ABSTRACT

DATA ASSIMILATION FOR CONDUCTANCE-BASED NEURONAL MODELS

by
Matthew Moye

This dissertation illustrates the use of data assimilation algorithms to estimate unobserved variables and unknown parameters of conductance-based neuronal models. Modern data assimilation (DA) techniques are widely used in climate science and weather prediction, but have only recently begun to be applied in neuroscience. The two main classes of DA techniques are sequential methods and variational methods. Throughout this work, twin experiments, where the data is synthetically generated from output of the model, are used to validate use of these techniques for conductance-based models observing only the voltage trace. In Chapter 1, these techniques are described in detail and the estimation problem for conductance-based neuron models is derived. In Chapter 2, these techniques are applied to a minimal conductance-based model, the Morris-Lecar model. This model exhibits qualitatively different types of neuronal excitability due to changes in the underlying bifurcation structure and it is shown that the DA methods can identify parameter sets that produce the correct bifurcation structure even with initial parameter guesses that correspond to a different excitability regime. This demonstrates the ability of DA techniques to perform nonlinear state and parameter estimation, and introduces the geometric structure of inferred models as a novel qualitative measure of estimation success.

Chapter 3 extends the ideas of variational data assimilation to include a control term to relax the problem further in a process that is referred to as nudging from the geoscience community. The nudged 4D-Var is applied to twin experiments from a more complex, Hodgkin-Huxley-type two-compartment model for various

time-sampling strategies. This controlled 4D-Var with nonuniform time-samplings is then applied to voltage traces from current-clamp recordings of suprachiasmatic nucleus neurons in diurnal rodents to improve upon our understanding of the driving forces in circadian ($\sim$ 24-hour) rhythms of electrical activity.

In Chapter 4 the complementary strengths of 4D-Var and UKF are leveraged to create a two-stage algorithm that uses 4D-Var to estimate fast timescale parameters and UKF for slow timescale parameters. This coupled approach is applied to data from a conductance-based model of neuronal bursting with distinctive slow and fast time-scales present in the dynamics. In Chapter 5, the ideas of identifiability and sensitivity are introduced. The Morris-Lecar model and a subset of its parameters are shown to be identifiable through the use of numerical techniques. Chapter 6 frames the selection of stimulus waveforms to inject into neurons during patch-clamp recordings as an optimal experimental design problem. Results on the optimal stimulus waveforms for improving the identifiability of parameters for a Hodgkin-Huxley-type model are presented. Chapter 7 shows the preliminary application of data assimilation for voltage-clamp, rather than current-clamp, data and expands on voltage-clamp principles to formulate a reduced assimilation problem driven by the observed voltage. Concluding thoughts are given in Chapter 8.

DATA ASSIMILATION FOR CONDUCTANCE-BASED NEURONAL MODELS

by
Matthew Moye

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology and
Rutgers, The State University of New Jersey – Newark
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Mathematical Sciences

Department of Mathematical Sciences
Department of Mathematics and Computer Science, Rutgers-Newark

May 2020

APPROVAL PAGE

DATA ASSIMILATION FOR CONDUCTANCE-BASED NEURONAL MODELS

Matthew Moye

Dr. Casey O. Diekman, Dissertation Advisor — Date
Associate Professor of Mathematics, New Jersey Institute of Technology

---

Dr. Horacio G. Rotstein, Committee Member — Date
Professor of Biological Sciences, New Jersey Institute of Technology

---

Dr. Victor V. Matveev, Committee Member — Date
Professor of Mathematics, New Jersey Institute of Technology

Dr. Richard O. Moore, Committee Member — Date
Director of Programs and Services, Society of Industrial and Applied Mathematics

---

Dr. Timothy Sauer, Committee Member — Date
Professor of Mathematics, George Mason University

BIOGRAPHICAL SKETCH

Author: Matthew Moye

Degree: Doctor of Philosophy

Date: May 2020

Undergraduate and Graduate Education:

- Doctor of Philosophy in Mathematical Sciences
  New Jersey Institute of Technology, Newark, NJ, 2020
- Bachelors of Science in Quantitative Biology
  University of Delaware, Newark, DE, 2014

Major: Mathematical Sciences

Presentations and Publications:

Matthew J. Moye, Casey O. Diekman,"Data Assimilation for Conductance-Based Neuronal Models", Quantitative Biology and Medicine at Exeter Seminar University of Exeter, Exeter, UK, January 2020.

Matthew J. Moye, "Estimating Input for Striatal Neurons from In Vivo Recordings", IBM Research Intern Poster Session, IBM Thomas J Watson Research Center, Yorktown Heights, NY, August 2019.

Matthew J. Moye, Casey O. Diekman, "Data Assimilation Methods for Neuronal State and Parameter Estimation", Research Training Group Tutorial Workshop on Parameter Estimation for Biological Models, NC State University, Raleigh, NC, July 2018.

Matthew J. Moye, Casey O. Diekman, "Data Assimilation Methods for Neuronal State and Parameter Estimation". J. Math. Neurosc. (2018) 8: 11. https://doi.org/10.1186/s13408-018-0066-8

Matthew J. Moye, Casey O. Diekman, "Data Assimilation Methods for Neuronal State and Parameter Estimation", Institute for Brain and Neuroscience Research Graduate Student/Post-Doctoral Research Showcase, New Jersey Institute of Technology Newark, NJ, March 2018.

Matthew J. Moye, Casey O. Diekman, "Data Assimilation and Electrophysiological Modeling of Mammalian Circadian Clock Neurons" The 3rd International Conference on Mathematical Neuroscience, Boulder, CO, May 2017.

Matthew J. Moye, Casey O. Diekman, "Data Assimilation and Electrophysiological Modeling of Mammalian Circadian Clock Neurons" SIAM Conference on Applications of Dynamical Systems, Snowbird, UT, May 2017

Matthew J. Moye, Casey O. Diekman, "Data Assimilation in Circadian Cells", Dynamical Systems and Data Analysis in Neuroscience: Bridging the Gap, Mathematical Biosciences Institute, Columbus, OH, October 2016.

ACKNOWLEDGMENT

First and foremost, I would like to thank my dissertation advisor, Dr. Casey Diekman, for everything he has done for me in these past several years. My professional trajectory might have gone an entirely different way had he not taken the time and effort to discuss his research ideas with me during my first year in graduate school. He has been the most engaging and spirited mentor that one could have, as well as great friend.

I would also like to thank my committee members for their continued investment in my success. Dr. Horacio Rotstein has always showed a sincere curiosity in my work, and computational neuroscience in general, so I would like to thank him for helping me think about problems from many different angles and pushing me to connect with the broader community. Dr. Victor Matveev was the professor for my first course in computational neuroscience and I want to truly thank him for his commitment to comprehensive teaching excellence. Dr. Richard Moore provided wonderful insight into data assimilation and was a constant source of enthusiasm for applied mathematics. Dr. Timothy Sauer paved the way for much of my work in nonlinear filtering and I would like to thank him for discussions early during my PhD.

I doubt I would have made it this far without the friends I made at NJIT. While by no means a comprehensive list, I want to give a special shout-out to Andrew, Mahdi, Malik, and RJ for the laughs, the Smash Bros, the White Mana, and everything in between. To Soheil and Emel, thanks for the insightful discussions on mathematical neuroscience and for helping maintain a wonderful office atmosphere. I would also like to thank my graduate student mentor, Amin, for his friendship and being a role model for dedication to quality research.

Thank you to Dr. Beatriz Baño Otalora, Dr. Mino Belle, and Dr. Robert Lucas for sharing their SCN data and for the receptive collaboration. I want to extend thanks to Tyrus Berry and Franz Hamilton for helpful discussions about the UKF and for sharing code. Thanks also to Dr. Henry Abarbanel and his students Dr. Nirag Kadakia and Dr. Paul Rozdeba, amongst others, for discussions on variational data assimilation and for sharing code as well. I want to also thank my funding sources; this work was partially supported by NSF DMS grants 1412877 and 1555237, and U.S. Army Research Office grant W911NF-16-1-0584.

Lastly, I would like to thank my family for their encouragement. My parents, Bill and Dee, and my sister, Tina, have always believed in me and pushed me to succeed in whatever I tried. To my girlfriend, Callan, I want to thank her for building me up every day and instilling in me the motivation I needed to complete this work.

TABLE OF CONTENTS

Chapter **Page**

1 INTRODUCTION 1

1.1 Parameter Estimation and Data Assimilation 1

1.1.1 The Parameter Estimation Problem 1

1.1.2 Data Assimilation 3

1.2 Nonlinear Data Assimilation Methods 8

1.2.1 Nonlinear Filtering 8

1.2.2 Variational Methods 15

2 DATA ASSIMILATION APPLIED TO THE MORRIS-LECAR MODEL . 19

2.1 Twin Experiments 19

2.1.1 Recovery of Bifurcation Structure 19

2.1.2 Morris-Lecar Model 20

2.1.3 Results with UKF 23

2.1.4 Results with 4D-Var 29

2.2 Application to Bursting Regimes of the Morris-Lecar Model 38

2.2.1 Results with UKF 40

2.2.2 Results with 4D-Var 42

2.3 Discussion and Conclusions of Twin Experiments 44

3 CONTROLLED 4D-VAR AND APPLICATION TO CIRCADIAN NEURONS 51

3.1 4D-Var Formulation 51

3.2 Twin Experiment: Two-Compartment Hodgkin-Huxley Model 62

3.2.1 Two-Compartment NaKL 62

3.2.2 Twin Experiments on Two Compartment NaKL 65

3.3 Application to Rhabdomys Circadian Neurons 73

3.3.1 Behavioral Responses to Hyperpolarizing Pulses 86

3.3.2 Behavioral Responses to Depolarizing Pulses 89

TABLE OF CONTENTS
(Continued)

Chapter **Page**

4 TWO STAGE ESTIMATION COMBINING 4D-VAR AND UKF 98

5 IDENTIFIABILITY OF CONDUCTANCE-BASED MODELS 107

6 OPTIMAL DESIGN OF INJECTED CURRENTS 114

6.1 The Optimal Injected Current Problem 114

6.2 Optimal Current for Hodgkin-Huxley Type Model 123

7 DATA ASSIMILATION FOR VOLTAGE-CLAMP RECORDINGS 127

7.1 Real Time Estimation with Voltage-Clamp Data 127

7.2 Voltage-Forcing and Decoupling the System 130

7.2.1 Pushing the Synchronization 136

8 CONCLUDING THOUGHTS 138

APPENDIX A CH2: MATLAB CODE 139

APPENDIX B CH5: MATLAB CODE 149

REFERENCES 164

LIST OF TABLES

Table **Page**

2.1 Morris-Lecar Parameter Values 21

2.2 UKF Parameter Estimates at End of Estimation Window, with Observed Data from Bifurcation Regime 't' and Initial Parameter Guesses Corresponding to Bifurcation Regime 'g' 29

2.3 Bounds Used During 4D-Var Estimation for the Results Shown in Tables 2.4 and 2.5 32

2.4 4D-Var Parameter Estimates at the End of the Optimization for each Bifurcation Regime 32

2.5 4D-Var Parameter Estimates at the End of the Optimization for Each Bifurcation Regime, Derivative Provided 32

2.6 Bounds Used During 4D-Var Estimation for the Results Shown in Tables 2.7 and 2.8 33

2.7 4D-Var Parameter Estimates at the End of the Optimization for each Bifurcation Regime with Loose Parameter Bounds 33

2.8 4D-Var Parameter Estimates at the End of the Optimization for each Bifurcation Regime with Hessian Provided 33

2.9 Parameters for Bursting in the Modified Morris-Lecar Model 41

2.10 UKF Parameter Estimates for each Bursting Regime 42

2.11 4D-Var Parameter Estimates for each Bursting Regime 44

2.12 Comparison of the Sequential (UKF) and Variational (4D-Var) Approaches to Data Assimilation 46

3.1 Bounds of Parameters Used in Estimation of Two Compartment NaKL Model 67

3.2 Bounds of Parameters Used in Estimation of SCN Model 80

4.1 BRS Estimated Model Parameters from Two-Stage DA Strategy. 105

5.1 Matrix of ORC for Morris-Lecar Model Evaluated for Hopf Parameters . 113

6.1 Relative Error for Various Currents for Estimating NaKL Model 124

LIST OF TABLES
(Continued)

Table **Page**

6.2 NaKL Estimated Model Parameters from Strong 4D-Var with Nudging for Various Applied Currents 126

7.1 Voltage-Clamp UKF Results for I_{Na} 129

LIST OF FIGURES

Figure **Page**

1.1 Unscented transformation for the Morris-Lecar model. 10

1.2 Example of iterative estimation in UKF. 15

1.3 Example cost function for 4D-Var. 18

2.1 Three different excitability regimes of the Morris-Lecar model. 22

2.2 State estimates for UKF. This example corresponds to initializing with parameters from the HOPF regime and attempting to correctly estimate those of the SNIC regime. The noisy observed voltage V and true unobserved gating variable n are shown in blue, and their UKF estimates are shown in red. 25

2.3 Parameter estimates for UKF. This example corresponds to initializing with parameters from the HOPF, SNIC, and HOMO regimes and attempting to correctly estimate those of the SNIC, HOMO, and HOPF regimes (left to right column, respectively). The blue curves are the estimates from the UKF, with ± 2 standard deviations from the mean based on the filter estimated covariance shown in red. The gray lines indicate the true parameter values. 26

2.4 Bifurcation diagrams for UKF twin experiments. The gray lines correspond to the true diagrams, and the blue dotted lines correspond to the diagrams produced from the estimated parameters in 2.2. 27

2.5 UKF state estimates of n for the Morris-Lecar model with 100 different initial guesses of the state sampled from $\mathcal{U}(0, 1)$, with all other parameters held fixed. 29

2.6 **A:** UKF results from runs of the t:SNIC/g:HOPF twin experiment for various parameter combinations of λ and α_I. 30

2.7 Example of 4D-Var assimilation initializing with parameters from the Hopf regime but observational data from the SNIC regime. 34

2.8 Bifurcation diagrams for 4D-Var twin experiments. The gray lines correspond to the true diagrams, and the blue dotted lines correspond to the diagrams produced from the estimated parameters in Table 2.4. 35

2.9 Example parameter estimation with 4D-Var initializing with Hopf parameter regime and estimating parameters of SNIC regime. 36

LIST OF FIGURES
(Continued)

Figure **Page**

2.10 **A:** Sparsity pattern for the Hessian of the cost function for the Morris-Lecar equations for $N + 1 = 2001$ time points. The final eight rows (and symmetrically the last eight columns) depict how the states at each time depend upon the parameters. **B:** The top left corner of the Hessian shown in (A). 36

2.11 **A** shows the logarithm of the value of the cost function for a twin experiment initialized with parameters from the Hopf regime but observational data from the SNIC regime. **B** shows the bifurcation diagrams produced from parameter estimates for selected iterations . . 37

2.12 Bursting model bifurcation diagrams and trajectories. 40

2.13 Bifurcation diagrams for UKF twin experiments for the bursting Morris-Lecar model. The gray lines correspond to the true diagrams, and the blue dotted lines correspond to the diagrams produced from the estimated parameters in Table 2.10. 43

2.14 Bifurcation diagrams for 4D-Var twin experiments for the bursting Morris-Lecar model. The gray lines correspond to the true diagrams, and the blue dotted lines correspond to the diagrams produced from the estimated parameters in Table 2.11. 44

3.1 Circuit diagram of two-compartment NaKL model. Current is injected into the soma, where a probe will measure the voltage, V_m. 64

3.2 Example simulations of two-compartment NaKL model. Left panel: First row depicts the somatic compartment, the second row depicts the axonal compartment, and final row are the stimulating applied currents. Right panel: voltage speed plots for these compartments for steady-firing. 64

3.3 Bifurcation diagrams for the two compartment NaKL model **A** and one compartment model **B**. 65

3.4 Dynamics of our data used for twin experiment of the two compartment NaKL model. The only observed quantity, shown in blue, is the somatic voltage perturbed by white noise. The remaining unobserved variables, shown in red, are contained in the axonal compartment. 66

3.5 Statistics of 50kHz sampling 2 compartment twin experiments using 100ms ramp injected current. For visual purposes, a scaling factor $\alpha = 100$ is applied. 69

3.6 $R(t)$ for a preferred local minima. Left panel is $R(t)$ over time. Right panel is a histogram for the range of $R(t)$ values. 70

LIST OF FIGURES
(Continued)

Figure **Page**

3.7 Simulations of the two compartment NaKL model over a longer time window than the orignal 100ms assimilation window. The estimated parameter sets from the control and non-controlled 4D-var closely align for the data even beyond the assimilation window. 70

3.8 Comparison of $R(t)$ for a local minima exhibiting preferential dependence upon the control to match the observed voltage. 71

3.9 Observations of the two compartment NaKL model using different sampling rates. 71

3.10 Comparison of prediction windows for data generated by using coarser sampling. The top panel uses a collocation time difference and observational time grid of .1 ms. The bottom grid uses a collocation time difference of .02 ms and an observational time difference of 1 ms. 73

3.11 Rhabdomys electrical variation over day/night cycle. Colormap corresponds to ZT of data collection, illustrating 5 seconds of data. Data was recorded by Mino Belle and Beatriz Baño-Otálora in the laboratory of Robert Lucas at the University of Manchester. 78

3.12 Sampling strategy for SCN data. Black corresponds to full data used, whereas gray corresponds to downsampling used. Red 'x' shows the midpoint for the high-resolution sampling. 82

3.13 SCN data used for assimilation. Blue corresponds to full current-clamp data, whereas orange corresponds to training data used for assimilation. 83

3.14 SCN data assimilation for training data for cell exhibiting rebound from hyperpolarization. 85

3.15 Bifurcation diagram and FI curve for estimated model of Rhabdomys cell exhibiting rebound. 85

3.16 SCN rebound mechanism. 87

3.17 SCN rebound speed plot. 88

3.18 SCN data assimilation for training data for cell exhibiting delay from hyperpolarization. 91

3.19 SCN delay mechanism. 92

3.20 SCN data assimilation for training data for cell which is non-adapting when exposed to depolarizing pulses. 93

3.21 SCN non-adapting mechanism. 94

LIST OF FIGURES
(Continued)

Figure **Page**

3.22 SCN data assimilation for training data for cell which is adapting-firing when exposed to depolarizing pulses. 95

3.23 SCN adapting-firing mechanism. 96

3.24 SCN data assimilation for training data for cell which is adapting-silent when exposed to depolarizing pulses. 96

3.25 SCN adapting-silent mechanism. 97

4.1 pre-Bötzinger complex neuron bursting time series. 99

4.2 Utilizing UKF and 4D-Var for long time-series data. 100

4.3 Dynamics for BRS Model 1 at the onset of the burst. 103

4.4 Burst for BRS Model 1. 104

4.5 Burst comparison between BRS Model 1 and estimated model via two-stage strategy 105

6.1 Voltage and I_{app} versus t and $\frac{dV}{dt}$ vs V for the NaKL model. The red is under constant applied forcing and the grey is forced with the scaled output of the Lorenz '63 model. 118

6.2 Ellipsoid representation of the covariance matrix, $\mathbf{\Sigma}$. 123

6.3 A and D-optimal applied currents derived using OED strategies for NaKL model. 125

6.4 Other nominal currents used as comparisons for success at estimating parameters of NaKL model. 125

7.1 Voltage clamp data for I_{Na}. 129

7.2 Full time course of voltage clamp data from I_{Na}. 130

7.3 Recreation of voltage-clamp data using the estimated and initial parameter sets of Table 7.1 compared to the true data. 131

7.4 Simulation of HH model using the estimated and initial parameter sets of Table 7.1 compared to the true data. 131

CHAPTER 1

INTRODUCTION

1.1 Parameter Estimation and Data Assimilation

1.1.1 The Parameter Estimation Problem

The goal of conductance-based modeling is to be able to reproduce, explain, and predict the electrical behavior of a neuron or networks of neurons. Conductance-based modeling of neuronal excitability began in the 1950s with the Hodgkin-Huxley model of action potential generation in the squid giant axon [56]. This modeling framework uses an equivalent circuit representation for the movement of ions across the cell membrane:

$$C\frac{dV}{dt} = I_{\text{app}} - \sum_{\text{ion}} I_{\text{ion}} \tag{1.1}$$

where V is membrane voltage, C is cell capacitance, I_{ion} are ionic currents, and I_{app} is an external current applied by the experimentalist. The ionic currents arise from channels in the membrane that are voltage- or calcium-gated and selective for particular ions, such sodium (Na^+) and potassium (K^+). For example, consider the classical Hodgkin-Huxley currents:

$$I_{\text{Na}} = g_{\text{Na}} m^3 h (V - E_{\text{Na}}) \tag{1.2}$$

$$I_{\text{K}} = g_{\text{K}} n^4 (V - E_{\text{K}}). \tag{1.3}$$

The maximal conductance g_{ion} is a parameter that represents the density of channels in the membrane. The term $(V - E_{\text{ion}})$ is the driving force, where the equilibrium potential E_{ion} is the voltage at which the concentration of the ion inside and outside

of the cell is at steady state. The gating variable m is the probability that one of three identical subunits of the sodium channel is "open", and the gating variable h is the probability that a fourth subunit is "inactivated". Similarly, the gating variable n is the probability that one of four identical subunits of the potassium channel is open. For current to flow through the channel, all subunits must be open and not inactivated. The rate at which subunits open, close, inactivate, and de-inactivate depends on the voltage. The dynamics of the gating variables are given by

$$\frac{dx}{dt} = \alpha_x(V)(1-x) + \beta_x(V)x \tag{1.4}$$

where $\alpha_x(V)$ and $\beta_x(V)$ are nonlinear functions of voltage with several parameters.

The parameters of conductance-based models are typically fit to voltage-clamp recordings. In these experiments, individual ionic currents are isolated using pharmacological blockers and one measures current traces in response to voltage pulses. However, many electrophysiological datasets consist of current-clamp rather than voltage-clamp recordings. In current-clamp, one records a voltage trace (e.g. a series of action potentials) in response to injected current. Fitting a conductance-based model to current-clamp data is challenging because the individual ionic currents have not been measured directly. In terms of the Hodgkin-Huxley model, only one state variable (V) has been observed, and the other three state variables (m, h, and n) are unobserved. Conductance-based models of neurons often contain several ionic currents, and therefore more unobserved gating variables and more unknown or poorly known parameters. For example, a model of HVC neurons in the zebra finch has 9 ionic currents, 12 state variables, and 72 parameters [79]. An additional difficulty in attempting to fit a model to a voltage trace is that if one performs a least-squares minimization between the data and model output, then small

differences in the timing of action potentials in the data and the model can result in large error [74]. Data assimilation methods have the potential to overcome these challenges by performing state estimation (of both observed and unobserved states) and parameter estimation simultaneously.

