. Suppose you have a set of labeled data points $\mathbf{X} = \{\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_i,...,\mathbf{x}_m\}$ with their corresponding label $\mathbf{Y} = \{y_1,y_2,...,y_i,...,y_m\}$. Since this is a binary classified, $\mathbf{y}_i$ can take two values: -1 or $+1$. W

$$P(\mathbf{y}(n)|\mathbf{x}(n)) = P(\eta) \propto e^{-\frac{1}{2}\eta^T\cdot\mathbf{R}_m\cdot\eta}, \tag{2.6}$$

where $\mathbf{R}_m^{-1}$ is an L-by-L covariance matrix. We also assume that the measurement of the system at any point in time depends *only* on the state at that time and not an previous state of the system. That is $P(\mathbf{y}(n)|\mathbf{x}(n),\mathbf{x}(n-1),...,\mathbf{x}(1)) = P(\mathbf{y}(n)|\mathbf{x}(n))$. For our use case, these covariance matrices (and the corresponding precision matrices) are diagonal. We denote non-zero diagonal elements of these matrices as $R_f(a)$ and $R_m(l)$.

Now that we have defined these probability distributions, we can proceed to defining our conditional probability distribution. $P(\mathbf{X}|\mathbf{Y})$ can be rewritten using Baye's rules. For viewing clarity, we define an alternate notation: the path up to time t_n is $\mathbf{X}_n$ and the state at time t_n is $\mathbf{x}_n$.

$$P(\mathbf{X}|\mathbf{Y}) = P(\mathbf{X}_n|\mathbf{Y}_n) = \frac{P(\mathbf{X}_n, \mathbf{Y}_n)}{P(\mathbf{Y}_n)} \tag{2.7}$$

We can rewrite any path probability distribution $P(\mathbf{X}_n)$ as $P(\mathbf{x}_n, \mathbf{X}_{n-1})$ and use Baye's theorem to obtain

$$P(\mathbf{X}_n|\mathbf{Y}_n) = \frac{P(\mathbf{Y}_{n-1})P(\mathbf{y}_n, \mathbf{X}_n|\mathbf{Y}_{n-1})}{P(\mathbf{y}_n|\mathbf{Y}_{n-1})P(\mathbf{Y}_{n-1})} \cdot \frac{P(\mathbf{X}_n|\mathbf{Y}_{n-1})}{P(\mathbf{X}_n|\mathbf{Y}_{n-1})}. \tag{2.8}$$

We multiply the expression by $\frac{P(\mathbf{X}_n|\mathbf{Y}_{n-1})}{P(\mathbf{X}_n|\mathbf{Y}_{n-1})}$ in order to obtain a familiar expression. We use the definition of conditional mutual information,

$$CMI(a,b|c) = \log\left[\frac{(P(a,b|c)}{P(a|c)P(a|c)}\right]. \tag{2.9}$$

This is Shannon's conditional mutual information [35] telling us how many bits (for $\log_2$) we know about a when observing b conditioned on c. For us $a = \{\mathbf{y}_n\}, b = \{\mathbf{x}_n, \mathbf{X}_{n-1}\}, c = \{\mathbf{Y}_{n-1}\}$. We define a recursion relation

$$\begin{aligned} P(\mathbf{X}_n|\mathbf{Y}_n) = & \frac{P(\mathbf{y}_n, \mathbf{x}_n, \mathbf{X}_{n-1}|\mathbf{Y}_{n-1})}{P(\mathbf{y}_n|\mathbf{Y}_{n-1})P(\mathbf{x}_n, \mathbf{X}_{n-1}|\mathbf{Y}_{n-1})} \cdot \\ & P(\mathbf{x}_n|\mathbf{x}_{n-1})P(\mathbf{X}_{n-1}|\mathbf{Y}_{n-1}) \qquad (2.10) \\ = & \exp[CMI(\mathbf{y}_n, \mathbf{x}_n, \mathbf{X}_{n-1}|\mathbf{Y}_{n-1})] \cdot \\ & P(\mathbf{x}_n|\mathbf{x}_{n-1})P(\mathbf{X}_{n-1}|\mathbf{Y}_{n-1}). \qquad (2.11) \end{aligned}$$

To get equation 2.10, we have to use the transfer probability assumption defined earlier. Using this recursion relation (equation 2.11) to move backwards through the observation window from $t_F = t_0 + N\Delta t$ through the measurements at times τ_k to the start of the window at t_0, we may write, up to factors independent of $\mathbf{X}$

$$P(\mathbf{X}|\mathbf{Y}) = \left\{ \prod_{k=1}^{F} P(\mathbf{y}_k|\mathbf{X}_k) \prod_{n=0}^{F-1} P(\mathbf{x}_{n+1}|\mathbf{x}_n) \right\} P(\mathbf{x}_0). \tag{2.12}$$

Here, the iterations of k and n need not be the same. If we now write $P(\mathbf{X}|\mathbf{Y}) \propto \exp[-A(\mathbf{X})]$ where $A(\mathbf{X})$, the negative of the log likelihood, we call the action, then conditional expected values for functions along the path $\mathbf{X}$ are defined by

$$E[G(\mathbf{X})|\mathbf{Y}] = \langle G(\mathbf{X}) \rangle = \frac{\int d\mathbf{X}\, G(\mathbf{X}) \exp[-A(\mathbf{X})]}{\int d\mathbf{X} \exp[-A(\mathbf{X})]}. \tag{2.13}$$

The action takes the convenient expression

$$A(\mathbf{X}) = \left\{ -\sum_{k=1}^{F} \log[P(\mathbf{y}_k|\mathbf{X}_k)] - \sum_{n=0}^{N} \log[P(\mathbf{x}_{n+1}|\mathbf{x}_n)] \right\} - \log[P(\mathbf{x}_0)], \tag{2.14}$$

which is the sum of the terms which modify the conditional probability distribution when an observation is made at $t = \tau_k$ and the sum of the stochastic version of $\mathbf{x}_n \to \mathbf{x}_{n+1} - \mathbf{f}(\mathbf{x}_n, \mathbf{q})$ and finally the distribution when the observation window opens at t_0.

In the language of Bayesian Inference, this is a Maximum a Posteriori estimation where $P(\mathbf{X}|\mathbf{Y})$ is our posterior distribution, the first term of our action is our likelihood function, and the transition probabilities are our prior distribution.

The action simplifies to what we call the ‘standard model’ of data assimilation when (1) observations $\mathbf{y}$ are related to their model counterparts via Gaussian noise with zero mean and diagonal precision matrix $\mathbf{R}_m$, and (2) model errors are associated with Gaussian errors of mean zero and diagonal precision matrix $\mathbf{R}_f$:

$$A(\mathbf{X}) = \sum_{k=1}^{F} \sum_{l=1}^{L} \frac{R_m(l)}{2}(x_{\{l,k\}} - y_{\{l,k\}})^2 + \sum_{n=0}^{F-1} \sum_{a=1}^{D} \frac{R_f(a)}{2}(x_{\{a,n+1\}} - f_a(\mathbf{x}_n, \mathbf{q}))^2. \tag{2.15}$$

If we have knowledge of the distribution $P(\mathbf{x}_0)$ at t_0 we may add it to this action. If we have no knowledge of $P(\mathbf{x}_0)$, we take its distribution to be uniform over the dynamic range of the model variables.

Our challenge is to perform integrals such as equation (2.13). The evaluation of this integral is computationally expensive. Fortunately, in most applications, we need not know the full probability distribution. Instead we seek to sufficiently approximate this integral such that it yields us the expected value of our properties of interest.

In most systems, we expect that the dominant contribution to the expected value comes from the maxima of $P(\mathbf{X}|\mathbf{Y})$ or, equivalently the minima of $A(\mathbf{X})$. At such minima, the two contributions to the action, the measurement error and the model error, balance each other to accomplish the explicit transfer of information from the former to the latter.

We note that when $\mathbf{f}(\mathbf{x}(n), \mathbf{q})$ is nonlinear in $\mathbf{X}$, the expected value integral equation (2.13) is not Gaussian. So, some thinking is in order to approximate this high dimensional integral. The two generally useful methods available for evaluating this kind of high-dimensional integral are Laplace's method [62, 63] and Monte Carlo techniques [81, 59, 76]. They will be addressed in this order.

2.1.4 Laplace's Method

Laplace's method [62] is a method for approximating integrals of the form

$$H(x) = \int e^{Mh(x)} g(x) dx \approx e^{Mh(x^*)} g(x^*) \int e^{-\frac{1}{2}M|h''(x^*)|(x-x^*)^2} dx \tag{2.16}$$

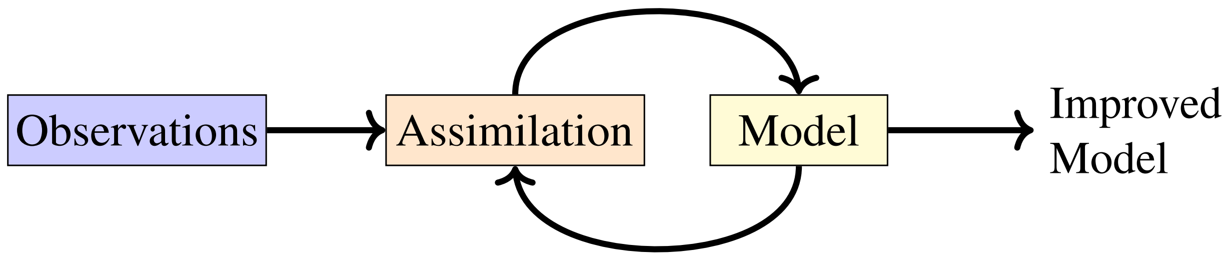

Figure 2.2: The workflow of data assimilation. The goal of data assimilation is to use real observations of a dynamical system to inform a physical model of that system.

where that $h(x)$ and $g(x)$ are smooth real functions, M is real value. As M increases, the solution to the integral becomes more dependent upon the maximum value of $h(x)$. The integral is approximated by performing a second order Taylor expansion of $h(x)$ around it's global maximizer, x^*. For our purposes, this amounts to finding the minimum of the action in equation 2.15.

To locate the minima of the action $A(\mathbf{X}) = -\log[P(\mathbf{X}|\mathbf{Y})]$ we must seek extremum paths $\mathbf{X}^{(j)};\ j = 0, 1, \ldots$ such that $\partial A(\mathbf{X})/\partial \mathbf{X}|_{\mathbf{X}^{(j)}} = 0$, and then check that the second derivative at $\mathbf{X}^{(j)}$, the Hessian, is a positive definite matrix in path coordinates. Both these conditions ensure that the path $X^{(j)}$ found is a minima of $A(\mathbf{X})$.

Laplace's method [62] expands the action around the $\mathbf{X}^{(j)}$ seeking the path $\mathbf{X}^{(0)}$ with the smallest value of $A(\mathbf{X})$. The contribution of $\mathbf{X}^{(0)}$ to the integral equation 2.13 is approximately $\exp\left[A(\mathbf{X}^{(1)}) - A(\mathbf{X}^{(0)})\right]$ bigger than that of the path with the next smallest action.

This sounds more or less straightforward, however, finding the global minimum of a nonlinear function such as $A(\mathbf{X})$ is an NP-complete problem [74]. In a practical sense, this means this problem is not solvable in a realistic amount of time. However there is an attractive feature of the form of $A(\mathbf{X})$ that permits us to accomplish more.

We now discuss two algorithmic approaches to implementing Laplace's method.

2.1.5 Precision Annealing for Laplace's Method

Looking at equation 2.15, we see that if the precision of the model is zero, $R_f = 0$, the action is quadratic in the L measured variables $x_l(n)$ and independent of the remaining states. The global minimum of such an action comes with $x_l(\tau_k) = y_l(\tau_k)$ and any choice for the remaining states and parameters. We can initialize our search such that we choose a path where $x_l(\tau_k) = y_l(\tau_k)$ and sample the remaining state variables and parameter values from a uniform distribution covering their dynamic range. We call this path $\mathbf{X}_{\text{init}}$. Because we are not enforcing model dynamics ($R_f = 0$), the minimum of $A(\mathbf{X})$ is very degenerate. We approach this by initializing multiple paths, namely N_I of them. We continue to call the collection of N_I paths $\mathbf{X}_{\text{init}}$. In practice this is usually anywhere from 10 to a couple hundred paths.

The next step is to increase R_f from $R_f = 0$ to a small value R_{f0}. Each state variable in the problem has its own value for R_{f0}, $R_{f0}(n)$. For the following explanation, will refer to R_f generally, as the value of $R_{f0}(n)$ does not change our analysis. Each of the N_I paths in $\mathbf{X}_{\text{init}}$ is used as an initial condition for our numerical optimization program chosen to find the minima of $A(\mathbf{X})$. This gives us N_I paths $\mathbf{X}_0$. The action is evaluated on each of these paths N_I, $A(\mathbf{X}_0)$, and recorded.

Next, we increase $R_f = R_{f0} \to R_{f0}\alpha$; $\alpha > 1$, and now use the N_I paths $\mathbf{X}_0$ as the initial conditions for our numerical optimization program chosen to find the minima of $A(\mathbf{X})$, we arrive at N_I paths $\mathbf{X}_1$. Evaluate $A(\mathbf{X}_1)$ on all N_I paths $\mathbf{X}_1$, and record the values. R_f is increased again to $R_{f0}\alpha^2$ and the same steps are repeated. We continue in this manner, increasing R_f to $R_f = R_{f0}\alpha^{\beta}$; $\beta = 0, 1, \ldots$, then using the selected numerical optimization program to arrive at N_I paths $\mathbf{X}_{\beta}$. Evaluate $A(\mathbf{X}_{\beta})$ on all N_I paths $\mathbf{X}_{\beta}$. We do this until we reach a maximum value of β that we choose, β_{max}. The choice of β_{max} depends in part on the choice of α. We must make $R_f = R_{f0}\alpha^{\beta_{\text{max}}}$ sufficiently large such that $R_f \to \infty$. This scaling is important as we seek to find a path and set of parameters such that the action becomes independent of the model error, or the model error is approximately zero. We can observe when this happens by plotting the values of the action as a function of $\beta = \log_\alpha \left[\frac{R_f}{R_{f0}}\right]$. We display all N_I values of $A(\mathbf{X}_\beta)$ versus β for all $\beta = 0, 1, 2, \ldots\beta_{\max}$. If we see that the values of the action 'level out', or remain constant with an increasing R_f, we know that we have found a model estimation which reflects the dynamics of the system.

We call this method *precision annealing* (PA) [107, 109, 108, 83]. It slowly turns up the precision of the model collecting paths at each R_f that emerged from the degenerate global minimum at $R_f = 0$. In practice it is able to track N_I possible minima of $A(\mathbf{X})$ at each R_f. When not enough information is presented to the model, that is L is too small, there are many local minima at all R_f. This is a manifestation of the NP-completeness of the minimization of $A(\mathbf{X})$ problem. None of the minima may dominate the expected value integral of interest.

As L increases, and enough information is transmitted to the model, for large R_f one minimum appears to stand out as the global minimum, and the paths associated with that smallest minimum yields good predictions. We note that there are always paths, not just a single path, as we have a distribution of paths, N_I of which are sampled in the PA procedure, within a variation of size $1/\sqrt{R_m}$ [95].

2.1.6 "Nudging" within Laplace's Method

In meteorology, one approach to data assimilation is to add a term to the deterministic dynamics which move the state of a model towards the observations [6]

$$x_a(n+1) = f_a(\mathbf{x}(n), \mathbf{q}) + u(n)(y_l(n) - x_l(n))\delta_{al}, \tag{2.17}$$

where $u(n) > 0$ and vanishes except where a measurement is available. This is referred to as 'nudging'. $u(n)$ is a penalty term which penalizes the system if the model state is too far from observations. The penalty term can be seen in the action previous defined. In continuous time, the action is defined as

$$A(\mathbf{x}(t)) = \int_{t0}^{t_F} dt\, \mathcal{L}(\mathbf{x}(t), \dot{\mathbf{x}}, t) \tag{2.18}$$

where the Lagrangian is

$$\begin{aligned}\mathcal{L}(\mathbf{x}, \dot{\mathbf{x}}, t) = &\sum_{l=1}^{L} \frac{R_m(t,l)}{2}(x_l(t) - y_l(t))^2 \\ &+ \sum_{a=1}^{D} \frac{R_f(a)}{2}(\dot{\mathbf{x}}_a(t) - F_a(\mathbf{x}(t), \mathbf{q}))^2. \end{aligned}\tag{2.19}$$

The extremum of this action is given by the Euler-Lagrange equations of the calculus of variations [40]

$$\begin{aligned}\left[\delta_{ab}\frac{d}{dt} + \frac{\partial F_b(\mathbf{x}(t))}{\partial x_a(t)}\right]&\left[\dot{x}_b(t) - F_b(\mathbf{x}(t))\right] \\ &= \frac{R_m(l,t)}{R_f(a)}\delta_{al}(x_l(t) - y_l(t)), \end{aligned}\tag{2.20}$$

in which the right hand side is the 'nudging' term appearing in a natural manner. Approximating the operator $\delta_{ab}\frac{d}{dt} + \frac{\partial F_b(\mathbf{x}(t))}{\partial x_a(t)}$ we can rewrite this Euler-Lagrange equation in 'nudging' form

$$\frac{dx_a(t)}{dt} = F_a(\mathbf{x}(t)) + u(t)\delta_{al}(x_l(t) - y_l(t)). \tag{2.21}$$

This method is different from precision annealing in that precision anneal casts our problem as an unconstrained optimization where the dynamics of our system appear in the action, which can be otherwise known as the cost or objective function. The nudging method is a constrained optimization problem where we which to minimize the measurement error and penalty term, equation 2.22, with the constraint that all estimates follow the dynamics described in equation 2.21. This is achieved through the use of lagrange multipliers.

$$C = \sum_{k=1}^{F} \sum_{l=1}^{L} (x_l(k) - y_l(k))^2 + u(k)^2 \tag{2.22}$$

Both the precision annealing and nudging methods in combination with numerical optimization tools will be used to address problems in this dissertation.

2.1.7 Monte Carlo Methods

Optimization methods focus their approach on finding the minima of an objective function. For convex problems, this is relatively straight forward. However, the dynamics of our system are non-linear, making the the landscape of our objective function much more complicated. Rather than have a single global minima, the cost function contains many local minima. Due to the way objective functions search the space, it is possible for them to ’get stuck’ in a local minima. Monte Carlo techniques do not have this issues as they can move about the search space in any direction with some probability.

Monte Carlo methods [71, 59, 83, 81] are well covered in the literature. Rather than go into detail about these methods, I will discuss why such methods may be beneficial over the precision annealing Laplace’s Approximation approach. The approach detail above requires the evaluation of very high dimensional derivatives of $A(\mathbf{X})$. Depending on the optimization method chosen, one may need to calculate an $(D \times N)$ by $(D \times N)$ dimensional Hessian. There are many approaches to handing this hessian. One can symbolically calculate all non-zero elements prior to optimization [108], or using an approximation method such as L-BFGS-B [92]. Even such approaches are insufficient depending on the problem [48]. The use of precision annealing in combination Metropolis-Hastings Monte Carlo techniques addresses the difficulties associated with large matrices for the Jacobians and Hessians required in variation principles [105]. It does this by using random walkers to move around in $\mathbf{X}$ space. Each step the random walker takes is a proposal for a new position in this space. The proposal is then either accepted or rejected based on a ratio of probabilities. If the proposal is accepted, this new state is recorded and future proposals start from this new state. If the proposal is rejected, the old state is recorded and future proposal start from the old state. In this way, the probability distribution is sampled and the expected value of the integral can be computed after sufficiently may steps have been taken.

Metropolis-Hastings Monte Carlo techniques can suffer in high dimensional systems. The random nature of the proposals can result in a low acceptance rate of proposals and a slow exploration of $\mathbf{X}$ space. In theses types of high-dimensional problems, one can employ the use of Hamiltonian Monte Carlo in conjunction with precision annealing. Hamiltonian Monte Carlo adds an additional set of variables $\mathbf{P}$ and searches in $(\mathbf{X},\mathbf{P})$ space instead of just $\mathbf{X}$ space. The set of variables $\mathbf{P}$ are canonical conjugate of $\mathbf{X}$ and allow for the rules of classical mechanics to move around in $(\mathbf{X},\mathbf{P})$ space.

The Hamiltonian can be defined as

$$H(\mathbf{X},\mathbf{P}) = -\log\left(P(\mathbf{X},\mathbf{P}|\mathbf{Y})\right) = A(\mathbf{X}) + h(\mathbf{P}) \tag{2.23}$$

where $h(\mathbf{P}) = -\log(P(\mathbf{P}))$. Proposals are then made following Hamilton's equations

$$\frac{d\mathbf{X}(s)}{ds} = \frac{\partial H(\mathbf{X}(s),\mathbf{P}(s))}{\partial \mathbf{P}(s)};\ \frac{d\mathbf{P}(s)}{ds} = -\frac{\partial H(\mathbf{X}(s),\mathbf{P}(s))}{\partial \mathbf{X}(s)} \tag{2.24}$$

where s plays the role of time in classical mechanics and labels the movement of $\mathbf{X}$ and $\mathbf{P}$. Using these rules to move around in this space has the benefit of allowing large movement in $\mathbf{X}$ space with a high probability of acceptance, ideally 1 if the Hamiltonian is perfectly conserved. Practically, equations 2.24 need to be discretized which prevents perfect conservation of the Hamiltonian. The use of symplectic integrators allows for a high proposal acceptance rate and fast exploration of the phase space [34, 48].

For the purposes of estimating physical parameters of our system of interest, equation

2.25 demonstrates that the expected value is unchanged for an arbitrary choice of the canonical momentum.

$$\int d\mathbf{X}d\mathbf{P}G(\mathbf{X})P(\mathbf{P})P(\mathbf{X}|\mathbf{Y}) = \frac{d\mathbf{X}d\mathbf{P}G(\mathbf{X})e^{-H(\mathbf{X},\mathbf{P})}}{\int d\mathbf{X}d\mathbf{P}e^{-H(\mathbf{X},\mathbf{P})}} = \langle G(\mathbf{X})\rangle \tag{2.25}$$

While HMC solves the issue of slow phase space search, one drawback of this method is that it still requires the calculation of a Jacobian. For some problems, the use of PAMC may still be preferred.

In addition to avoiding the calculation of large matricies, Monte Carlo techniques have other advantages. In problems which do not have a clearly dominant global maxima, we cannot use Laplace's approximation. Monte Carlo techniques sample the entire probability distribution, allowing for a more accurate evaluation of equation 2.13.

2.1.8 Validation

To reiterate what we have just explained, we have a time series of measurements $\mathbf{Y}$, and wish to find a model, $\mathbf{F}$ that describes the observations. We do this by approximating a probability distribution $P(\mathbf{X}|\mathbf{Y})$, while estimating both the true state of the system $\mathbf{X}$ and the constant parameters of the model $\mathbf{q}$. We must choose one of the above techniques to perform this task. In this way, DA is used to transfer information from the measurements to our model. The general workflow of DA can be seen in figure 2.2.

If the transfer methods are successful and, according to some metric of success, we arrange matters so that at the measurement times τ_k, the L model variables $\mathbf{x}(t)$ associated with $\mathbf{y}(\tau_k)$ are such that $x_l(\tau_k) \approx y_l(\tau_k)$, we are *not* finished. We have then only demonstrated that the model is consistent with the known data $\mathbf{Y}$. The next step is to use the model, completed by the estimates of the $\mathbf{q}$ and the state of the model at t_F, $\mathbf{x}(t_F)$, to predict forward for $t > t_F$. We then compare this prediction with the measurements $\mathbf{y}(\tau_r)$ for $\tau_r > t_F$. If these two things generally agree, dependent of course on the metric of success we choose, we have succeeded in finding a good model to represent the physical system we're investigating.

As a small aside, the same overall setup applies to supervised machine learning networks [2] where the observation window is called the training set; the prediction window is called the test set, and prediction is called generalization.

2.1.9 Twin Experiment

Prior to applying DA to a physical system, we must ask whether our algorithm is even capable of yielding the correct parameter and state variable values given experimental conditions. There are multiple ways to verify this. One such way is by calculating conditional Lyapunov exponents (CLE). Lyapunov exponents are a metric which characterize the divergence of infinitesimally close trajectories. If at some time, there are two paths separated by $|\delta\mathbf{Z}_0|$, the rate at which this separation grows is $\approx e^{\lambda t}$. In a practical sense, this means that a system with positive lyapunov exponents, λ, are difficult to predict. Conditional lyapunov exponents are similar, except that the measured data is coupled to the model of the system. That is, the measured data drives the model. We can calculate what the lyapunov exponents are under the condition that some measured data exists (that's what makes CLE's conditional). If we find that there are positive CLE's we know our algorithm will be a poor predictor of the model with only the current amount of measured data. This calculation has been done on a 20-dimensional Lorenz96 model [58], using nudging methods. However, this calculation is much more difficult when using the precision annealing methods laid out earlier in the chapter. Instead we test our chosen methods by designing an experiment in which we have full knowledge and control of the model and experimental data.

Suppose we want to model the neuron of a mouse. We spend time looking at experiments and reading papers and design a model, F, that we believe is an accurate representation of this particular neuron. Once we have our model, the next step is to generate synthetic data an appropriate numerical solver. We pick a starting point, and integrate that model forward. This gives us our ground truth. In this set up, we know the model, the parameters (because we chose them), and every value of every state variable at every time. But the goal is to verify whether our algorithm is capable of producing the correct parameters *under experimental conditions*. This means we take the measurable state variables and add noise. The noise amount should be in line with what we expect from the experimental tools taking the measurements. Then we set aside the unmeasured state variables, and apply our DA method of choice to this noisy synthetic data using the same model used to generate the data. We ask our DA method to estimate the same parameters as we would in a 'real' experiment. Then we obtain our results. Since we know the model entirely, we can directly compare parameter estimations *and* the prediction. If DA is incapable of estimating the correct model under these ideal conditions, we know that what we have is insufficient and we must work to alter the experiment in some way. The twin experiment is also an opportunity to tune the hyper parameters of our methods. The two major hyper-parameters being α and β, but we can also tune the relative important we place on the differential equations for each state variable by tuning the relative strength of R_{f0} for each state variable. Once we have obtained favorable results from the twin experiment, we can apply the methods to real data.

Chapter 3

Neurological Background

The neuron is a complex biological structure. The cell body, or soma, is responsible for maintaining neuron functions and contains the cell's genetic information in its nucleus. One or multiple axons propagate electrical signal down its long, tail-like structure, surrounded by segments of myelin sheath. The myelin sheath functions like an insulator and does not allow ions to cross the cell membrane at those regions, but instead only at junctions between the myelianated sections, allowing for the quick propagation of electrical signal down the axon. Axon terminals make synaptic connections with the dendrites of other neurons. Dendrites branch out from the cell body and receive signal from other neurons through synapses. The varying lengths of dendrites and finite speed of signal propagation vary the time it takes a signal to reach a post-synaptic neuron. Depending on the parameters of interest and the research question one wants to answer, models of varying complexity can be used.

3.0.1 Charge Conservation

The membrane of the neuron acts like and is modeled as a simple capacitor. Embedded in the cell membrane, there are ion channels which allow the passage of one or more types ions. The current of ions induce change in the potential difference across the membrane. This relationship can be expressed as

$$C_m \frac{dV}{dt} = \sum_{\text{all currents}} I \tag{3.1}$$

where C_m is the capacitance of the membrane, V is the potential difference across the membrane ($V_{\text{inside}} - V_{\text{outside}}$), and I represents any current which passes through the membrane.

The choice of what ionic currents to include on the right hand side depends upon the the type of neuron being modelled. If the goal is to create a biologically realistic model of a spiking neuron (as not all neurons spike), that model would need to include a sodium I_{Na} and potassium current I_K, as these two current in combination create voltage spikes. As a neuron receive stimulus, Na^+ ions pass into the cell through open sodium channels, creating small increases of potential, called graded potentials. If a threshold potential is reached, a rush of Na^+ ions enter the cell, thereby causing a sharp increase. The voltage is brought back down by the movement of K^+ ions outside the cell. These two interactions create a spike in the voltage, called an action potential. In addition to the sodium and potassium currents, simple biophysical models frequently include a leak current, I_L. This current is meant to capture the current of any ions the membrane is permeable to, but frequently captures the passage of Cl^- ions across the membrane.

Stimulus of a neuron comes from the synaptic connections made with other neurons. Synapses are directional, so a neuron receive stimulus from any neuron which connects to its dendrites, but not ones which its connected to through its axon terminals. For the purposes of this notation, I_{syn} is used to represent the sum of all synaptic currents. In laboratory experiments, one may wish to characterize a neuron's response to specific current waveforms. This current is designed by the experimenter and injected into the cell body through an electrode. This current is denoted as I_{inj}, the injected current. Our equation now reads

$$C_m \frac{dV}{dt} = I_{Na} + I_K + I_L + I_{inj} + I_{syn}. \tag{3.2}$$

3.1 Goldman-Hodgkin-Katz First Order Approximation

The next question to answer is one chooses to define the currents. Ion channels are not always open, but gated by specific conditions. Channels can be ligand-gated, closed until specific ions or molecules bind to it, or voltage-gated, closed until a threshold voltage is reached. Due to limitations on computational power, simplifications must be made. It would be unrealistic to model every protein in every channel embedded in the membrane, though molecular dynamics experiments do this on a smaller scale. Instead, we focus on what happens to the neuron as a whole.

On either side of the neuron, there are differing amounts of concentration of each ion. This creates a concentration gradient, that when the channels are open, ions move down freely. However, since these ions are charged, there is also movement corresponding to the electrical field the ions experience. For any given ion, the ion flux through the membrane is the sum of these two effects: (1) the concentration gradient due to unequal concentrations of ion on either side of the membrane, and (2) the electric field due to separation of charges across the membrane. The ion flux of a through the membrane is as follows

$$J_a = J_{\text{diffusion}} + J_{\text{electric drift}} = -D\nabla[a] + z\mu[a](-\nabla V) \tag{3.3}$$

where D is a diffusion coefficient, μ is the ion a's mobility, z is the valence of the ion, and V is the voltage across the membrane. To get the ionic current across the membrane, we assume that ions do not interact and the gradient of the voltage is constant. Integrating over the width of the membrane, and multiplying by the charge of the ion gives the Goldman-Hodgkin-Katz (GHK) [41] current equation

$$I = P_a z_a F \frac{z_a F V}{RT} \frac{[a]_i - [a]_o e^{-z_a F V/RT}}{1 - e^{-z_a F V/RT}}. \tag{3.4}$$

Here, the current is written using molar constants, but could easily be changed to ion values $\left(\frac{RT}{zF} = \frac{k_B T}{q}\right)$. P_a is the permeability of the membrane to ion a, which depends on the mobility of the ion, the width of the membrane, and the relative solubility of the ion in the membrane. F is Faraday's constant (96485 C/mol), R is the gas constant (8.314 J/mol·K) and z_a is the valence of the ion. $[a]_i$ and $[a]_o$ denote the intracellular and extracellular concentrations of the membrane, respectively.

Ionic currents can pass into or outside the cell depending on the membrane voltage and when the corresponding ionic channels are open. It is useful to define an equilibrium potential at which membrane current for a given ion equals zero. This voltage is called the Nernst or reversal potential and is found by setting equation 3.4 to zero. The reversal potential for ion a is

$$E_a = \frac{RT}{z_a F} ln \frac{[a]_o}{[a]_i}. \tag{3.5}$$

For the sake of model simplicity, the GHK equation is rarely used to model currents. Instead, a first order approximation to 3.4 is used, expanding around the Nernst potential

$$I = P \frac{z^2 F^2}{RT} \frac{[a]_o \cdot [a]_i}{[a]_i - [a]_o} ln \frac{[a]_o}{[a]_i} (E_a - V) = \tilde{g}_a (E_a - V) \tag{3.6}$$

where $\tilde{g}_a$ is the variable conductance of the ion across the membrane. For most ion currents,

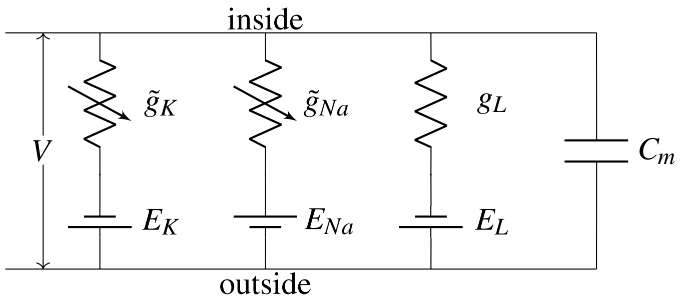

Figure 3.1: A model of a neuron. Here, we model the neuron's membrane as a capacitor and its various ion channels as ionic currents. The batteries labeled E_a are used to represent to reversal potential in equation 3.5. The resistors labeled $\tilde{g}_K$ and $\tilde{g}_{Na}$ represent the conductance of each ion, which can change over time, denoted by the variable resistors. g_L is a constant value because it captures ions passing through the membrane outside of ion channels.

this approximation is acceptable. The ions passing through the membrane are sufficiently small in number as compared to the concentrations inside or outside the membrane, so $[a]_i$ and $[a]_o$ are taken to be constant. This model breaks down when the concentration of an ion is noticeably changed by the current of the ion through the membrane. In these instances, a model of the full GHK is appropriate.

Figure 3.1 shows the model of a neuron in detail, using the approximations from equation 3.6, excluding currents from external sources. $\tilde{g}_{Na}$ and $\tilde{g}_K$ are modeled as variable resistors because ion channels open and close, changing to overall conductance over time. The full equation representing our model in figure 3.1 is

$$C_m \frac{dV}{dt} = \tilde{g}_{Na}(E_{Na} - V) + \tilde{g}_K(E_K - V) + g_L(E_L - V). \tag{3.7}$$

Figure 3.1 is an example of a single-compartment model, meaning we neglect the geometry of the neuron. However, incorporating the geometry of a neuron may be necessary to preserve neuron spiking behavior. In these cases, we must use a multi-compartment model. Frequently, multi-compartment models are used to capture behavior localized to dendrites. In these instances, one or multiple dendritic compartments are connected to a somatic compartment. An example of

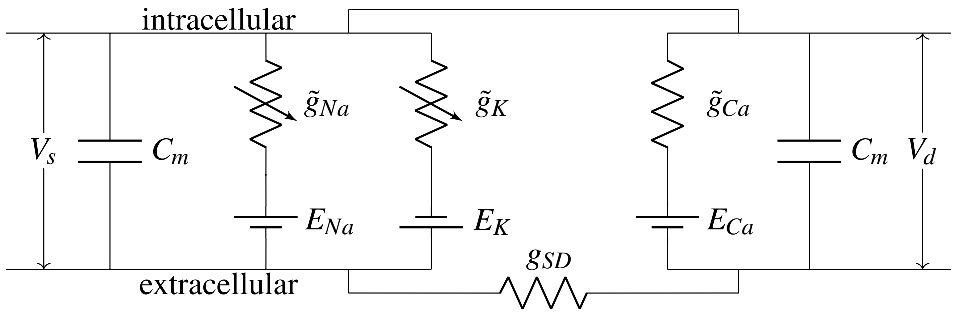

Figure 3.2: A two compartment neuron model. The left compartment corresponds to the soma of the neuron, where a sodium (Na^+) and a potassium (K^+) current are modeled. The right compartment corresponds to a a dendrite where a calcium (Ca^{2+}) current is modeled. The soma and dendrite are connected ohmically, with a constant conductance g_{SD}.

what a multi-compartment model might look like is shown in figure 3.2.

3.1.1 Hodgkin-Huxley Model

This dissertation will mainly focus on the usage of biophysical single-compartment models. While there are many neuron models which can empirically recreate the properties of a neuron, the usage of biologically realistic models can give us insight into how neural networks operate.

All neuron models in this dissertation are based off a relatively simple biophysical model, the Hodgkin-Huxley (HH) model. This model was developed by Alan Hodgkin and Andrew Huxley in 1952 through their study of a giant squid neuron [50].

The general form of a HH current is

$$I_a(t) = g_a m_a^{integer1}(t) h_a^{integer2}(t)(E_a - V(t)), \tag{3.8}$$

where E_a is the equilibrium Nernst potential [54, 98]. The gating variables $\{m(t), h(t)\}$ lie between zero and one and represents the probability of an ion channel to be open and inactivated, respectively. $m(t)$ is an activating gating variable which describes the opening of the ion channel. As $m(t)$ increases, the corresponding ionic current will also increase for a constant potential. $h(t)$

is an inactivating gating variable. As $h(t)$ increases, the corresponding ionic current will decrease for a constant potential. $h(t)$ does not describe the return of an ion channel to its closed or deactivated state, but instead describes the blocking of ions through the channel while open. The integer exponents to these gating variables are found by fitting data and their physical meaning is unclear. g_a represent the maximal conductance, which corresponds to when all ion channels are open. g_a is related to the overall conductance, $\tilde{g}_a$ by $\tilde{g}_a = g_a m_a^{interger1}(t) h_a^{integer2}(t)$.

The gating variables in the HH have their own time dependence and corresponding differential equations. Gating variable equations take the form of

$$\frac{dw(t)}{dt} = \alpha_w(V)(1-w) + \beta_w(V)w \tag{3.9}$$

where $w(t)$ is a place holder for the gating variables $\{m(t), h(t)\}$, $\alpha_w(V)$ is the channel opening rate constant, and $\beta_w(V)$ is the closing channel rate constant. $\alpha(V)_w$ and $\beta_x(V)$ are exponential and sigmoidal functions which can be difficult to work with numerically. Instead, it is easier to define these functions in terms of the gating variable steady state value, $w_\infty(V)$ and a relaxation time $\tau_w(V)$. These kinetic variables are expressed in terms of rate constants using $w_\infty(V) = \frac{\alpha_w(V)}{\alpha_w(V)+\beta_w(V)}$ and $\tau_w(V) = \frac{1}{\alpha_w(V)+\beta_w(V)}$. The full equations for the gating variables are

$$\frac{dw(t)}{dt} = \frac{w_\infty(V) - w(t)}{\tau_w(V(t))} \tag{3.10}$$

$$w_\infty(V) = \frac{1}{2}\left[1 + \tanh\left(\frac{V-\theta_w}{dV_w}\right)\right] \tag{3.11}$$

$$\tau_w(V) = t_{w,1} + t_{w,2}\left[1 - \tanh^2\left(\frac{V-\theta_{w,t}}{dV_{w,t}}\right)\right] \tag{3.12}$$

where the functions for $w_\infty(V)$ and $\tau_w(V)$ are approximations to the sigmoidal functions that match for the range of the neuron's behavior. The constants θ_w, dV_w, $\theta_{w,t}$, and $dV_{w,t}$ are chosen with this in mind.

Now that the general form of Hodgkin-Huxley like neurons has been established. We can write out the full Hodgkin-Huxley model for the giant squid axon

$$\frac{dV}{dt} = \frac{1}{C_m}\left[g_{Na}m^3h(E_{Na}-V)+g_Kn^4(E_K-V)+g_L(E_L-V)+I_{ext}\right] \tag{3.13}$$

$$\frac{dm}{dt} = \frac{m_\infty(V)-m(t)}{\tau_m(V)} \tag{3.14}$$

$$\frac{dh}{dt} = \frac{h_\infty(V)-h(t)}{\tau_h(V)} \tag{3.15}$$

$$\frac{dn}{dt} = \frac{n_\infty(V)-n(t)}{\tau_n(V)}, \tag{3.16}$$

where I_{ext} is any current received from outside the neuron. This could be an injected current by an experimentalist, or a synaptic current from another neuron. The gating variable $n(t)$ is the activating gating variable for the potassium current, commonly labeled as $n(t)$. Keeping consistent with the form presented in equation 3.8, $n = m_K$ where $integer1_K = 4$.

Chapter 4

Birdsong System

The focus of this chapter is using Data Assimilation to characterize neurons in the avian song system of the zebra finch. The avian song system is an attractive model to study because (1) juvenile zebra finch learn their song through auditory feedback [27, 16, 36, 73], making this system a good model for learning complex behavior [36, 101, 96] and (2) the manner in which songbirds learn and reproduce their songs is analogous to the learning of speech in humans [32]. Zebra finch, in particular, are an attractive organism to study because they sing a single short, stereotyped motif for their entire adult lives. Unlike other song birds species which can learn a multitude of songs throughout their adult lives, the learning period for a zebra finch is limited to the first year of its life, after which it does not significantly change its learned song[20].

Present research on this system involves the investigation of both single cell and network dynamics and function responsible for learning and production of songs in zebra finch. Previous studies have demonstrated that single neuron membrane dynamics play an important role in encoding the zebra finch song [53]. This chapter will focus on these single neuron dynamics.

The song system starts with auditory pathways. This is where acoustic stimulation in the ear is transformed into neuron spiking. Beyond the auditory pathway, two neural pathways are principally responsible for song acquisition and production in zebra finch. The first is the

Anterior Forebrain Pathway (AFP) which modulates learning. The second is a posterior pathway responsible for directing song production: the Song Motor Pathway (SMP) [16, 73, 77]. The HVC nucleus in the avian brain uniquely contributes to both of these [73].

There are two principal classes of projection neurons which extend from HVC: neurons which project to the SMP pathway and $\mathrm{HVC_X}$ neurons extend to the AFP [73, 38]. $\mathrm{HVC_{RA}}$ neurons connect to the robust nucleus of the arcopallium ($\mathrm{HVC_{RA}}$) a brain region which indirectly stimulates the vocal respiratory muscles. $\mathrm{HVC_X}$ neurons project to Area X, believed to be responsible for inter-syllable song variability, which has been demonstrated to play a role in adult learning [57].

These two classes of projection neurons combined with classes of HVC interneurons ($\mathrm{HVC_I}$), make up the three broad classes of neurons within HVC. Figure 4.1 [17] displays these structures in the avian brain.

In order to characterize these neuron classes, single neuron data must be obtained. *In vitro* (outside the living organism) or *in vivo* (inside the living organism) observations of neurons can be obtained through patch-clamp techniques making intracellular voltage recordings. In this patch clamp technique, a pipette attaches to the membrane of the neuron and suction is applied until the membrane tears. This allows the pipette to have access to the inside of the whole cell. Within this pipette is a electrode capable of simultaneously injecting current and measuring the whole-cell voltage response [47].

From this data, one can establish the physical parameters of the system [27]. Traditionally this is done using neurochemicals to block selected ion channels and measuring the response properties of others [49]. Single current behavior is recorded and parameters are found through mathematical fits of the data. This procedure has its drawbacks, of course. There are various technical problems with the choice of channel blockers. Many of even the modern channel blockers are not subtype specific [11] and may only partially block channels [39]. Data assimilation can be used to address such problems and has been detailed in chapter 2.

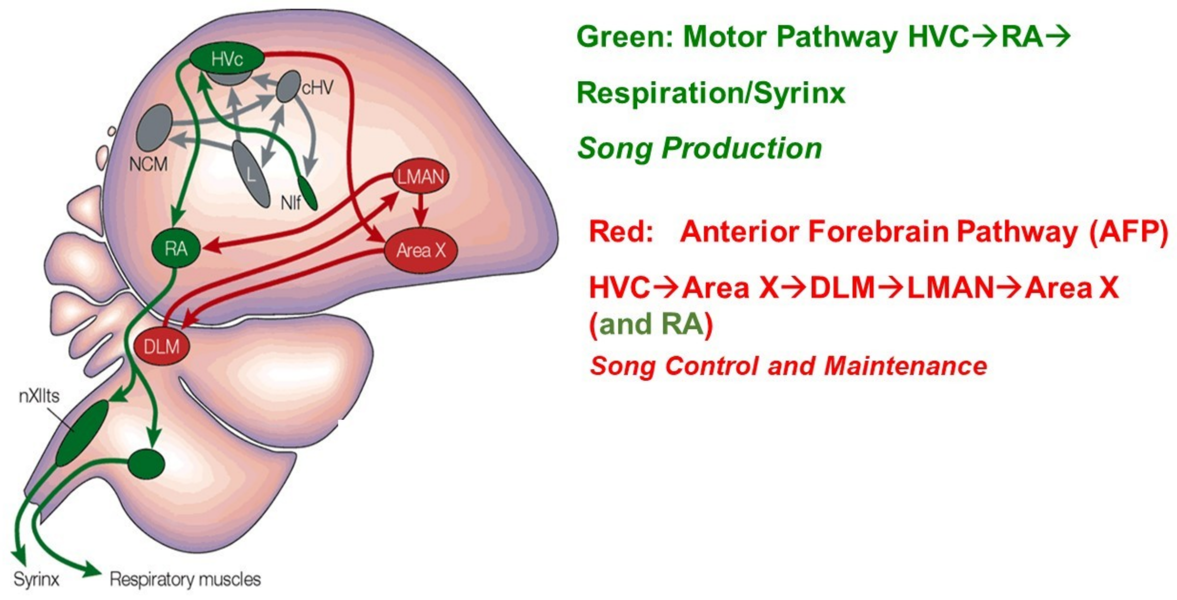
HVc
cHV
NCM
L
NIf
LMAN
RA
Area X
DLM
nXIIts
Syrinx
Respiratory muscles
Green: Motor Pathway HVC→RA→
Respiration/Syrinx
Song Production
Red: Anterior Forebrain Pathway (AFP)
HVC→Area X→DLM→LMAN→Area X
(and RA)
Song Control and Maintenance

Figure 4.1: A drawing of the Song Production Pathway and the Anterior Forebrain Pathway of avian songbirds. Parts of the auditory pathways are shown in grey. Pathways from the brainstem that ultimately return to HVC are not shown. The Anterior Forebrain Pathway is responsible for modulating learning of songs. The Song Production Pathway is responsible for directing song production. The HVC network image is taken from [17].

Data assimilation has previously be used to characterize two of the three neuron classes, HVC_{RA} [55] and HVC_{I} [19]. In this chapter, we turn towards characterizing the last of these three neuron classes, HVC_{X}. The bulk of this chapter is an extension of the work in [27]. This chapter is organized as follows: First, the HVC_{X} model is presented in detail in section 4.1.1. Next, results of applying precision annealing methods on twin experiments and measured laboratory data are located in sections 4.2.1 and 4.2.2 Lastly, the results of applying nudging methods on measured data of multiple neurons is presented in section 4.2.3.

4.1 Methods

4.1.1 HVC$_X$ Neuron Model

The equations describing the HVC$_X$ neuron dynamics are heavily based off the work in [27]. These equations are of Hodgkin-Huxley (HH) form for a neuron without spatial considerations, or a single-compartment model as explained in chapter 3. The model does not consider synaptic input to the neuron and only captures activity induced by the experimenter through an injected current.

The dynamical variables include the observable quantities: voltage across the cell membrane, $V(t)$ and the intracellular concentration of $[Ca^{2+}]_{in}(t)$. $V(t)$ is directly connected to action potentials or voltage spikes that communicate among cells in a network; the time scale of these spikes is a few ms. The voltage recordings are obtained through *in vitro* patch clamp measurements, detailed in section 4.1.2. $[Ca^{2+}]_{in}(t)$ provides a slow background modulation that raises the cells potential (depolarizes the cell) or lowers it (hyperpolarizes the cell) on time scales as long as 10's of ms. Changes in calcium concentrations are measurable by injecting a dye into neurons which fluoresces in response to a two photon laser near the infrared spectrum. A series of images can be taken and this imaging data can be using to infer changes in the calcium concentration and neuron activity [88]. Although, it is possible to obtain this data, the calcium concentration was not used in the analysis detail in this chapter.

The voltage equation relates the capacitance of the cell membrane C_m as it separates concentrations of ions within and without the cells to the various currents which contain the nonlinear voltage dependence of the permeability of ions to passing into and out of the cell. The model represents these ion currents: $\{Na^+, K^+, Ca^{2+}\}$ in several different ways.