1.1.2 Data Assimilation

Data assimilation can broadly be considered to be the optimal integration of observations from a system to improve estimates of a model output describing that system. Data assimilation (DA) is used across the geosciences, such as in studying land hydrology and ocean currents, as well as studies of climates of other planets [24, 50, 93]. An application of DA familiar to the general public is its use in numerical weather prediction [9]. In the earth sciences, the models are typically high dimensional partial differential equations (PDEs) that incorporate dynamics of the many relevant governing processes, and the state system is a discretization of those PDEs across the spatial domain. These models are nonlinear and chaotic, with interactions of system components across temporal and spatial scales. The observations are sparse in time, contaminated by noise, and only partial with respect to the full state space.

In neuroscience, models can also be highly nonlinear and potentially chaotic. When dealing with network dynamics or wave propagation, the state space can be quite large, and there are certainly components of the system for which one would not have time course measurements [57]. As mentioned above, if one has a biophysical model of a single neuron, and measurements from a current-clamp protocol, the only quantity in the model that is actually measured is the membrane voltage. The question then becomes how does one obtain estimates of the full system state?

To begin, we assume we have a model to represent the system of interest and a way to relate observations we have of that system to the components of the model. Additionally, we allow, and naturally expect, there to be errors present in the model

and measurements. To start, let's consider first a general model with linear dynamics and a set of discrete observations which depend linearly on the system components:

$$x_{k+1} = Fx_k + \omega_{k+1}, \qquad x_k \in \mathbb{R}^L \tag{1.5}$$

$$y_{k+1} = Hx_{k+1} + \eta_{k+1}, \quad y_{k+1} \in \mathbb{R}^M. \tag{1.6}$$

In this state-space representation, x_k is interpreted as the state of the system at some time t_k, and y_k are our observations. For application in neuroscience, we can take $M \ll L$ as few state variables of the system are readily observed. F is our model which maps states x_k between time points t_k and t_{k+1}. H is our observation operator which describes how we connect our observations y_{k+1} to our state space at t_{k+1}. The random variables ω_{k+1} and η_{k+1} represent model error and measurement error respectively. A simplifying assumption is that our measurements are diluted by Gaussian white noise, and that the error in the model can be approximated by Gaussian white noise as well. Then $\omega_k \sim \mathcal{N}(0, Q_k)$ and $\eta_k \sim \mathcal{N}(0, R_k)$, where Q_k is our model error covariance matrix and R_k is our measurement error covariance matrix. We will assume these distributions for the error terms for the remainder of the dissertation.

We now have defined a stochastic dynamical system where we have characterized the evolution of our states and observations therein based upon assumed error statistics. The goal is now to utilize these transitions to construct methods to best estimate the state x over time. To approach this goal, it may be simpler to consider the evaluation of *background* knowledge of system compared to what we actually observe from a measuring device. Background knowledge, in our context, would be some representative model based on our physical understanding of the system. Consider

the following cost function [6]:

$$C(x) = \frac{1}{2} \|y - Hx\|_R^2 + \frac{1}{2} \left\|x - x^b\right\|_{P^b}^2 \tag{1.7}$$

where $\|z\|_A^2 = z^T A^{-1} z$. P^b acts to give weight to certain background components x^b, and R acts in the same manner to the measurement terms. The model or background term acts to regularize the cost function. Specifically, trying to minimize $\frac{1}{2} \|y - Hx\|_R^2$ is underdetermined with respect to the observations unless we can observe the full system, and the measurement term aims to inform the problem of the unobserved components. We are minimizing over state components x. In this way, we balance the influence of what we think we know about the system, such as from a model, compared to what we can actually observe. The cost function is minimized from

$$\nabla C = (H^T R^{-1} H + (P^b)^{-1}) x^a - (H^T R^{-1} y + (P^b)^{-1} x^b) = 0 \tag{1.8}$$

This can be restructured as

$$x^a = x^b + K(y - Hx^b) \tag{1.9}$$

where

$$K = P^b H^T (H P^b H^T + R)^{-1} \tag{1.10}$$

The optimal Kalman gain matrix, K, acts as a weighting of the confidence of our observations to the confidence of our background information given by the model.

If the background uncertainty is relatively high or the measurement uncertainty is relatively low, K is larger, which more heavily weights the *innovation* $y - Hx^b$.

The minimum of the cost functional (1.7), which is explicitly given by equation (1.9) can be interpreted as the solution of a single time step in our state space problem (1.5)-(1.6). In the DA literature, minimizing this cost function independent of time is referred to as 3D-Var. However, practically we are interested in problems resembling:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} \|y_k - Hx_k\|^2_{R_k} + \frac{1}{2}\sum_{k=0}^{N-1} \|x_{k+1} - Fx_k\|^2_{P^b_k} \tag{1.11}$$

where formally the background component x^b has now been replaced with our model. Now we are concerned with minimizing over an observation window with $N+1$ time points. **Variational methods**, specifically "weak 4D-Var", seek minima of equation (1.11) either by formulation of an adjoint problem [7], or directly from numerical optimization techniques.

Alternatively, **sequential data assimilation** aims to use information from previous time points $t_{k-l}, t_{k-l+1}, \ldots t_k$ (or, in the case of the classical Kalman filter, just t_k), to optimally estimate the state at t_{k+1}. The classical Kalman filter utilizes the form of equation (1.10), which minimizes the trace of the posterior covariance matrix of the system at step $k+1$, P^a_{k+1}, to update the state estimate and system uncertainty.

The Kalman filtering algorithm takes the following form:

Our *analysis* estimate, $\hat{x}^a_k$ from the previous iteration, is mapped through the linear model operator F to obtain our *forecast* estimate $\hat{x}^f_{k+1}$:

$$\hat{x}^f_{k+1} = F_k \hat{x}^a_k. \tag{1.12}$$

The observation operator, H, is applied to the forecast estimate to generate the measurement estimate, $\hat{y}^f_{k+1}$:

$$\hat{y}^f_{k+1} = H_{k+1}\hat{x}^f_{k+1}. \tag{1.13}$$

The forecast estimate covariance P^f_{k+1} is generated through calculating the covariance from the model and adding it with the model error covariance Q_k:

$$P^f_{k+1} = F_k P^a_k F^T_k + Q_k. \tag{1.14}$$

Similarly, we can construct the measurement covariance estimate by calculating the covariance from our observation equation and adding it to the measurement error covariance R_k:

$$P^y_{k+1} = H_{k+1}P^f_{k+1}H^T_{k+1} + R_k. \tag{1.15}$$

The Kalman gain is defined analogously to equation (1.10):

$$K_{k+1} = P^f_{k+1}H^T_{k+1}(P^y_{k+1})^{-1}. \tag{1.16}$$

The covariance and the mean estimate of the system are updated through a weighted sum with the Kalman gain:

$$P^a_{k+1} = (I - K_{k+1}H_{k+1})P^f_{k+1} \tag{1.17}$$

$$\hat{x}^a_{k+1} = \hat{x}^f_{k+1} + K_{k+1}(y_{k+1} - \hat{y}^f_{k+1}). \tag{1.18}$$

These equations can be interpreted as a predictor-corrector method, where the predictions of the state estimates are $\hat{x}^f_{k+1}$ with corresponding uncertainties P^f_{k+1} in the *forecast*. The correction, or *analysis*, step linearly interpolates the forecast predictions with observational readings.

1.2 Nonlinear Data Assimilation Methods

1.2.1 Nonlinear Filtering

For nonlinear models, the Kalman equations need to be adapted to permit nonlinear mappings in the forward operator and the observation operator:

$$x_{k+1} = f(x_k) + \omega_{k+1}, \qquad \omega_k \in R^L \tag{1.19}$$

$$y_{k+1} = h(x_{k+1}) + \eta_{k+1}, \quad \eta_{k+1} \in R^M. \tag{1.20}$$

Our observation operator for voltage data remains linear: $h(x) = Hx = [\mathbf{e_1}\ 0 \dots 0]x$, where $\mathbf{e_j}$ is the jth elementary basis vector, is a projection onto the voltage component of our system. Note that $h(x)$ is an operator, not to be confused with the inactivation gate in equation (1.2). Our nonlinear model update, $f(x)$ in equation (1.19), is taken as the forward integration of the dynamical equations between observation times.

Multiple platforms for adapting the Kalman equations exist. The most straightforward approach is the Extended Kalman filter (EKF) which uses local linearizations of the nonlinear operators in equation (1.19)–(1.20) and plugs these into the standard Kalman equations. By doing so, one preserves Gaussianity of the state space. Underlying the data assimilation framework is the goal of understanding the distribution, or statistics of the distribution, of the states of the system given the observations:

$$p(x|y) \propto p(y|x)p(x). \tag{1.21}$$

The Gaussianity of the state space declares the posterior conditional distribution, $p(x|y)$, to be a normal distribution by the product of Gaussians being Gaussian, and the statistics of this distribution lead to the Kalman update equations [7]. However, the EKF is really only suitable when the dynamics are nearly linear between observations, and can result in divergence of the estimates [66].

Rather than trying to linearize the transformation to preserve Gaussianity, where this distributional assumption is not going to be valid for practical problems anyway, an alternative approach is to preserve the nonlinear transformation and try to estimate the first two moments of transformed state [66]. The Unscented Kalman Filter (UKF) approximates the first two statistics of $p(x_k|y_k)$ by calculating sample means and variances, which bypasses the need for Gaussian integral products. The UKF uses an ensemble of deterministically selected points in the state space whose collective mean and covariance are that of the state estimate and its associated covariance at some time. The forward operator $f(x)$ is applied to each of these *sigma points*, and the mean and covariance of the transformed points can then be computed to estimate the nonlinearly transformed mean and covariance. Figure 1.1 depicts this "unscented" transformation. The sigma points accurately estimate the true statistics both initially (Figure 1.1 A) and after nonlinear mapping (Figure 1.1 B). **A:** Initial data where blue corresponds to sampling points from a normal distribution of the V, n state-space and the red circles are the sigma points. Blue corresponds to the true uncertainty and mean of the sampled distribution. Magenta corresponds to the statistics of the sigma points. **B:** Illustrates the forward operator $f(x)$ acting on each element of the left panel where $f(x)$ is the numerical integration of the Morris-Lecar equations (2.1)–(2.5) between observation times.

In the UKF framework, as with all DA techniques, one is attempting to estimate the states of the system. The standard set of states in conductance-based models includes the voltage, the gating variables, and any intracellular ion concentrations

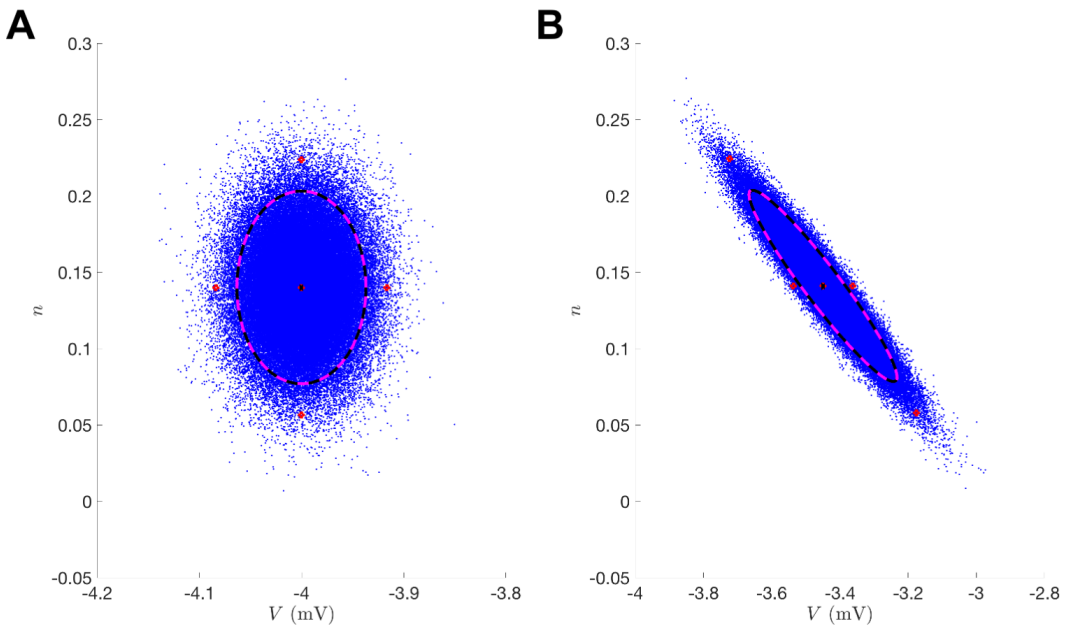


Figure 1.1 Unscented transformation for the Morris-Lecar model.

not taken to be stationary. To incorporate parameter estimation, parameters $\boldsymbol{\theta}$ to be estimated are promoted to states whose evolution is governed by the model error random variable:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \omega_{k+1}^{\boldsymbol{\theta}} \qquad \omega_k^{\boldsymbol{\theta}} \in R^D. \tag{1.22}$$

This is referred to as an "artificial noise evolution model", as the random disturbances driving deviations in model parameters over time robs them of their time-invariant definition [49, 75]. We found this choice to be appropriate for convergence and as a tuning mechanism. An alternative is to zero out the entries of Q_k corresponding to the parameters in what is called a "persistence model" where $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k$ [36]. However, changes in parameters can still occur during the analysis stage.

We declare our augmented state to be comprised of the states in the dynamical system as well as parameters $\boldsymbol{\theta}$ of interest:

$$\text{Augmented State} : \mathbf{x} = (V, \mathbf{q}, \boldsymbol{\theta})^{\mathrm{T}} \quad \mathbf{q} \in \mathbb{R}^{L-1}, \ \boldsymbol{\theta} \in \mathbb{R}^{D} \tag{1.23}$$

where q represents the additional states of the system besides the voltage. The filter requires an initial guess of the state $\hat{x}_0$ and covariance P_{xx}.

An ensemble of σ points are formed whose position and weights are determined by λ, which can be chosen to try to match higher moments of the system distribution [66]. λ, to the best of our understanding, is introduced purely as a scaling parameter, and certain implementations have found success by merely setting $\lambda = 0$. Practically, this algorithmic parameter can be chosen to spread the ensemble for $\lambda > 0$, shrink the ensemble for $-N < \lambda < 0$, or to have the mean point completely removed from the ensemble by setting it to zero. The ensemble is formed on lines 80-82 of *UKF_Step.m*. The individual weights can be negative, but their cumulative sum is 1.

$$\sigma \text{ Points} : X_j = \hat{x}_k^a \pm \left(\sqrt{(N+\lambda) P_{xx}} \right)_j, \quad j = 1 \ldots 2N, \quad X_0 = \hat{x}_k^a \tag{1.24}$$

$$\text{Weights} : W_j = \frac{1}{2\,(N+\lambda)}, \quad j = 1 \ldots 2N, \quad W_0 = \frac{\lambda}{N+\lambda} \tag{1.25}$$

$$\tag{1.26}$$

We form our background estimate $\hat{x}_{k+1}^b$ by applying our map $f(x)$ to each of the ensemble members,

$$\tilde{X}_j = f(X_j) \tag{1.27}$$

and then computing the resulting mean:

$$\text{Forecast Estimate}: \quad \hat{x}^b_{k+1} = \sum_{j=0}^{2N} W_j \tilde{X}_j. \tag{1.28}$$

We then propagate the transformed sigma points through the observation operator:

$$\tilde{Y}_j = h(\tilde{X}_j) \tag{1.29}$$

and compute our predicted observation $\hat{y}^b_{k+1}$ from the mapped ensemble:

$$\text{Measurement Estimate}: \hat{y}^b_{k+1} = \sum_{j=0}^{2N} W_j \tilde{Y}_j. \tag{1.30}$$

We compute the background covariance estimate by calculating the variance of the mapped ensemble and adding the process noise Q_k:

$$\text{Background Cov. Est.}: P^f_{xx} = \sum_{j=0}^{2N} W_j (\tilde{X}_j - \hat{x}^b_{i+k})(\tilde{X}_j - \hat{x}^b_{i+k})^{\mathrm{T}} + Q_k \tag{1.31}$$

and do the same for the predicted measurement covariance with the addition of R_k:

$$\text{Predicted Meas. Cov.}: P_{yy} = \sum_{j=0}^{2N} W_j (\tilde{Y}_j - \hat{y}^b_{k+1})(\tilde{Y}_j - \hat{y}^b_{k+1})^{\mathrm{T}} + R_{k+1}. \tag{1.32}$$

The Kalman gain is computed by matrix multiplication of the cross-covariance:

$$\text{Cross-Cov.}: \quad P_{xy} = \sum_{j=0}^{2N} W_j(\tilde{X}_j - \hat{x}^b_{k+1})(\tilde{Y}_j - \hat{y}^b_{k+1})^{\mathrm{T}} \tag{1.33}$$

with the predicted measurement covariance:

$$\text{Kalman Gain}: \quad K = P_{xy}P_{yy}^{-1}. \tag{1.34}$$

When only observing voltage, this step is merely scalar multiplication of a vector. The gain is used in the *analysis*, or update step, to linearly interpolate our background statistics with measurement corrections. The update step for the covariance is:

$$P^a_{xx} = P^b_{xx} - KP^{\mathrm{T}}_{xy}, \tag{1.35}$$

and the mean is updated to to interpolate the background estimate with the deviations of the estimated measurement term with the observed data y_{k+1}:

$$\hat{x}^a_{k+1} = \hat{x}^b_{k+1} + K(y_{k+1} - \hat{y}^b_{k+1}). \tag{1.36}$$

An implementation of this algorithm is shown in the Appendix with the parent function *UKFML.m* and one time step of the algorithm computed in *UKF_Step.m*.

The analysis step is performed on line 124 of *UKF_Step.m*. Some implementations also include a redistribution of the sigma points about the forecast estimate using the background covariance prior to computing the cross-covariance P_{xy} or the

predicted measurement covariance P_{yy} [107]. So, after equation (1.31), we redefine $\tilde{X}_j$, $\tilde{Y}_j$ in equation (1.27) as:

$$\tilde{X}_j = \hat{x}^b_{k+1} \pm \left(\sqrt{(N+\lambda)P_{xx}}\right)_j, \quad j = 1 \dots 2N$$
$$\tilde{Y}_j = h(\tilde{X}_j)$$

The above is shown in lines 98-117 in *UKF_Step*. A particularly critical part of using a filter, or any DA method, is choosing the process covariance matrix Q_k and the measurement covariance matrix R_k. The measurement noise may be intuited based upon knowledge of one's measuring device, but the model error is practically impossible to know *a priori*. Work has been done to use previous innovations to simultaneously estimate Q and R during the course of the estimation cycle [12], but this becomes a challenge for systems with low observability (such as is the case when only observing voltage). Rather than estimating the states and parameters simultaneously as with an augmented state space, one can try to estimate the states and parameters separately. For example, [101] used a shooting method to estimate parameters and the UKF to estimate the states. This study also provided a systematic way to estimate an optimal covariance inflation Q_k. For high-dimensional systems where computational efficiency is a concern, an implementation which efficiently propagates the square root of the state covariance has been developed [108].

Figure 1.2 depicts how the algorithm operates. Between observation times, the previous analysis (or best estimate) point is propagated through the model to come up with the predicted model estimate. The Kalman update step interpolates this point with observations weighted by the Kalman gain. The red circles are the result of forward integration through the model using the previous best estimates. The green are the estimates after combining these with observational data. The blue stars

depict the true system output (without any noise), and the magenta stars are the noisy observational data with noise generated by Equation (2.7) and $\epsilon = 0.1$.

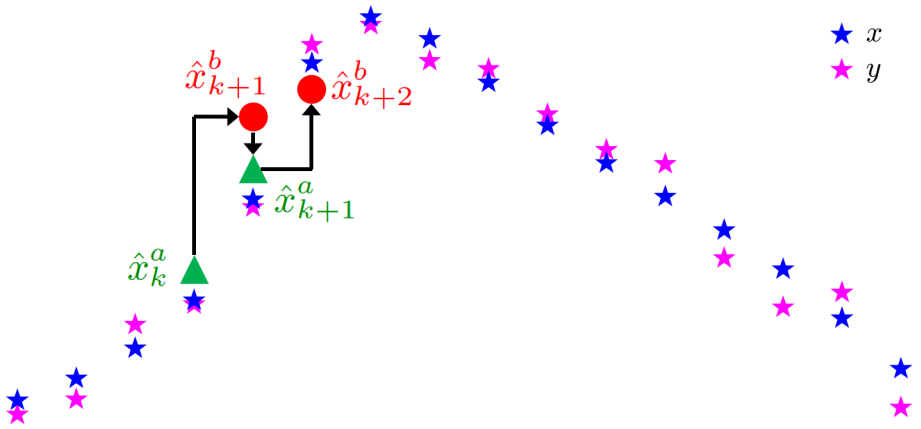


Figure 1.2 Example of iterative estimation in UKF.

1.2.2 Variational Methods

In continuous time, variational methods aim to find minimizers of functionals which represent approximations to the probability distribution of a system conditioned on some observations. As our data is available only in discrete measurements, it is practical to work with a discrete form similar to Equation (1.7) for nonlinear systems:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} \|y_k - h(x_k)\|^2_{R_k} + \frac{1}{2}\sum_{k=0}^{N-1} \|x_{k+1} - f(x_k)\|^2_{P^b_k}\,. \tag{1.37}$$

We assume the states follow the state-space description in Equations (1.19)–(1.20) with $\omega_k \sim \mathcal{N}(0,\, Q)$ and $\eta_k \sim \mathcal{N}(0,\, R)$, where Q is our model error covariance matrix and R is our measurement error covariance matrix. As an approximation, we impose Q, R to be diagonal matrices, indicating that there is assumed to be no correlation between errors in different states. Namely, Q, contains only the

assumed model error variance for each state-space component, and R is just the measurement error variance of the voltage observations. These assumptions simplify the cost function to:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} R^{-1}\left(y_k - V_k\right)^2 + \frac{1}{2}\sum_{l=1}^{L}\sum_{k=0}^{N-1} Q_{l,l}^{-1}\left(x_{l,k+1} - f_l(x_k)\right)^2 \tag{1.38}$$

where $V_k = x_{1,k}$. For the current-clamp data problem in neuroscience, one seeks to minimize Equation (3.4) in what is called the "weak 4D-Var" approach. An example implementation of weak 4D-Var is shown in *w4DvarML.m* in the Appendix. An example of the cost function with which to minimize over is given in the child function *w4dvarobjfun.m*. Each of the x_k are mapped by $f(x)$ on line 108. Alternatively, "strong 4D-Var" forces the resulting estimates to be consistent with the model, $f(x)$. This can be considered the result of taking $Q \to \mathbf{0}$, which yields the nonlinearly constrained problem:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} R^{-1}\left(y_k - V_k\right)^2 \tag{1.39}$$

such that

$$x_{k+1} = f(x_k), \quad k = 0 \ldots N. \tag{1.40}$$

The rest of this chapter will be focused on the weak case, Equation (3.4), where we can define the argument of the optimization as:

$$\mathbf{x} = [x_{1,1},\ x_{1,2},\ \ldots x_{1,N},\ x_{2,1}, \ldots x_{L,N},\ \theta_1,\ \theta_2, \ldots \theta_D] \tag{1.41}$$

resulting in an $(N+1)L+D$ dimensional estimation problem. An important aspect of the scalability of this problem is that the Hessian matrix,

$$\mathbf{H}_{i,j} = \frac{\partial^2 C}{\partial x_i \partial x_j} \tag{1.42}$$

is sparse. Namely, each state at each discrete time has dependencies based upon the model equations, and the chosen numerical integration scheme. At the heart of many gradient-based optimization techniques lies a linear system involving the Hessian and the gradient, $\nabla C(\mathbf{x}_n)$, of the objective function that is used to solve for the next candidate point. Specifically, Newton's method for optimization is:

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \mathbf{H}^{-1}\nabla C(\mathbf{x}_n) \tag{1.43}$$

Therefore, if $(N+1)L+D$ is large, then providing the sparsity pattern is advantageous when numerical derivative approximations, or functional representations of them, are being used to perform minimization with a derivative-based method. One can calculate these derivatives by hand, symbolic differentiation, or automatic differentiation.