For our choice of ion currents we follow the results of experimental data [33, 72, 61] and generally reproduce the model listed in [27]. These models were developed based on the upon the literature and properties found in neuron spiking. HVC$_X$ spiking properties include fast rectifying current, sag in response to hyperpolarizing current, and spike frequency adaption in response to depolarizing current. Each of these properties can be explained by specific ion channel types.

$$
\begin{aligned}
C\frac{dV(t)}{dt} &= I_{Na}(t) + I_K(t) + I_L(t) + I_{CaT}(t) + I_{CaL}(t) \\
&\quad + I_A(t) + I_{SK}(t) + I_h(t) + I_{Nap}(t) + I_{injected}(t)
\end{aligned}
\tag{4.1}
$$

$I_{Na}(t)$ and $I_K(t)$ are the standard HH currents we saw before in chapter 3. The combination of these two currents is responsible for the majority of spiking behavior. $I_L(t)$ is a leak current meant to capture the behavior of all leaky currents in the neuron. $I_{CaT}(t)$ is a low threshold T-type calcium current that causes rebound depolarization in cooperation with $I_h(t)$. $I_h(t)$ is activated during hyperpolarization and responsible for the observed voltage sag in response to a hyperpolarizing current. $I_{CaL}(t)$ is a high threshold L-type calcium current. $I_{CaL}(t)$ works in conjunction with $I_{SK}(t)$, a calcium concentration dependent potassium current, to create shifting frequencies in neuron spiking (frequency adaptation). $I_A(t)$ is an A-type potassium current. It is different from the other potassium current I_K in that I_K is considered a delayed-rectifier current which slowly inactivates the neuron, while I_A is a fast inactivating current. $I_{Nap}(t)$ is a persistent sodium current. Unlike I_{Na}, the persistent sodium current doesn't significantly inactivate, creating a long duration current into the neuron [56]. From the model presented in [27], we eliminate $I_{KNa}(t)$, a sodium dependent potassium current, and rewrite all sigmoidal functions as hyperbolic tangents.

Chapter 3 details the approximations we make to establish the general form of a Hodgkin-Huxley neuron. Most currents in this neuron can be modelled in this fashion. However, this is not so for the Ca^{2+} currents. The intracellular concentration of Ca^{2+} within the cell is on the order of $10-100$ nM. During a voltage spike 10^6 ions pass into the cell, noticeably changing the calcium concentration. This change in intracellular calcium concentration breaks one of the assumptions detailed on chapter 3 for Hodgkin-Huxley approximations to currents. The use of the Goldman-Hodgkin-Katz (GHK) form for all calcium currents in necessary. We rewrite the GHK form below,

$$I_a(t) = -P_a z_a^2 F^2 \left(\frac{[a]_{in}(t) - [a]_{out} e^{-z_a FV(t)/RT}}{1 - e^{-z_a FV(t)/RT}} \right), \tag{4.2}$$

for ion a. z_a is the charge on the ion, F the Faraday constant, R the gas constant, T the temperature, and P_a the permeability of the cell membrane to ion a.

The two types of calcium currents written in GHK form are

$$I_{CaL}(t) = g_{CaL} V s_\infty^2(V) \left(\frac{[Ca]_{out}}{e^{2FV(t)/RT} - 1} \right) \tag{4.3}$$

$$I_{CaT}(t) = g_{CaT} V(t) [a_T]_\infty^3(V) [b_T]_\infty^3(r_T) \left(\frac{[Ca]_{out}}{e^{2FV(t)/RT} - 1} \right) \tag{4.4}$$

$$b_{T_\infty}(r_T) = \frac{1}{1 + e^{\left(\frac{r_T - \theta_b}{\sigma_b}\right)}} - \frac{1}{1 + e^{\left(\frac{-\theta_b}{\sigma_b}\right)}} \tag{4.5}$$

$$\tau_{r_T}(V) = \tau_{r_0} + \frac{\tau_{r_1}}{1 + e^{\left(\frac{V - \theta_{r_T}}{\sigma_{r_T}}\right)}} \tag{4.6}$$

where a_T and b_T are instantaneous activating and inactivating gating variables, respectively. r_T is a slow gating variable whose form is consistent with previously described gating variable form, $w(t)$. The forms presented here for the calcium currents are not quite GHK form, instead the $[a]_{in}(t)$ term is dropped. These gating variables $w(t)$ satisfy a first order kinetic equation, equation 3.10. Unlike the form detailed in equation 3.11, $w_\infty(V)$ will be expressed in a different manner shown below

$$w_\infty(V) = \frac{1}{2}\left[1 - \tanh\left(\frac{V - \theta_w}{2\sigma_w}\right)\right]. \tag{4.7}$$

All gating variables except $h_\infty(V)$, appearing in $I_{Na}(t)$, follow this functional form [27]. Instead, $h(t)$ uses the rate based kinetic form described in equation 3.9, with the rates displayed in equations 4.8-4.9. θ_w is the half-activation voltage and σ_w controls the slope of the activation function. The forms detailed in equations 3.11 and 4.7 are functionally identical. The parameters in 3.11 have been scaled and the anti-symmetric nature of tanh() has been taken advantage of to detail a simplified form.

$$\alpha_h(V) = C_{\alpha h} e^{\left(\frac{V-\theta_{\alpha h}}{\sigma_{\alpha h}}\right)} \tag{4.8}$$

$$\beta_h(V) = \frac{C_{\beta h}}{1 + e^{\left(\frac{V-\theta_{\beta h}}{\sigma_{\beta h}}\right)}} \tag{4.9}$$

For fast gating variables, such as $m(t)$ of I_{Na}, $mp(t)$ of I_{Nap} (equation 4.11), $s(t)$ of I_{CaL}, and $a(t)$ of I_A (equation 4.12), we replace the time-dependent equations their behavior after a sufficiently long time, $w_\infty(V(t))$. The time constants, $\tau_w(V)$, for n and hp (corresponding to the activating and inactivating gating variables for I_K and I_{Nap}, respectively) are given below, where $\bar{\tau}_w$ is an average time constant.

$$\tau_w(V) = \frac{\bar{\tau}_w}{\cosh\left(\frac{V-\theta_w}{2\sigma_w}\right)} \text{ for n or hp} \tag{4.10}$$

$$I_{Nap} = g_{Nap} mp_\infty(V) hp(E_{Na} - V) \tag{4.11}$$

$$I_A = g_A a_\infty(V) e(E_K - V) \tag{4.12}$$

$$\tag{4.13}$$

The Ca^{2+} gated potassium current and mass conservation equation for Ca^{2+} are given below.

$$I_{SK} = g_{SK} k_\infty [Ca]_{in}(t)(E_K - V) \tag{4.14}$$

$$k_\infty([Ca^{2+}]_{in}(t)) = \frac{[Ca^{2+}]_{in}^2(t)}{[Ca^{2+}]_{in}^2(t) + k_s^2} \tag{4.15}$$

$$\frac{d[Ca^{2+}]_{in}(t)}{dt} = \phi[\varepsilon(I_{CaT}(t) + I_{CaL}(t)) + k_{Ca}(b_{Ca} - [Ca^{2+}]_{in}(t))] \tag{4.16}$$

$k_\infty([Ca^{2+}]_{in}(t))$ is the steady-state activation function based on on the intracellular calcium concentration, and k_s is a dissociation constant which measures the rate of calcium unbinding from the potassium channel (i.e., no longer activating the potassium channel). The constant ϕ represents the fraction of free Ca^{2+} ions, whereas ε is a constant which combines the effects of buffers, cell volume, and molar charge [27]. K_{Ca} is a rate constant related to calcium pumps and b_{Ca} is the basal level of Ca^{2+} inside the neuron.

The last current in the model is I_h. It is comprised of a fast component, r_f and slow component r_s. Both components obey the gating variable differential form in equation 3.10. I_h and the associated time constants can be seen below,

$$I_h = g_h \left[k_r r_f + (1 - k_r) r_s\right] (E_h - V) \tag{4.17}$$

$$\tau_{rs}(V) = \tau_{rs_0} + \tau rs_1 \left(1 - \tanh^2\left(\frac{V - \theta_{trs}}{\sigma_{trs}}\right)\right) \tag{4.18}$$

$$\tau_{rf} = \frac{p_{r_f}}{\frac{-7.4\left(V - \theta_{trf_1}\right)}{e^{\frac{V - \theta_{trf_1}}{-0.8}} - 1} + 65 e^{\frac{V - \theta_{trf_2}}{-23}}}, \tag{4.19}$$

where k_r is a tuned parameter which controls the sag seen in the hyperpolarized region.

Table 4.1: Parameter values used to numerically generate the HVC$_X$ data. The source of these values comes from [27]. Data was generated using an adaptive Runge-Kutta method, and can be seen in Fig. (4.2A) and Fig. (4.2B).

Parameter	Value	Parameter	Value	Parameter	Value
g_{Na}	450 nS	g_L	2 nS	k_r	0.3
E_{Na}	50 mV	E_L	-70 mV	θ_{mp}	-40 mV
g_K	50 nS	g_{Nap}	1 nS	σ_{mp}	-6 mV
E_K	-90 mV	g_{CaL}	19 nS	θ_s	-20 mV
g_{CaT}	2.65 nS	θ_m	-35 mV	σ_s	-0.05 mV
g_{SK}	6 nS	σ_m	-5 mV	θ_{hp}	-48 mV
g_h	4 nS	θ_n	-30 mV	σ_{hp}	6 mV
E_h	-30 mV	σ_n	-5 mV	$\bar{\tau}_{hp}$	1000 ms
C_m	100 pF	$\bar{\tau}_n$	10 ms	θ_e	-60 mV
$\theta_{\alpha h}$	- 15 mV	$\sigma_{\alpha h}$	-18 mV	$C_{\alpha h}$	0.128 ms^{-1}
$\theta_{\beta h}$	-27 mV	$\sigma_{\beta h}$	-5 mV	$C_{\beta h}$	4 ms^{-1}
θ_a	-20 mV	θ_{rf}	-105 mV	σ_e	5 mV
σ_a	-10 mV	σ_{rf}	5 mV	τ_e	20 ms
θ_{rs}	-105 mV	θ_{aR}	-65 mV	θ_b	0.4 mV
σ_{rs}	25 mV	σ_{aT}	-7.8 mV	σ_b	-0.1 mV
θ_{rT}	-67 mV	θ_{rrT}	68 mV	ϕ	0.1
σ_{rT}	2 mV	σ_{rrT}	2.2 mV	ε	0.0015 $\frac{\mu M}{pA\cdot ms}$
τ_{r_0}	200 ms	τ_{r_1}	87.5 ms	p_{r_f}	100
k_{Ca}	0.3 ms^{-1}	b_{Ca}	0.1 μM	k_s	0.5 μM
τ_{rs_0}	0.1 ms	τ_{rs_1}	193 ms	θ_{trs}	-80 mV
σ_{trs}	-21 mV	θ_{trf_1}	-70 mV	θ_{trf_2}	-56 mV
σ_{trf_1}	-0.8 mV	σ_{trf_2}	-23 mV		

4.1.2 Experimental Data

Data was obtained by Arij Daou at the Margoliash lab. A fully detailed procedure for how data is obtained can be seen in [27]. *In vitro* observations of HVC$_X$ neurons have been obtained through brain slice preparation [27]. Brain slice preparation allows for the study of individual neurons or synapses in isolation from the rest of the network. The brain region of interest is sliced and recordings are taken using the patch clamp method described above. A pre-determined current waveform is injected into neurons and the responding membrane voltage is measured.

The pre-determined current waveform is selected on criteria established through successful use of these techniques on other neuron models [70]. In order to obtain good estimation results, we must choose a forcing or stimulus with the model in mind: the dynamical range of the neuron must be thoroughly explored. This suggests a few key properties of the stimulus:

- The current waveform of $I_{injected}(t)$ must have sufficient amplitudes ($\pm$) and must be applied sufficiently long in time that it explores the full range of the neuron variation.
- The frequency content of the stimulus current must be a low enough that it does not exceed the low-pass cutoff frequency associated with the RC time constant of the neuron. This cutoff is typically in the neighborhood of 50-100Hz.
- The current must explore all time scales expressed in the neuron's behavior.

4.1.3 Variational Annealing Twin Experiment on HVC$_X$ Neuron Model

A twin experiment is a synthetic numerical experiment meant to mirror the conditions of a laboratory experiment. We use our mathematical model with some informed parameter choices in order to generate synthetic numerical data. Noise is added to observable variables in the model, here that is $V(t)$. These data are then put through our SDA procedure to estimate parameters and unobserved states of the model. The neuron model is now calibrated or completed.

Using the parameters and the full state of the model at the end t_F of an observation window $[t_0, t_F]$, we take a current waveform $I_{injected}(t \geq t_F)$ to drive the model neuron and predict the time course of all dynamical variables in the prediction window $[t_F, ...]$. This validation of the model is the critical test of our SDA procedure, here PA. In a laboratory experiment we have no specific knowledge of the parameters in the model and, by definition, cannot observe the unobserved state variables; here we can do that. So, 'fitting' the observed data within the observation window $[t_0, t_F]$ is not enough, we must reproduce all states for $t \geq t_F$ to test our SDA methods.

We assume that the neuron has a resting potential of -70 mV and set the initial values for the voltage and each gating variable accordingly. We assume that the internal calcium concentration of the cell is $C_{in} = 0.1\ \mu M$. We use an integration time step of 0.02 ms and integrate forward in time using an adaptive Runge-Kutta method [81]. Noise is added to the voltage time course by sampling from a Gaussian distribution $\mathcal{N}(0,2)$ in units of mV.

The waveform used to generate the data was chosen to have three key attributes: (1) It is strong enough to cause spiking in the neuron, (2) it dwells a long time in a hyperpolarizing region to capture the effects of I_H, and (3) its overall frequency content is low enough to not be filtered out by the neuron. On this last point, a neuron behaves like an RC circuit, it has a cut off frequency limited by the time constant of the system. Any input current which has a frequency higher than that of the cut off frequency won't be 'seen' by the neuron. The time constant is taken to be the time it takes to spike and return back to 37% above its resting voltage. We chose a current where the majority of the power spectral density exists below 50 Hz.

The first two seconds of our chosen current waveform is a varying hyperpolarizing current. In order to characterize I_h(t) and $I_{CaT}(t)$, it is necessary to thoroughly explore the region where the current is active. $I_h(t)$ is only activated when the neuron is hyperpolarized. The activation of $I_h(t)$ de-inactivates $I_{CaT}(t)$, thereby allowing its dynamics to be explored. Our waveform reaches up to 300 pA for brief periods of time as this is the level needed to cause spiking for the short duration. This spiking is necessary to characterize $I_{Na}(t)$ and $I_K(t)$.

The parameters used to generate the data are located in Table (4.1). These starting parameters are taken from many laboratory experiments performed on these neurons. The collection of all such parameters is listed in [27]. The injected current and noisy generated membrane voltage response may be seen in figures Fig. (4.2A) and Fig. (4.2B).

In order to mimic the experiments performed in [27], we chosen to estimate the maximal conductance values and reversal potentials of key ionic currents. While these maximal conductance values are less known, reversal potential for a given neuron are fairly well defined. The estimate of these potentials functions as another way to evaluate the success of our methods in parameter estimation. We also the capacitance of the neuron. More specifically, we instructed our

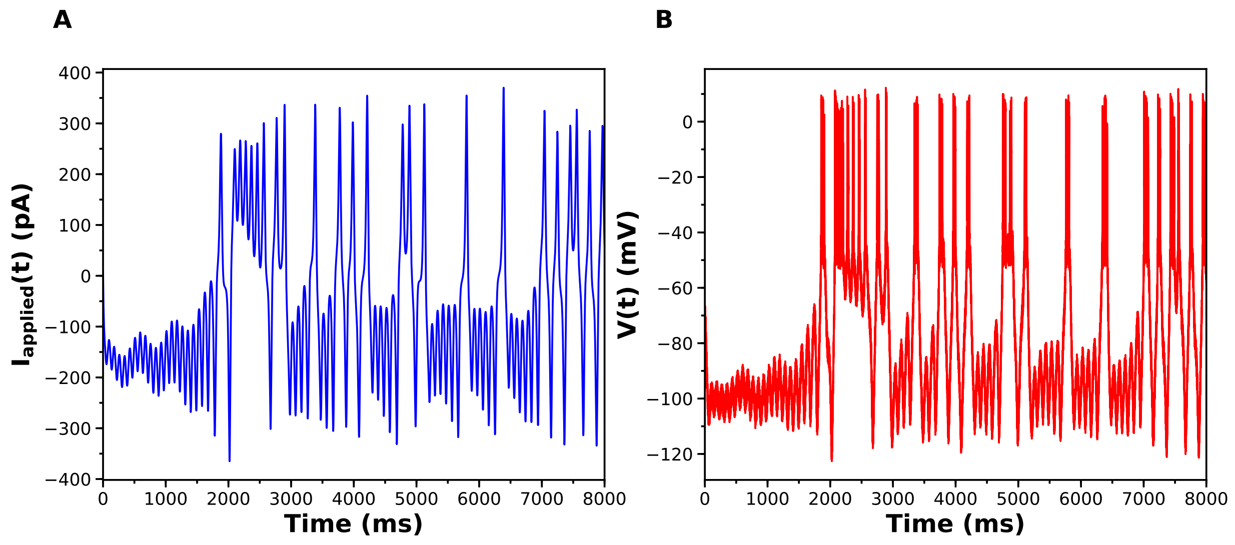

Figure 4.2: Generated Data of HVC using the model described and the current shown. (A) The current 'injected' into the neuron. (B) The voltage data of a neuron after integrating the model using and adaptive Runge-Kutta method. Noise of $\mathcal{N}(0, 1 \text{ mV})$ is added to each data point to emulate the noise expected in the experiment.

methods to estimate the inverse capacitance and estimate the ratio $g' = \frac{g}{C_m}$ instead of g and C_m independently. Estimating g and C_m separately can present a challenge to numerical procedures.

For all state variables and parameter estimates, we set bounds on the possible values. These values are informed partially based on model design, for example, state variables are designed to only take values within the range $[0,1]$, and partially due to biological experiments. We know roughly what is an acceptable maximal conductance for each ionic current from multiple biological experiments meant to measure these parameters. We also know the voltage ranges are neuron is capable of exploring before 'frying' it.

The numbers chosen for the data assimilation procedure in this paper are $\alpha = 1.4$ and β ranging from 1 to 80. $R_{f,0,V} = 10^{-4}$ for voltage and $R_{f,0,j} = 1$ for all gating variables. These numbers are chosen because we are not normalizing the data in any way and the voltage range is 100 times large than the gating variable range. Choosing a single $R_{f,0}$ value would result in the gating variable equations being less enforced than the voltage equation by a factor of 10^4. The α and β numbers are chosen because we seek to make $\frac{R_f}{R_{f0}}$ sufficiently large. $N_I = 25$ paths are initialized. The voltage estimates are initialized to their measured values and unmeasured state variables are sampled from a uniform distribution over the allowed chosen ranges.

Within our computational capability we can reasonably perform estimates on 50,000 data points. This captures a second of data when $\Delta t = 0.02\,\mathrm{ms}$. However, there are time constants in the model neuron which are on order 1 second. In order for us to estimate the behavior of these parameters accurately, we need to see multiple instances of the full response, and therefore need a window on the order of 2-4 seconds. We can obtain this by downsampling the data. We know from previous results that downsampling can lead to better estimations [18]. We take every ith data point, depending on the level of downsampling we want to do. In this twin experiment, we downsampled by a factor of 4 to incorporate 4 seconds of data in the estimation window.

Optimization was performed by an interior point optimization algorithm included in an open-source software called IPOPT (ver 3.12.8) [106]. IPOPT allows for the choice of linear solver. We used HSL ma97 linear solver [1]. For each N_I paths, a minima is found at every β value.

4.1.4 Variational Annealing on HVC$_X$ Neuron Data

Using the procedure detailed above, we proceeded to assimilate the voltage data obtained from zebra finch HVC$_X$ to the model described in 4.1.1. We used all the same tuned hyper-parameters. However, applying the same downsampling and time window resulted in very long convergence times and poor estimations. As a result, we decreased the the number of data points and changed the downsampling factor to 3 instead of 4, resulting in estimation on a shorter time window. Instead of 50000 data points spanning 4.0 s, we used 40000 data points spanning 2.4s. All other hyper parameters were kept the same. The injected current and measured voltage from one HVC$_X$ neuron data can be seen in figure 4.3. All other parameters were kept the same.

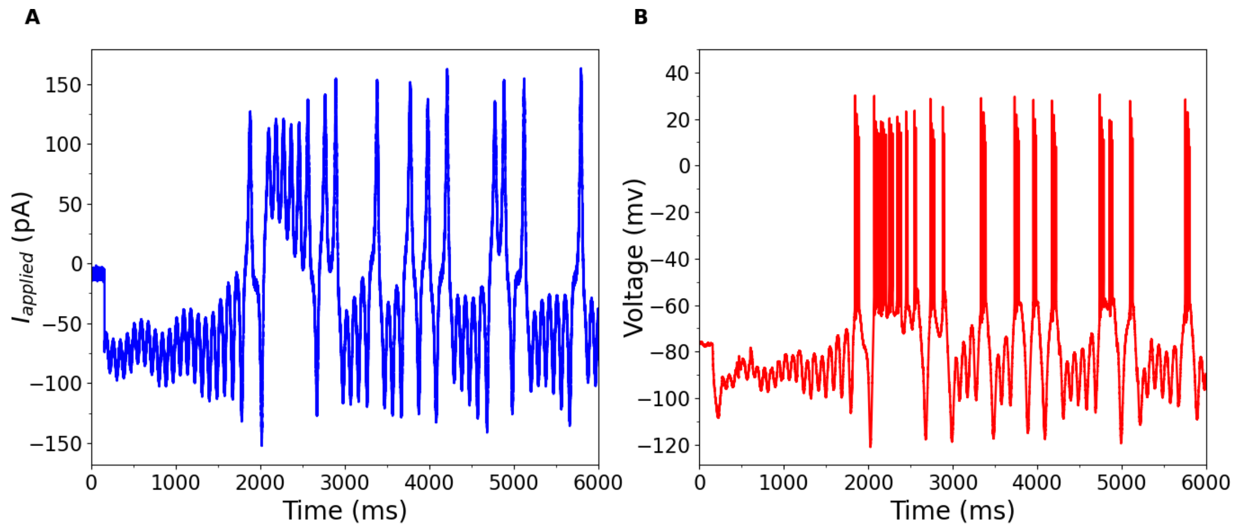

Figure 4.3: Data Recorded *in vitro* of a HVC_X neuron. Brain slices were taken from the HVC in Zebra finch. Current clamp experiment was performed where a pre-designed current waveform was injected into the neuron (A). The resulting voltage response was measured at a 50kHz sampling rate (B).

4.1.5 Nudging Methods HVC_X Neuron Data

In the field of optimization, there is a theorem called the ”no free lunch theorem”, which, when broadly translated, means there is no one optimization method to rule them all. In other words, if an optimization performs well on one class of problems, it is likely at the expense of performance on another class of problems. There are no shortcuts. With that motivation, we employ the “nudging” method described in equation 2.17 on the same HVC_XX problem. In this section we use data from multiple HVC_X neurons, using a variety of different current waveforms.

The baseline characteristics of an ion channel are set by the properties of the cell membrane and the complex proteins penetrating the membrane forming the physical channel. Differences among birds would then come from the density or numbers of various channels as characterized by the maximal conductances. If such differences were identified, it would promote further investigation of the biologically exciting proposition that these differences arise in relation to some aspect of the song learning experience of the birds [26].