A feature of the most common derivative-based methods is assured convergence to local minima. However, our problem is non-convex due to the model term, which leads to the development of multiple local minima in the optimization surface as depicted in Figure 1.3. Figure 1.3 A depicts a surface generated by taking the logarithm of $C(\alpha,\beta)$, where $C(\alpha,\beta) = C\left(\mathbf{x}_0(1-\alpha)(1-\beta) + \alpha x_{\min,\mathrm{d}} + \beta x_{\min,\mathrm{s}}\right)$ so that at $\alpha = \beta = 0$, $\mathbf{x} = \mathbf{x}_0$ (magenta circle), and at $\alpha = 1$ and $\beta = 0$, $\mathbf{x} = x_{\min,\mathrm{d}}$ for the deeper minima (gray square), and similarly for the shallower minima (gray diamond). Figure 1.3 B shows the contour plot of the surface shown in (A). For the

results in this tutorial we will only utilize local optimization tools, but see Section 2.3 for a brief discussion of some global optimization methods with stochastic search strategies.

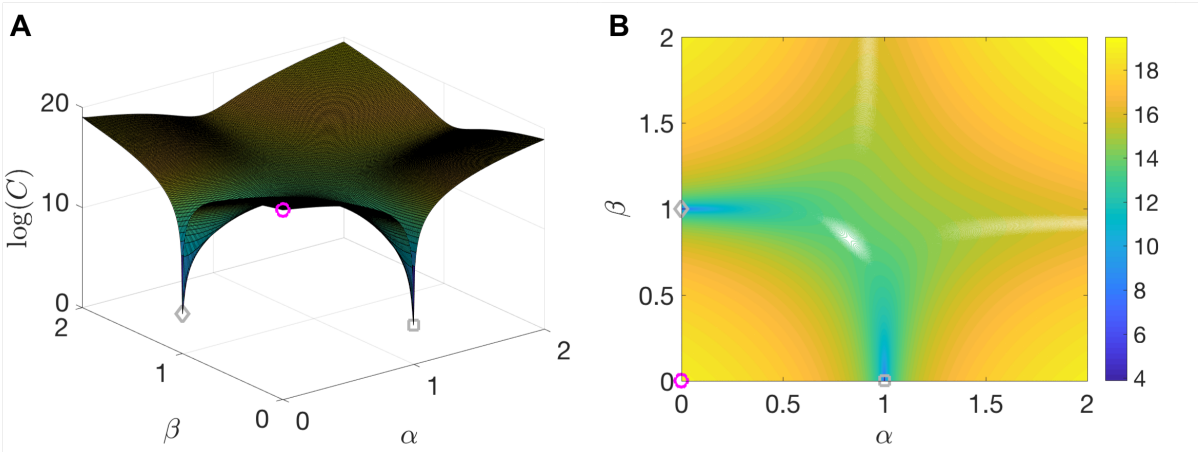


Figure 1.3 Example cost function for 4D-Var.

This chapter focuses on the introduction of Data Assimilation as a means to optimally incorporate model uncertainty and measurement uncertainty to estimate a systems' dynamics. In UKF, observations are sequentially introduced and estimates of the system states are sequentially improved. 4D-Var operates on a fixed time window and estimates the system simultaneously. We will apply it to a minimal conductance-based model, the Morris-Lecar model, in the following chapter.

CHAPTER 2

DATA ASSIMILATION APPLIED TO THE MORRIS-LECAR MODEL

2.1 Twin Experiments

Data assimilation is a framework for the incorporation of system observations into an estimation problem in a systematic fashion. Unfortunately, the methods themselves do not provide a great deal of insight into the tractability of unobserved system components of specific models. There may be a certain level of redundancy in the model equations, and degeneracy in the parameter space leading to multiple potential solutions [98]. Also, it may be the case that certain parameters are non-identifiable if, for instance, a parameter can be completely scaled out [113]. Some further work on identifiability is ongoing [102, 103].

Before applying a method to data from a real biological experiment, it is important to test it against simulated data where the ground truth is known. In these experiments, one creates simulated data from a model, and then tries to recover the true states and parameters of that model from the simulated data alone.

2.1.1 Recovery of Bifurcation Structure

In conductance-based models, as well as in real neurons, slight changes in a parameter value can lead to drastically different model output or neuronal behavior. Sudden changes in the topological structure of a dynamical system upon smooth variation of a parameter are called *bifurcations*. Different types of bifurcations lead to different neuronal properties, such as the presence of bistability and subthreshold oscillations [42]. Thus, it is important for a neuronal model to accurately capture the bifurcation dynamics of the cell being modeled [63]. In this chapter, we ask whether or not the models estimated through data assimilation match the bifurcation structure of the model that generated the data. This provides a qualitative measure of success or

failure for the estimation algorithm. Since bifurcations are an inherently nonlinear phenomenon, our use of topological structure as an assay emphasizes how nonlinear estimation is a fundamentally distinct problem from estimation in linear systems.

2.1.2 Morris-Lecar Model

The Morris-Lecar model, first used to describe action potential generation in barnacle muscle fibers, has become a canonical model for studying neuronal excitability [82]. The model includes an inward voltage-dependent calcium current, an outward voltage-dependent potassium current, and a passive leak current. The activation gating variable for the potassium current has dynamics, whereas the calcium current activation gate is assumed to respond instantaneously to changes in voltage. The calcium current is also non-inactivating, resulting in a 2D model. The model exhibits multiple mechanisms of excitability: for different choices of model parameters, different bifurcations from quiescence to repetitive spiking occur as the applied current is increased [42].

Three different bifurcation regimes—Hopf, saddle-node on an invariant circle (SNIC), and homoclinic—are depicted in Figure 2.1 and correspond to the parameter sets in Table 2.1. For a given applied current in the region where a stable limit cycle (corresponding to repetitive spiking) exists, each regime displays a distinct firing frequency and action potential shape. The bifurcation diagrams in the top row depict stable fixed points (red), unstable fixed points, including saddles, (black), stable limit cycles (blue), and unstable limit cycles (green). Gray dots indicate bifurcation points, and the dashed gray lines indicate the value of I_{app} corresponding to the traces shown for V (middle row) and n (bottom row). **A:** As I_{app} is increased from 0 or decreased from 250 nA, the branches of stable fixed points lose stability through subcritical Hopf bifurcation and unstable limit cycles are born. The branch of stable limit cycles that exists at $I_{\text{app}} = 100$ nA eventually collides with these unstable limit cycles and is

destroyed in a saddle-node of periodic orbits (SNPO) bifurcation as I_{app} is increased or decreased from this value. **B:** As I_{app} is increased from 0, a branch of stable fixed points is destroyed through saddle-node bifurcation with the branch of unstable fixed points. As I_{app} is decreased from 150 nA, a branch of stable fixed points loses stability through subcritical Hopf bifurcation and unstable limit cycles are born. The branch of stable limit cycles that exists at $I_{\text{app}} = 100$ nA is destroyed through a SNPO bifurcation as I_{app} is increased and a SNIC bifurcation as I_{app} is decreased. **C:** Same as (B), except that the stable limit cycles that exist at $I_{app} = 36$ nA are destroyed through a homoclinic orbit bifurcation as I_{app} is decreased.

Table 2.1 Morris-Lecar Parameter Values

	Hopf	SNIC	Homoclinic
ϕ	.04	.067	.23
g_{Ca}	4	4	4
V_3	2	12	12
V_4	30	17.4	17.4
g_{K}	8	8	8
g_{L}	2	2	2
V_1	-1.2	-1.2	-1.2
V_2	18	18	18

The equations for the Morris-Lecar model are:

$$\begin{aligned} C_m \frac{dV}{dt} &= I_{app} - g_{\text{L}}(V - E_{\text{L}}) - g_{\text{K}} n (V - E_{\text{K}}) \\ &\quad - g_{\text{Ca}} m_\infty(V)(V - E_{\text{Ca}}) \; = f_V^\star(V, n; \boldsymbol{\theta}) \end{aligned} \tag{2.1}$$

$$\frac{dn}{dt} = \phi(n_\infty(V) - n)/\tau_n(V) \; = f_n^\star(V, n; \boldsymbol{\theta}) \tag{2.2}$$

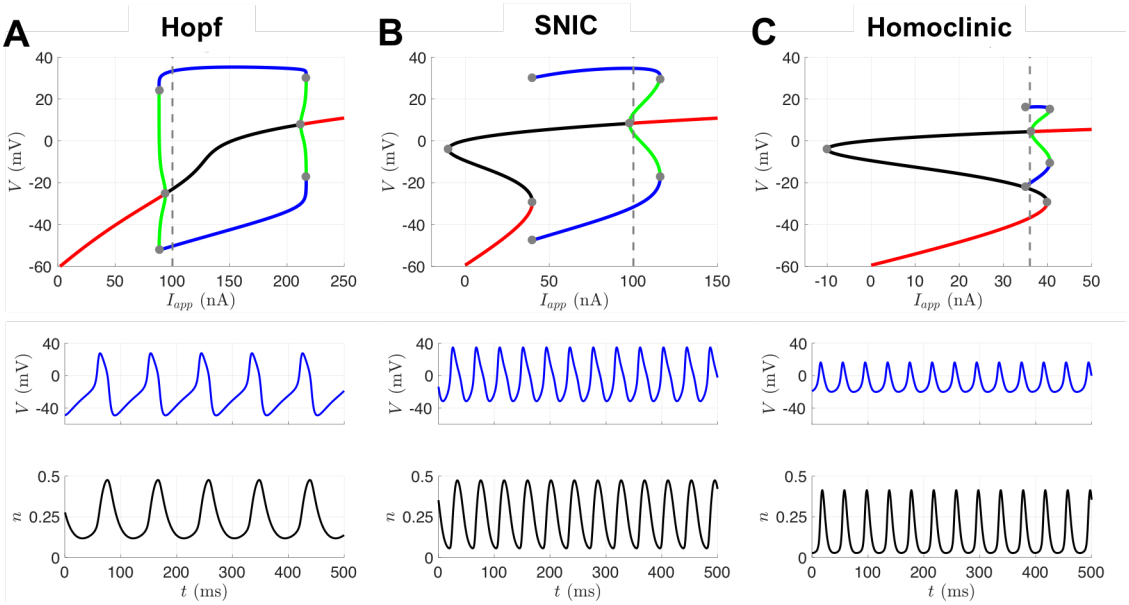


Figure 2.1 Three different excitability regimes of the Morris-Lecar model.

with

$$m_\infty = \frac{1}{2}[1 + \tanh((V - V_1)/V_2)] \tag{2.3}$$

$$\tau_n = 1/\cosh((V - V_3)/2V_4) \tag{2.4}$$

$$n_\infty = \frac{1}{2}[1 + \tanh((V - V_3)/V_4)] \tag{2.5}$$

The eight parameters that we will attempt to estimate from data are g_{L}, g_{K}, g_{Ca}, ϕ, V_1, V_2, V_3, and V_4. We are interested in whether the estimated parameters yield a model with the desired mechanism of excitability. Specifically, we will conduct twin experiments where the observed data is produced by a model with parameters in a certain bifurcation regime, but the data assimilation algorithm is initialized with parameter guesses corresponding to a different bifurcation regime. We then assess whether or not a model with the set of estimated parameters undergoes the same bifurcations as the model that produced the observed data. This approach provides

an additional qualitative measure of estimation accuracy, beyond simply comparing the values of the true and estimated parameters.

2.1.3 Results with UKF

The UKF was tested on the Morris-Lecar model in an effort to simultaneously estimate V and n along with the 8 parameters in Table 2.1. For all simulations, $C = 20$, $E_{\text{Ca}} = 120$, $E_{\text{K}} = -84$, and $E_{\text{L}} = -60$. For the Hopf and SNIC regime, $I_{\text{app}} = 100$; for the homoclinic regime, $I_{\text{app}} = 36$. Data was generated via a modified Euler scheme, a 2nd order Runge-Kutta method, at observation points every 0.1 ms, where we take the step-size, Δt, as 0.1 as well:

$$\begin{aligned} \tilde{x}_{k+1} &= x_k + \Delta t f^\star(t_k, x_k) \\ x_{k+1} &= x_k + \frac{\Delta t}{2}\left(f^\star(t_k, x_k) + f^\star(t_{k+1}, \tilde{x}_{k+1})\right) \\ &= f(x_k). \end{aligned} \tag{2.6}$$

The UKF is a particularly powerful tool when a lot of data is available; the computational complexity in time is effectively the same as the numerical scheme of choice, whereas the additional operations at each time point are $O((L+D)^3)$ [115]. $f(x)$ in Equation (1.19) is taken to be the Morris-Lecar equations (2.1)–(2.2), acting as $f^\star(t_k, x_k)$, integrated forward via modified Euler as in Equation (2.6), and is given on line 126 of *UKFML.m*. The function *fXaug.m*, shown later in the Appendix, represents our augmented vector field. Our observational operator H is displayed on line 136 of *UKFML.m*. To reiterate, the states to be estimated in the Morris-Lecar model are the voltage and the potassium gating variable. The 8 additional parameters are promoted to members of state space with trivial dynamics resulting in a 10-D estimation problem.

These examples were run using 20 seconds of data which is 200,001 time points. During this time window, the Hopf, SNIC, and homoclinic models fire 220, 477, and 491 spikes, respectively. Such a computation for a 10-D model takes only a few minutes on a laptop computer. R can be set to 0 when one believes the observed signal to be completely noiseless, but even then it is commonly left as a small number to try to mitigate the development of singularities in the predicted measurement covariance. We set our observed voltage to be the simulated output using modified Euler with additive white noise at each time point:

$$V_{\text{obs}}(t) = V_{\text{true}}(t) + \eta(t) \tag{2.7}$$

where $\eta \sim \mathcal{N}(0, (\epsilon\sigma_{\text{true}})^2)$ is a normal random variable whose variance is equal to the square of the standard deviation of the signal scaled by a factor ϵ, which is kept fixed at 0.01 for these simulations. R is taken as the the variance of η. The initial covariance of the system is $\alpha_I I$, where I is the identity matrix and α_I is 0.001. The initial guess for n is taken to be 0. Q is fixed in time as a diagonal matrix with diagonal $10^{-7}\,[\max(V_{\text{obs}}) - \min(V_{\text{obs}}), 1, |\theta_0|]$, where θ_0 represents our initial parameter guesses. We set $\lambda = 5$, however this parameter was not especially influential for the results of these runs, as discussed further below. These initializations are displayed in the body of the parent function *UKFML.m*.

Figure 2.2 shows the state estimation results when the observed voltage is from the SNIC regime but the UKF is initialized with parameter guess corresponding to the Hopf regime. Initially, the state estimate for n and its true, unobserved dynamics have great disparity. As the observations are assimilated over the estimation window, the states and model parameters adjust to produce estimates which better replicate the observed, and unobserved, system components. In this way, information from the

observations is transferred to the model. The evolution of the parameter estimates for this case is shown in the first column of Figure 2.3, with ϕ, V_3, and V_4 all converging to close to their true values after 10 seconds of observations. The only difference in parameter values between the SNIC and homoclinic regimes is the value of the parameter ϕ. The second column of Figure 2.3 shows that when the observed data is from the homoclinic regime but the initial parameter guesses are from the SNIC regime, the estimates of V_3 and V_4 remain mostly constant near their original (and correct) values, whereas the estimate of ϕ quickly converges to its new true value. Finally, the third column of Figure 2.3 shows that all three parameter estimates evolve to near their true values when the UKF is presented data from the Hopf regime but initial parameter estimates from the homoclinic regime.

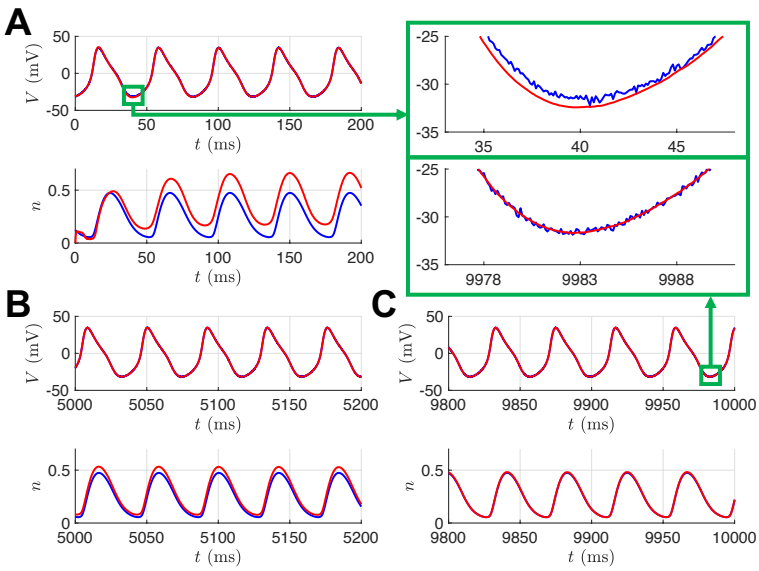


Figure 2.2 State estimates for UKF. This example corresponds to initializing with parameters from the HOPF regime and attempting to correctly estimate those of the SNIC regime. The noisy observed voltage V and true unobserved gating variable n are shown in blue, and their UKF estimates are shown in red.

Table 2.2 shows the parameter estimates at the end of the estimation window for all of the nine possible twin experiments. Promisingly, a common feature of the results is the near recovery of the true value of each of the parameters. However, the

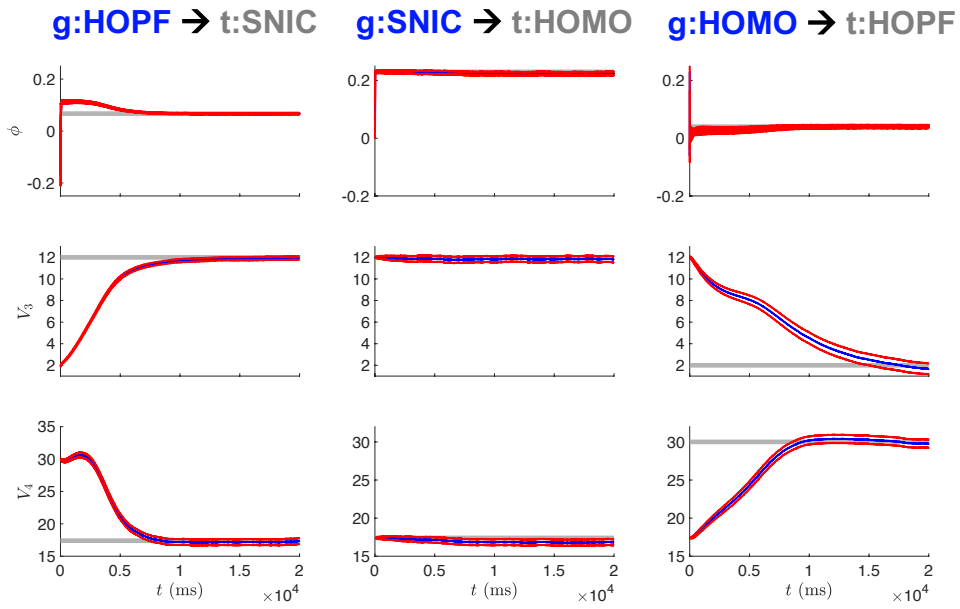


Figure 2.3 Parameter estimates for UKF. This example corresponds to initializing with parameters from the HOPF, SNIC, and HOMO regimes and attempting to correctly estimate those of the SNIC, HOMO, and HOPF regimes (left to right column, respectively). The blue curves are the estimates from the UKF, with ± 2 standard deviations from the mean based on the filter estimated covariance shown in red. The gray lines indicate the true parameter values.

estimated parameter values alone do not necessarily tell us about the dynamics of the inferred model. To assess the inferred models, we generate bifurcation diagrams using the estimated parameters and compare them to the bifurcation diagrams for the parameters that produced the observed data. Figure 2.4 shows that the SNIC and homoclinic bifurcation diagrams were recovered quite exactly. The Hopf structure was consistently recovered, but with shifted regions of spiking and quiescence, and minor differences in spike amplitude.

To check the consistency of our estimation, we set 100 initial guesses for n across its dynamical range as samples from $\mathcal{U}(0, 1)$. Figure 2.5 shows that the state estimates for n across these initializations quickly approached very similar trajectories. We confirmed that after the estimation cycle was over, the parameter estimates for all 100 initializations were essentially identical to the values shown in Table 2.2. In this

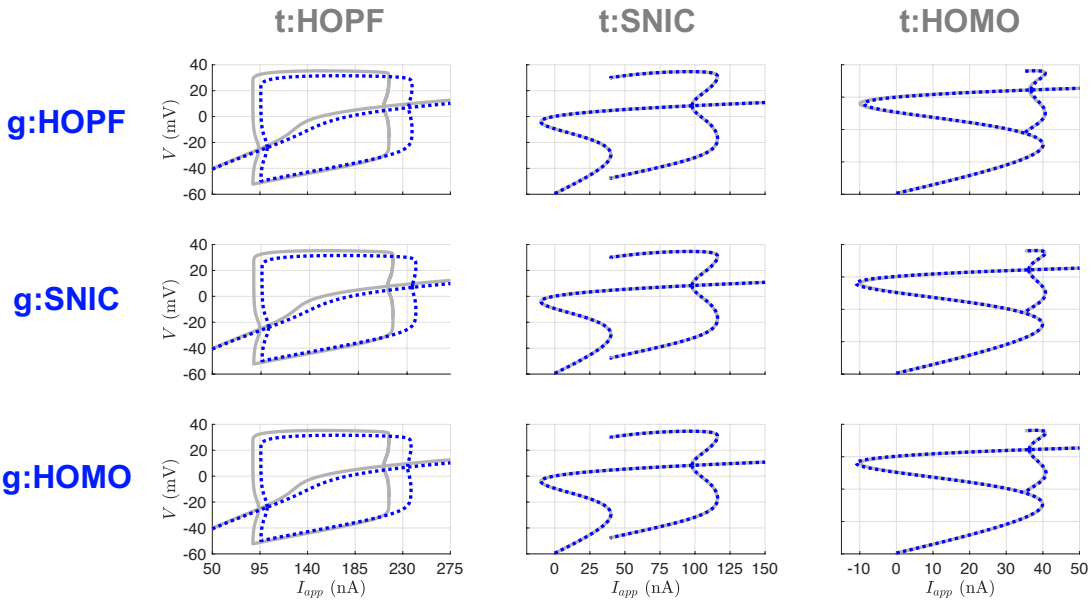


Figure 2.4 Bifurcation diagrams for UKF twin experiments. The gray lines correspond to the true diagrams, and the blue dotted lines correspond to the diagrams produced from the estimated parameters in 2.2.

paper, we always initialized the UKF with initial parameter values corresponding to the various bifurcation regimes, and did not explore the performance for randomly selected initial parameter guesses. For initial parameter guesses that are too far from the true values, it is possible that the filter would converge to incorrect parameter values or fail outright before reaching the end of the estimation window.

Additionally, we investigated the choices of certain algorithmic parameters for the UKF, namely λ and α_I. Figure 2.6A shows suitable ranges of these parameters, with the color indicating the root mean squared error of the parameters at the end of the cycle compared to their true values. We found this behavior to be preserved across our 9 twin experiment scenarios. Notably, this shows that our results in Table 2.2 were generated using an initial covariance, $\alpha_I = 0.001$, that was smaller than necessary.

By increasing the initial variability, the estimated system can converge to the true dynamics more quickly, as shown for $\alpha_I = 0.1$ in Figure 2.6B. The color scale represents the root mean squared error of the final parameter values at $T = 200,001$

from the parameters of the SNIC bifurcation regime. Gray indicates the filter failed outright before reaching the end of the estimation window. **B:** Parameter estimates over time for the run with $\lambda = 5, \alpha_I = 0.1$. The parameters (especially ϕ and V_3) approach to their true values more quickly than corresponding runs with smaller initial covariances; see column 1 of Figure 2.3 for parameter estimates with $\lambda = 5, \alpha_I = 0.001$. **C:** Same as (A), but with a modification to the numerical integration scheme that restricts the gating variable n to remain within its biophysical range of 0 to 1. The value of λ does not have a large impact on these results, except for when $\alpha_I = 1$. Here the filter fails before completing the estimation cycle, except for a few cases where λ is small enough to effectively shrink the ensemble spread and compensate for the large initial covariance. For example, with $\lambda = -9$, we have $N - 9 = 1$ and therefore the ensemble spread in Equation (1.25) is simply $X_j = \hat{x}_k^a \pm \sqrt{P_{xx}}$. For even larger initial covariances ($\alpha_I > 1$), the filter fails regardless of the value of λ. We noticed that in many of the cases that failed, the parameter estimate for ϕ was becoming negative (which is unrealistic for a rate) or quite large ($\phi > 1$), and that the state estimate for n was going outside of its biophysical range of 0 to 1. When the gating variable extends outside of its dynamical range it can skew the estimated statistics and the filter may be unable to recover. The standard UKF framework does not provide a natural way of incorporating bounds on parameter estimates, and we do not apply any for the results presented here. However, we did find that we can modify our numerical integration scheme to prevent the filter from failing in many of these cases, as shown in Figure 2.6C. Specifically, if n becomes negative or exceeds 1 after the update step, then artificially setting n to 0 or 1 in the Modified Euler method (2.6) before proceeding can enable the filter to reach the end of the estimation window and yield reasonable parameter estimates.

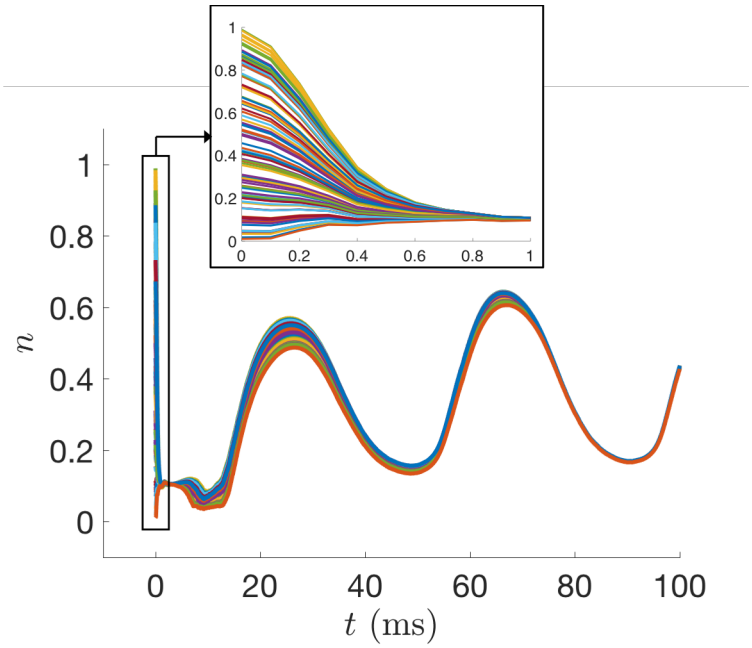


Figure 2.5 UKF state estimates of n for the Morris-Lecar model with 100 different initial guesses of the state sampled from $\mathcal{U}(0,1)$, with all other parameters held fixed.

2.1.4 Results with 4D-Var

The following results illustrate the use of weak 4D-Var. One can minimize the cost function in Equation (3.4) using a favorite choice of optimization routine. For the following examples, we will consider a local optimizer by using interior point optimization with MATLAB's built-in solver *fmincon*. At the heart of the solver is a Newton-step which uses information about the Hessian, or a conjugate gradient step

Table 2.2 UKF Parameter Estimates at End of Estimation Window, with Observed Data from Bifurcation Regime 't' and Initial Parameter Guesses Corresponding to Bifurcation Regime 'g'

	t:HOPF			**t:SNIC**			**t:HOMO**		
	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO
ϕ	0.040	0.40	0.040	0.067	0.067	0.067	0.237	0.224	0.224
g_{Ca}	4.017	4.019	4.025	4.001	4.000	4.001	4.112	3.874	3.877
V_3	1.612	1.762	1.660	11.931	11.937	11.912	11.751	11.784	11.772
V_4	29.646	29.832	29.771	17.343	17.337	17.342	17.739	16.806	16.815
g_{K}	7.895	7.926	7.892	7.970	7.971	7.958	7.929	7.854	7.850
g_{L}	2.032	2.027	2.033	2.003	2.004	2.003	2.025	1.967	1.968
V_1	-1.199	-1.195	-1.189	-1.193	-1.193	-1.190	-1.064	-1.346	-1.341
V_2	18.045	18.053	18.067	17.991	17.991	17.991	18.179	17.734	17.740

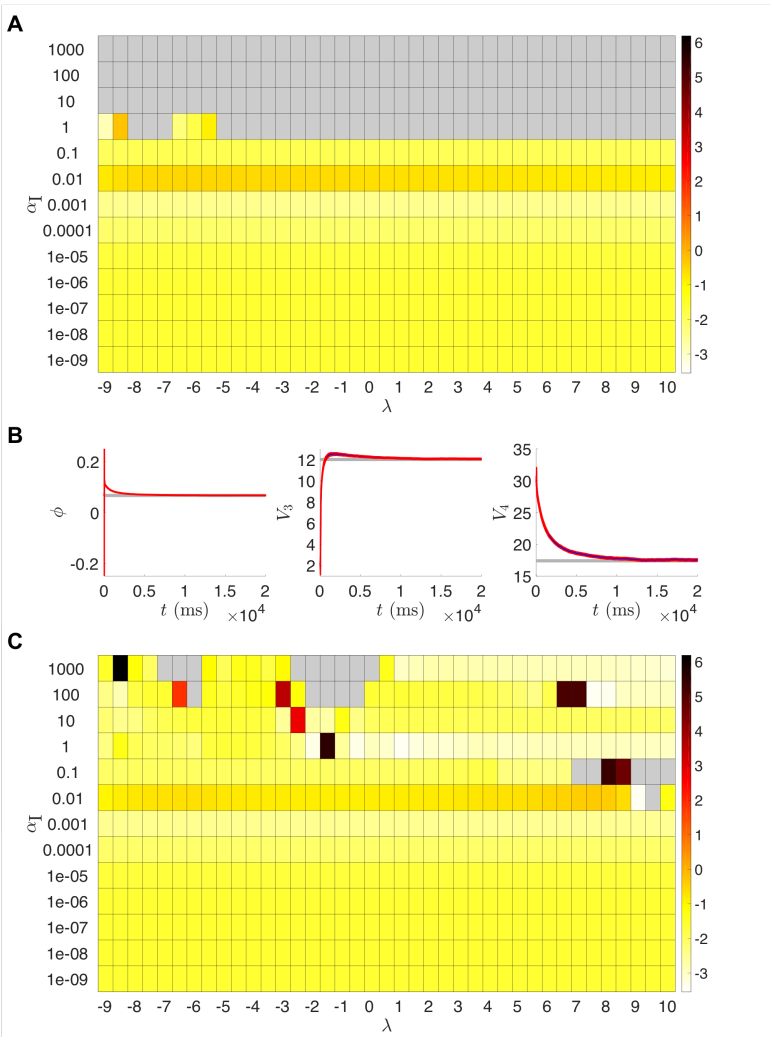


Figure 2.6 **A:** UKF results from runs of the t:SNIC/g:HOPF twin experiment for various parameter combinations of λ and α_I.

using gradient information [21,22,114]. The input we are optimizing over conceptually takes the form of

$$\mathbf{x} = [V_0, V_1, \ldots V_N, n_0, n_1, \ldots n_N, \theta_1, \theta_2, \ldots \theta_D] \tag{2.8}$$

resulting in an $(N+1)L+D$ dimensional estimation problem where $L = 2$. There are computational limitations with memory storage and the time required to sufficiently

solve the optimization problem to a suitable tolerance for reasonable parameter estimates. Therefore, we can not be cavalier with using as much data with 4D-Var as we did with the UKF, as that would result in a $(200001)2 + 8 = 400,010$ dimensional problem. Using Newton's method, Equation (1.43), on this problem would involve inverting a Hessian matrix of size $(400,010)^2$, which according to a rough calculation would require over 1 TB of RAM. Initialization of the optimization is shown on line 71 of *w4DVarML.m*.

The estimated parameters are given in Table 2.4. These results were run using $N = 2001$ time points. To simplify the search space, the parameter estimates were constrained between the bounds listed in Table 2.3. These ranges were chosen to ensure that the maximal conductances, the rate ϕ, and the activation curve slope V_2 all remain positive. We found that running 4D-Var with even looser bounds (Table 2.6) yielded less accurate parameter estimates (Tables 2.7 and 2.8).The loose parameter bounds in Table 2.6 were used for these trials. The white noise perturbations or the 4D-Var trials were the same as those from the UKF examples. Initial guesses for the states at each time point are required. For these trials, V is initialized as V_{obs}, and n is initialized as the result of integration of its dynamics forced with V_{obs} using the initial guesses for the parameters, i.e. $n = \int_t f_n(V_{\text{obs}}, n; \theta_0)dt$. The initial guesses are generated beginning on line 38 of *w4DvarML.m*. We impose that Q^{-1} in Equation (3.4) is a diagonal matrix with entries $\alpha_Q[1,\ 100^2]$ to balance the dynamical variance of V and n. The scaling factor α_Q represents the relative weight of the model term compared to the measurement term. Based on preliminary tuning experiments, we set $\alpha_Q = 100$ for the results presented.

Figure 2.7 depicts the states produced by integrating the model with the estimated parameters across different iterations within the interior-point optimization. Over iteration cycles, the geometry of spikes as well as the spike time alignments eventually coincide with the noiseless data, V_{true}. Figure 2.9 shows the evolution

Table 2.3 Bounds Used During 4D-Var Estimation for the Results Shown in Tables 2.4 and 2.5

	Lower Bound	Upper Bound
ϕ	0	1
g_{Ca}	0	10
V_3	-20	20
V_4	.1	35
g_{K}	0	10
g_{L}	0	5
V_1	-10	20
V_2	.1	35

Table 2.4 4D-Var Parameter Estimates at the End of the Optimization for each Bifurcation Regime

	t:HOPF			**t:SNIC**			**t:HOMO**		
	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO
ϕ	0.040	0.037	0.039	0.069	0.067	0.066	0.414	0.218	0.230
g_{Ca}	4.000	3.890	3.976	4.024	4.000	4.045	9.037	3.813	3.999
V_3	2.000	3.404	3.241	12.695	12.000	12.076	7.458	13.022	12.004
V_4	30.000	29.085	30.122	18.759	17.400	16.990	28.365	17.165	17.403
g_{K}	8.000	8.386	8.287	8.284	8.000	8.009	9.817	8.472	8.002
g_{L}	2.000	2.016	2.021	1.930	2.000	2.071	3.140	1.941	2.000
V_1	-1.200	-1.335	-1.250	-1.078	-1.200	-1.179	2.872	-1.419	-1.202
V_2	18.000	17.619	17.911	18.091	18.000	18.162	24.769	17.712	18.000

of the parameters across the entire estimation cycle. The blue traces are noiseless versions of the observed voltage data (left column) or the unobserved variable n (right column) from the model that produced the data. The red traces are the result of integrating the model with the estimated parameter sets at various points during the course of the optimization. **A:** Initial parameter guesses. **B:** Parameter values after 100 iterations. **C:** Parameter values after 1,000 iterations. **D:** Parameter values after 30,000 iterations (corresponds to t:SNIC/g:HOPF column of Table 2.4). The

Table 2.5 4D-Var Parameter Estimates at the End of the Optimization for Each Bifurcation Regime, Derivative Provided

	t:HOPF			**t:SNIC**			**t:HOMO**		
	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO
ϕ	0.039	0.039	0.039	0.066	0.067	0.066	0.230	0.230	0.230
g_{Ca}	3.889	3.889	3.889	4.002	4.035	4.002	4.014	4.019	4.014
V_3	1.971	1.971	1.971	11.825	12.176	11.825	12.321	12.320	12.320
V_4	29.533	29.533	29.533	17.071	17.342	17.071	17.615	17.633	17.616
g_{K}	8.050	8.050	8.050	7.923	8.057	7.923	8.157	8.158	8.157
g_{L}	1.928	1.928	1.928	2.027	2.038	2.027	1.996	1.997	1.996
V_1	-1.301	-1.301	-1.301	-1.232	-1.165	-1.232	-1.154	-1.148	-1.153
V_2	17.600	17.600	17.600	18.004	18.126	18.004	18.050	18.057	18.050

Table 2.6 Bounds Used During 4D-Var Estimation for the Results Shown in Tables 2.7 and 2.8

	Lower Bound	Upper Bound
ϕ	0	∞
g_{Ca}	0	∞
V_3	$-\infty$	∞
V_4	0.1	∞
g_{K}	0	∞
g_{L}	0	∞
V_1	$-\infty$	∞
V_2	0.1	∞

Table 2.7 4D-Var Parameter Estimates at the End of the Optimization for each Bifurcation Regime with Loose Parameter Bounds

	t:HOPF			**t:SNIC**			**t:HOMO**		
	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO
ϕ	0.040	0.041	0.040	0.066	0.067	0.066	0.406	0.225	0.229
g_{Ca}	4.011	3.959	3.989	4.016	4.035	4.040	8.623	3.992	3.983
V_3	2.210	13.479	6.284	12.497	12.176	12.102	7.453	14.333	12.197
V_4	29.917	37.854	32.748	17.589	17.342	16.998	27.569	18.593	17.464
g_{K}	8.046	10.857	8.989	8.192	8.057	8.021	9.543	9.213	8.092
g_{L}	2.026	1.806	1.959	2.009	2.038	2.067	3.029	1.960	1.990
V_1	-1.222	-1.188	-1.208	-1.171	-1.165	-1.188	2.604	-1.198	-1.212
V_2	18.030	17.921	17.979	18.087	18.126	18.148	24.260	18.089	17.985

parameter bounds in Table 2.3 were used for these trials. Hessian information was not provided to the optimizer. For the UKF, the "plateauing" effect of the parameter estimates seen in Figure 2.3 indicates confidence that they are conforming to being constant in time. With 4D-Var, and in a limiting sense of the UKF, the plateauing effect indicates the parameters are settling into a local minimum of the cost function. In Figure 2.8 we show the bifurcation diagrams of the estimated models from our

Table 2.8 4D-Var Parameter Estimates at the End of the Optimization for each Bifurcation Regime with Hessian Provided

	t:HOPF			**t:SNIC**			**t:HOMO**		
	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO	g:HOPF	g:SNIC	g:HOMO
ϕ	0.039	0.039	0.039	0.066	0.066	0.066	0.571	0.560	0.549
g_{Ca}	3.889	3.889	3.889	4.002	4.002	4.002	831.907	911.887	913.350
V_3	1.971	1.971	1.971	11.825	11.825	11.825	826.608	896.717	822.366
V_4	29.533	29.533	29.533	17.071	17.071	17.071	1695.018	1816.501	1813.829
g_{K}	8.050	8.050	8.050	7.923	7.923	7.923	847.999	932.249	885.392
g_{L}	1.928	1.928	1.928	2.027	2.027	2.027	0.024	0.026	0.118
V_1	-1.301	-1.301	-1.301	-1.232	-1.232	-1.232	53.706	54.172	53.913
V_2	17.600	17.600	17.600	18.004	18.004	18.004	75.855	76.135	76.111

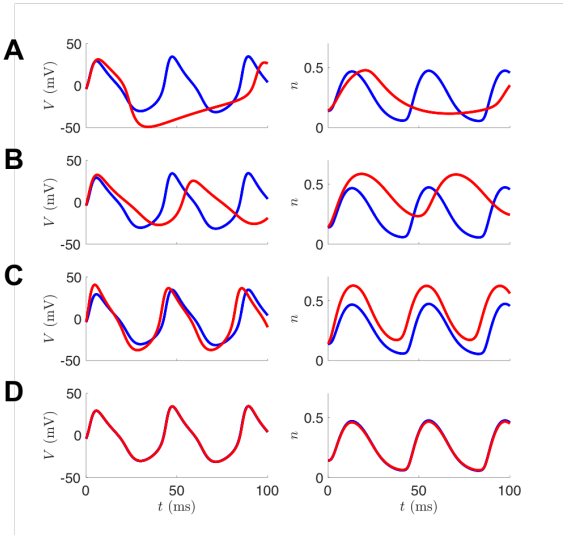


Figure 2.7 Example of 4D-Var assimilation initializing with parameters from the Hopf regime but observational data from the SNIC regime.

4D-Var trials. Notice, and shown explicitly in Table 2.4, when initializing with the true parameters, the correct model parameters are recovered as our optimization routine is confidently within the basin of attraction of the global minimum. In the UKF, comparatively, there is no sense of stopping at a local minimum. Parameter estimates may still fluctuate even when starting from their true values unless the variances of the state components fall to very low values, and the covariance Q_k can be tuned to have a baseline variability in the system. The parameter sets for the SNIC and homoclinic bifurcation regimes only deviate in the ϕ parameter, and so our optimization had great success estimating one from the other. The kinetic parameters (V_3 and V_4) for the Hopf regime deviate quite a bit from the SNIC or homoclinic. Still, the recovered bifurcation structures from estimated parameters associated with trials involving HOPF remained consistent with the true structure.

A drawback of the results shown in Table 2.4 is that for the default tolerances in *fmincon*, some runs took more than two days to complete on a dedicated core. Figure 2.9 shows that the optimal solution had essentially been found after 22,000 iterations, however the optimizer kept running for several thousand more iterations

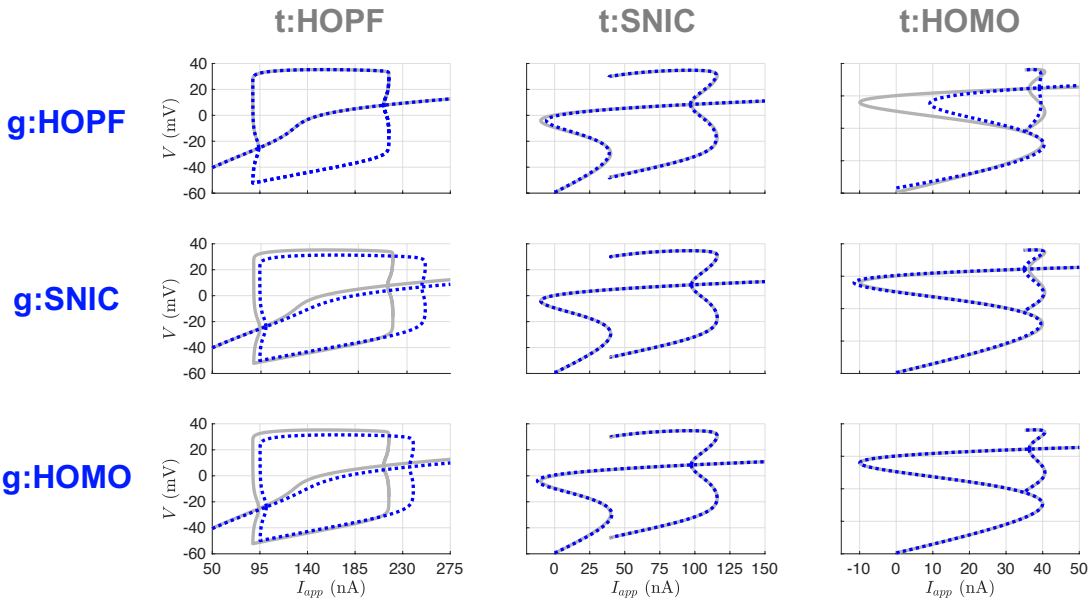


Figure 2.8 Bifurcation diagrams for 4D-Var twin experiments. The gray lines correspond to the true diagrams, and the blue dotted lines correspond to the diagrams produced from the estimated parameters in Table 2.4.

before the convergence tolerances were met. Rather than attempting to speed up these computations by adjusting the algorithmic parameters associated with this solver for this specific problem, we decided to try to exploit the dynamic structure of the model equations using automatic differentiation (AD). AD deconstructs derivatives of the objective function into elementary functions and operations through the chain rule. We used the MATLAB AD tool ADiGator, which performs source transformation via operator overloading, and has scripts available for simple integration with various optimization tools, including *fmincon* [119]. For the same problem scenario and algorithmic parameters, we additionally passed in generated gradient and Hessian functions to the solver. For this problem, the Hessian structure is shown in Figure 2.10. Note that we are using a very simple scheme in the modified Euler method in Equation (2.6) to perform numerical integration between observation points, and the states at $k+1$ only have dependencies upon those at k, and on the parameters. Higher order methods, including implicit methods, can be employed

naturally since the system is being estimated simultaneously. A tutorial specific to collocation methods for optimization has been developed [69].