The questions we asked was whether we could, using SDA, identify differences in biophys-

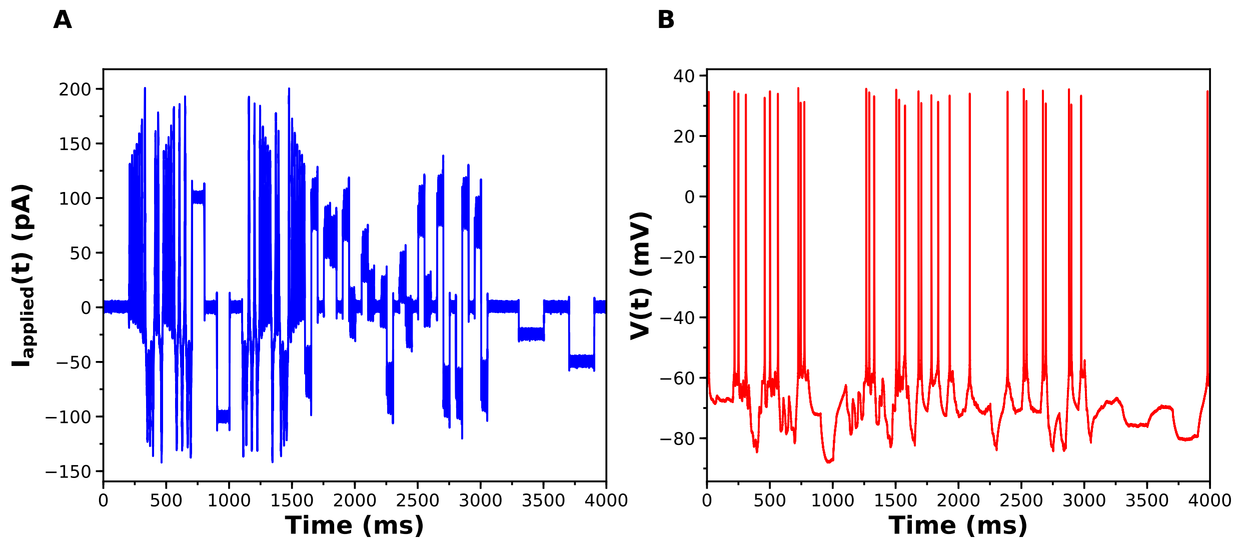

Figure 4.4: **(A)** One of the library of stimuli used in exciting voltage response activity in an HVC$_X$ neuron. The cell was prepared *i*n vitro, and a single patch clamp electrode injected Iinjected(t) (this waveform) and recorded the membrane potential. **(B)** The voltage response. One of the library of stimuli used in exciting voltage response activity in an HVC$_X$ neuron. The cell was prepared in vitro, and a single patch clamp electrode injected $I_{\text{injected}}(t)$ (this waveform) and recorded the membrane potential.

ical characteristics of the birds. This question is motivated by prior biological observations. Using the same HVC_X model as before, we estimated the maximal conductances $\{g_{Na}, g_K, g_{CaT}, g_{SK}, g_h\}$ while holding other parameters constant.

We estimated the parameters of nine total neurons, four belonging to one bird, and five belonging to another non-sibling bird. Each neuron received multiple different waveforms designed according to the criteria laid out in section 4.1.2. Figure 4.4 shows an example of one such waveform. Within the lab, each current waveform was presented to each neuron multiple times. For every current and voltage response measurement, our nudging SDA method was applied. The time steps chosen for each estimation windowed dependended on the the current. However, estimation was performed on windows encompassing 1 - 3 seconds of data with no downsampling. Each run started with the forcing parameter at $u = 100$ with $N_I = 5$ initial paths. Like with variational annealing, the initial guesses for the state and parameter estimates are sampled from a uniform distribution over the bounds allowed chosen ranges.

4.2 Results

4.2.1 Variational Annealing on a Twin Experiment of HVC_X

The results of this experiment can be seen in figure 4.5. Here we present a plot of the action as a function of $\beta = \log[R_f/R_{f0}]$. As explained in chapter 2, we are looking for the value of the action to level off as a function of R_f. This leveling off is an indication that our model estimate reflects the dynamics of the system [108]. Once this has been established, we can then explore how well our parameter estimations perform when integrating them forward as predictions of the calibrated model. Looking at the action plot in figure (4.5), we can see there is a region in which the action appears to level off, around $\beta = 40$. It is in this region where we begin to look for our parameter estimates.

We examine all solutions around this region of β and utilize their parameter estimates in our predictions. We compare our forecast to the "real" data from our synthetic experiment. We choose which paths to predict by first isolating which 5 of the 25 paths corresponded to the lowest action value for each β. We then took the last time step in the estimated path, and used that to predict forward using the estimated parameters in the dynamical equations. This is done for the 5 lowest action paths for each β, using β values ranging from 25 to 45. We focused on this β value range as that is where the action plot levels off. Each one of the 25 initial paths found the same minima. Small differences in parameter estimates between paths can be attributed to the normalized threshold of error we allow to determine when the optimization is complete.

We evaluate how good each prediction is by using calculating the correlation coefficient between estimates and the synthetic data. This metric is chosen instead of a simple root mean square error because slight variations in spike timings yield a high amount of error even if the general spiking pattern is correct. correlation coefficient can account for these variations in spike timing as it is invariant under shifts in the data.

The prediction plot and parameters for the prediction with the highest correlation coeffi-

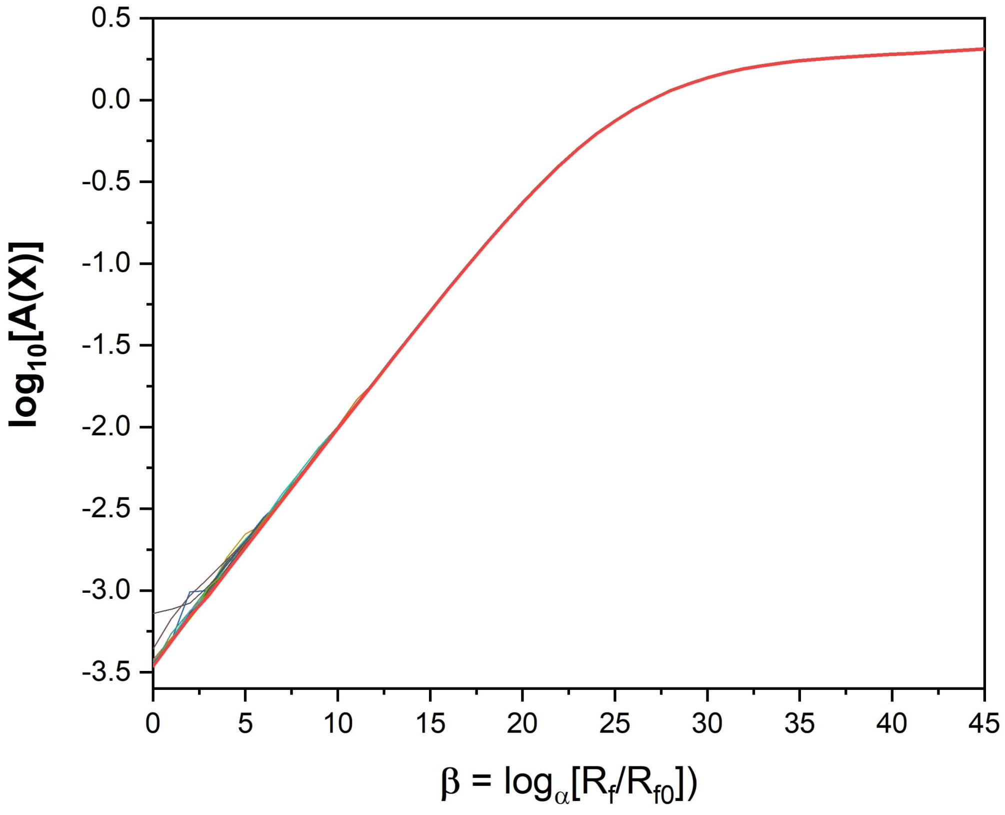

Figure 4.5: Action levels for the HVC$_X$ model twin experiment. We see that the action rises to a "plateau" where it becomes quite independent of R_f. The calculation of the action uses PA with $\alpha = 1.4$ and $R_{f,0} = R_m$. $N_I = 25$ initial choices for the path X_{init} were used in this calculation. For small R_f one can see the slight differences in action level associated with local minima of $A(X)$. The relative leveling off of the action is a sign that the optimization is becoming independent of the model error.

cient can be seen in figure (4.6) and table (4.2). The voltage trace in red is the estimated voltage after data assimilation is completed. It is overlayed on the synthetic input data in black. The blue time course is a prediction, starting at the last time point of the red estimated $V(t)$ trace and using the parameter estimates associated with the estimated path displayed.

The red curve matches the computed voltage trace quite well. There is no significant deviation in the frequency of spikes, spike amplitudes, or the hyperpolarized region of the cell. Looking at the prediction window, we can see that there is some deviation in the hyperpolarized

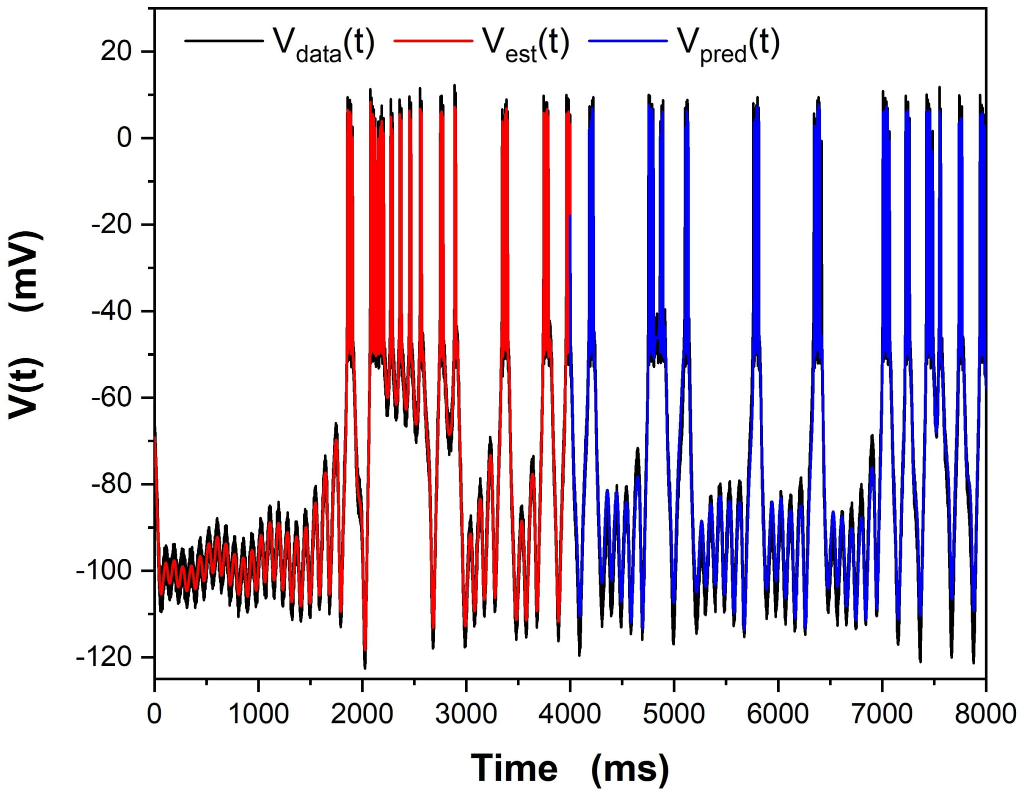

Figure 4.6: Results of the "twin experiment" using the model HVC$_X$ neuron described in section 4.1.1. Noise was added to data developed by solving the dynamical equations. The noisy $V(t)$ was presented to the precision annealing SDA calculation along with the $I_{\text{injected}}(t)$ in the observation window $t_0 = 0$ ms, $t_F = 4000$ ms. From all estimates, we take the path the lowest action (all paths are approximately the same solution) and plot it in red. The noisy model voltage data is shown in black. For $t \geq 4000$ ms we show the predicted membrane voltage, in blue, generated by solving the HVC$_X$ model equations using the parameters estimated during SDA within the observation window.

voltage trace after 7000 ms. Additionally, our our predicted voltage does not become nearly as hyperpolarized as the real data. This is an indication that our parameter estimates for currents activated in this region are not entirely correct.

The estimated parameters are shown in table 4.2. We can see there is a general trend in the way parameters are estimated. The conductances tend to be estimated as higher than their true value, and the reversal potentials tend to be estimated as lower their true values. One exception is g_{SK}, thought to be responsible for spike frequency adaptation. In order for this mechanism to occur, the neuron needs to be stimulated for enough time for the timing between spikes to shift. This is not something we considered in the design of this current, and will need to be considered in future experiment designs.

Table 4.2: Parameter estimates from the path with the 'best' prediction. The best prediction is chosen by finding the highest correlation coefficient between the predicted voltage and 'real' voltage.

Parameter	Bounds	Estimate	Actual Value	Units
g'_{Na}	0.1, 10	4.98	4.5	nS/pF
E_{Na}	1, 100	43.2	50	mV
g'_{K}	0.01, 5	0.907	0.5	nS/pF
E_{K}	-140, -10	-127.4	-90	mV
g'_{CaT}	0.001, 1	0.0326	0.0265	nS/pF
g'_{SK}	0.001, 1	0.0373	0.06	nS/pF
g'_{h}	0.001, 1	0.0432	0.04	nS/pF
E_{h}	-100, -1	-44.1	-30	mV
C_{inv}	0.001, 0.5	0.011	0.01	pF^{-1}

Of particular interest is the parameters surrounding I_h. This current was specifically designed to explore the hyperpolarized region of the neuron in order to accurately estimate this current. We can see that E_h is estimated as being significantly lower than its true value. This may be an indication that the waveform needs adjusting. Previous experiments have shown that a chaotic current is more successful than square wave pulses at characterizing neurons, however, these neuron models did not include currents which are only active where the neuron is hyperpolarized like I_h. Due to its long time constant, a current similar to a square wave pulse may be better than one with small oscillations. Alternatively, it may be the amplitude of these oscillations that are the problem. I_h, at its highest current value, is only 4 pA, whereas our input current oscillates with an amplitude of at least 20 pA in the hyperpolarizing region. Adjusting the current waveform to include smaller amplitude or no oscillations in the hyperpolarizing region may allow us to better estimate the parameters of I_h. Despite these, we still are able to reproduce neuron behavior quite well. All parameters fall within a reasonable physical range and none hit the boundaries we set. This is an indication that variational annealing is capable of parameterizing this model and can be applied to biological data.

4.2.2 Variational Annealing on Biological Data from HVC$_\mathrm{X}$ Neurons

The results of this experiment can be seen in figure 4.7. We can see the action does not level off like we would expect, and what happened in our twin experiment. This is an indication that the estimated path for all state variables is not consistent with the mathematical model and its estimated parameters. We know from section 4.2.1, that our SDA methods are capable of parametrizing the HVC$_\mathrm{X}$ neuron model described in section 4.1.1, provided the model is an accurate representation of the data. This brings into question whether mathematical model is a sufficient representation of the biological neuron.

Figure 4.8 shows our estimates and prediction plots overlayed on the true biological data. We note that the hyperpolarized neuron response matches quite well in both estimation and prediction windows. However, in both the estimation and prediction windows, the neuron responses spikes for too long. The estimated parameters are displayed in table 4.3, along with the parameters we used to generate the data in the twin experiment. The parameters we chose were based on literature, and should be reasonably close to the 'true' values.

Here we can see that both g_{Na} and E_h hit the search boundary. What is interesting to note is that, despite the incorrectly estimated E_h, the voltage response in the hyperpolarized regions are reasonable in the prediction window. This is an indication that the response here is governed

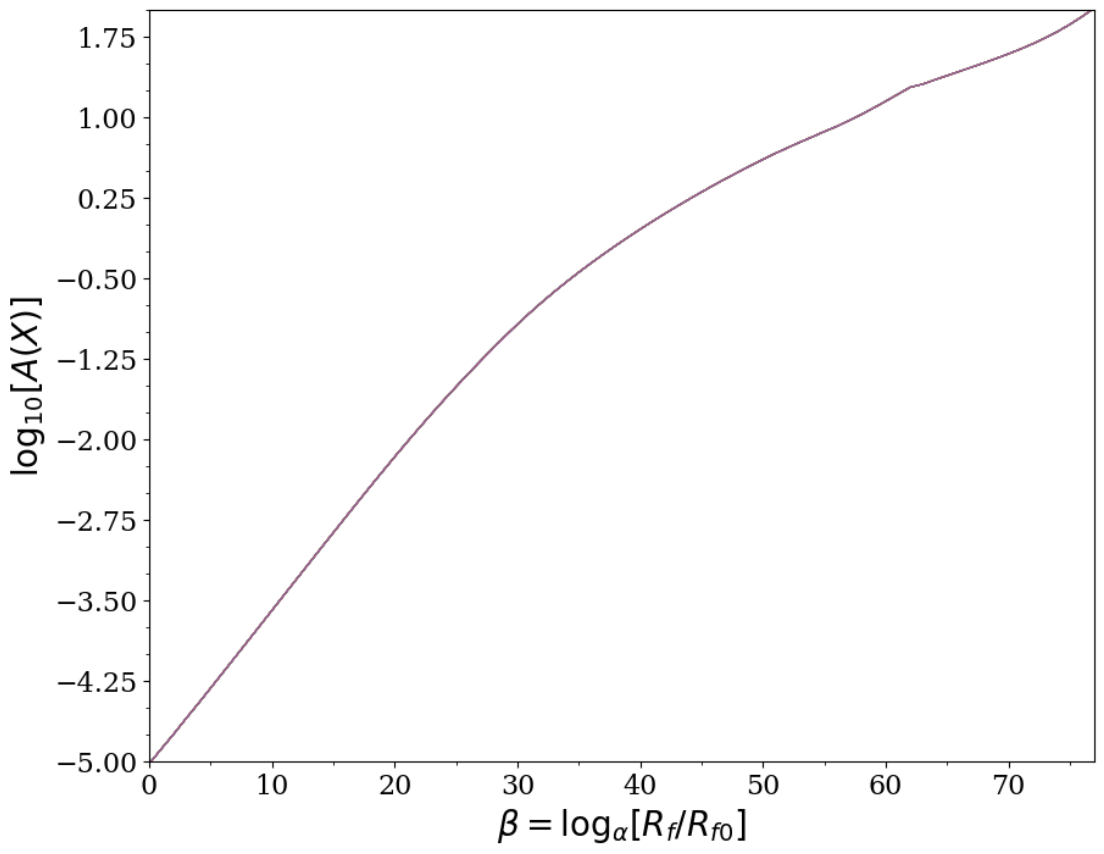

Figure 4.7: Action levels for the HVC_X model using biological data. We don't see the action level off and become independent of R_f. This is an indication that our methods have not found a solution consistent with the equations of our model. The calculation of the action uses PA with $\alpha = 1.4$ and $R_{f,0} = R_m$. $N_I = 25$ initial choices for the path X_{init} were used in this calculation.

by the amplitude of oscillations of the input current, and the effects of I_h are well captured by the current in this region. This is in agreement with what we found in the twin experiment.

Most other parameters appear to deviate significantly from the twin experiment values, most surprising are the parameters associated with I_{Na} and I_K. These two currents are responsible for the majority of spiking behavior.

Generally speaking, increasing g_{Na} can increase the amplitude and shift the timing of spikes. Increasing g_K can narrow the width of spikes and alter the number of burst spikes for short duration stimuli. Figure 4.9 displays the synthetic (red) and biological (blue) neuron response for the same current. We can see that the voltage response in the synthetic data is different than the biological data in a few key ways: (1) within a spike burst, the synthetic data is missing the initial spike, (2) the synthetic data is missing a second burst of spikes at time $t = 2150$ ms, and (3) the synthetic data spikes are wider and have a lower amplitude. This is an indication that our model is wrong in some way. SDA functions by trying to find a parameter set which will make our model reproduce the blue voltage trace. However, we only allow conductances and reversal potentials to vary. If we want to make or synthetic neuron spikes sooner, with a higher amplitude, and narrower spikes, increasing both g_{Na} and g_K are the only ways we are allowed to do so. This is reflected in our parameter estimates, table 4.3. Both g_{Na} and g_K are significantly higher than we would expect. g_{Na} hits the upper bound of our search region, which is an unrealistic value. g_K does not hit the upper bound of our search, but is much larger than previous experiments suggested it would be. The differences in the model displayed in figure 4.9 may account for the the poor parameter estimation.

Another potential source for this model error is the removal of the sodium dependent potassium current. From the model described in [27], we dropped the sodium dependent potassium current, I_{KNa}. This current, in addition to I_{SK}, can be responsible for spike-frequency adaptation. With some guidance from the authors of [27], we chose to keep I_{SK} over I_{KNa} for the sake of model simplicity.

4.2.3 Nudging on Multiple HVC$_X$ Neurons

The main result of this experiment can be seen in figure 4.10. This figure displays the parameter estimates for $\{g_{Na}, g_{CaT}, g_{SK}\}$ in two birds. In this experiment, we ran multiple current waveforms through each of the nine neurons. Each neuron was stimulated with a single waveform more than once. Figure 4.10 displays the best parameter estimates for all nine neurons, corresponding to a single waveform. In this example, the current waveform was run through

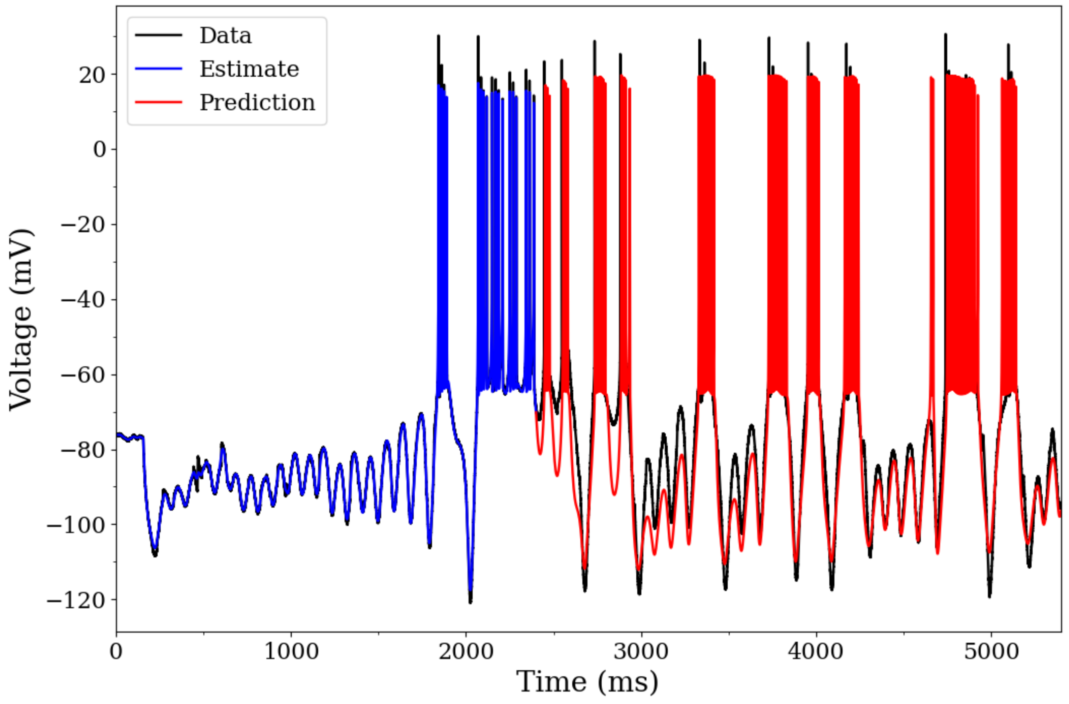

Figure 4.8: Results of the experiment using the model HVC$_X$ neuron described in section 4.1.1 and biological data taken in lab. Noise was added to data developed by solving the dynamical equations. The noisy $V(t)$ was presented to the precision annealing SDA calculation along with the $I_{\text{injected}}(t)$ in the observation window $t_0 = 0$ ms, $t_F = 4000$ ms. The noisy model voltage data is shown in black, and the estimated voltage is shown in red. For $t \geq 4,000$ ms we show the predicted membrane voltage, in blue, generated by solving the HVC$_X$ model equations using the parameters estimated during SDA within the observation window.

each of the nine neurons four times, so there are a total 36 parameter estimates in this plot. The parameter estimates for each run were found by calculating the expected value of each parameter, excluding all paths which had a high final value for the cost function.