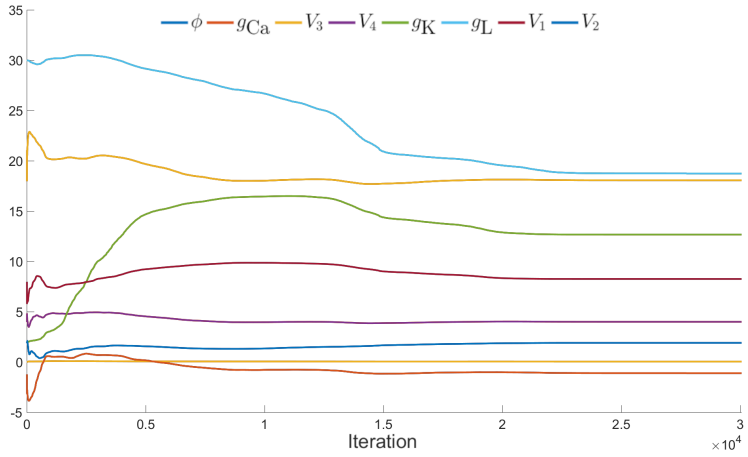


Figure 2.9 Example parameter estimation with 4D-Var initializing with Hopf parameter regime and estimating parameters of SNIC regime.

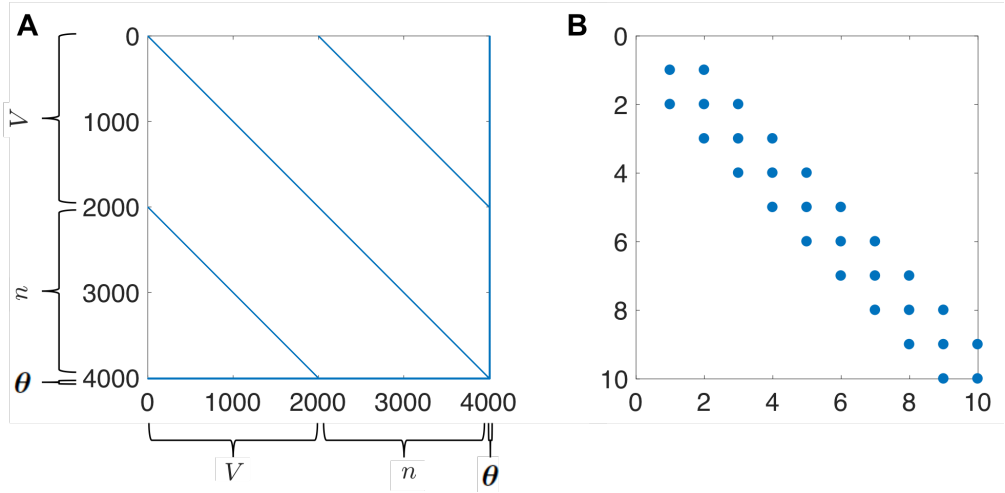


Figure 2.10 **A:** Sparsity pattern for the Hessian of the cost function for the Morris-Lecar equations for $N + 1 = 2001$ time points. The final eight rows (and symmetrically the last eight columns) depict how the states at each time depend upon the parameters. **B:** The top left corner of the Hessian shown in (A).

The results are shown in Table 2.5. The loose parameter bounds in Table 2.6 were used for these trials. Hessian information was provided to the optimizer. Each twin experiment scenario took, at most, a few minutes on a dedicated core. These

trials converged to the optimal solution in many fewer iterations than the trials without using the Hessian. Since convergence was achieved within a few dozen iterations, we decided to inspect how the bifurcation structure of the estimated model evolved throughout the process for the case of HOPF to SNIC. Figure 2.11 shows that by Iteration 10, the objective function value has decreased greatly, and parameters that produce a qualitatively correct bifurcation structure have been found. The optimization continues for another 37 iterations, and explores other parts of parameter space that do not yield the correct bifurcation structure, before converging very close to the true parameter values. The iterates were generated from *fmincon* with provided Hessian and gradient functions. For Figure 2.11**B**, the blue is the initial bifurcation structure, the gray is the true bifurcation structure for the parameters that generated the observed data, the red is the bifurcation structure of the iterates, and the green is the bifurcation structure of the optimal point determined by *fmincon*.

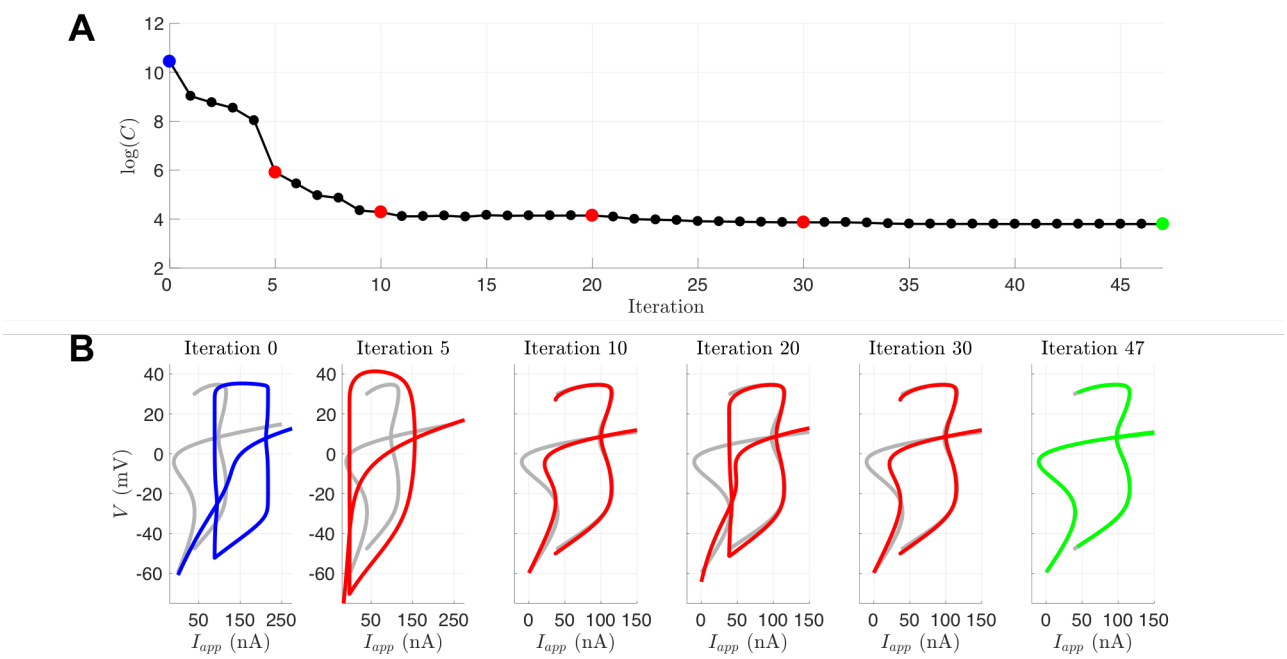


Figure 2.11 **A** shows the logarithm of the value of the cost function for a twin experiment initialized with parameters from the Hopf regime but observational data from the SNIC regime. **B** shows the bifurcation diagrams produced from parameter estimates for selected iterations

We also wished to understand more about the sensitivity of this problem to initial conditions. We initialized the system with the voltage states as those of the observation, the parameters as those of the initializing guess bifurcation regime, and the gating variable $[n_0, n_1, \ldots n_N]$ to be i.i.d. from $\mathcal{U}(0,1)$. The results confirm our suspicions that multiple local minima exist. For 100 different initializations of n, for the problem of going from SNIC to HOPF, 63 were found to fall into a deeper minima, yielding better estimates and a smaller objective function value, while 16 fell into a shallower minima, and the rest into three different even shallower minima. While one cannot truly visualize high-dimensional manifolds, one can try to visualize a subset of the surface. Figure 1.3 shows the surface that arises from evaluating the objective function on a linear combination of the two deepest minima and an initial condition, $\mathbf{x}_0$, which eventually landed in the shallower of the two minima as points in 4010-dimensional space.

Again, these results, at best, can reflect only locally optimal solutions of the optimization manifold. The 4D-Var framework has been applied to neuroscience using a more systematic approach to finding the global optimum. In [122], a population of initial states $\mathbf{x}$ are optimized in parallel, and an outer loop incorporating an *annealing* algorithm. The annealing parameter relates the weights of the two summations in Equation (3.4), and the iteration proceeds by increasing the weight given to the model error compared to the measurement error.

2.2 Application to Bursting Regimes of the Morris-Lecar Model

Many types of neurons display burst firing, consisting of groups of spikes separated by periods of quiescence. Bursting arises from the interplay of fast currents that generate spiking and slow currents that modulate the spiking activity. The Morris-Lecar model can be modified to exhibit bursting by including a calcium-gated potassium (K_{Ca})

current that depends on slow intracellular calcium dynamics [95]:

$$\begin{aligned} C_m \frac{dV}{dt} &= I_{app} - g_{\mathrm{L}}(V - E_{\mathrm{L}}) - g_{\mathrm{K}} n (V - E_{\mathrm{K}}) \\ &\quad - g_{\mathrm{Ca}} m_\infty(V)(V - E_{\mathrm{Ca}}) - g_{\mathrm{KCa}} z (V - E_{\mathrm{K}}) \end{aligned} \tag{2.9}$$

$$\frac{dn}{dt} = \phi(n_\infty(V) - n)/\tau_n(V) \tag{2.10}$$

$$\frac{dCa}{dt} = \epsilon\left(-\mu I_{\mathrm{Ca}} - Ca\right) \tag{2.11}$$

$$z = \frac{Ca}{Ca + 1}. \tag{2.12}$$

Bursting can be analyzed mathematically by decomposing models into fast and slow subsystems and applying geometric singular perturbation theory. Several different types of bursters have been classified based on the bifurcation structure of the fast subsystem. In square-wave bursting, the active phase of the burst is initiated at a saddle-node bifurcation and terminates at a homoclinic bifurcation. In elliptic bursting, spiking begins at a Hopf bifurcation and terminates at a saddle-node of periodic orbits bifurcation. The voltage traces produced by these two types of bursting are quite distinct, as shown in Figure 2.12. The bifurcation diagrams (top row) depict stable fixed points (red), unstable fixed points (black), stable limit cycles (blue), and unstable limit cycles (green) of the fast subsystem (V, n) with bifurcation parameter z. The gray curves are the projection of the 3-D burst trajectory (V, second row; n, third row; Ca, fourth row) onto the (V, z) plane, where z is a function of Ca. **A**: During the the quiescent phase of the burst, Ca and therefore z are decreasing and the trajectory slowly moves leftward along the lower stable branch of fixed points until reaching the saddle-node bifurcation or "knee", at which point spiking begins. During spiking, Ca and z are slowly increasing and the trajectory oscillates

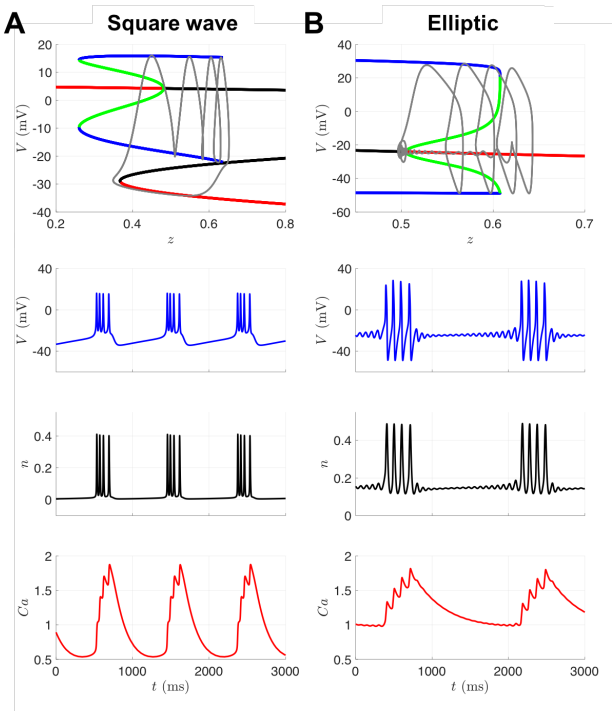


Figure 2.12 Bursting model bifurcation diagrams and trajectories.

while traveling rightward until the stable limit cycle is destroyed at a homoclinic bifurcation and spiking ceases. **B:** During the quiescent phase of the burst, z is decreasing and the trajectory moves leftward along the branch of stable fixed points with small-amplitude decaying oscillations until reaching the Hopf bifurcation, at which point the oscillations grow in amplitude to full spikes. During spiking, z is slowly increasing and the trajectory oscillates while traveling rightward until the stable limit cycle is destroyed at a saddle-node of periodic orbits bifurcation and spiking ceases. The particular parameters are given by Table 2.9. For square-wave bursting $I_{\text{app}} = 45$, and for elliptic bursting $I_{\text{app}} = 120$. All other parameters are the same as in Table 2.1.

2.2.1 Results with UKF

We conducted a set of twin experiments for the bursting model to address the same question as we did for the spiking model: from a voltage trace alone, can

Table 2.9 Parameters for Bursting in the Modified Morris-Lecar Model

	Square-wave	Elliptic
ϕ	0.23	0.04
g_{Ca}	4	4.4
V_3	12	2
V_4	17.4	30
g_{K}	8	8
g_{L}	2	2
V_1	-1.2	-1.2
V_2	18	18
g_{KCa}	0.25	0.75
ϵ	0.005	0.005
μ	0.02	0.02

DA methods estimate parameters that yield the appropriate qualitative dynamical behavior? Specifically, we simulated data from the square-wave (elliptic) bursting regime, and then initialized the UKF with parameter guesses corresponding to elliptic (square-wave) bursting. As a control experiment we also ran the UKF with initial parameter guesses corresponding to the same bursting regime as the observed data. The observed voltage trace included additive white noise generated following the same protocol as in previous trials. We used 200,001 time points with observations at every 1 ms. Between observations, the system is integrated forward using substeps of 0.025 ms. For the square-wave burster, this includes 215 bursts with four spikes per burst, and 225 bursts with two spikes for the elliptic burster.

The small parameters ϵ and μ in the calcium dynamics equation were assumed to be known and were not estimated by the UKF. Thus, for the bursting model, we are estimating one additional state variable (Ca) and one additional parameter ($g_{\text{app}}KCa$) than was the case for the spiking model. Table 2.10 shows the UKF parameter estimates after initialization with either the true parameters or the parameters producing the other type of bursting. The results for either case are quite consistent and fairly close to their true values for both types of bursting. Since small changes in parameter values can affect bursting dynamics, we also computed bifurcation diagrams for these estimated parameters and compared them to their true

counterparts. Figure 2.13 shows that in all four cases, the estimated models have the same qualitative bifurcation structure as the models that produced the data. The recovered parameter estimates were insensitive to initial conditions for n and Ca, with 100 different initializations for these state variables sampled from $\mathcal{U}(0,1)$ and $\mathcal{U}(0,5)$, respectively. Note, most predominantly in the top right panel, the location of the bifurcations are relatively sensitive to small deviations in certain parameters, such as g_{KCa}. Estimating g_{KCa} is challenging due to the algebraic degeneracy of estimating both terms involved in the conductance $G_{\text{KCa}} = g_{\text{KCa}} Ca/(Ca+1)$, and the inherent time-scale disparity of the Ca dynamics compared to V and n. If one had observations of calcium, or full knowledge of its dynamical equations, this degeneracy is immediately alleviated. To address difficulties in the estimation of bursting models, an approach has been developed that separates the estimation problem into two stages based on timescales, first estimating the slow dynamics with the fast dynamics blocked and then estimating the fast dynamics with the slow parameters held fixed [67]

Table 2.10 UKF Parameter Estimates for each Bursting Regime

	t:Square-wave		**t:Elliptic**	
	g:Square-wave	g:Elliptic	g:Square-wave	g:Elliptic
ϕ	0.214	0.215	0.040	0.040
g_{Ca}	3.758	3.767	4.396	4.398
V_3	12.045	12.023	1.603	1.685
V_4	16.272	16.316	29.582	29.639
g_{K}	7.955	7.952	7.866	7.889
g_{L}	1.974	1.972	2.015	2.017
V_1	-1.514	-1.511	-1.120	-1.199
V_2	17.640	17.624	18.010	18.015
g_{KCa}	0.251	0.251	0.767	0.763

2.2.2 Results with 4D-Var

We also investigated the utility of variational techniques to recover the mechanisms of bursting. For these runs, we took our observations to be coarsely sampled at 0.1 ms, and our forward mapping is taken to be one step of modified Euler between

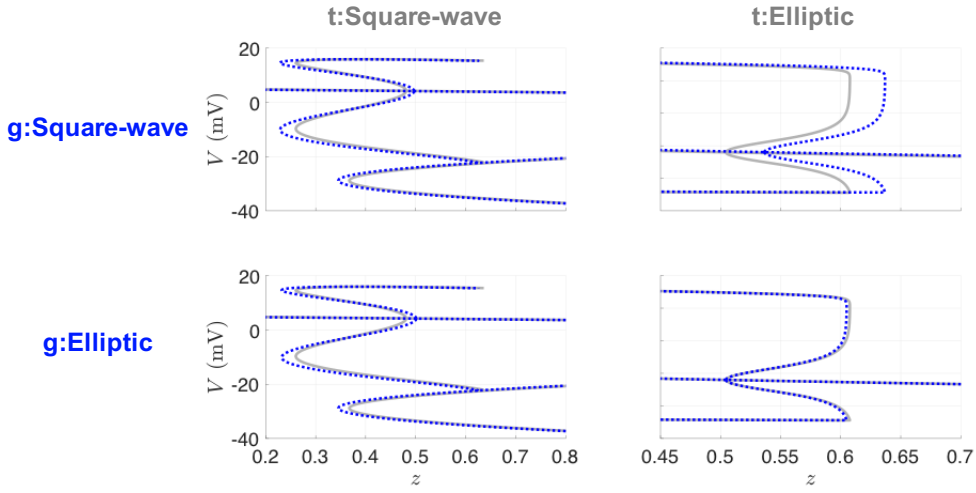


Figure 2.13 Bifurcation diagrams for UKF twin experiments for the bursting Morris-Lecar model. The gray lines correspond to the true diagrams, and the blue dotted lines correspond to the diagrams produced from the estimated parameters in Table 2.10.

observation times, as was the case for our previous 4D-Var Morris-Lecar results. We used 10,000 time points, which is 1 burst for the square wave burster, and 1 full burst plus another spike for the elliptic burster. We used the *L-BFGS-B* method [123], as we found it to perform faster for this problem than *fmincon*. This method approximates the Broyden-Fletcher-Goldfarb-Shanno (BFGS) Quasi-Newton algorithm using a limited memory (L) inverse Hessian approximation, with an extension to handle bound constraints (B). It is available for Windows through the OPTI toolbox [33] or through a nonspecific operating system MATLAB MEX wrapper [10, 23]. We supplied the gradient of the objective function, but allowed the solver to define the limited-memory Hessian approximation for our $30,012$ dimensional problem. The results are captured in Table 2.11. We performed the same tests with providing the Hessian; however, there was no significant gain in accuracy or speed. The value for g_{KCa} for initializing with the square wave parameters and estimating the elliptical parameters is quite off, which reflects our earlier assessment for the value in observing calcium dynamics. Figure 2.14 shows that we are still successful in recovering the true bifurcation structure.

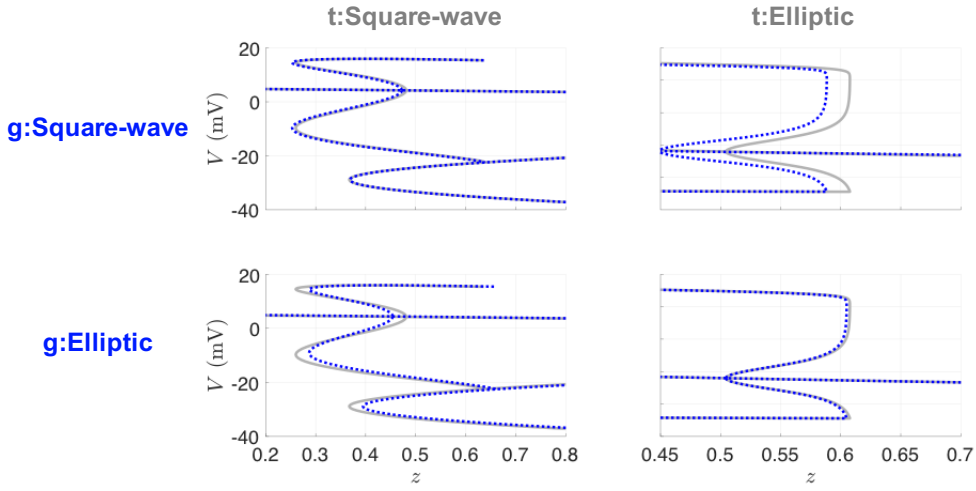


Figure 2.14 Bifurcation diagrams for 4D-Var twin experiments for the bursting Morris-Lecar model. The gray lines correspond to the true diagrams, and the blue dotted lines correspond to the diagrams produced from the estimated parameters in Table 2.11.

Table 2.11 4D-Var Parameter Estimates for each Bursting Regime

	t:Square-wave		**t:Elliptic**	
	g:Square-wave	g:Elliptic	g:Square-wave	g:Elliptic
ϕ	0.230	0.260	0.037	0.040
g_{Ca}	4.009	4.509	4.244	4.412
V_3	12.009	11.920	6.667	1.971
V_4	17.437	19.581	32.605	30.026
g_{K}	8.006	8.244	9.485	8.002
g_{L}	2.003	2.068	1.979	2.009
V_1	-1.187	-0.627	-1.307	-1.172
V_2	18.029	18.754	17.469	18.049
g_{KCa}	0.250	0.237	0.554	0.741

2.3 Discussion and Conclusions of Twin Experiments

Data assimilation is a framework by which one can optimally combine measurements and a model of a system. In neuroscience, depending on the neural system of interest, the data we have may unveil only a small subset of the overall activity of the system. For the results presented here, we used simulated data from the Morris-Lecar model with distinct activity based upon different choices for model parameters. We assumed access only to the voltage and the input current, which corresponds to the expected data from a current-clamp recording.

We showed the effectiveness of standard implementations of the Unscented Kalman Filter and weak 4D-Var to recover spiking behavior and, in many circumstances, near-exact parameters of interest. We showed that the estimated models undergo the same bifurcations as the model that produced the observed data, even when the initial parameter guesses do not. Additionally, we are also provided with estimates of the states and uncertainties associated with each state and parameter, but for sake of brevity these values were not always displayed. The methods, while not insensitive to noise, have intrinsic weightings of measurement deviations to account for the noise of the observed signal. Results were shown for mild additive noise. We also extended the Morris-Lecar model to exhibit bursting activity, and demonstrated the ability to recover these model parameters using the UKF.