The maximal conductances from one bird are shown in blue and from the other bird, in red. There is a striking difference between the distributions of maximal conductances. Neurons from each bird occupy a small but distinct region of the parameter space. This clear separation of parameter estimates was reproduced for multiple different stimuli and is not stimulus specific (not shown). This result and its implications for birdsong learning, and more broadly for neuroscience,

Table 4.3: Parameter estimates from the best prediction. The best prediction is chosen by finding the highest correlation coefficient between the predicted voltage and measured voltage. Also shown are the parameters used to generate the data in the twin experiment. These parameters were chosen from literature surrounding experiments on these neurons, and were believe to be near the ’true’ parameters.

Parameter	Bounds	Estimation	Twin Experiment	Units
g'_{Na}	0.1, 12	12.0	4.5	nS/pF
E_{Na}	1.0,100.0	29.4	50	mV
g'_{K}	0.01, 5	1.45	0.5	nS/pF
E_K	-150.0,-1.0	-130.2	-90	mV
g'_{CaT}	0.001, 1	0.156	0.0265	nS/pF
g'_{SK}	0.001, 1	0.032	0.06	nS/pF
g'_{h}	0.001,1	0.058	0.04	nS/pF
E_h	-100.0,-1.0	-1.0	-30	mV
C_m	0.001, 0.5	0.022	0.01	pF^{-1}

are described in Daou and Margoliash [26].

One limitation of the present result is that the SDA estimates for g_{SK} for a subset of the neurons/observations for Bird One reach the bounds of the observation window. The reasons for this likely similar to what is explained in the previous section: limitations on regions explored by the injected current, discrepancies between the model and governing dynamics of the neuron, and our cost function.

4.3 Conclusions

There are several insights that can be made from the analysis in this chapter related to how we approach the parametrization of neurons.

Section 4.2.1 has shown that variational annealing is capable of parametrizing the HVC_X model. Although our experiment produced reasonable results, more can be done to change the input to better fit this model. Particularly, our twin experiment has shown that refinement is needed to our current inputs. Two things we can change are the inclusion of regions of constant current in both the depolarizing and hyperpolarizing regions. A constant depolarizing current will better

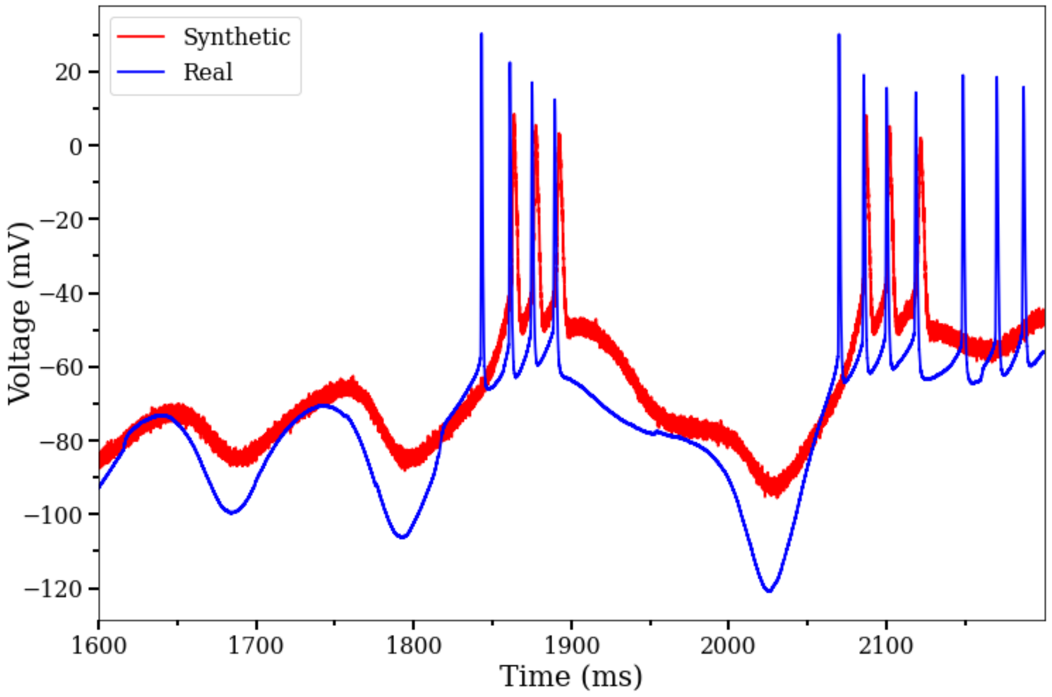

Figure 4.9: A section of the voltage response, overlaying the twin experiment described above and the real neuron response data. One can clearly see the response as different. Our model does not respond to the current with nearly as much amplitude, we have less spikes in a burst, and our model spikes are wider.

allow use to characterize currents associated with spike-frequency adaptation I_{CaT} and I_{SK}. A constant hyperpolarizing current may better allow use to characterize I_h. More generally, we need to make adjustments to our stimulating currents to account for the for these behaviors. A good stimulating current should include all properties described in section 4.1.2, and constant stimuli which both hyperpolarizes and depolarizes the neuron to captures spike-frequency adaptation and sag behavior.

When we applied our variational annealing technique to biological data, it did not produce an accurate forecast of the neuron response. Given our neuron model and successful twin experiment, we know that variational annealing is capable of characterizing a neuron with the same model under the same conditions. The results suggest that the model we chose was wrong.

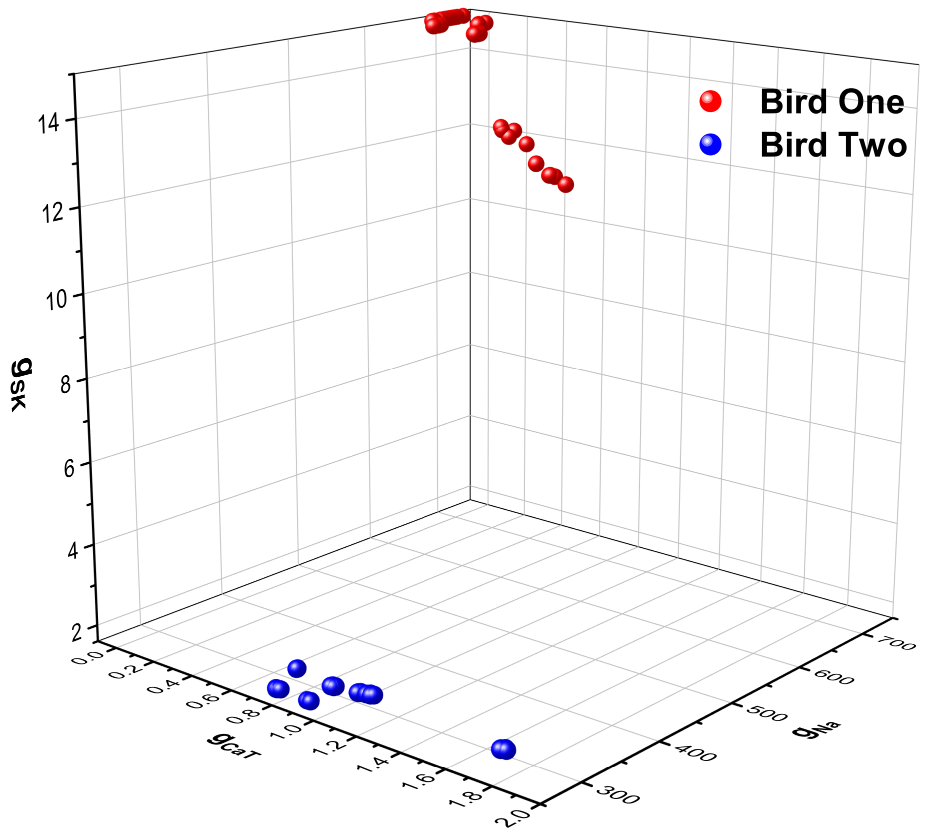

Figure 4.10: A three dimensional plot of three of the maximal conductances estimated from HVC_X cells using the stimulating current shown in Figure 4.4A. Membrane voltage responses from five neurons from one bird were recorded many times, and membrane voltage responses from four neurons from a second bird were recorded many times. One set of maximal conductances $\{g_{Na}, g_{CaT}, g_{SK}\}$ are shown. The estimates from Bird 1 are in red-like colors, and the estimates from Bird 2 are in blue-like colors.

One source of model error could be the parameters we chose to keep constant. The model in its entirely has 65 parameters, some of which are very difficult to measure in traditional laboratory experiments. Figure 4.9 shows clear differences between our model and the biological response. Multiple parameters that contribute to these types of responses so it is difficult to point to which parameter might be wrong by visualizing the data. Nonetheless, the demonstrates that our model for the neuron has some inconsistencies with the real thing.

Additionally, figure 4.10 shows that the maximal conductance parameters for two different birds are not the same. The variation of parameters across birds means that we shouldn't hold parameters constant. Although computationally more expensive, allowing more parameters to be estimated may improve performance.

One final note to touch on is the action itself. Within the action of variational annealing and the cost function in nudging, the measurement error term is the square error between the estimate and the measured experimental data. It has previously been shown for neuron models that the majority of the measurement error between estimate and measured data is due to small timing differences between the neuron spikes [95]. The square error metric heavily penalizes small time shifts between spikes in the 'estimate' and 'measured' time series. Functions to make the estimates prioritize the timing of spikes over things like amplitude. In the context of modeling neurons, we would favor a spike train that is slightly shifted in time with matching amplitudes over one which was exact with different amplitudes. We could improve the results of our experiment by using a different metric, for example dynamic time warping (DTW). DTW is a metric designed to measure the similarity between two time series, while allowing for non-linear 'warping' between the two signals. Although computationally intense, this would prioritize the shape of our time series rather than point by point matching.

Chapter 4 was adapted from work being prepared for publication of the material as it may appear in A. Miller, D. Li, J. Platt, A. Daou, D. Margoliash, and H.D.I. Abarbanel, Statistical Data Assimilation: Formulation and Examples From Neurobiology, with consent of the authors. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Winnerless Competition Networks

The focus of this chapter is to build and assess a model of the olfactory network in insects. The insect olfactory system is an attractive system to study because it is an example of a relatively simple neural system where learning takes place [24, 14]. Of particular interest is the way in takes odors, a chemical input, and nonlinearly transform it into a complex temporal pattern of electrical signals, and identifies the odors reliably [66, 65].

A image of the complete network of the olfactory system can be seen in figure 5.1. The network starts when odor molecules bind to receptors within the insect antenna, stimulating activity in the olfactory receptor neurons (ORNs). ORNs project to the Antennal Lobe (AL). Within the AL, there are synaptically dense regions, neuropils, where a group of neurons can collectively receive stimulus from the ORNs. These synaptically dense regions are called glomeruli and are comprised of two types of neurons: projection neurons (PNs) and local neurons (LNs). Projection neurons, aptly named, are excitatory neurons that "project" to the mushroom body (MB). Local neurons are inhibitory neurons that connect to other neurons within the AL. The interaction of PNs and LNs within the antennal encode the signal from the ORNs both spatially (which neurons are firing) and temporally (when they fire) [66, 64].

Kenyon cells (KC) within the mushroom body (MB) receive the spatiotemporal signal from the AL further change the signal by making it very sparse [80]. Depending on the insect, there are one to two orders of magnitude more KCs than neurons in the AL, allowing for roughly 2% of neurons to respond to a given odor. The kenyon cells regulate their firing by projecting to the giant gabaergic neuron (GGN) in the locust [44], or the anterior paired laternal (APL) neurons in drosophila [67]. As the kenyon cells attempt to fire, they stimulate the GGN/APL, and the GGN/APL is able to inhibit the spiking of kenyon cells, enforcing sparsity [79].

The KCs also project to a different region of the MB, called the β-lobe (BL). The BL neurons are the readout neurons of the olfactory system. The synaptic connections between the KC and βL neurons are plastic and follow a plasticity rule called spike timing dependent plasticity (STDP) [24]. STDP functions to further separate the odor representation [9].

Each component of the network, or layer, in the network does its part to transform the input signal into an easily read output that allows the insect to reliably distinguish between odors even in the presence of environmental noise [8]. This property has made the olfactory system to be very attractive for investigation and modeling in a machine learning setting [31, 29, 30, 13, 12, 93, 25, 10]. In this chapter, we will detail the important aspects of the olfactory network relevant to its ability to learn. We will then build a simplified model based of these properties. Lastly, we will explore the model's ability to classify odors.

5.1 Building A Simplified Network

In the previous section gave an overview of the insect olfactory network and briefly touched on the properties crucial to its function. In this section, we go into these properties in depth and build up a simplified model which approximates the networks full behavior. This simplified model is comprised of two parts (1) a winnerless competition network meant to replicate the behavior of the antennal lobe, and (2) a support vector machine (SVM), meant to replicate the behavior of the Mushroom Body. This model was built with the intention of

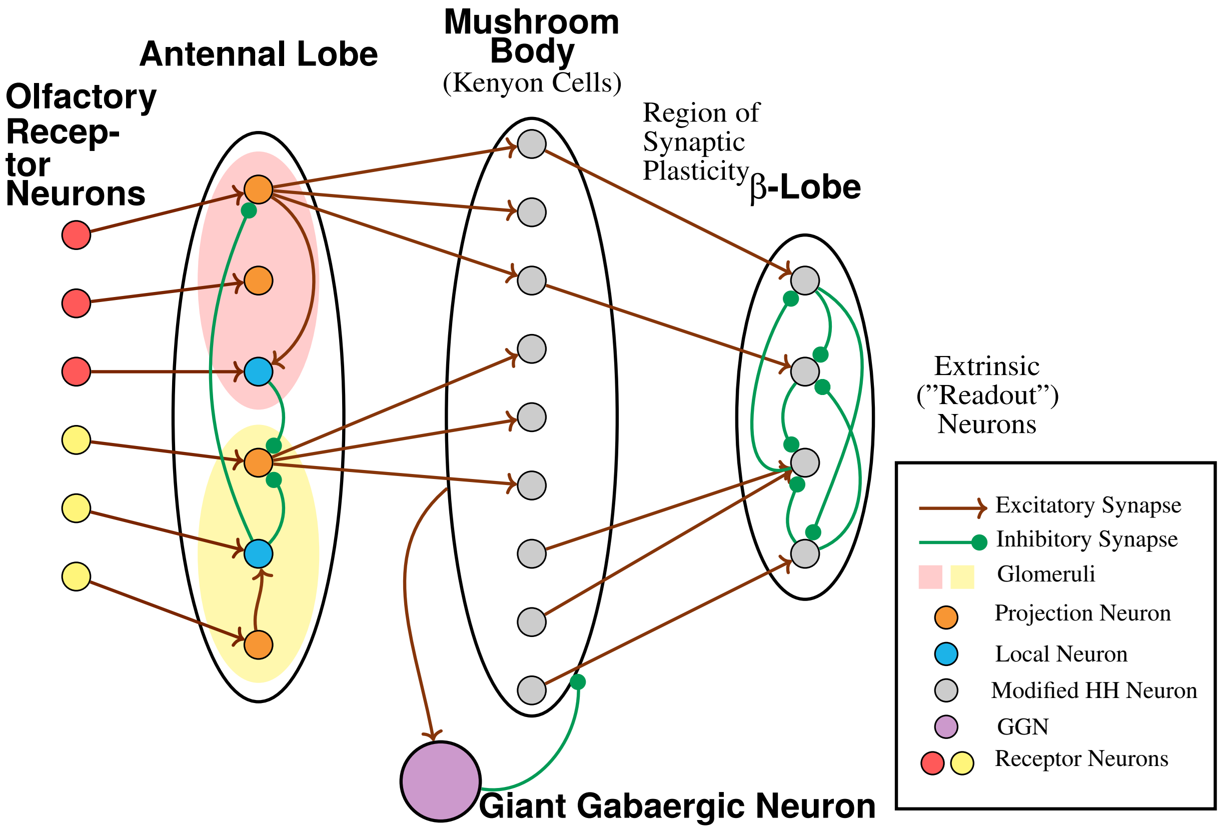

Figure 5.1: The neural circuits that perform the identification and classification of components in odors. The initial stage is composed of sensory neurons that are activated when a particular chemical component attaches to receptor neurons. These sensors produce electrical signals directed to the second stage called the antennal lobe (AL). Within the AL are excitatory projection neurons (PNs) and inhibitory interneurons (LNs). These neurons fire in such a way that encodes the input from the olfactory receptor neurons. The PNs carry AL activity forward to the next stage of olfactory recognition called the mushroom body (MB). The MB is comprised of a large number of Kenyon Cells (KC) whose activation is regulated by the Giant Gabaergic Neuron (GGN). The activity of of the KCs is projected to the β-Lobe, which are the readout neurons of this network. On the whole, the MB is suggested to act as a support vector machine in the biological olfactory network [51].

capturing all of the olfactory network's key properties which being the least computationally intensive.

5.1.1 Input

In order to investigate this network's ability to classify, we must create an input which emulates the role of the ORNs. An ORN's response to a given odor molecule depends on the olfactory receptor gene it contains. These genes can be narrowly tuned, only responding to a few molecules, or more broadly tuned, responding to many odor molecules. The are tens of thousands of ORNs in insects (90,000 in locusts [85], 60,000 in honey bees [60]), but these ORNs are not all unique. In each insect, there are on the order of 100 different olfactory receptor genes [91] which dictate how all of the ORNs fire. When no odor is present, ORNs fire spontaneously and the firing rate depends on ORN type [104, 28].

To look at the input the AL would receive, we modeled 200 NaKL neurons (equations 3.13 - 3.16) with spontaneous firing rates in response to stimuli and found that the collective activity of these neurons looks like a noisy DC signal. We also know that the concentration of an odor in air alters the firing rates of ORNs. As the odor concentration increases, ORN firing rate increases and more ORNs respond [46, 45]. This change in firing affects the collective ORN activity by changing its amplitude.

Using this information, we choose to represent the collective activity of all ORNs with a given receptor gene as a DC current. To differentiate between odors, we use the fact that all ORNs with a given olfactory receptor gene project to the same glomeruli within the AL. Therefore we can define an odor by which neurons within the AL receive stimulus, and how much stimulus (amplitude) they receive. This is an entirely spatial representation of odors. Section 5.2.5 goes over how we represent each odor as vector, where each element in the vector corresponds to a neuron in the AL and the values dictate the amplitude of a DC current.

In addition to investigating how well our network distinguishes between odors, we also want evaluate its response to mixtures of odors. When presenting a mixture of two known odors to an insect, only the ORNs which responded to the individual components are stimulated. No new ORNs are recruited in the presence of a mixture [15]. For this reason, we approximate mixtures of odors as a linear combination of the two ’base’ odors.

5.1.2 Substitution of the Antennal Lobe: Winnerless Competition Networks

The AL is comprised of PNs and LNs organized to receive stimulus in groups, called glomeruli. For the purposes of building a network, we focus on the activity of the PNs. All the information received from the ORNs must be encoded in PN activity because LN activity is not seen outside the AL. translating information, we focus on the activity of the PNs as PN activity is the only activity which continues on in our network.

When an odor is presented, a subset of AL neurons begin spiking in an odor-specific way. Since we seek to understand how the stimulus is encoded and passed along to other regions in the network, we focus primarily on the PNs.

The first key property we want to reproduce is the local field potential (LFP). The LFP is the average potential of all PNs and oscillates at frequencies ranging from 20-30Hz in locusts [66, 65]. The LFP works as a point of reference for the rest of the network. Within the AL, PNs have been found to exhibit transient synchrony with the LFP. In other words, PNs will spike in phase with the LFP for short durations after the onset of an odor. This transient synchrony of PN spiking with the LFP over many epochs of the LFP is reproducible and functions to continually increase the separation of similar odors over time [64, 13, 12, 103]. Additionally, the LFP creates cyclical integration windows where KCs are more sensitive to coincident PN spiking [80, 43, 42, 7].

The next property we want to reproduce is the long timescale (hundreds of miliseconds) odor-specific periods of spiking and quiescence in PN activity. The destruction of this patterning reduces odor separation [12, 13].

Lastly, we can investigate how the AL activity evolves over time by translating the PN activity into a higher dimensional phase space and observing its trajectory. When stimulated by distinct current inputs from a sensory input network, the biological AL produces trajectories in the network phase space following distinct patterns. There are three phases to its trajectory: (1) an ”on transient” immediately following the onset of a stimulus, (2) a fixed point which the network reaches and stays near after 1-2s, and an ”off transient” immediately following the offset of the stimulus. We can think of PN activity as a stimulus specific orbit in this phase space.

In order for a simpler model to function as a stand in for the insect antennal lobe, the model must reproduce the properties detailed above. These observed properties led to a suggestion of an AL network structure [87, 86, 5] called **winnerless competition networks** (WLC).

Winnerless competition behavior is characterized by sequential switching from one group to another, where one group is temporarily the ”winner”. In winnerless competition networks, the ”winner” is a subset of neurons which dominate firing at any given point in time. However, winnerless competition behavior has been observed in many dynamical systems, such as in models describing population levels in a predator-prey system, the Lotka-Volterra model [4],and in some studies investigating the convection of fluids in a turbulent rotating layer [23].

The first WLC network created was comprised of spiking neurons with only inhibitory connections. Investigation of the nine neuron WLC network was performed by Rabinovich [87, 86]. He found that the network, in response to stimulus, was defined by saddle points or saddle limit cycles connected by heteroclinic contours. These saddle points correspond to different neurons firing, and the heteroclinic contours between points correspond to the switching dynamics emphasized above.

A comparison of WLC network activity in higher dimensional phase space reveals that WLC reproduce the ”on” and ”off” transient dynamics of the antennal lobe, but do not reproduce the fixed point behavior. Analysis of each of the three phases showed that the largest distinct between odors happens during the transient phases and the fixed point alone was not sufficient for classification purposes. The reproduction of transient dynamics and sequential switching of neuron firing makes the WLC network an acceptable substitute to the insect antennal lobe [87, 86, 69].

5.1.3 Substitution of the Mushroom Body: Support Vector Machines

Now that we have a replacement for the AL, we must address the functionality of the rest of the network. Once the odor is encoded by the AL, the signal moves on the MB. KC receive the signal, where the GGN is responsible for enforcing KC spiking sparsity. The last stop for the odor is the BL neurons. These neurons are the read out neurons of the network. Their output identifies the received odor. There are strong inhibitory connections between the BL neurons, enforcing competition. Huerta proposed a mathematical equivalence between the function of the MB (KC and BL neurons) to a modified version of a Support Vector Machine (SVM) [52, 51]. For our purposes, we will use a regular multi class SVM. The remainder of this section will detail what a SVM is and the changes made by Huerta.

Support Vector Machines (SVMs) are a class of learning algorithms that are used in classification tasks. By default, the SVM is only capable of binary classification. Suppose you have a set of labeled data points $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_i, ..., \mathbf{x}_m\}$ with their corresponding label $\mathbf{Y} = \{y_1, y_2, ..., y_i, ..., y_m\}$. Since this is a binary classified, $\mathbf{y}_i$ can take two values: -1 or $+1$. We seek to draw a hyperplane between these two classes such that data points which fall on one side of the hyperplane correspond to one class and data points which fall on the other side of the hyperplane correspond to the other. Figure 5.2 shows the elements of how an SVM works. We choose to define an equation for the hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$. An SVM uses the data points closest to the dividing hyperplane to influence its position and orientation. These data points are called support vectors. We define two lines, along which the support vectors fall: $\mathbf{w} \cdot \mathbf{x} + b = -1$ for data points with label $y_i = -1$ and $\mathbf{w} \cdot \mathbf{x} + b = +1$ for data points with label $y_i = +1$. ± 1 was chosen because the scaling does not affect the placement of the hyperplane. The distance between these two hyper planes is $\frac{2}{||\mathbf{w}||}$. The goal of an SVM is to to orient the dividing hyperplane in such a way that maximizes the margin.