The UKF and 4D-Var approaches implemented here both attempt to optimally link a dynamic model of a system to observed data from that system, with error statistics assumed to be Gaussian. Furthermore, both approaches try to approximate the mean (and for the UKF also the variance) of the underlying, unassumed system distributions. The UKF is especially adept at estimating states over long time courses, and if the algorithmic parameters such as the model error can be tuned, then the parameters can be estimated simultaneously. Therefore, if one has access to a long series of data, then the UKF (or an Unscented Kalman Smoother, which uses more history of the data for each update step) is a great tool to have at one's disposal. However, sometimes one only has a small amount of time series data, or the tuning of initial covariance, the spread parameter λ, and the process noise Q_k associated with the augmented state and parameter system becomes too daunting. The 4D-Var approach sets the states at each time point and the parameters as optimization variables, transitioning the estimation process from one which iterates in time, to one which iterates up to a tolerance in a chosen optimization routine. The only tuning parameters are those associated with the chosen optimization routine, and

the weights, $Q_{l,l}^{-1}$, $l \in [1 \dots L]$, for the model uncertainty of the state components at each time. There are natural ways to provide parameter bounds in the 4D-Var framework, whereas this is not the case for the UKF. However, depending upon the implementation choices and the dimension of the problem (which is extremely large for long time series data), then the optimization may take a computing time scale of days to yield reasonable estimates. Fortunately, derivative information can be provided to the optimizer to speed up the 4D-Var procedure. Both the UKF and 4D-Var can provide estimates of system uncertainty in addition to estimates of the system mean. The UKF provides mean and variance estimates at each iteration during the analysis step. In 4D-Var, one seeks mean estimates by minimization of a cost function. It has been shown that for cost functions of the form in Equation (3.4), the system variance can be interpreted as the inverse of the Hessian evaluated at minima of Equation (3.4), and scales roughly as Q for large Q^{-1} [122]. The pros and cons of implementing these two DA approaches are summarized in Table 2.12.

Table 2.12 Comparison of the Sequential (UKF) and Variational (4D-Var) Approaches to Data Assimilation

	UKF	4D-Var
Implementation choices	initial covariance (P_{xx}) sigma points (λ) process covariance matrix (Q)	model uncertainty (Q^{-1}) type of optimizer / optimizer settings state and parameter bounds
Data requirements	**Pro:** can handle a large amount of data **Con:** may not find a good solution with a small amount of data	**Pro:** may find a good solution with a small amount of data **Con:** cannot handle a large amount of data
Run Time	Minutes	Days, hours, or minutes depending on choice of optimizer and settings
Scalability to Larger Models	Harder to choose Q EnKF may use smaller number of ensemble members	Search dimension is $(N+1)L+D$ Sparse Hessian can be exploited during optimization

The UKF and 4D-Var methodologies welcome the addition of any observables of the system, but current-clamp data may be all that is available. With this experimental data in mind, for a more complex system, the number of variables increases while the total number of observables will remain at unity. Therefore, it may be useful to assess *a priori* which parameters are structurally identifiable and

the sensitivity of the model to parameters of interest in order to reduce the estimation state space [88]. Additionally, one should consider what manner of applied current to use to aid in state and parameter estimation. In the results presented above, we used a constant applied current, but work has been done which suggests the use of complex time-varying currents that stimulate as many of the model's degrees of freedom as possible [3].

The results we presented are based on MATLAB implementations of the derived equations for the UKF and weak 4D-Var. Sample code is provided in the Appendix. Additional data assimilation examples in MATLAB can be found in [73]. The UKF has been applied to other spiking neuron models such as the FitzHugh-Nagumo model [111]. A sample of this code can be found in [99], as well as further exploration of the UKF in estimating neural systems. The UKF has been used on real data from pyramidal neurons to track the states and externally applied current [118], the connectivity of cultured neuronal networks sampled by a microelectrode array [52], to assimilate seizure data from hippocampal OLM interneurons [107], and to reconstruct mammalian sleep dynamics [101]. A comparative study of the efficacy of the EKF and UKF on conductance-based models has been conducted [71].

The UKF is a particularly good framework for the state dimensions of a single compartment conductance based model as the size of the ensemble is chosen to be $2(L + D) + 1$. When considering larger state dimensions, as is the case for PDE models, a more general Ensemble Kalman Filter (EnKF) may be appropriate. An introduction to the EnKF can be found in [43, 44]. An adaptive methodology using past innovations to iteratively estimate the model and measurement covariances Q and R has been developed for use with ensemble filters [12]. The Local Ensemble Tranform Kalman Filter (LETKF) [60] has been used to estimate the states associated with cardiac electrical wave dynamics [57]. Rather than estimating the mean and covariance through an ensemble, particle filters aim to fully construct the posterior

density of the states conditioned on the observations. A particle filter approach has been applied to infer parameters of a stochastic Morris-Lecar model [39], to assimilate spike train data from rat layer V cortical neurons into a biophysical model [80], and to assimilate noisy, model-generated data for other states to motivate the use of imaging techniques when available [62].

An approach to the variational problem which tries to uncover the global minima more systematically has been developed [122]. In this framework, comparing to Equation (3.4), they define for diagonal entries of Q^{-1} that

$$Q^{-1} = Q_0^{-1}\alpha^\beta$$

for $\alpha > 1$ and $\beta \geq 0$. The model term is initialized as relatively small, and over the course of an annealing procedure, β is incremented resulting in a steady increase of the model term's influence on the cost function. This annealing schedule is conducted in parallel for different initial guesses for the state space. The development of this variational approach can be found in [1] and it has been used to assimilate neuronal data from HVC neurons [67] as well as to calibrate a neuromorphic very large scale integrated (VLSI) circuit [116]. An alternative to the variational approach is to frame the assimilation problem from a probabilistic sampling perspective and use Markov Chain Monte-Carlo (MCMC) methods [70]. We will present an associated formulation using MCMC methods in Chaper 7, albeit without verifying results.

A closely associated variational technique, known as "nudging", augments the vector field with a control term. If we only have observations of the voltage, this manifests as:

$$\frac{dV}{dt} = f_V^\star(V, \mathbf{q}; \boldsymbol{\theta}) + u(V_{\text{obs}} - V)$$

The vector field with the observational coupling term is now passed into the strong 4D-Var constraints. The control parameter u may remain fixed, or be estimated along with the states [2, 106]. More details on nudging can be found [89]. A similar control framework has been applied to data from neurons of the stomatogastric ganglion [17]. We will delve into this controlled 4D-Var in Chapter 3.

Many other approaches outside the framework of data assimilation have been developed for parameter estimation of neuronal models, see [109] for a review. A problem often encountered when fitting models to a voltage trace is that phase shifts, or small differences in spike timing, between model output and the data can result in large root mean square error. This is less of an issue for data assimilation methods, especially sequential algorithms like UKF. Other approaches to avoid harshly penalizing spike timing errors in the cost function are to consider spikes in the data and model-generated spikes that occur within a narrow time window of each other as coincident [97], or to minimize error with respect to the dV/dt versus V phase-plane trajectory rather than $V(t)$ itself [109]. Another way to avoid spike mismatch errors is to force the model with the voltage data and perform linear regression to estimate the linear parameters (maximal conductances), and then perhaps couple the problem with another optimization strategy to access the nonlinearly-dependent gating parameters [54, 61, 74]. A dive into this technique will be presented in Chapter 7.

A common optimization strategy is to construct an objective function that encapsulates important features derived from the voltage trace, and then use a genetic algorithm to stochastically search for optimal solutions. These algorithms proceed by forming a population of possible solutions and applying biologically inspired evolution strategies to gradually increase the fitness (defined with respect to the objective function) of the population across generations. Multi-objective optimization schemes will generate a "Pareto front" of optimal solutions that are considered equally good.

A multi-objective non-dominated sorting genetic algorithm (NSGA-II) has recently been used to estimate parameters of the pacemaker PD neurons of the crab pyloric network [35, 46].

In this chapter, we compared the bifurcation structure of models estimated by DA algorithms to the bifurcation structure of the model that generated the data. We found that the estimated models exhibited the correct bifurcations even when the algorithms were initiated in a region of parameter space corresponding to a different bifurcation regime. This type of twin experiment is a useful addition to the field that specifically emphasizes the difficulty of nonlinear estimation and provides a qualitative measure of estimation success or failure. Prior literature on parameter estimation that has made use of geometric structure includes work on bursting respiratory neurons [104] and "inverse bifurcation analysis" of gene regulatory networks [41,76]. The work discussed in Chapters 1 and 2 appeared in Moye and Diekman (2018) [83].

The following chapter will expand on the ideas of variational data and apply controlled 4D-Var, i.e., nudging, to a two-compartmental Hodgkin-Huxley-type model. We will then apply this technique estimate novel models of circadian neurons of a diurnal species of rodent.

CHAPTER 3

CONTROLLED 4D-VAR AND APPLICATION TO CIRCADIAN NEURONS

We aim to do a deeper dive into the applicability of 4D-Var for more complex models in neuroscience. We start by exploring the formulations of 4D-Var and settle on using a controlled variant known as dynamical state and parameter estimation (DSPE). We apply this to a two compartmental model of a Hodgkin-Huxley type model to assess its performance. We then apply this technique to *in vitro* data from suprachiasmatic nucleus (SCN) neurons of the *Rhabdomys*, a diurnal rodent.

3.1 4D-Var Formulation

4D-Var has a few formulations so, to begin, for our state-space system:

$$x_{k+1} = f(x_k) + \omega_{k+1}, \qquad \omega_k \in R^L \tag{3.1}$$

$$y_{k+1} = h(x_{k+1}) + \eta_{k+1}, \quad \eta_{k+1} \in R^M \tag{3.2}$$

Let us remind ourselves of the 4D-Var problem:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} \|y_k - H(x_k)\|^2_{R_k} + \frac{1}{2}\sum_{k=0}^{N-1} \|x_{k+1} - f(x_k)\|^2_{P_k^b} \tag{3.3}$$

If one assumes the states follow the state space description in (3.1) and (3.2) with $\omega_k \sim \mathcal{N}(0,\, Q)$ and $\eta_k \sim \mathcal{N}(0,\, R)$ where Q is our model error covariance matrix and R is our measurement error covariance matrix, that these have no cross-covariance,

and only voltage data is observed, the cost function can be simplified a bit:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} R^{-1}\left(y_k - V_k\right)^2 + \frac{1}{2}\sum_{l=1}^{L}\sum_{k=0}^{N-1} Q_{l,l}^{-1}\left(x_{l,k+1} - f_l(x_k)\right)^2 \tag{3.4}$$

where $V_k = x_{1,k}$. (3.4) would be the equation one seeks to minimize for the current-clamp data problem in neuroscience in what is called *Weak 4d-var*. Alternatively, *Strong 4d-var* forces the resulting estimates to be consistent with the model, f. This can be considered the result of taking $Q \to \mathbf{0}$, which yields the nonlinearly constrained problem:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} R^{-1}\left(y_k - V_k\right)^2 \tag{3.5}$$

such that

$$x_{k+1} = f(x_k), \quad k = 0 \ldots N \tag{3.6}$$

where R^{-1} can now be scaled out completely. In the cost function in (3.4), the estimated voltage is expected to be consistent with dynamics for large model weighting Q^{-1}, but the dynamics cannot possibly reproduce the irregularity in the data, so, by default, it is not apparent what the utility of taking this limit may be. The problem becomes *far* worse conditioned as we have removed the capacity of our optimization problem to naturally account for the model errors which truly exist. However, more specifically, we prevent ourselves from having a regularization.

One exceptional avenue which has been explored is known as variational annealing [122]. When the coefficient of the model error in equation (3.3) is zero, the minima is widely degenerate as the model, and any unmeasured parameters or

variables, are noninfluential to the solution of the system, which is $x_k = H^{-1}(y_k)$. In variational annealing, $R_m \equiv Q^{-1}$ is incrementally increased such that the degeneracy breaks, and different admissible solutions amongst the degenerate landscape are tracked. The model coefficient is increased as $R_m^\beta = R_{m,0}\alpha^\beta$, for $\alpha > 1$ and $\beta \geq 0$. The term "annealing" comes from the perspective of active particles in the degenerate well slowly cooling to manufacture the multi-minima, nonconvex landscape. In keeping with the temperature theme, this could be considered either incremental "warm-starting" or "hot-starting" depending on how the optimization is implemented. For warm-starting, purely the solution for one value of β is retained before incrementing and solving for a larger value of β. For hot-starting, the dual variables and multipliers are initialized as their final values from the previous iteration if using an interior point method.

Another way to view variational annealing is through the use of slack variables to lift the optimization. With strong 4D-Var, the dynamics are imposed as equality constraints. If we would want to allow our model to be an imperfect match, we could impose inequality constraints such that $-\epsilon \leq x_{k+1} - f(x_k) \leq \epsilon$ for ϵ small enough to not completely override the dynamics between time points. Slack variables are decision variables added to the inequality constraints to translate them to equality constraints, although slack variables by traditional definition should not be negative lest that point is infeasible. We can use these pseudo-slack variables to lift our optimization problem to a higher dimension, where we incrementally impose that our slack is penalized in place of our model error term. The problem then becomes:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} R^{-1}\left(y_k - V_k\right)^2 + \frac{1}{2}\sum_{l=1}^{L}\sum_{k=0}^{N-1} Q_{l,l}^{-1}(s_{l,k+1})^2 \tag{3.7}$$

such that

$$x_{l,k+1} - f_l(x_k) - s_{l,k+1} = 0, \quad k = 0 \ldots N, \quad l = 1 \ldots L \tag{3.8}$$

with

$$-\epsilon_l \leq s_{l,k+1} \leq \epsilon_l \tag{3.9}$$

Which looks quite similar to 3.4, but the system dimension now has increased proportionally to the number of constraints (LN).

We will not present any results using strict variational annealing in this work, but we have validated many of our previous, and future results, through an implementation of the "pseudo-slack" annealing. If a problem is highly irregular, we include these slack variables for fixed, very large level of the slack coefficient $Q_{l,l}^{-1}$.

In addition, there is another mechanism to regularize the optimization problem which utilizes dynamical systems theory for synchronizing two systems. Pecora and Carrol coauthored a series of papers [92] describing extensions from a basic idea: suppose one has a driving system and a response system; under what conditions and to what degree does the response system synchronize to the driving system [91]? A motivating example uses the Lorenz 1963 equations, where the primes denote the response variables:

$$\frac{dx}{dt} = \sigma(y + z) \tag{3.10}$$

$$\frac{dy}{dt} = -xz + rx - y, \quad \frac{dy'}{dt} = -xz' + rx - y' \tag{3.11}$$

$$\frac{dz}{dt} = xy - bz, \quad \frac{dz'}{dt} = xy' - bz' \tag{3.12}$$

The response variables are taken to have different initial conditions than the driving variables. As time evolves to infinity, the response variables for this system will converge to their counterparts in the driving subsystem. However, by choosing a different driving variable (e.g., y), the response will not converge to the driving subsystem. The authors formalized this by developing what are referred to as "Conditional Lyapunov Exponents" (CLEs). Considering the Lorenz system above, one can develop a variational equation by setting $y' = y + \delta y$ and $z' = z + \delta z$, then subtract the $y - z$ subsystems:

$$\frac{\delta y}{dt} = -\delta y - (\delta z)x \tag{3.13}$$

$$\frac{\delta z}{dt} = x(\delta y) - b(\delta z) \tag{3.14}$$

For $\sigma = 10, b = 8/3, r = 60$, the CLEs of the system are (-1.81,-1.86). For this variational system, the unstable manifold is empty. For the overall problem we can say that the "synchronization manifold" is stable, and we expect the slave system to synchronize to the driving system. Further examples are given in [92].

Dynamical State and Parameter Estimation (DSPE) is a technique described in [2], with the premise being to stabilize the synchronization manifold of "nudging" problems from data assimilation. Further details can be found in [1]. For instance, recalling the Morris-Lecar example before, for equations (2.1) and (2.2), the controlled system becomes

$$\frac{dV}{dt} = f_V(n, V; \theta) + u(V_{\text{obs}} - V) \tag{3.15}$$

$$\frac{dn}{dt} = f_n(n, V; \theta) \tag{3.16}$$

where u can be taken as time dependent or fixed. The cost function then becomes:

$$C(x) = \frac{1}{2}\sum_{k=0}^{N} R^{-1}\left(y_k - V_k\right)^2 + \sum_{k=0}^{N} u_k^2 \tag{3.17}$$

This synchronization procedure has also been considered for specific function forms of u in the neuroscience context in [17] wherein they set up an optimal search strategy applied to real data. The nudging strategy in general has been used in geosciences primarily for state estimation [89]. As shown in [2,106] the control u acts to reduce conditional Lyapunov exponents.

The goal of DSPE is to define a high dimensional cost functional which weakly constrains the estimated states to the system observations, and strongly constrains the estimates to the controlled model dynamics while penalizing the control. Specifically, the procedure aims to minimize equation (3.17) subject to equations (3.15)-(3.16) in the case of Morris-Lecar. Without the control, the problem is explicitly formulated as a strong constraint 4D-Var. However, the basin of attraction for global minima along the optimization manifold is shallow. Also, while the minimization term itself is convex, the nonlinearities present in the model constraints generate a large degree of nonconvexity in the solution manifold. The intended effect of the nudging term is to smoothen the surface. Given that the system is so high dimensional and tightly-coupled, formally visualizing this surface is not achievable for our parameter estimation problems.

In the DSPE framework, parameters and states at each point in time are taken on equal footing. Namely, the solution space of the cost function is $(L+1)(N+1)+D$ where D is the number of fixed parameters to infer, L the number of dynamical variables, and we are additionally solving for the control $u(t)$ at each point in time. The control is penalized in (3.17) quadratically in an effort to reduce the impact of it

at the end of the optimization procedure. While having the control present enforces the data in the model equations, by minimizing it, one is attempting to recover back the minima subject to the uncontrolled model of the system. So, as $u \to 0$ over the course of the optimization, the physical system strong constraint is recovered.

There are theoretic underpinnings for this technique, but when working with complicated models and real data, the reality is that whatever technique "works" best and most reliably is the one that should be employed. Is strong 4D-var too ill-conditioned to work at all and the weak variant unconditionally better? Does nudging actually combat this ill-conditioning or does it merely introduce a higher-dimensionality into our workspace? What are the requirements on the bounds of our states and parameters?

To get started, let us be clear on the problem we are solving for. Here we are performing what is known as "direct transcription", where we transcribe our cost functional and dynamical equations into a nonlinear program. For simplicity, let's take $n = (L+1)(N+1) + D$

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & C(x) \\ \text{s.t.} \quad & g(x) = 0 \\ & x^L \leq x \leq x^U \end{aligned} \tag{3.18}$$

Where x is our combined state space at every time point, any controls, and parameters, and $C(x)$ is equation (3.17) if using a control, else the control term is excluded.

We must choose a particular transcription method to prescribe our equality constraints $g(x) = 0$. Prior to transcription, $g(x) = 0$ is the generalization of our system dynamics. Let's define our state vector as $\mathbf{x} = (V, \bar{\mathbf{x}})$ and our uncontrolled dynamics as:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}_{\mathbf{x}}(\mathbf{x}, \theta) \tag{3.19}$$

where we can separate out the terms with observations. We assume we only have observations of the voltage of one cell in one compartment (with natural generalizations to network/ multi-compartment descriptions):

$$\frac{dV}{dt} = f_V(\bar{\mathbf{x}}, V; \theta) \tag{3.20}$$

$$\frac{d\bar{\mathbf{x}}}{dt} = \mathbf{f}_{\bar{\mathbf{x}}}(\bar{\mathbf{x}}, V; \theta) \tag{3.21}$$

Then our controlled dynamics become

$$\frac{dV}{dt} = f_V(\bar{\mathbf{x}}, V; \theta) + u(V_{\text{obs}} - V) \tag{3.22}$$

$$\frac{d\bar{\mathbf{x}}}{dt} = \mathbf{f}_{\bar{\mathbf{x}}}(\bar{\mathbf{x}}, V; \theta) \tag{3.23}$$

Where it is understood that $u(t)$ appears only at observational times.

For our results with Morris-Lecar, we used what could be considered a multiple-shooting type approach, or very simple collocation, but incorporated them into the cost function in our model error term. More recently, we have formulated the constraints using either a multiple-shooting style approach using a fourth-order Runge-Kutta method (RK4) or Hermite-Simpson Collocation. We will assume measurements are taken uniformly at

$$t_k = t_0 + k\tau_{\text{obs}}$$

Neuroscience experimental machinery are fairly reliable, so we can operate under the assumption that no time points of data are missed. High resolution of our measurements are preferred so that we can have control and knowledge of the system at basically every knot point. However, there are circumstances where we may not have data with that level of precision or we may desire to downsample our data. For that reason, we will say that we have a set of times upon which our constraint equations are satisfied, namely

$$t_m = t_0 + m\tau_{\text{col}}$$

where we simply require that the ratio of these time differences is a positive integer.

$$\frac{\tau_{\text{obs}}}{\tau_{\text{col}}} \in \mathbb{N}$$

To reiterate, the constraints are what connect each our time points $[t_m, t_{m+1}]$ to one another. For a multiple shooting approach, we have an even further option of the "integration" time-step underpinning our RK4 method, compared to the length of time between segments in our transcription, compared to the observational sampling time. Each of these values will be explicitly mentioned whenever results are presented. More commonly, we choose to use a direct collocation method due to the stability options afforded to us for our highly complex, nonlinear problem. With collocation, implementation of implicit methods are effectively as simple as explicit methods. We choose to use Hermite-Simpson collocation which approximates the set of discrete integrations using Simpson's rule. We introduce midpoints in this fashion ($x_{k+\frac{1}{2}}$), which are approximated using Hermite interpolation.

$$x_{k+1} - x_k = \frac{1}{6} h_k \left(f_k + 4 f_{k+\frac{1}{2}} + f_{k+1} \right) \tag{3.24}$$

$$x_{k+\frac{1}{2}} = \frac{1}{2} \left(x_k + x_{k+1} \right) + \frac{h_k}{8} \left(f_k - f_{k+1} \right) \tag{3.25}$$

For $f_k = \mathbf{f_x}(\mathbf{x_k}, \hat{\theta})$, and $\hat{\theta}$ is the present estimate of θ, constant across our time window.

Here, we take the midpoint and endpoint conditions on equivalent footing for our constraints, $g(x) = 0$, in what is known as its "separated form". Therefore, we implement these equations so that $h_k = 2\tau_{\text{col}}$ based upon our previous notation, and we have LN equality constraints.

Often, additional constraints are imposed upon the control for completeness and smoothness. If so, this will result in the number of variables itself being increased by $N/2$ as the Hermite interpolation is imposed on the control, and additional variables for the "slope" of the control can be utilized.

$$u_{k+\frac{1}{2}} = \frac{1}{2} \left(u_k + u_{k+1} \right) + \frac{h_k}{8} \left(du_k - du_{k+1} \right) \tag{3.26}$$

where du represent the additional slope control variables [31, 105]

Through comparative experiments, we found no need for these additional constraints, but there may be circumstances in which they are important. In fact, we suspect that by imposing continuity, if the control is nonzero for even the lowest minima, the continuous structure of the control may inform which channels are missing in the model.