In order to maximize the margin, we need to minimize $||w||$ with the constraint that there are no data points that fall inside the margin. In other words, we expect that $\mathbf{w} \cdot \mathbf{x}_i + b \geq 1$ when $y_i = +1$ and $\mathbf{w} \cdot \mathbf{x}_i + b \leq -1$ when $y_i = -1$. We simplify the form and write out the goal in equation

$$\begin{gathered} \text{minimize } \frac{||w||}{2} \text{ such that} \\ \text{for all } \{\mathbf{x}_i, y_i\},\ y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq +1 \end{gathered} \tag{5.1}$$

We use Lagrange multipliers and optimization to find the optimal values of $\mathbf{w}$ and b for the decision boundary.

If data is not linearly separable data, SVM have something called the kernel trick where we can employ the use of a kernel, K. The idea behind the kernel trick is that, while our data may not be linearly separable in our n dimensional space (where n is the length of the vector $\mathbf{x}_i$), we can transform the data into a higher dimensional space where it is linearly separable. The transformation function is usually denoted as ϕ, and the kernel defines the inner product of the higher dimensional space, or $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$. This kernel trick is somewhat analogous to what the mushroom body does. There are roughly 1000 PNs in the AL, while there are 50000 KCs in the mushroom body. The significant dimensional increase allows for the KCs to have high specificity.

Once the decision boundary has been found, we can present new data to the trained SVM.The decision function for some data point $\mathbf{x}_k$ is $f(\mathbf{x}_k) = \mathbf{w} \cdot \mathbf{x}_k + b$, where we replace $\mathbf{x}_k$ with $\phi(\mathbf{x}_k)$ if a kernel was used. In the binary case, $f(\mathbf{x}_k) < 0$ corresponds to the class $y_k = -1$ and $f(\mathbf{x}_k) > 0$ corresponds to the class $y_k = +1$.

There are two configurations that allow for multiple classes when using an SVM: one-vs-one and one-vs-rest. In both of these configurations, we are training multiple binary SVMs. One-vs-one finds a dividing line between each class, needing to train $\frac{n!}{2!(n-2)!}$ SVMs, while one-vs-rest finds a dividing line between a class and every other class, needing to train n SVMs. Here we use the one-vs-rest configuration, training a total of L SVMs when there are L classes and L decision functions, f_l. A data point is identified as belong to one class by comparing all values $f_l(\mathbf{x}_k)$, and choosing the one yielding the greatest magnitude.

In [51] Huerta details an equivalence between something called Inhibitory Support Vector Machines (ISVM) and the function of the MB. The main difference between a regular SVM and ISVM is an additional inhibitory term in the decision function, regulated by scalar μ.

$$f_l(\mathbf{x}_k) = \mathbf{w_l} \cdot \phi(\mathbf{x}_k) - \mu \sum_{m}^{L} \mathbf{w_m} \phi(\mathbf{x}_k) \tag{5.2}$$

This inhibitory term makes it such that the SVMs inhibit each other, and each of the decision boundaries must adjust to classify the entire data set as well as possible. This formulation of the ISVM functions similarly to to the output in the BL neurons. BL neurons inhibit each other through inhibitory synapses, leading to a single active BL neuron at any given time.

For our purposes, initial testing found that the use of a kernel and the introduction of the inhibitory term was unnecessary for the success of the network. In this work, we use a multi-class linear SVM in a one-vs-rest configuration.

5.2 Methods

5.2.1 Neuron Models and Synapses

The base ingredients of any neural network, but more specifically the winnerless competition network, are neurons. The directed connections between neurons are called synapses. As we

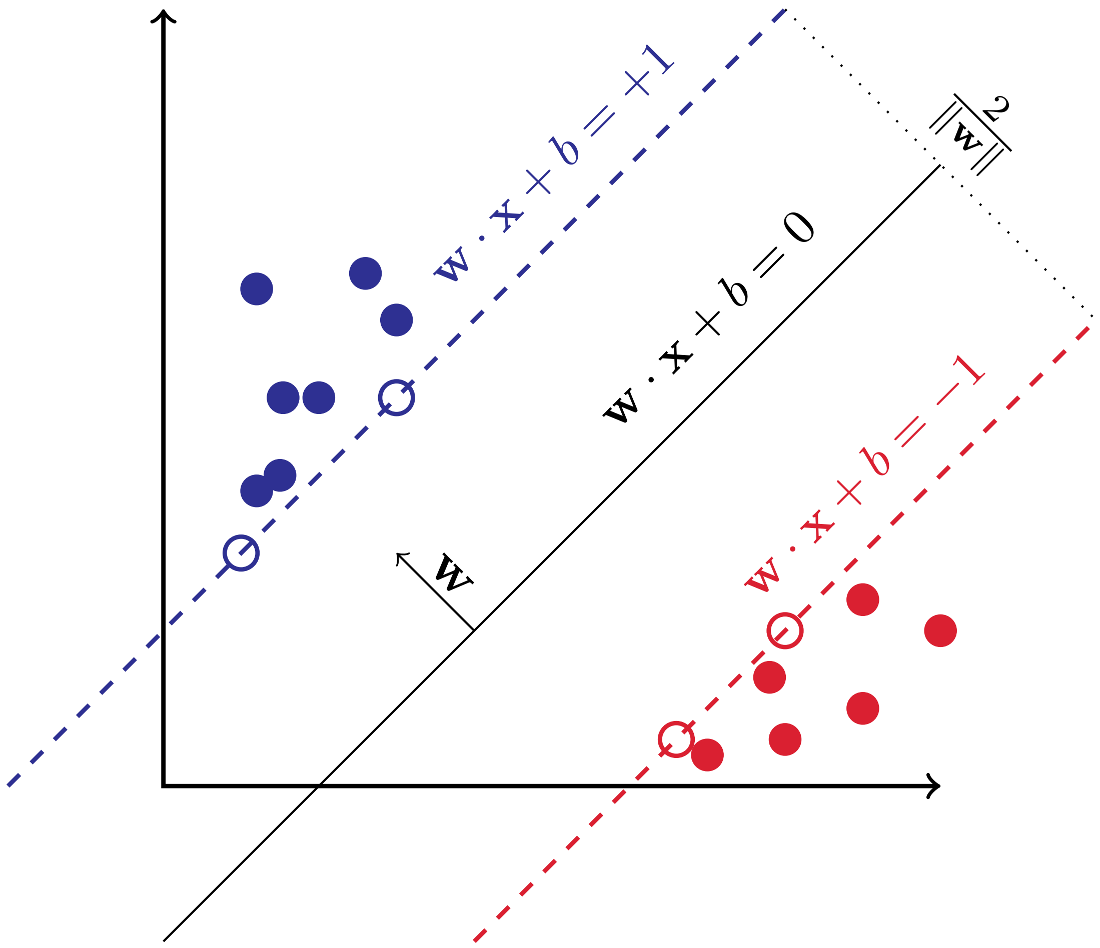
$\mathbf{w}\cdot\mathbf{x}+b=+1$
$\frac{2}{\|\mathbf{w}\|}$
$\mathbf{w}\cdot\mathbf{x}+b=0$
$\mathbf{w}\cdot\mathbf{x}+b=-1$
$\mathbf{w}$

Figure 5.2: An image of a Support Vector Machine. Here there are two classes, corresponding to the red and blue color, which we we give the labels -1 and $+1$, respectively. The goal of an SVM is to choose a dividing line which maximizes the distance away from either class using the closest data points, otherwise known as support vectors. We choose an expression for the dividing line, $\mathbf{w}\cdot\mathbf{x}+b=0$, and parallel lines to represent the or support vectors $\mathbf{w}\cdot\mathbf{x}+b=+1$ and $\mathbf{w}\cdot\mathbf{x}+b=-1$. Finally we are left to maximize the distance $\frac{2}{\|w\|}$ under the condition that no data points fall between the two support vector lines.

saw in chapter 3, we can model neurons with a set of by nonlinear differential equations. Within this chapter, we explore the properties of the WLC network using two different neuron models, the Hodgkin-Huxley model and the FitzHugh-Nagumo Model.

5.2.2 Hodgkin Huxley Neurons

Hodgkin Huxley model is a biologically realistic model formulated in 1952 during the study of giant squid neurons [50]. The basis of this model treats the neuron as a capacitor across which currents of different ions can pass. While there are many potential ion currents, the HH model focuses on a select few: a sodium current responsible for increasing the voltage across the neuron membrane, the potassium current, responsible for the sudden fall in the voltage of the membrane, and a leak current meant to capture the leak of ion across the membrane outside the dynamics of the sodium and potassium channels.

These currents are described in a conductance based model, which is a first order approximation of the more complete solution, balancing electrical forces and diffusion. The maximum conductance of each channel, the Nernst, or reversal, potential of each channel (a point of zero current of that ion, not necessarily zero current across the membrane) depending on the concentrations of each ion inside and outside the neuron, and the fraction of channels which have opened/close (gating variables). The full description of the model can be seen in chapter 3 section 3.1.1.

The parameters for equations 3.10 - 3.12 are listed in table 5.1. In this table, the conductance and capacitance terms have been combined into one term $g'_a = \frac{g_a}{C_m}$.

Table 5.1: The Parameters Used in conjunction with the Hodgkin-Huxley neuron model (equations 3.10 - 3.16).

Parameter	Data	Units	Parameter	Data	Units
g'_{Na}	120	nS/pF	E_{Na}	50	mV
g'_K	20	nS/pF	E_K	-77	mV
g'_L	0.3	nS/pF	E_L	-54.4	mV
θ_m	-40	mV	dV_m	0.0667	mV^{-1}
$t_{m,1}$	0.1	ms	$t_{m,2}$	0.4	ms
θ_h	-60	mV	dV_h	-0.0667	mV^{-1}
$t_{h,1}$	1	ms	$t_{h,2}$	7	ms
θ_n	-55	mV	dV_n	0.0333	mV^{-1}
$t_{n,1}$	1	ms	$t_{n,2}$	5	ms

5.2.3 Hodgkin Huxley Synapses

Neurons are connected and communicate through unidirectional synapses. When an action potential travels down the axon, to the axon terminals, neurotransmitters are released from the terminal into the synaptic cleft connecting the two neurons. These neurotransmitters then bind to the post synaptic neuron, and allow for the signal from one neuron to be communicated to the other. There are a few different neurotransmitters regularly seen, but our focus is on describing inhibitory synapses, as you will see in section 5.2.4. Gamma aminobutyric acid (GABA) is an inhibitory neurotransmitter. The equations governing a synapse can be seen in equations 5.3 -5.4.

$$\frac{dr}{dt} = \alpha_r \frac{T_m}{1+e^{(V_{\text{pre}}-V_p)/K_p}}(1-r) - \beta_r r \tag{5.3}$$

$$I_{syn} = g_{Ntr}(V - E_{gaba}) \tag{5.4}$$

Here, r is a gating variable which scales the transmission as a function of time. $\alpha_r T_m$ and β_r opening an closing rates, respectively. Note here that the opening rate is proportional to the maximum neurotransmitter concentration, T_m. V_{pre} is the voltage of the pre-synaptic neuron. V_p is the half activation voltage of the synapse and K_p dictates the rate of change of activation with respect to voltage. Equation 5.4 is the inhibitory current the post-synpatic neuron receives from one other neuron. g_{Ntr} is the maximal conductance for the synapse and E_{gaba} is the reversal potential related to the neurotransmitter, GABA.

FitzHugh-Nagumo Model

The second neuron model used in the network is the FitzHugh Nagumo (FHN) model. This is a two dimensional model based on the Hodgkin Huxley model discovered in 1952. The purpose of this model was to recreate/isolate the mathematical properties of the Hodgkin-Huxley model [37]. The HH model has 4 state variables that can be group into two time scales. V and

m are on one 'fast' time scale, and h and n are on a slow time scale. The behavior of h and n is closely related, $h \approx 1 - n$. We can take advantage of these properties and reduce our neuron model to 2 state variables: a voltage variable, V, with fast excitation and a recovery variable, w, that provides slower negative feedback. Because this model is much simpler, it is computationally more efficient, while still capturing important spiking characteristics of the HH model. Equations for this model are shown below in equations 5.5-5.7.

$$\frac{dV}{dt} = \frac{1}{\tau_1}\left(V - \frac{V^3}{3} - w + I_{inj} - z(V - \nu) + 0.35 \text{ mV}\right) \tag{5.5}$$

$$\tau_3 \frac{dw}{dt} = V - bw + a \tag{5.6}$$

$$\frac{dz}{dt} = \frac{1}{\tau_2}(I_{syn} - z) \tag{5.7}$$

Equation 5.5 up to and including the injected current, I_{inj} as well as equation 5.6 are the original equations developed by FitzHugh and Nagumo.

This model is originally dimensionless, but to put it in the context of neurondynamics, we will give the voltage-like variable, V, units of mV and the injected current, I_{inj} units of pA. Time is in units of ms. These dimensions are chosen to be comparable to biological neurons. Here there is no term for membrane capacitance, so to correct the dimensions of current, I_{inj} is multiplied by $1M\Omega$.

When this neuron receives a sufficiently large injected current, V will rapidly increase to roughly 1 mV. The combination of the cubic term in equation 5.5 and the recovery variable, w, rapidly work to decrease the voltage back to its resting potential of -1 mV. The time constants, $\tau_1 = 0.08$ ms and $\tau_3 = 1$ ms, control to rate at which this happens. a and b are constant parameters with values $a = 0.7$ and $b = 0.8$. 0.35 mV was added to the voltage equation to ensure WLC properties in this network.

In an insect antennal lobe, there are two classes of neurons: inhibitory local neurons and excitatory projection neurons. Our modified model of FitzHugh-Nagumo neurons performs the role of both. Equation 5.7 and its corresponding term in equation 5.5 describe the inhibitory synapses for this network. z is a something similar to conductance while $\nu = -1.5$ mV is the reversal potential of the synapse. The time constant $\tau_2 = 3.1$ ms controls the rate at which the synaptic conductance changes. I_{syn} is the total synaptic current received by a given neuron. The total synaptic current is described by equations 5.8 and 5.9

$$I_{syn} = \sum_{\text{all } V_{pre}} g_{synAL} G(V) \tag{5.8}$$

$$G(V) = \frac{1}{1 + e^{-1000 V_{pre}}} \tag{5.9}$$

where $G = 1$ for $V > 0$ mV and $G = 0$ otherwise. When a pre-synaptic neuron is depolarized, it produces a constant synaptic current received by the post-synaptic neuron. g_{synAL} is the strength of synaptic inhibition and set to $g_{synAL} = 0.1$. This value is chosen by us to make sure the inhibition is not so weak that it has no effect, but not so strong that a few neurons dominate spiking. Values both too small and too large tend to produce winner-take-all networks. We chose to use a steep sigmoid function for $G(V)$ in place of a step function for integration purpose, which can be seen in equation 5.9.

5.2.4 Network Connectivity

In order to replicate the size of the AL in locusts, we use a network with 1000 neurons, using one of the two neuron models described above. The initial conditions of each neuron are set to be near their values in the absence of stimulus, randomly sampled from a narrow gaussian distribution. The connectivity of this network is based entirely on the empirical properties needed to reproduce WLC behavior. We use a large randomly connected network of either HH or FHN neurons, with a connection probability of 0.5. Each network was tested by verifying the properties detailed in section 5.1.2. Like neurons, the initial conditions of the synapse state variables are randomly chosen near their rest values.

All network modelling was performed by using the Brian2 Python package [99]. Brian2 is an open source simulator for spiking neural networks.

5.2.5 Robustness to Noise in a Winnerless Competition Network with HH and FHN Neurons

One of the more attractive features of the insect olfactory network is its ability to perform even in the presence of noisy, turbulent air. Here we investigate our networks ability to correctly classify odors in the presence of noise.

As we described in section 5.1.1, an odor is defined by injecting a DC current into a unique subset of neurons, which we express as a vector $\mathbf{I}_j$. Within the insect olfactory network, the number of ORNs that respond to a given odor can vary wildly, depending on the molecule and its concentration. To find out how many neurons should be in this subset, we varied the fraction of neurons which received injected current and looked at the network response. We started with as low as 5% and went as high as 50% of neurons stimulated. The amplitude of the injected current, I_0, was set such that it stimulated action potentials in isloated FHN neurons. There are two properties we checked for in the network activity. First, we look at the average neuron potential to verify that the LFP has some oscillatory nature, as the LFP plays a significant role in the insect's ability to learn. Next, we looked at overall network spiking activity to ensure that a pattern of spiking and quiescence . We found that, if we injected roughly one-third of all neurons with current, the pattern was preserved.

Additionally, we found that WLC network activity is sensitive to the amplitude of the input current, I_0. If we elect to choose an I_0 that is too large, the inhibitory network is too weak to combat it, and we end up with a winner-take-all arrangement, where the winners are neurons which receive this injected current. If I_0 is too small, we don't stimulate enough activity within the network to encode the odor. I_0 was set to 150 pA, chosen empirically to preserve WLC behavior.

The input base current vectors $\mathbf{I}_{\text{base\_}j}(t)$ are selected by drawing the components of each $\mathbf{I}_{\text{base\_}j} : \{I_1, I_2, \ldots, I_N\}$ from a uniform random distribution including a decision threshold such that $\approx$ 1/3 of the values are set to I_0. The rest are set to 0. This gives N/3 distinct stepwise DC currents: *off* before t_0, then *on* with amplitude I_0, and then *off* again after t_M. The base odors $\mathbf{I}_j$ are not orthogonal to one another.

In order to create noisy version of odors, we take the base odor $\mathbf{I}_{\text{base\_}j}$, and add a small amount of noise for each neuron. This is chosen by sampling from a uniform distribution, and scaling the number by the strength of the original odor I_0, and an additional parameter η.

Each of the K noisy current stimuli $\mathbf{I}_j$ are $j = 1, 2, \ldots, K$ currents in N-dimensional neuron space is given by a distinct input vector

$$\mathbf{I}_j(t) = \theta(t - t_0)\theta(t_M - t)\left[\mathbf{I}_{\text{base\_}j} + I_0 \eta \xi\right]. \tag{5.10}$$

All elements of ξ are sampled from a uniform distribution ranging from -1 to 1, $\xi_i = \mathcal{U}(-1, 1)$.

The noise changes the amplitude of the DC current injected (or not injected) into each neuron. Neurons not originally receiving current could receive a small positive or negative injected current. Neurons originally receiving $I_0 = 150$ pA, will have the strength of the DC current changed. We create additional noisy odors for a given base odor by resampling the noise using the above procedure.

The network tested is comprised of $N = 1000$ FHN neurons, randomly connected with a probability of 0.5. 500 base odors we randomly chosen and stimulated the network for 100 ms each, with an integration time step of 0.05 ms.

We take the output of this network and feed it through a one-vs-rest multiclass SVM to train it. The SVM is trained in 1000-dimensional space, corresponding to the voltages of each of the 1000 neurons. Each time point is a data point. Once the SVM is trained on the base odors, we test the network by measuring the networks ability to classify noisy odors. For a given η, we create two noisy odors per base odor and run them through the WLC network for 100 ms. The voltage outputs of each noisy odor are taken and ran through the SVM. For a given odor, each time point is labeled as belonging to one of 500 odors. We then classify the whole time series by counting the labels and choosing the odor that most frequently appears. We repeat this procedure, varying η from $\eta = \frac{\sqrt{3}}{4}$ to $\eta = 5\sqrt{3}$.

We then repeat this procedure, building a $N = 1000$ neuron WLC network with HH neurons and connection probability of 0.5 and $I_0 = 150$ pA. The network is trained on 15 base odors, and tested on 10 randomly sampled noisy odors per base odor, for a total of 150 noisy odors per η. η is varied from $\eta = 0$ to $\eta = 8$ and each stimulus is presented to the network for 125 ms each.

5.2.6 Winnerless Competition Network with FHN Neurons in the Presence of Mixtures

In nature, an insect is not exposed to single odors in isolation, but rather a mixture of these odors. When presented with a mixture of odors, the insect must be able to identify the individual components. It is for this reasoning that we explore how the WLC network responds to odor mixtures.

In order to create mixtures of odors, we take two base odors created in the way laid out by section 5.2.5. We then assume that a mixture of odors is a linear combination of the two base odors as shown in equation 5.11.

$$\mathbf{I}_{\text{mixture}}(t) = \alpha \mathbf{I}_{\text{base\_1}}(t) + (1-\alpha)\mathbf{I}_{\text{base\_2}}(t) \tag{5.11}$$

α is a scaling factor that takes a value between 0 and 1.

A 1000 neuron network was created, as described in section 5.2.5. The WLC+SVM network was trained in the same fashion as above on 50 distinct base odors presented for 100 ms each. From these 50 odors, two were selected to create the mixtures. α is varied from 0 to 1, increasing by increments of 0.01. Each of these mixtures was presented to the network for 100 ms. We then extend this analysis to a mixture of 3 odors.

5.3 Results

5.3.1 Robustness to Noise in a Winnerless Competition Network with HH and FHN Neurons

In order to visualize the spatiotemporal encoding of the WLC network in we reduce the dimensions of this $N = 1000$ dimensional space by projecting it into 3 dimensional space through the use of Principal Component Analysis (PCA)[82]. PCA is a procedure used to reduce the dimensions in a dataset by calculating a covariance matrix and transforming the data into a lower dimensional space corresponding to the eigenvectors with the largest eigenvalues, known as principle components. Because the matrix is real and symmetric, the eigenvectors which make up this lower dimensional space are orthogonal.

For our purposes, we take the neuron voltage outputs $V_j(t);\ j = 1, 2, \ldots, N$ of five different base odors concatenate them into one large data set. We then use PCA on this dataset to project the data into 3 dimensional space. This projection can be seen in figures 5.4 and 5.3, where each data point represents a point in time. For both a WLC network of FHN and HH neurons, the separation between odors in the PCA space is clear. Figure 5.3 shows the full trajectory from rest. For each of the five odors, the network initially starts at rest. It then shoots off into different directions in this space, where it orbits. The geometric image of this orbit is a heteroclinic contour. Not displayed here is the trajectory when the stimulus is removed. Figure 5.4 displays the heteroclinic contours without the initial trajectory. This trajectory was removed in order to better visualize the trajectory in this space. Both figures display a clear separation of network activity in response to differing odors in this 3 dimensional space.

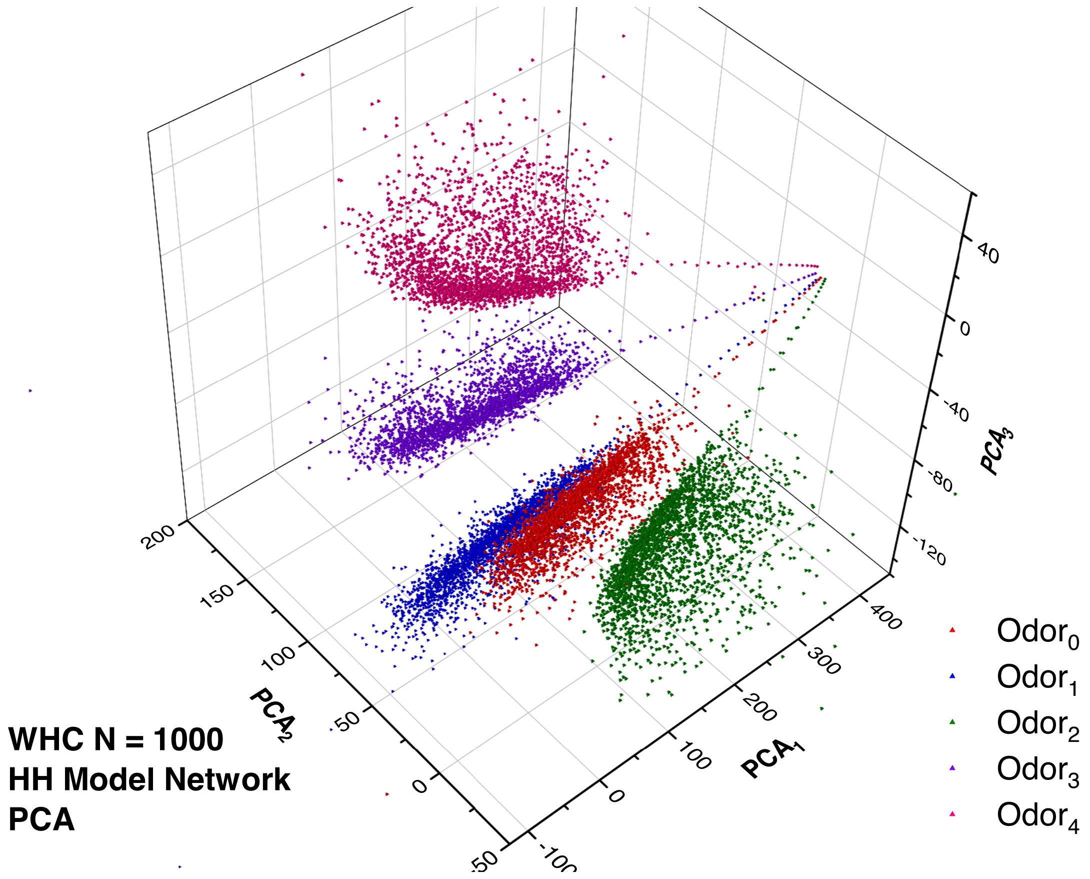

Figure 5.3: Five distinct odors represented by stimulating currents were presented to the 1000 model WLC network built with Hodgkin-Huxley (here) or Fitz-Hugh Nagumo biophysical neurons [54, 98, 102] connected by inhibitory synaptic processes. A PCA projection was performed on a concatenation of the five distinct, 1000 dimensional voltage signals produced by these stimuli. The projection displayed is into the three dimensional space spanned by the PCA eigenvectors with the largest singular values. In this very low dimensional space the separation of the input stimuli in space (neuron number) and time is clear. Before the stimulation is turned on, all signals are at rest. When stimulation begins they move rapidly to different regions of PCA projected state space and remain there while the stimulation persists.