Boundary constraints $x^L \leq x \leq x^U$ must be considered. While often completely arbitrary, we have found that how these boundaries are specified can be of paramount

importance. For the states, we specify that the voltage is within a survivable and plausible range based on prior knowledge of the system and the variance in the observations. The gating variables are restricted to their dynamic range between 0 and 1. As for the parameters, especially for real data, we shouldn't necessarily know what appropriately tight boundaries are. As a rule of thumb, if it is possible to parameterize the model in a systematic and symmetric way, it may be easier to construct meaningful bounds. Also, it is advisable to keep the parameters within a bounding box which prevents blowup of the dynamics such as divisions by zero. The maximal conductances are positive valued, and the sign of the slope for the steady-state gating functions should dictate if they are activating (positive) or inactivating (negative). Background knowledge of the passive properties of the system, such as the capacitance and reversal potentials, can be informed from isolating step protocols by the electrophysiologist or voltage-clamp data if that is available. The summation of this knowledge forges our prior if considering the problem from the Bayesian perspective, although by nature of this direct implementation, we are assuming a flat prior for the parameter variances within their bounding box.

For the remaining results to come, we have implemented 4D-Var in a framework with CasADi in MATLAB [5]. The "cas" comes from "computer algebra system", in which the implementation of mathematical expressions resembles that of any other symbolic toolbox, and the "AD" for algorithmic (automatic) differentiation. These expressions are easily then used for generating derivatives through breaking the expressions into a number of atomic operations with explicit chain rules, with natural extensions to vector and matrix-valued functions. CasADi data-types are all sparse matrices, and low-level expressions (SX expressions) are stored as directed acyclic graphs where their numerical evaluation is conducted using virtual machines. For nonlinear programming problems, MX expressions are then built from function evaluations to create a hierarchy of functions to prevent blow-up of memory. CasADi

will generate the gradient and Hessian information through AD which are then passed to the solver of choice. We elect to solve the optimization problem with IPOPT (Interior Point OPTimize) [112]. The high-dimensional linear algebra calculations are done using the linear solver MUMPS (MUltifrontal Massively Parallel sparse direct Solver) which is readily distributed with CasADi and interfaced with IPOPT.

3.2 Twin Experiment: Two-Compartment Hodgkin-Huxley Model

3.2.1 Two-Compartment NaKL

There has been plenty of preceding work done on the applicability of 4D-var and its variants on spiking neuronal models led primarily by Abarbanel et al. A standard model they have used to validate their work against for simulated data, with extensions used for real data, is based on the Hodgkin-Huxley model [106]. Here, opening of sodium channels leads to an influx of sodium ions into the neuron driven by a natural electrochemical gradient which generates the upstroke of the action potential. A slower inactivation of this channel prevents further influx of sodium, and the opening of potassium channels drives the cell nearer to its resting membrane potential in the down-stroke of the action potential.

$$\begin{aligned} C\frac{dV}{dt} =& I_{\text{inj}} - g_{\text{Na}} m^3 h \left(V - E_{\text{Na}}\right) \\ & - g_K n^4 \left(V - E_K\right) - g_L \left(V - E_L\right) \\ \frac{da}{dt} =& \frac{a_\infty(V) - a}{\tau_a(V)}, \quad a = \{m, h, n\} \\ a_\infty(V) =& \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{V - v_a}{dv_a}\right) \\ \tau_a(V) =& \tau_{a0} + \tau_{a1}\left(1 - \tanh^2\left(\frac{V - v_a}{dv_a}\right)\right) \end{aligned} \tag{3.27}$$

We have validated similar results of this NaKL model for state and parameter estimation using our framework, and these results will be presented later in the optimal injected current section.

A natural extension, which arises in many situations, is that we have measurements from the soma, but that the active channels and origination of the action potential are located in the axonal initial segment. As an aside, detailed multi-compartmental models are a less explored avenue in data assimilation, but the techniques are completely amenable to spatial incorporation. However, for 4D-Var, at least, for a detailed large multi-compartment model, the number of variables for the problem will increase on the order of number of compartments. These large scale optimizations will be put off, for the time being.

Our two-compartment variant of the NaKL model is:

$$\begin{aligned}
C_s \frac{dV_s}{dt} =& I_{\text{inj}} - g_{L,s}\left(V - E_L\right) - g_{s,ax}\left(V_s - V_{ax}\right) \\
C_{ax} \frac{dV_{ax}}{dt} =& I_{\text{inj}} - g_{\text{Na}} m^3 h \left(V_{ax} - E_{\text{Na}}\right) \\
& - g_K n^4 \left(V_{ax} - E_K\right) \\
& - g_{L,ax}\left(V_{ax} - E_L\right) - g_{ax,s}\left(V_{ax} - V_s\right) \\
\frac{da}{dt} =& \frac{a_\infty(V) - a}{\tau_a(V)}, \quad a = \{m, h, n\} \\
a_\infty(V) =& \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{V - v_a}{dv_a}\right) \\
\tau_a(V) =& \tau_{a0} + \tau_{a1}\left(1 - \tanh^2\left(\frac{V - v_a}{dv_a}\right)\right)
\end{aligned} \tag{3.28}$$

where ax refers to the axonal compartment and s refers to the passive somatic compartment. Figure 3.1 depicts the schematic for this model. Figure 3.2 shows some characteristic plots of this model for $C_s = C_{ax} = 1$ and $g_{s,ax} = g_{ax,s} = 1$, and the rest of the parameters are the same as the default model. Figure 3.2 shows a typical run

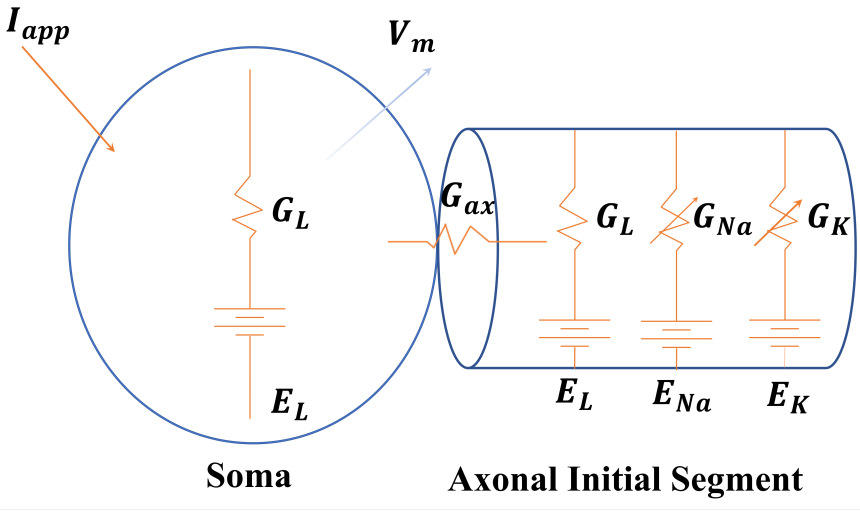


Figure 3.1 Circuit diagram of two-compartment NaKL model. Current is injected into the soma, where a probe will measure the voltage, V_m.

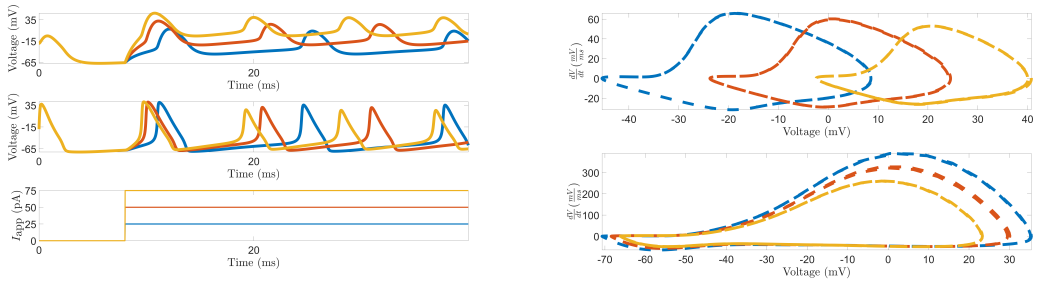


Figure 3.2 Example simulations of two-compartment NaKL model. Left panel: First row depicts the somatic compartment, the second row depicts the axonal compartment, and final row are the stimulating applied currents. Right panel: voltage speed plots for these compartments for steady-firing.

of the model. The top left two rows of Figure 3.2 show the somatic compartment and axonal compartment, respectively, and the bottom row is the applied current used for each of these runs. The somatic compartment average voltage is highly affected by the applied current. These compartments are tightly coupled so the action potentials fire synchronously. The action potentials themselves have highly differing shape, as the somatic compartment acts as a filter for some of the nonlinearities, and the substantial difference in rate of change for the two compartments is visualized in the phase-plane trajectory plots on the right two rows in Figure 3.2.

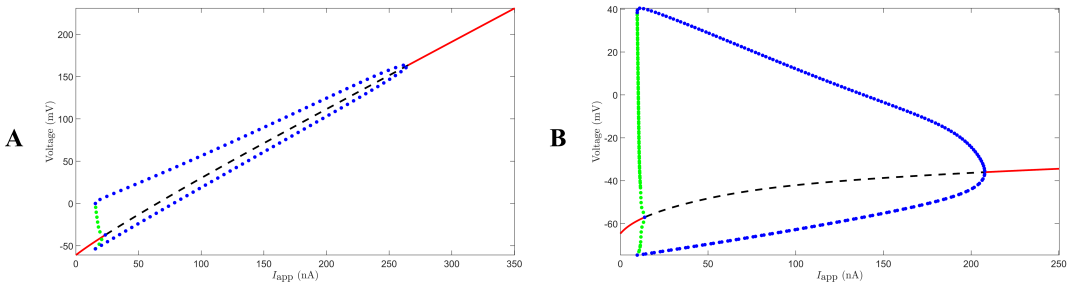


Figure 3.3 Bifurcation diagrams for the two compartment NaKL model **A** and one compartment model **B**.

While this extension is small, there is a subtlety that I feel requires further consideration: the nonlinear dynamics are *completely* isolated to the additional, unobserved compartments, and the somatic compartment, acting as a passive filter, is the lens by which we are trying to observe this information.

We will present some results on strong 4D-Var for twin experiments of this two-compartment model. We will assume we have knowledge of the passive parameters (capacitance and reversal potentials), that $g_{L,ax} = g_{L,s}$, and that $g_{s,ax} = g_{ax,s}$. We will use a series of upward and downward sloping ramps as our applied current. We make this choice because it mimics the derivation of a bifurcation diagram with the applied current as the bifurcation parameter, and we allow the system to traverse a broad range of dynamics to ideally traverse each of the gating variables dynamical ranges. The bifurcation diagram for this system is shown in Figure 3.3 B. The somatic voltage value is reflected on the y-axis and the applied current, acting as the bifurcation parameter, is varied along the x-axis. Spiking emerges through a subcritical Hopf bifurcation and the system undergoes depolarization block via a supercritical Hopf bifurcation after injecting several hundred nA of current.

3.2.2 Twin Experiments on Two Compartment NaKL

We motivate these experiments with a series of questions: Can we uncover the global minimum for this problem given that we are observing from a passive compartment?

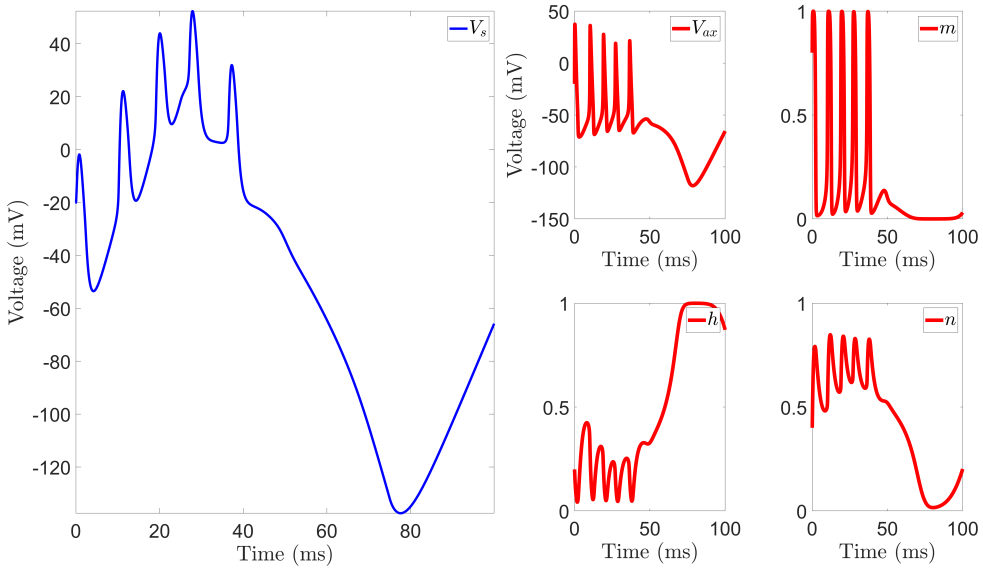


Figure 3.4 Dynamics of our data used for twin experiment of the two compartment NaKL model. The only observed quantity, shown in blue, is the somatic voltage perturbed by white noise. The remaining unobserved variables, shown in red, are contained in the axonal compartment.

What value, from a practical standpoint, does the control bring? If the control case eases our transition to a minima during the optimization iterations, does its contribution vanish in the end? If we can't uncover the true parameters, do our yielded parameter sets have similar dynamical properties, and what does this say about the underlying degeneracy of the problem? What can we say with regard to uncertainty quantification? What sampling rates are necessary to achieve sufficient precision of our results?

We will use data which is originally sampled at 50 kHz riddled with additive white noise which leaves the signal to noise ratio around 43 dB calculated from

$$SNR = 10 \log \left(\frac{\sum V_{\text{somatic}}(t)^2}{\sum \epsilon(t)^2} \right) \tag{3.29}$$

Table 3.1 Bounds of Parameters Used in Estimation of Two Compartment NaKL Model

	LB	UB
C_{soma}	1.00	1.00
C_{axon}	1.00	1.00
E_{Na}	50.00	50.00
E_{K}	-77.00	-77.00
E_{Leak}	-54.40	-54.40
G_{Na}	1.00	1000.00
G_{K}	1.00	1000.00
G_{Leak}	0.00	10.00
G_{axon}	0.00	10.00

	LB	UB
vm	-100.00	0.00
dvm	10.00	100.00
$tm0$	0.01	10.00
$tm1$	0.01	10.00
vh	-100.00	0.00
dvh	-100.00	-10.00
$th0$	0.01	10.00
$th1$	0.01	10.00
vn	-100.00	0.00
dvn	10.00	100.00
$tn0$	0.01	10.00
$tn1$	0.01	10.00

The full dynamics of the system can be observed in Figure 3.4. By sending the system to a hyperpolarized regime, we can fully exercise the inactivation gating variable for sodium, and uncover broader details about the potassium activation variable.

The answer to the first question we posed, "can we recover the global minimum?", simply is *yes*! We can systematically recover parameter sets which are quite close to the true values of the data. To test the fidelity of our approach when encountering noise, we used 10 different observations generated by ten different realizations of our noise. We used the parameter bounds given by Table 3.1. Parameters with equivalent lower and upper bounds are assumed to be fixed and known. The results when sampling the data at 50kHz and using this as our collocation time step are shown in Figure 3.5. The error bars represent two standard deviations drawn from the statistics of ten estimated models from our ten different noise realizations, and the height of the bar is the mean of these ten model parameters. This figure shows when using the control and when not using the control, within two standard deviations, each of the parameters were successfully identified from the deepest minima using our technique. Interestingly, when *not* using the control, the means of the estimates for many of these parameters even more closely align with the true values. Herein lies the question: is the more tightly coupled standard deviation a significant enough justification for using the control? I would argue, at least in this

circumstance, it is useful, but not conclusive. From various experiments, what we can claim, is that the control regularizes the first-order optimality conditions for the log barrier problem in IPOPT. The infeasibility of the dual problem is calculated as:

$$||\nabla C(x) + \nabla g(x)\lambda_g - \lambda_b||_\infty \tag{3.30}$$

where here λ represents the Lagrangian multipliers for the equality and bound constraints, respectively, and sufficiently reducing the dual infeasibility is one of the stopping criterion for IPOPT [112].

The control has, in general, prevented this quantity from blowing up, which stalls, potentially indefinitely, the search for minima. The drawback is that sometimes the solver pushes the difficulty of finding the solution onto merely increasing the control; such local minima are improper given that the control must vanish to return to the default model dynamics. We utilize a measure of the influence of the control described in [106].

$$R(t) = \frac{f_V(V_s(t), V_{ax}(t), m(t), h(t), n(t))^2}{f_V(V_s(t), V_{ax}(t), m(t), h(t), n(t))^2 + (u(t)(y(t) - V_s(t))^2} \tag{3.31}$$

When $R(t) \approx 1$, the control term has little to no relative contribution to the dynamics. Conversely, when $R(t) \approx 0$, the control term dominates the dynamics and everything else is trivial to generating data which matches the observations.

Let's focus in on one realization for the time being. For the deepest minima for the control case in this circumstance, the $R(t)$ plot can be shown in the left panel of Figure 3.6. For most time points, our $R(t)$ is quite close to one, indicating that the control is no longer contributing significantly to the dynamics. Conversely, Figure 3.8 depicts a local minima which, while reducing the cost function to the same order

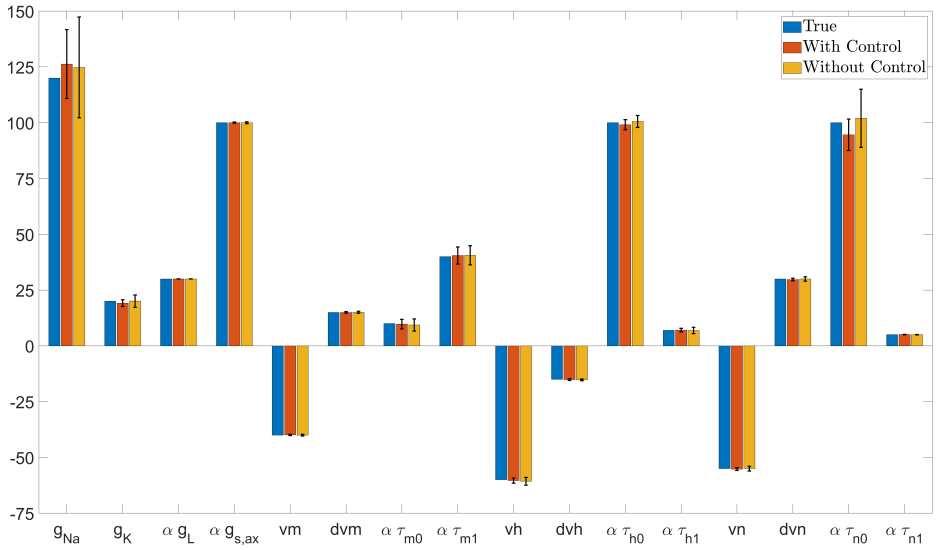


Figure 3.5 Statistics of 50kHz sampling 2 compartment twin experiments using 100ms ramp injected current. For visual purposes, a scaling factor $\alpha = 100$ is applied.

of magnitude as our deeper minima, has depended thoroughly upon control in order to do so. The resulting parameters, now in absence of the control, do not generate behavior matching the observations.

We anticipate that our model should perform well at reconstructing the assimilated dynamics given the relatively tight fit of the parameters. As a check of certainty we can plot the simulations generated by our optimal parameter sets using the same control current, and also predict for a target window beyond our original 100ms observation. For a real experiment, one might assimilate for only some portion of the overall data. Then using the estimated parameters from that assimilation, one can predict beyond to see how well the estimated model matches the data not used for the estimation. In other fields, such as machine learning and statistical inference, these are referred to as training data sets and validation data sets. Figure 3.7 shows how well the sets from our 50 kHz sampling perform. Here, our series of ramps was generated by randomly sampling target applied currents every 25 ms and interpolating between them.

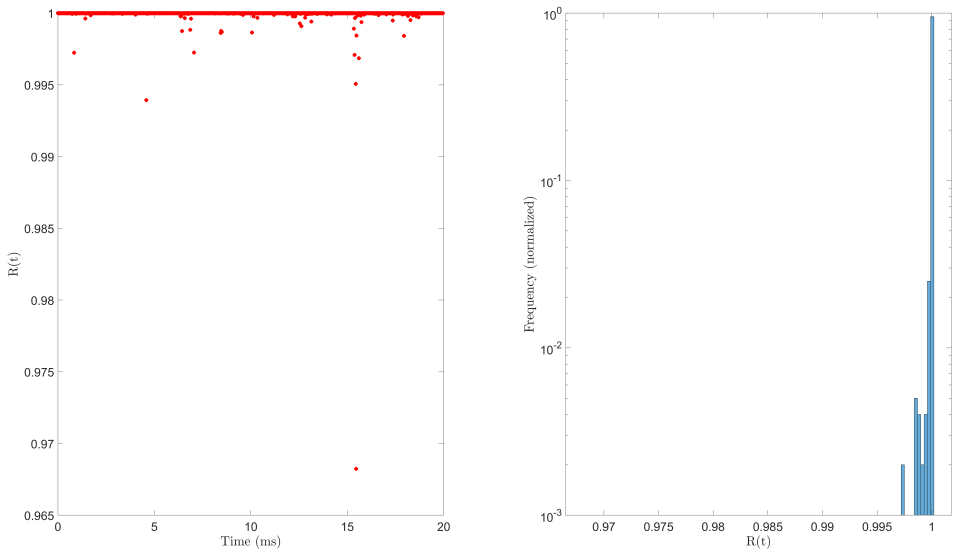


Figure 3.6 $R(t)$ for a preferred local minima. Left panel is $R(t)$ over time. Right panel is a histogram for the range of $R(t)$ values.

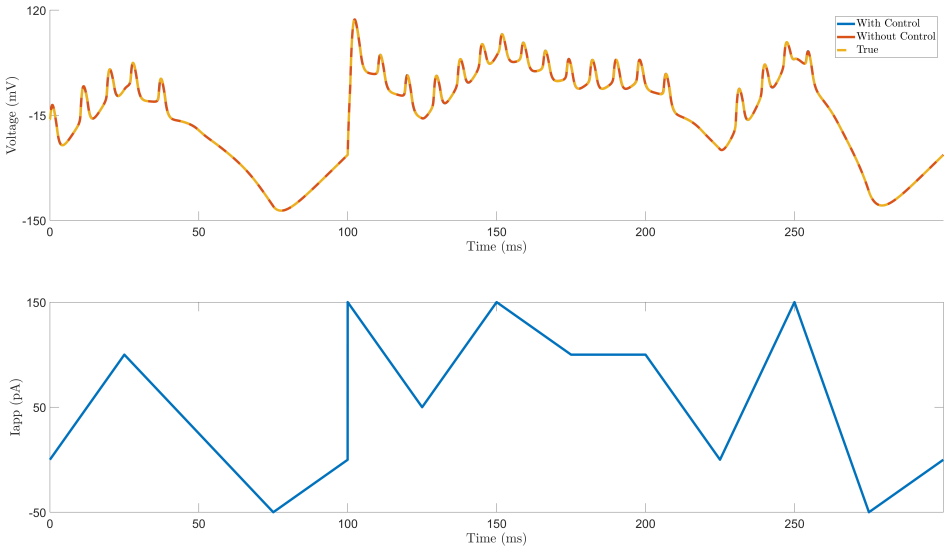


Figure 3.7 Simulations of the two compartment NaKL model over a longer time window than the orignal 100ms assimilation window. The estimated parameter sets from the control and non-controlled 4D-var closely align for the data even beyond the assimilation window.