Next we turn to the full network's ability to correctly classify in the presence of noise. The FHN neuron WLC+SVM network, was trained on 500 base odors presented to the network for 100 ms each. We test the networks ability to classify new odors by presenting 1000 noisy odors for each η value, two noisy odors per base odor. η was varied from $\eta = 0$ (no noise) to

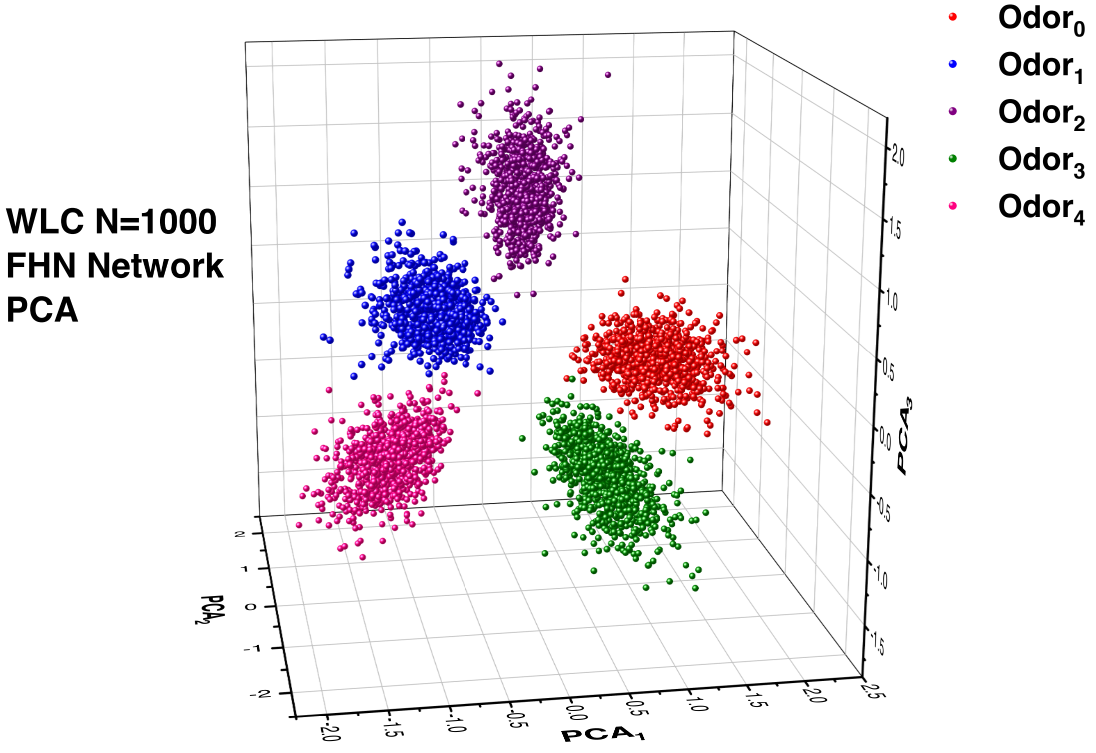

Figure 5.4: Five distinct odors represented by stimulating currents were presented to the 1000 model WLC network built with Fitz-Hugh Nagumo biophysical neurons [54, 98, 102] connected by inhibitory synaptic processes. A PCA projection was performed on a concatenation of the five distinct, 1000 dimensional voltage signals produced by these stimuli. The projection displayed is into the three dimensional space spanned by the PCA eigenvectors with the largest singular values. In this very low dimensional space the separation of the input stimuli in space (neuron number) and time is clear. Before the stimulation is turned on, all signals are at rest. When stimulation begins they move rapidly to different regions of PCA projected state space and remain there while the stimulation persists.

$\eta = 5\sqrt{3}$. For each presentation of an odor, all time points were labelled as belonging to one of the 500 original odors, and the time series of the odor presentation was classified based on the label which appeared the most.

The results can be seen in figure 5.5. We plot the classification accuracy against a noise-to-signal ratio (inverse of the signal-to-noise). Here we can see that this network architecture performs well even in the presence of significant feature noise. This level of performance is surprising. Because we add noise by changing the amplitudes of the injected current at each neuron and odors are primarily spatial, we would expect that once the noise was strong enough to add or remove an amplitude of I_0, this would distort the odor enough that the WLC network would see it as a different input. However, the SVM is still accurate under these conditions. A similar curve is displayed in figure 5.6, corresponding the the HH WLC+SVM network. Here η ranges from $\eta = 0$ to $\eta \approx 8.3$. Adjusting for the factor of $\sqrt{3}$, the FHN networks perform similarly in response to noise.

5.3.2 Winnerless Competition Networks with FHN Neurons in the Presence of Mixtures

In order to test the networks response to mixtures, we train a $N = 1000$ FHN neuron WLC+SVM network on 50 base odors. We then sample two base odors from the 50, which we will denote as $I_{\text{base\_1}}$ and $I_{\text{base\_2}}$, and create a mixture $\mathbf{I}_{\text{mixture}}(\alpha) = \alpha\mathbf{I}_{\text{base\_1}} + (1-\alpha)\mathbf{I}_{\text{base\_2}}$; $0 \leq \alpha \leq 1$. This mixture was then presented to the trained network for 125 ms. Each data point in the 1000 dimensional voltage time series output of this mixture was classified as belonging to one of the 50 base odors. More mixtures are created by varying α from $\alpha = 0$ (entirely odor $\mathbf{I}_{\text{base\_1}}$) to $\alpha = 1$ (entirely odor $\mathbf{I}_{\text{base\_2}}$) in increments of 0.01. For our purposes, we express the classification of odor in terms of a probability shown in equation 5.12.

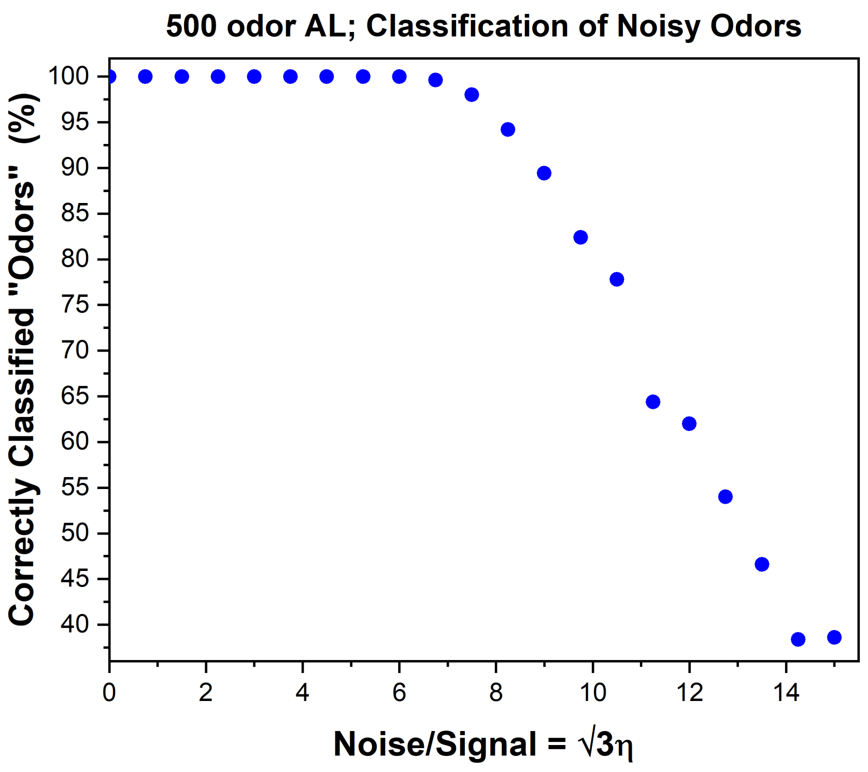

Figure 5.5: The classification accuracy of noisy currents presented to a WLC+SVM trained on 500 baseline ($\eta = 0$) odors—see equation (5.10). Multiple trials of each noisy ($\eta > 0$) odor were presented to the WLC+SVM network (with FHN neurons at the nodes) for $t_M - t_0 = 100$ ms. The odor is labeled by taking the highest probability of Eq.(5.12)

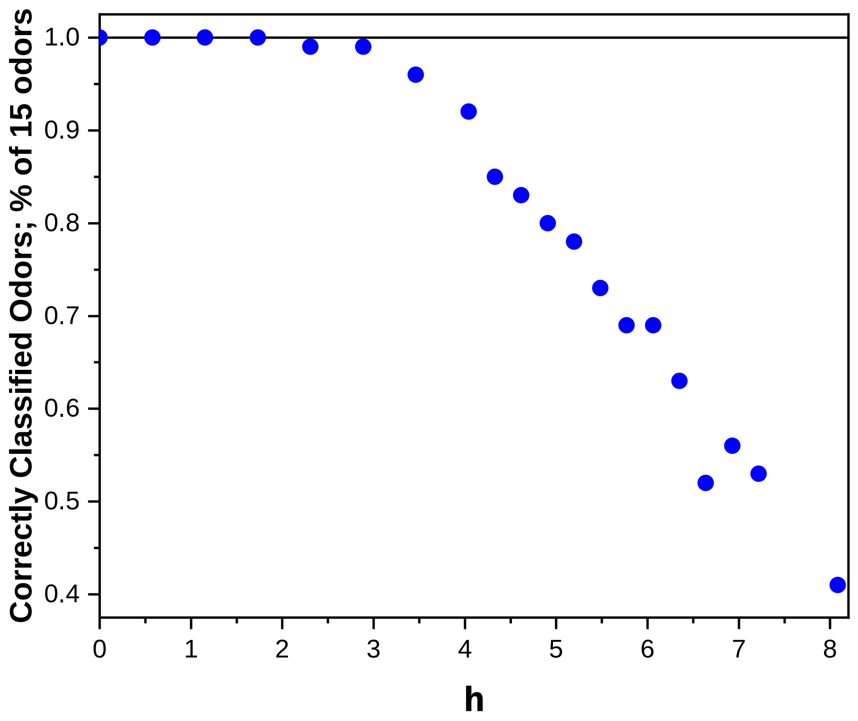

Figure 5.6: The classification accuracy of noisy odors presented to a HH WLC+SVM network trained on 15 odors. Multiple trials of each noisy odor having $\eta \neq 0$ in Eq.(5.10) were presented to the network for 125 ms with the label assigned as described in equation.(5.12). The classification accuracy is found by comparing this prediction to the label of the “baseline” odor equation.(5.10) to which the noise is added. Data for FHN neurons is similar and is presented in Fig (5.5).

$$P_{I\text{base\_}k}(Odor) = \frac{\text{Time in SVM region k}}{\text{Total time in all SVM regions}}. \tag{5.12}$$

Figure(5.7) shows the probability of $\mathbf{I}_{\text{base\_1}}$ as a function of α. At $\alpha = 0$ we expect the probability to be essentially zero, and at $\alpha = 1$, it should very close to unity. The data in this figure was fit with a sigmoid function, given by the expression in 5.13, with $a = 14.6$. The dots in red show the calculation for $P_{I_{\text{base\_1}}}$ at each α value.

$$P_{I_{\text{base\_1}}}(\alpha) = \frac{1}{2}\left[1 - \tanh a(0.5-\alpha)\right] = \frac{e^{a\alpha}}{e^{a\alpha} + e^{a(1-\alpha)}}, \tag{5.13}$$

with $a = 14.6$.

For each α value, $\mathbf{I}_{\text{mixture}}(\alpha)$ was entirely contained within the regions corresponding to $\mathbf{I}_{\text{base\_1}}$ or $\mathbf{I}_{\text{base\_2}}$, with some very small leakage into the other 48 odors classes. This experiment was performed two additional times by resampling the odors contained in the mixture. The results (not shown) are similar to figure 5.7.

If the network is presented with a mixture of two unknown (but previously experienced) odors, we can recover the concentration α by calculating $P_{I_{\text{base\_}k}}(Odor)$ and inverting equation (5.13); thus the fraction of each pure odor can be written as

$$\alpha = \frac{1}{2} + \frac{1}{a}\tanh^{-1}[2P_{I_{\text{base\_}k}}(\alpha) - 1].$$

We extend this analysis to a mixture of three odors by sampling three new odors from the initial 50 and using the same network used for a 2 base odor mixture. The equation for the a mixture of three odors can be seen in equation 5.14 where $0 \leq \alpha \leq 1$, $0 \leq \beta \leq 1$, $\alpha + \beta \leq 1$.

$$\mathbf{I}_{mix}(\alpha,\beta) = \alpha\mathbf{I}_{\text{base\_1}} + \beta\mathbf{I}_{\text{base\_2}} + (1-\alpha-\beta)\mathbf{I}_{\text{base\_3}} \tag{5.14}$$

Each data point in the voltage time series output is labeled. We increment α and β by

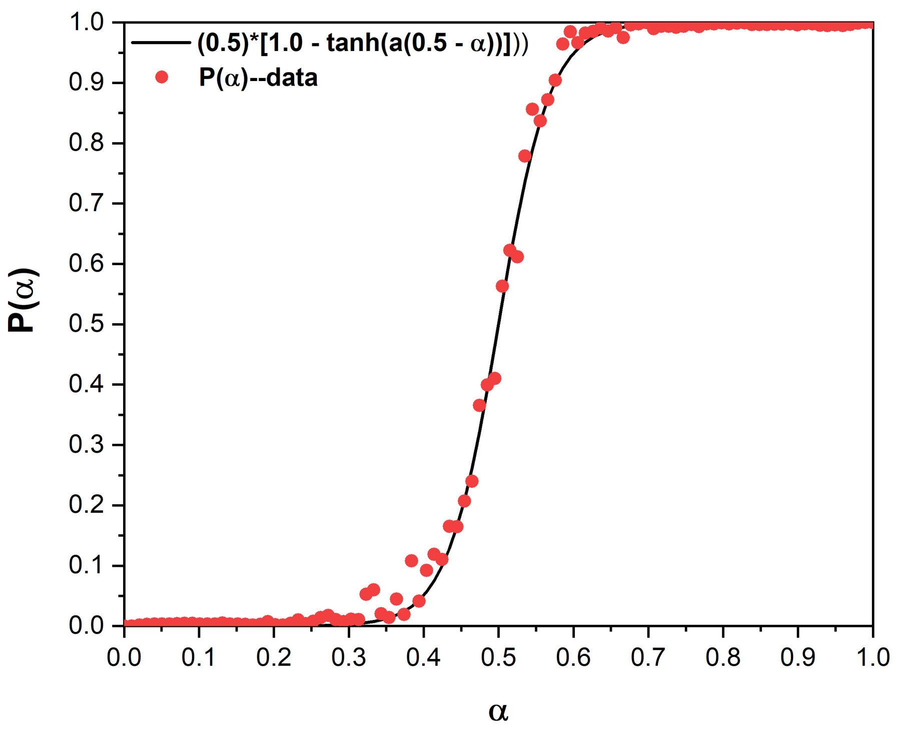

Figure 5.7: A WLC+SVM network was trained on 50 pure odors I_j; $j = 1, 2, \ldots, 50$ and the mixture $\alpha I_1 + (1-\alpha)I_2$ presented. Sigmoid curves were fit to the data. The input mixture varies linearly in α, while the classification probability follows a non-linear sigmoid. The MB enhances the separation in the AL and creates sharp boundaries between odors. The robustness to noise of the WLC+SVM network is due in part to the sharp boundary between classes of mixtures as shown here.

0.04 until all combinations of values of α and β are explored.

The results can be seen in figure 5.8. The red dots are given by taking the output of the WLC network and calculating equation 5.12 for one base odor, which we call $\mathbf{I}_{\text{base\_1}}$. The blue dots represent the fit to the data, given by

$$P_{I_{\text{base\_1}}}(\alpha,\beta) = \frac{e^{\sigma\alpha}}{e^{\sigma\alpha} + e^{\sigma\beta} + e^{\sigma(1-\alpha-\beta)}} \tag{5.15}$$

with $\sigma = 20$. Here we see similar results to the two odor case.

We can generalize the ability to to identify relative odor concentrations of mixtures. If we have a linear mixture of K previously seen base odors ($\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_K\}$)

$$\mathbf{I}_{mix}(\boldsymbol{\alpha}) = \sum_{k=1}^{K} \alpha_k I_k, \tag{5.16}$$

the output from the WLC+SVM network, using equation 5.12, will be matched accurately by the expression

$$P_{I_{\text{base\_}j}}(\boldsymbol{\alpha}) = \frac{e^{\sigma\alpha_j}}{\sum_{k=1}^{K} e^{\sigma\alpha_k}}; \;\; \sum_{k=1}^{K} \alpha_k = 1. \tag{5.17}$$

We can invert this equation for each of the K odors, to get K α_j values as a function of the observed $P_{I_{\text{base\_}j}}(\boldsymbol{\alpha})$.

A casual observer may ask why the WLC network (and therefore the AL) is necessary at all. An SVM, if trained on the vectors of our base odors, $I\mathbf{I} + \text{base\_}k$, is capable of classifying these odors correctly with comparable performance to our WLC+SVM network. However, an SVM alone will classify a mixture, I_{mixture} into one of the previously seen 50 base odors. Without the voltage time series response from the WLC network, the ability to separate mixtures of learned base odors is totally lost.

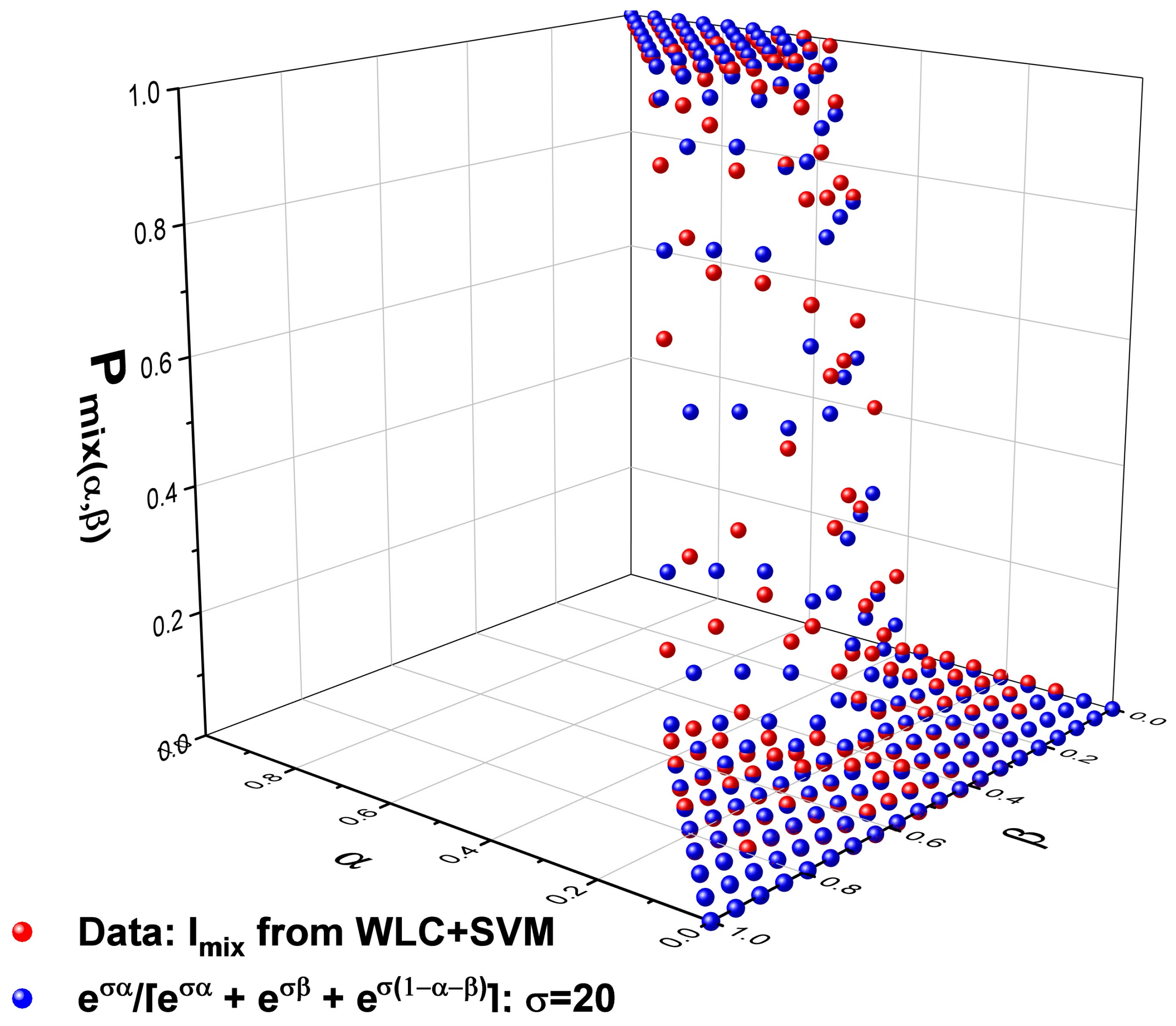

Figure 5.8: Mixture of three odors (currents): $I_{mix}(\alpha,\beta) = \alpha I_1 + \beta I_2 + (1-\alpha-\beta)I_3$ after processing by a 1000-dimensional WLC + SVM network (red dots). Fit to data (blue dots) is given by fitting equation 5.15.

5.4 Conclusions

Section 5.2.5 demonstrates that our WLC+SVM network structure is capable of separating spatial classes and is particularly robust to high levels of noise using both FHN and HH neurons. While an impressive result, there is still room for improvement. When compared to the classification ability of an SVM alone, the WLC+SVM network performs comparable to the SVM. This result is not surprising. An SVM is a modern tool used in machine learning specifically for classification purposes of non-time dependent signals. In the above experiments, we use constant inputs meant to replicate the collective behavior of the ORNs. However, the strength of the AL and presumable the WLC network comes from its ability to handle noisy, time varying signals.

Even in controlled laboratory experiments where turbulence is minimized, the ORN response to an odor is dynamic [89, 75, 68]. This manifests itself in the ORNs by odor specific time-delays between odor presentation and ORN activity. ORN response is dependent on many factors, including the concentration and its rate of change [75]. Previous studies have found that time delays of ORN response are essential for complex spatiotemporal encoding within the antennal lobe [89], but how the AL accounts for this information is still an open question [78]. One way in which we can improve upon these experiments is by testing the network's capability to encode time-dependent signals.

A general improvement that can be made to the WLC network is to introduce lateral excitation. As there are no excitatory synapses in this network to replicate projection neurons, a mechanism is needed to allow for spiking in neurons that do not receive direct stimulus from odors. To accomplish this, 0.35 mV to the voltage equation. This allows our neurons, when isolated, to spike without input. However, depending on the input, the inhibitory effect from the directly stimulated neurons completely prevents the activity of non-stimulated neurons. This behavior is necessary as it has been shown that the recruitment of non-stimulated neurons through lateral excitation improve the antennal lobe's ability to distinguish similar odors [94, 8].

Additionally, the way in which we input data into the SVM could be changed to drastically reduce training time. There is a somewhat recently developed technique for handling multivariate time series, called reservoir computing. The reservoir computing approach uses a recurrent (inter-connected) neural network to process sequential data. That data is then processed in the usual manner, depending on the desired application. Examples include the use of linear regression or a linear classifier such as a SVM. In these implementations, the linear models trained do not use the entire time series, but instead use the last state of the network. This is because interconnected networks have the capability to embed all the information necessary to reconstruct the original input [100]. However, this implementation may compromised the ability of the network to distinguish mixtures.

Section 5.3.2 demonstration the network's ability to identify and recover individual components of mixtures presented to the network. This network allows for the identification of mixtures of classes given a single data point, impossible with an SVM alone. This WLC+SVM network arrangement provides additional functionality over standard classifications methods. However there is still more to be explored. While there are no additional neurons recruited in the presence of a mixture odors, a linear combination representation may not be sufficient. Competitive binding between molecules functions to normalize the ORN response. That is, the strength of ORN response does not depend on the number of components within the mixture, and is largely explained by competitive antagonism [90, 97]. This means a mixture of odors cannot be explained by a linear combination of the base odors.

Chapter 5 was adapted from work being prepared for publication of the material as J. Platt, A. Miller, L.Fuller, H.D.I. Abarbanel, Machine Learning Classification Informed by a Functional Biophysical System, with consent of the authors. The dissertation author was one of the primary investigators and authors of this paper.

Bibliography

[1] Computational Mathematics Group at the STFC Rutherford Appleton Laboratory. HSL, a collection of fortran codes for large-scale scientific computation. see http://www.hsl.rl.ac.uk.

[2] H. D. I. Abarbanel, P. J. Rozdeba, and Sasha Shirman. Machine learning: Deepest learning as statistical data assimilation problems. *Neural Computation*, 30(8):2025–2055, August 2018.

[3] Henry D. I. Abarbanel. *Predicting the Future: Completing Models of Observed Complex Systems*. Springer, 2013.

[4] Valentin Afraimovich, Irma Tristan, Ramon Huerta, and Mikhail I. Rabinovich. Winnerless competition principle and prediction of the transient dynamics in a Lotka–Volterra model. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 18(4):043103, October 2008.

[5] VALENTIN S. Afraimovich, MIKHAIL I. Rabinovich, and PABLO Varona. Heteroclinic contours in neural ensembles and the winnerless competition principle. *International Journal of Bifurcation and Chaos*, 14(04):1195–1208, 2004.

[6] R. Anthes. Data assimilation and initialization of hurricane prediction models. *J. Atmos. Sci.*, 31, 1974.

[7] Collins Assisi and Maksim Bazhenov. Frontiers | A mechanism that governs a transition from coincidence detection to integration in olfactory network. 2010.

[8] Collins Assisi, Mark Stopfer, and Maxim Bazhenov. Excitatory Local Interneurons Enhance Tuning of Sensory Information. *PLOS Computational Biology*, 8(7):e1002563, July 2012.

[9] Collins Assisi, Mark Stopfer, and Maxim Bazhenov. Optimality of sparse olfactory representations is not affected by network plasticity. *PLOS Computational Biology*, 16(2):e1007461, February 2020.

[10] Collins Assisi, Mark Stopfer, Gilles Laurent, and Maxim Bazhenov. Adaptive regulation of sparseness by feedforward inhibition. *Nature Neuroscience*, 10(9):1176–1184, September 2007.

[11] Sharan K Bagal, Brian E Marron, Robert M Owen, R Ian Storer, and Nigel A Swain. Voltaged gated sodium channels as drug discovery targets. *Channels (Austin)*, 9(6):360–366, December 2015.

[12] Maxim Bazhenov, Mark Stopfer, Mikhail Rabinovich, Henry D.I. Abarbanel, Terrence J. Sejnowski, and Gilles Laurent. Model of Cellular and Network Mechanisms for Odor-Evoked Temporal Patterning in the Locust Antennal Lobe. *Neuron*, 30(2):569–581, May 2001.

[13] Maxim Bazhenov, Mark Stopfer, Mikhail Rabinovich, Ramon Huerta, Henry D.I. Abarbanel, Terrence J. Sejnowski, and Gilles Laurent. Model of Transient Oscillatory Synchronization in the Locust Antennal Lobe. *Neuron*, 30(2):553–567, May 2001.

[14] JS de Belle and M. Heisenberg. Associative odor learning in Drosophila abolished by chemical ablation of mushroom bodies. *Science*, 263(5147):692–695, February 1994.

[15] Leonardo Belluscio and Lawrence C. Katz. Symmetry, Stereotypy, and Topography of Odorant Representations in Mouse Olfactory Bulbs. *Journal of Neuroscience*, 21(6):2113–2122, March 2001.

[16] Johan J. Bolhuis, Kazuo Okanoya, and Constance Scarff. Twitter evolution: Converging mechanisms in birdsong and human speech. *Nature Reviews Neuroscience*, 11:747–759, 2010.

[17] Michael S. Brainard and Allison J. Doupe. Auditory feedback in learning and maintenance of vocal behaviour. *Nature Reviews Neuroscience*, 1, 2000.

[18] Daniel Breen. *Characterizing Real World Neural Systems Using Variational Methods of Data Assimilation*. PhD thesis, University of California San Diego, 2017.

[19] Daniel Breen, Sasha Shirman, Eve Armstrong, Nirag Kadakia, and Henry Abarbanel. Hvci neuron properties from statistical data assimilation. 2016.

[20] Eliot A. Brenowitz and Michael D. Beecher. Song learning in birds: diversity and plasticity, opportunities and challenges. *Trends in Neurosciences*, 28(3):127–132, March 2005.

[21] Eliot A Brenowitz, Daniel Margoliash, and Kathy W Nordeen. An introduction to birdsong and the avian song system. page 6.

[22] Zachary Daniel Burkett, Nancy F Day, Todd Haswell Kimball, Caitlin M Aamodt, Jonathan B Heston, Austin T Hilliard, Xinshu Xiao, and Stephanie A White. FoxP2 isoforms delineate spatiotemporal transcriptional networks for vocal learning in the zebra finch. *eLife*, 7:e30649, January 2018.

[23] F. H. BUSSE and K. E. HEIKES. Convection in a Rotating Layer: A Simple Case of Turbulence. *Science*, 208(4440):173–175, 1980.

[24] Stijn Cassenaer and Gilles Laurent. Hebbian STDP in mushroom bodies facilitates the synchronous flow of olfactory information in locusts. *Nature*, 448(7154):709–713, August 2007.

[25] J.-Y. Chen, E. Marachlian, C. Assisi, R. Huerta, B. H. Smith, F. Locatelli, and M. Bazhenov. Learning Modifies Odor Mixture Processing to Improve Detection of Relevant Components. *Journal of Neuroscience*, 35(1):179–197, January 2015.

[26] Arij Daou and Daniel Margoliash. Intrinsic neuronal properties represent song and error in zebra finch vocal learning. *Nature Communications*, 11(1):1–17, February 2020.

[27] Arij Daou, Matthew Ross, Frank Johnson, Richard Hyson, and Richard Bertram. Electrophysiological characterization and computational models of hvc neurons in the zebra finch. *Journal of Neurophysiology*, 110:1227–1245, 2013.

[28] Marien de Bruyne, Kara Foster, and John R. Carlson. Odor Coding in the Drosophila Antenna. *Neuron*, 30(2):537–552, May 2001.

[29] Charles B. Delahunt and J. Nathan Kutz. Putting a bug in ML: The moth olfactory network learns to read MNIST. *arXiv:1802.05405 [cs, q-bio]*, February 2018. arXiv: 1802.05405.

[30] Charles B. Delahunt, Jeffrey A. Riffell, and J. Nathan Kutz. Biological Mechanisms for Learning: A Computational Model of Olfactory Learning in the Manduca sexta Moth, with Applications to Neural Nets. *arXiv:1802.02678 [cs, q-bio]*, February 2018. arXiv: 1802.02678.

[31] Charles Boise Delahunt. Smart as a Bug: A Computational Model of Learning in the Moth Olfactory Network, with Applications to Neural Nets. page 117.

[32] Allison J. Doupe and Patricia K. Kuhl. Birdsong and human speech: Common themes and mechanisms. *Annual Review of Neuroscience*, 22(1):567–631, 1999. PMID: 10202549.

[33] Patrick Dutar, Huan M. Vu, and David J. Perkel. Multiple cell types distinguished by physiological, pharmacological, and anatomic properties in nucleus hvc of the adult zebra finch. *Journal of Neurophysiology*, 80(4):1828–1838, 1998. PMID: 9772242.

[34] Zheng Fang, Adrian S. Wong, Kangbo Hao, Alexander J. A. Ty, and Henry D. I. Abarbanel. Precision annealing Monte Carlo methods for statistical data assimilation and machine learning. *Physical Review Research*, 2(1):013050, January 2020.

[35] Robert M. Fano. *Transmission of Information; A Statistical Theory of Communication*. MIT Press, 1961.

[36] Michale S. Fee and Constance Scharff. The songbird as a model for the generation and learning of complex sequential behaviors. *ILAR Journal*, 51(4):362–377, 2010.

[37] Richard FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical journal*, 1(6):445, 1961.

[38] Eric S. Fortune and Daniel Margoliash. Parallel pathways and convergence onto hvc and adjacent neostriatum of adult zebra finches (taeniopygia guttata). *Journal of Comparative Neurology*, 360(3):413–441, 1995.

[39] Roman V Frolov and Matti Weckström. Ion channels as therapeutic targets, part a. *Advances in Protein Chemistry and Structural Biology*, 103:1–386, 2016.

[40] I. M. Gelfand and S. V. Fomin. *Calculus of Variations*. Dover Publications, Inc., 1963.

[41] David E. Goldman. Potential, impedance, and rectification in membranes. *The Journal of General Physiology*, 27(1):37–60, 1943.

[42] Eyal Gruntman and Glenn C. Turner. Integration of the olfactory code across dendritic claws of single mushroom body neurons. *Nature Neuroscience*, 16(12):1821–1829, December 2013.

[43] Nitin Gupta, Swikriti Saran Singh, and Mark Stopfer. Oscillatory integration windows in neurons. *Nature Communications*, 7(1):1–10, December 2016.

[44] Nitin Gupta and Mark Stopfer. Olfactory Coding: Giant Inhibitory Neuron Governs Sparse Odor Codes. *Current Biology*, 21(13):R504–R506, July 2011.

[45] Elissa A. Hallem and John R. Carlson. Coding of Odors by a Receptor Repertoire. *Cell*, 125(1):143–160, April 2006.

[46] Elissa A Hallem, Michael G Ho, and John R Carlson. The Molecular Basis of Odor Coding in the Drosophila Antenna. *Cell*, 117(7):965–979, June 2004.

[47] O. P. Hamill, A. Marty, E. Neher, B. Sakmann, and F. J. Sigworth. Improved patch-clamp techniques for high-resolution current recording from cells and cell-free membrane patches. *Pflügers Archiv*, 391(2):85–100, Aug 1981.

[48] Kangbo Hao. *Data Assimilation and Precision Annealing Monte Carlo Method in Nonlinear Dynamical Systems*. PhD thesis, University of California San Diego, 2020.

[49] Erick O. Hernández-Ochoa and Martin F. Schneider. Voltage clamp methods for the study of membrane currents and sr ca^{2+} release in adult skeletal muscle fibres. *Progress in Biophysics and Molecular biology*, 108:98–118, 2012.

[50] A. L. Hodgkin and A. F. Huxley. The components of membrane conductance in the giant axon of *Loligo*. *The Journal of Physiology*, 116(4):473–496, April 1952.

[51] R. Huerta. Learning pattern recognition and decision making in the insect brain. pages 101–119, La Herradura, Spain, 2013.

[52] Ramón Huerta, Thomas Nowotny, Marta García-Sanchez, H. D. I. Abarbanel, and M. I. Rabinovich. Learning Classification in the Olfactory System of Insects. *Neural Computation*, 16(8):1601–1640, August 2004.

[53] Dezhe Z. Jin, Fethi M. Ramazanoglu, and H. Sebastian Seung. Intrinsic bursting enhances the robustness of a neural network model of sequence generation by avian brain area HVC. *Journal of Computational Neuroscience*, 23:283–299, 2007.

[54] Daniel Johnston and Samuel Miao-Sin Wu. *Foundations of Cellular Neurophysiology*. Bradford Books, MIT Press, 1995.

[55] Nirag Kadakia, Eve Armstrong, Daniel Breen, Arij Daou, Daniel Margoliash, and Henry Abarbanel. Nonlinear statistical data assimilation for hvcra neurons in the avian song system. *Biological Cybernetics*, 110(6):417–434, 2016.

[56] T. Kiss. Persistent Na-channels: origin and function. A review. *Acta Biologica Hungarica*, 59 Suppl:1–12, 2008.

[57] Satoshi Kojima, Mimi H. Kao, Allison J. Doupe, and Michael S. Brainard. The Avian Basal Ganglia Are a Source of Rapid Behavioral Variation That Enables Vocal Motor Exploration. *The Journal of Neuroscience*, 38(45):9635–9647, November 2018.

[58] Mark Kostuk. *Synchronization and statistical methods for the data assimilation of HVc neuron models*. PhD thesis, University of California San Diego, 2012.

[59] Mark Kostuk, Bryan A. Toth, C. Daniel Meliza, Daniel Margoliash, and Henry D. I. Abarbanel. Dynamical estimation of neuron and network properties II: path integral Monte Carlo methods. *Biological Cybernetics*, 106(3):155–167, March 2012.

[60] Sabine Krofczik, Randolf Menzel, and Martin P. Nawrot. Rapid Odor Processing in the Honeybee Antennal Lobe Network. *Frontiers in Computational Neuroscience*, 2, January 2009.

[61] Michinori Kubota and Ikuo Taniguchi. Electrophysiological characteristics of classes of neuron in the hvc of the zebra finch. *Journal of Neurophysiology*, 80(2):914–923, 1998. PMID: 9705478.

[62] P. S. Laplace. Memoir on the probability of causes of events. *Mathématique et de Physique,Tome Sixiéme*, pages 621–656, 1774.

[63] P.S. Laplace. Memoir of the probability of causes of events. *Statistical Science*, 1:365–378, 1986. Translation to English by S. M. Stigler.

[64] Gilles Laurent. Olfactory network dynamics and the coding of multidimensional signals. *Nature Reviews Neuroscience*, 3(11):884, November 2002.

[65] Gilles Laurent and Hananel Davidowitz. Encoding of Olfactory Information with Oscillating Neural Assemblies. *Science*, 265(5180):1872–1875, September 1994.

[66] Gilles Laurent, Katrina MacLeod, Mark Stopfer, and Michael Wehr. Spatiotemporal Structure of Olfactory Inputs to the Mushroom Bodies. *Learning & Memory*, 5(1):124–132, May 1998.

[67] Andrew C. Lin, Alexei Bygrave, Alix de Calignon, Tzumin Lee, and Gero Miesenböck. Sparse, Decorrelated Odor Coding in the Mushroom Body Enhances Learned Odor Discrimination. *Nature neuroscience*, 17(4):559–568, April 2014.

[68] Carlotta Martelli, John R. Carlson, and Thierry Emonet. Intensity Invariant Dynamics and Odor-Specific Latencies in Olfactory Receptor Neuron Response. *Journal of Neuroscience*, 33(15):6285–6297, April 2013.

[69] Ofer Mazor and Gilles Laurent. Transient Dynamics versus Fixed Points in Odor Representations by Locust Antennal Lobe Projection Neurons. *Neuron*, 48(4):661–673, November 2005.

[70] C. Daniel Meliza, Mark Kostuk, Hao Huang, Alain Nogaret, Daniel Margoliash, and Henry D. I. Abarbanel. Estimating parameters and predicting membrane voltages with conductance-based neuron models. *Biological Cybernetics*, 108(4):495–516, August 2014.

[71] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys.*, 21:1087–1092, June 1953.

[72] Richard Mooney. Different subthreshold mechanisms underlie song selectivity in identified hvc neurons of the zebra finch. *Journal of Neuroscience*, 20(14):5420–5436, 2000.

[73] Richard Mooney. Neuronal mechanisms for learned birdsong. *Learning and Memory*, 16:655–669, 2009.

[74] K. G. Murty and S. N. Kabadi. Some np-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39:117–129, 1987.

[75] Katherine I. Nagel and Rachel I. Wilson. Biophysical mechanisms underlying olfactory receptor neuron dynamics. *Nature neuroscience*, 14(2):208–216, February 2011.

[76] Radford Neal. Mcmc using hamiltonian dynamics. In Andrew Gelman, Galin Jones, and Xiao-Li Meng, editors, *Handbook of Markov Chain Monte Carlo*, chapter 5. Chapman and Hall; CRC Press, 2011.

[77] Fernando Nottebohm, Tegner M. Stokes, and Christiana M. Leonard. Central control of song in the canary, serinus canarius. *Journal of Comparative Neurology*, 165(4):457–486, 1976.

[78] Mario Pannunzi and Thomas Nowotny. Odor Stimuli: Not Just Chemical Identity. *Frontiers in Physiology*, 10, 2019.

[79] Maria Papadopoulou, Stijn Cassenaer, Thomas Nowotny, and Gilles Laurent. Normalization for Sparse Encoding of Odors by a Wide-Field Interneuron. *Science (New York, N.Y.)*, 332(6030):721–725, May 2011.

[80] Javier Perez-Orive, Ofer Mazor, Glenn C. Turner, Stijn Cassenaer, Rachel I. Wilson, and Gilles Laurent. Oscillations and sparsening of odor representations in the mushroom body. *Science (New York, N.Y.)*, 297(5580):359–365, July 2002.

[81] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing, Third Edition*. Cambridge University Press, 2007.

[82] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical recipes: the art of scientific computing*. Cambridge University Press, 2007.

[83] John C. Quinn. *A path integral approach to data assimilation in stochastic nonlinear systems*. PhD thesis, University of California San Diego, 2010.

[84] John C. Quinn and Henry D. I. Abarbanel. Data Assimilation using a GPU Accelerated Path Integral Monte Carlo Approach. *Journal of Computational Physics*, 230(22):8168–8178, September 2011. arXiv: 1103.4887.

[85] Rodrigo Quian Quiroga and Stefano Panzeri. *Principles of Neural Coding*, page 237. CRC Press, May 2013.

[86] M. Rabinovich, A. Volkovskii, P. Lecanda, R. Huerta, H. D. I. Abarbanel, and G. Laurent. Dynamical Encoding by Networks of Competing Neuron Groups: Winnerless Competition. *Physical Review Letters*, 87(6):068102, July 2001.

[87] M. I. Rabinovich, R. Huerta, A. Volkovskii, H. D. I. Abarbanel, M. Stopfer, and G. Laurent. Dynamical coding of sensory information with competitive networks. *Journal of Physiology-Paris*, 94(5):465–471, December 2000.

[88] Vahid Rahmati, Knut Kirmse, Dimitrije Marković, Knut Holthoff, and Stefan J. Kiebel. Inferring Neuronal Dynamics from Calcium Imaging Data Using Biophysical Models and Bayesian Inference. *PLOS Computational Biology*, 12(2):e1004736, February 2016.

[89] Baranidharan Raman, Joby Joseph, Jeff Tang, and Mark Stopfer. Temporally Diverse Firing Patterns in Olfactory Receptor Neurons Underlie Spatiotemporal Neural Codes for Odors. *The Journal of Neuroscience*, 30(6):1994–2006, February 2010.

[90] Gautam Reddy, Joseph D Zak, Massimo Vergassola, and Venkatesh N Murthy. Antagonism in olfactory receptor neurons and its implications for the perception of odor mixtures. *eLife*, 7:e34958, April 2018.

[91] H. M. Robertson and K. W. Wanner. The chemoreceptor superfamily in the honey bee, Apis mellifera: Expansion of the odorant, but not gustatory, receptor family. *Genome Research*, 16(11):1395–1403, October 2006.

[92] Paul Joseph Rozdeba. *Nonlinear Inference in Partially Observed Physical Systems and Deep Neural Networks*. PhD thesis, UC San Diego, 2018.

[93] Pavel Sanda, Tiffany Kee, Nitin Gupta, Mark Stopfer, and Maxim Bazhenov. Classification of odorants across layers in locust olfactory pathway. *Journal of Neurophysiology*, 115(5):2303–2316, May 2016.

[94] Yuhua Shang, Adam Claridge-Chang, Lucas Sjulson, Marc Pypaert, and Gero Miesenböck. Excitatory Local Circuits and Their Implications for Olfactory Processing in the Fly Antennal Lobe. *Cell*, 128(3):601–612, February 2007.

[95] Sasha Shirman. *Strategic Monte Carlo and Variational Methods in Statistical Data Assimilation for Nonlinear Dynamical Systems*. PhD thesis, University of California San Diego, 2018.

[96] Kristina Simonyan, Barry Horwitz, and Erich Jarvis. Dopamine regulation of human speech and bird song: A critical review. *Brain and Language*, 122:142–150, 2012.

[97] Vijay Singh, Nicolle R. Murphy, Vijay Balasubramanian, and Joel D. Mainland. Competitive binding predicts nonlinear responses of olfactory receptors to complex mixtures. *Proceedings of the National Academy of Sciences*, 116(19):9598–9603, May 2019.

[98] D. Sterratt, B. Graham, A. Gillies, and D. Willshaw. *Principles of Computational Modelling in Neuroscience*. Cambridge University Press, 2011.

[99] Marcel Stimberg, Romain Brette, and Dan FM Goodman. Brian 2, an intuitive and efficient neural simulator. *eLife*, 8:e47314, August 2019.

[100] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to Sequence Learning with Neural Networks. page 9, 2014.

[101] Ikuko Teramitsu, Lili C. Kudo, Sarah E. London, Daniel H. Geschwind, and Stephanie A. White. Parallel foxp1 and foxp2 expression in songbird and human brain predicts functional interaction. *Journal of Neuroscience*, 24(13):3152–3163, 2004.

[102] Bryan A Toth, Mark Kostuk, C Daniel Meliza, Daniel Margoliash, and Henry DI Abarbanel. Dynamical estimation of neuron and network properties i: variational methods. *Biological cybernetics*, 105(3-4):217–237, 2011.

[103] Michael Wehr and Gilles Laurent. Odour encoding by temporal sequences of firing in oscillating neural assemblies. *Nature*, 384(6605):162–166, November 1996.

[104] Rachel I. Wilson. Early Olfactory Processing in *Drosophila* : Mechanisms and Principles. *Annual Review of Neuroscience*, 36(1):217–241, July 2013.

[105] Adrian S. Wong, Kangbo Hao, Zheng Fang, and Henry D. I. Abarbanel. Precision Annealing Monte Carlo Methods for Statistical Data Assimilation: Metropolis-Hastings Procedures. *arXiv:1901.04598 [physics, stat]*, January 2019. arXiv: 1901.04598.

[106] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, March 2006.

[107] J. Ye, N. Kadakia, P. J. Rozdeba, H. D. I. Abarbanel, and J. C. Quinn. Improved variational methods in statistical data assimilation. *Nonlinear Processes in Geophysics*, 22(2):205–213, 2015.

[108] Jingxin Ye. *Systematic Annealing Approach for Statistical Data Assimilation*. PhD thesis, University of California San Diego, 2016.

[109] Jingxin Ye, Daniel Rey, Nirag Kadakia, Michael Eldridge, Uriel I. Morone, Paul Rozdeba, and Henry Abarbanel. Systematic variational method for statistical nonlinear state and parameter estimation. *Phys. Rev. E*, 92:052901, November 2015.