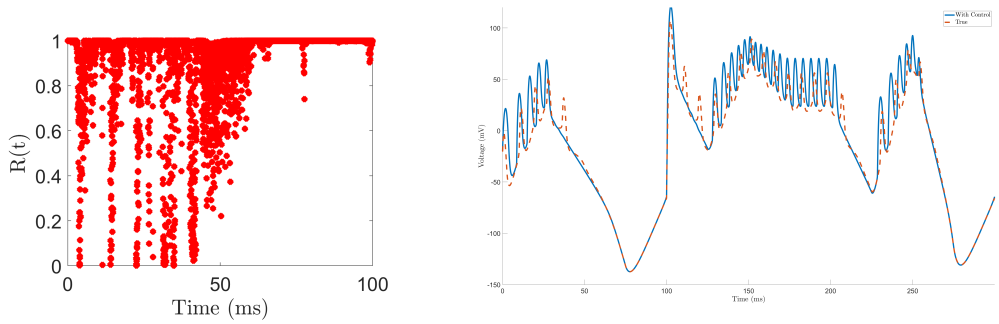


Figure 3.8 Comparison of $R(t)$ for a local minima exhibiting preferential dependence upon the control to match the observed voltage.

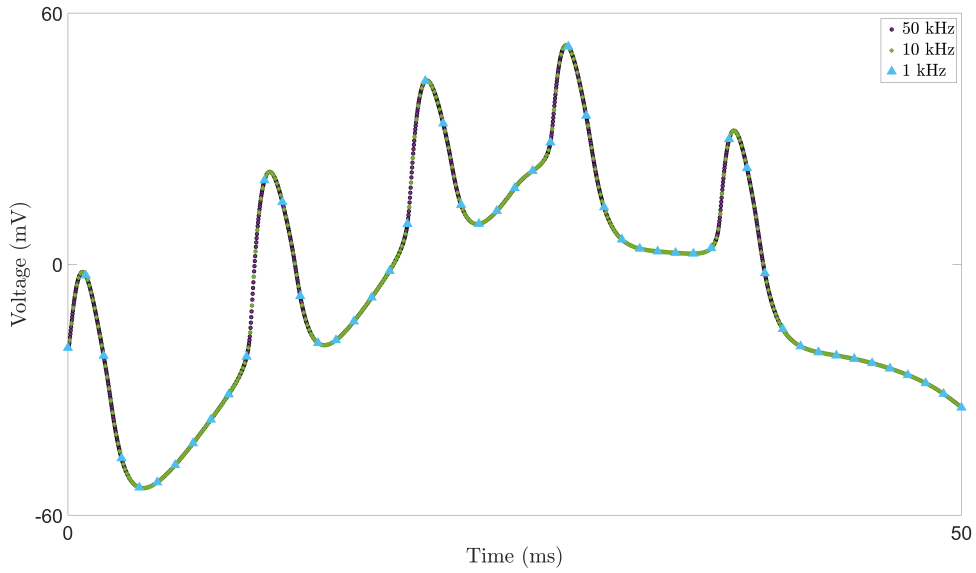


Figure 3.9 Observations of the two compartment NaKL model using different sampling rates.

Two cases we wanted to explore further were the requirement for fine collocation sampling and the requirement for fine observational sampling. An observational sampling restriction might result from some experimental or otherwise limitation, whereas the collocation sampling would merely sacrifice stability and precision of the result in exchange for a decrease in problem size by the corresponding factor. Figure 3.9 illustrates what the sampling looks like for 50kHz compared to 10kHz and 1 kHz.

The top panel of Figure 3.10 illustrates the prediction window by downsampling the observed data to 10 kHz and also using a 10kHz collocation discretization. The

difference in the resulting somatic voltage time series is minimal, with a substantial increase in computational speed. Such a downsampling also allows us to potentially integrate much longer time windows of data, although the precision of our technique will naturally be degraded by using a larger value of h_k in our collocation.

We also wanted to test a limiting case of observational information, i.e., 1 kHz sampling rate. With regard to neuronal data, action potentials may only be characterized by a handful of points with this sampling rate. Still, as shown in the bottom panel of Figure 3.10, at least for this model while using a fine collocation time grid of 50kHz, the data is sufficient for generating dynamics akin to the true voltage during even the prediction window of our data. Natural extensions to real data, beyond merely these substantially coarse grids, are constructing adaptive grids for data to optimally incorporate the most informational time points and downsampling elsewhere.

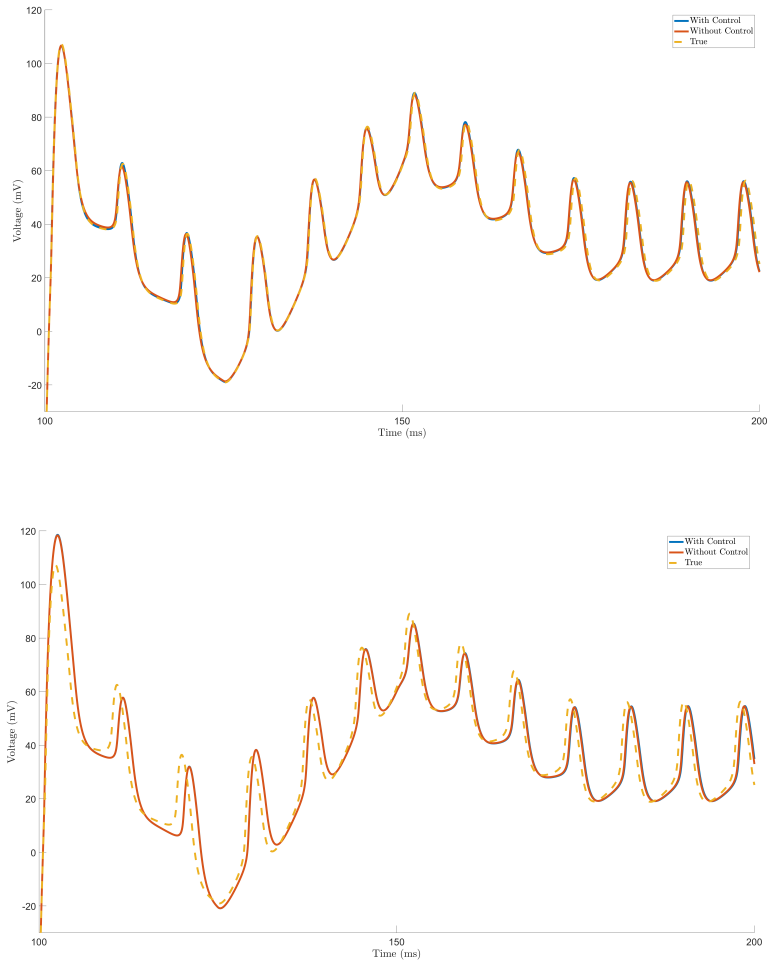


Figure 3.10 Comparison of prediction windows for data generated by using coarser sampling. The top panel uses a collocation time difference and observational time grid of .1 ms. The bottom grid uses a collocation time difference of .02 ms and an observational time difference of 1 ms.

3.3 Application to Rhabdomys Circadian Neurons

Organ systems including the heart, liver, and pancreas contain oscillators whose day/night rhythms are synchronized by central pacemaker neurons. These neurons dwell in the suprachiasmatic nucleus (SCN) in the hypothalamus. The SCN is subdivided into a ventral (core) and a dorsal (shell) region, with approximately 20,000 neurons in total. The core cells integrate external input projected from the retina as well as the geniculohypothalamic tract and projections from the raphe nuclei [81]. The neurons are tightly compacted with small diameter and the suprachiasmatic

nucleus orients itself atop the optic chasm. The dorsal shell exhibits robust circadian oscillations of gene expression [51, 85, 121] , and secretes arginine vasopressin (AVP) or prokineticin 2 (PK2), along with GABA. Interestingly, 60-70% of SCN neurons can fire rhythmically when dissociated from the rest of the network [8, 117].

SCN neurons have the ability to generate autonomous circadian (∼24 hour) rhythms in neuronal firing frequency. Other tissues synchronize to the SCN via direct and indirect pathways. Therefore, these cells are referred to as pacemakers, as they are the metronome by which a collection of oscillators act in concert.

Intracellular Gene Regulatory Feedback Loop The expression of "clock genes" within SCN neurons operates on roughly a 24 hour period. These rhythms are the result of a transcriptional and translational feedback loop. The transcription factors CLOCK and BMAL bind to the E-Box to activate the transcription of the Period (Per) and Cryptopchrome (Cry) genes. The mRNA transcripts are then translated which increases levels of their associated proteins. These proteins dimerize and ultimately suppress the E-box activation. Of course, that leads to lower levels of PER and CRY proteins, and the lower dimer levels allow the transcription rate to increase again [30].

Therefore, within the nucleus of these cells there exists an intrinsic gene expression cycle with a circadian rhythm, and a circadian rhythm in terms of electrical activity of the neuron at the membrane level.

Electrophysiology A mammalian SCN neuron has a set of characteristic currents to drive changes in its membrane potential, as well as induce changes in ionic concentrations. The changes in voltage act on a timescale of milliseconds. Any downstream network effects, or changes in gene expression due to calcium uptake or other neurotransmitters can happen across hours. The summation of currents with specific windows of activation and inactivation culminates in the rhythmic, or silent,

membrane potential. There is persistent sodium current which acts as an excitatory drive, leading to depolarization. A hyperpolarization-actived cyclic nucleotide-gated (HCN) ion is another excitatory drive with slow dynamics responding to hyperpolarizations of the membrane. Additionally, in response to hyperpolarization, the hyperpolarization-activated conductance (I_H) creates a depolarizing voltage sag. A T-type calcium current has low-voltage activation and inactivation, and an L-type calcium current is a mediator for subthreshold activity. The sodium-potassium pump maintains the resting membrane potential at the cost of cellular ATP. The amplitude of the fast delayed rectifying (FDR) potassium current plays an important role in circadian modulation. An A-type potassium (I_A) current is involved in the timing of action potentials and subthreshold responses. Calcium-activated potassium channels (BK channels) play a role in the nightly hyperpolarization of cells and repolarization of the membrane after an action potential [30].

SCN Modeling History The first conductance-based mammalian model for SCN neurons was developed by Sim and Forger [29]. The model included Hodgkin-Huxley like sodium and potassium currents. Potassium channel properties were fit to data from rat SCN slices [16]. The sodium channel is fit to voltage-clamp data, and an L-type calcium channel is included which was also fit to voltage-clamp data, assuming both activation and inactivation of the channel [64]. The single-compartment model assumes that total cell capacitance and conductance are used for parameters, and the capacitance, is recorded as 5.7 pF [64]. Their model is capable of explaining distinct behaviors of SCN neurons, including spontaneous firing, responses to random inhibitory postsynaptic currents (IPSCs), and the contributions of individual ionic currents during an action potential [29].

Later on, the model was adapted to try to further understand day/night differences in firing activating of SCN neurons [11]. The typical bifurcation parameter

for conductance-based models is the applied current. However, we are frequently expecting to measure spontaneous activity of these cells. For SCN neurons, the bifurcation parameters of interest are their maximal conductances. Some ion channel genes have expression which operates with a circadian rhythm. Therefore, we expect some of the maximal conductances to change over the course of the day-night cycle. Circadian variations in potassium conductances and calcium conductances, as well as intracellular calcium concentration through the reversal potential, were able to account for changes in firing rate and resting membrane potential. They were also able to account for the emergence of depolarized low amplitude oscillations recorded in a particular subset of SCN neurons expressing the clock gene *per1* in the vicinity of Hopf bifurcation points of the model [37]

The model was developed from [11] :

$$\begin{aligned} C\frac{dV}{dt} &= I_{app} - I_{\mathrm{Na}} - I_{\mathrm{Ca}} - I_{\mathrm{K}} - I_{\mathrm{leak}} \\ &= I_{app} - g_{\mathrm{Na}} m_{\mathrm{Na}}^3 h_{\mathrm{Na}} (V - E_{\mathrm{Na}}) - g_{\mathrm{K}} n^4 (V - E_{\mathrm{K}}) \\ &\quad - g_{\mathrm{CaL}} m_{\mathrm{Ca}} h_{\mathrm{Ca}} (V - E_{\mathrm{Ca}}) - g_{\mathrm{leak}} (V - E_{\mathrm{Na}}) \end{aligned} \tag{3.32}$$

The gating variable dynamics are all expressible by

$$\frac{da}{dt} = \frac{a_\infty(V) - a}{\tau_a(V)} \tag{3.33}$$

for $a = m_i, h_i, n$.

The steady-state activation functions were described by:

$$
\begin{aligned}
m_{\mathrm{Na},\infty}(V) &= \frac{1}{1+\exp\left(-(V+35.2)/8.1\right)} \\
h_{\mathrm{Na},\infty}(V) &= \frac{1}{1+\exp\left((V+62)/4\right)} \\
n_{\infty}(V) &= \frac{1}{\left(1+\exp\left(V-14\right)/-17\right)^{.25}} \\
m_{\mathrm{Ca}(V),\infty} &= \frac{1}{1+\exp\left(-(V+25)/7.5\right)} \\
h_{\mathrm{Ca}(V),\infty} &= \frac{1}{1+\exp\left((V+260)/65\right)}
\end{aligned}
$$

with rates

$$
\begin{aligned}
\tau_{m,\mathrm{Na}} &= \exp\left(-(V+286)/160\right) \\
\tau_{h,\mathrm{Na}} &= .51+\exp\left(-(V+26.6)/7.1\right) \\
\tau_{n} &= \exp -(V-67)/68 \\
\tau_{m,\mathrm{Ca}} &= 3.1 \\
\tau_{h,\mathrm{Ca}} &= \exp -(V-444)/220
\end{aligned}
$$

SCN Recordings Most rodents are nocturnal. Interestingly, the Four-striped grass mice, *Rhabdomys pumilio*, of Southern Africa are diurnal. Also, they have enhanced activity during the mornings periods in the evening, with a lull during the midday [100]. These species are then of great intrigue to study, as humans are diurnal, and

we desire to understand a bit more about the similarities and differences between the developed models based on nocturnal species.

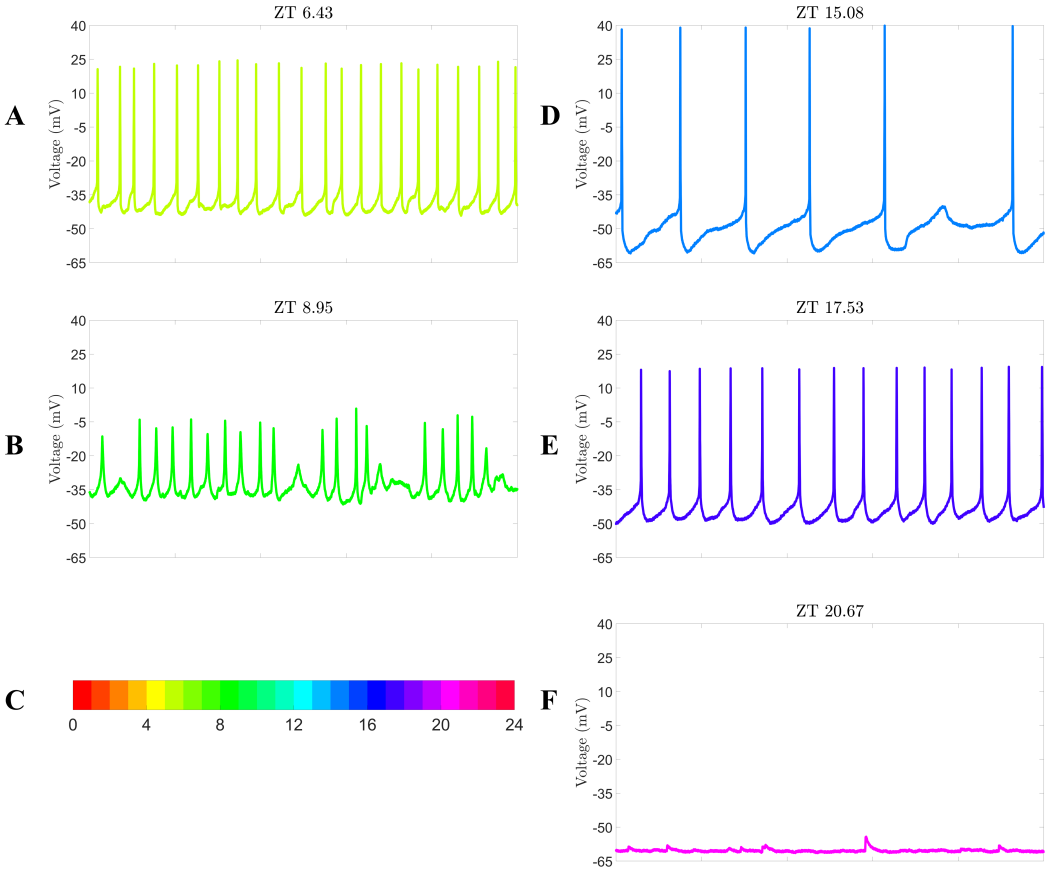


Figure 3.11 Rhabdomys electrical variation over day/night cycle. Colormap corresponds to ZT of data collection, illustrating 5 seconds of data. Data was recorded by Mino Belle and Beatriz Baño-Otálora in the laboratory of Robert Lucas at the University of Manchester.

Our hope is to be able to explain some of these various behaviors based on the activity of individual currents within our model. We anticipate there will be degree of heterogeneity amongst these cells, so we endeavor to parameterize them individually, and restrict conclusions of causality of behaviors to hypothesis. Some typical day/night variations are shown in Figure 3.11. Day is characterized by Zeitgeber Time (ZT) between 0 and 12, whereas night starts at 12 and runs to ZT 24. Each panel contains 5 seconds of spontaneous current-clamp data. During the

day, the resting membrane potential (RMP) is more depolarzized than during the evening, as depicted by Figure 3.11 A, B. During periods of the day, some cells exhibit deploarized low amplitude membrane oscllations, as shown in Figure 3.11 B. Slower, large amplitude firing, ∼1-2 Hz, is characteristic of early evening, 3.11 D, with a slight increase in RMP and firing rate, with a lower amplitude, later in the evening Figure 3.11 E. Bouts of time in evening are characterized by quiescence, with a hyperpolarized RMP, Figure 3.11 F. The colormap for this figure, organized by ZT, is shown in Figure 3.11 C.

Modeling Rhabdomys SCN A fault with the original version the SCN model is that the structure is asymmetric with huge ranges of parameter values, which creates complications when constructing our optimization problem. We aim to fit to current-clamp data of the Rhabdomys using the same types of currents, but expressing their kinetics uniformly as in equation 3.28.

Additionally, we separate the leak into sodium and potassium components to investigate the role each may play in altering the resting membrane potential of cells in day versus night, as was done in [37]. Lastly, we will approximate the sodium activation as instantaneous, as was previously done to reduce this model [29]. Conversely, we will allow the inactivation of sodium to have a wide range of permissible time constant values, as persistent sodium is known to play a role in maintaining the pace of firing [53], and thus our sodium channel functionally plays the role of transient generation of the upstroke of the action-potential, but also in possibly governing certain sub-threshold properties with the remaining persistence compensated by the sodium leak. The full system is shown in equation 3.34.

$$
\begin{aligned}
C\frac{dV}{dt} =& I_{\text{inj}} - I_{\text{Na}} - I_{\text{K}} - I_{\text{Ca}} - I_{\text{Na,Leak}} - I_{\text{K,Leak}} \\
=& I_{\text{inj}} - g_{\text{Na}} m_{Na}^3 h_{Na} \left(V - E_{\text{Na}}\right) \\
& - g_K n^4 \left(V - E_K\right) - g_{Ca} m_{Ca} h_{Ca} \left(V - E_{\text{Ca}}\right) \\
& - g_{\text{Na,Leak}} \left(V - E_{Na}\right) - g_{\text{K,Leak}} \left(V - E_K\right) \\
\frac{da}{dt} =& \frac{a_\infty(V) - a}{\tau_a(V)}, \quad a = \{m_i, h_i, n\} \\
a_\infty(V) =& \frac{1}{2} + \frac{1}{2} \tanh\left(\frac{V - v_a}{dv_a}\right) \\
\tau_a(V) =& \tau_{a0} + \tau_{a1}\left(1 - \tanh^2\left(\frac{V - v_a}{dv_a}\right)\right)
\end{aligned}
\tag{3.34}
$$

Table 3.2 Bounds of Parameters Used in Estimation of SCN Model

	LB	UB
C	0.8 C_{rec}	1.2 C_{rec}
E_{Na}	40.00	50.00
E_{K}	-100.00	-80.00
E_{Ca}	54.00	130.00
G_{Na}	0.10	500.00
G_{K}	0.10	300.00
G_{Ca}	0.01	300.00
$G_{\text{Leak,Na}}$	0.00	10.00
$G_{\text{Leak,K}}$	0.00	10.00
vm_{Na}	-50.00	0.00
dvm_{Na}	5.00	50.00
vh_{Na}	-70.00	0.00
dvh_{Na}	-50.00	-5.00
$th0_{\text{Na}}$	0.01	40.00
$th1_{\text{Na}}$	0.01	400.00
vht_{Na}	-70.00	0.00
$dvht_{\text{Na}}$	5.00	50.00

	LB	UB
vn	-70.00	0.00
dvn	5.00	50.00
$tn0$	0.01	10.00
$tn1$	0.01	40.00
vnt	-70.00	0.00
$dvnt$	5.00	50.00
vm_{Ca}	-40.00	0.00
dvm_{Ca}	5.00	50.00
$tm0_{\text{Ca}}$	0.01	40.00
$tm1_{\text{Ca}}$	0.01	40.00
vmt_{Ca}	-70.00	0.00
$dvmt_{\text{Ca}}$	5.00	50.00
vh_{Ca}	-70.00	0.00
dvh_{Ca}	-50.00	-5.00
$th0_{\text{Ca}}$	0.10	1000.00
$th1_{\text{Ca}}$	0.10	1000.00
vht_{Ca}	-70.00	0.00
$dvht_{\text{Ca}}$	5.00	50.00

These neurons fire fairly slowly, between 1-8 Hz while firing spontaneously, with irregular firing patterns. The cells may exhibit slow firing adaptation to depolarizing pulses, delays to return to firing from hyper-polarizing pulses, and slow return to normal firing from depolarizing pulses. The firing rate difference alone is a factor

of 10 between the NaKL model for $I_{app} > 25$, as the NaKL model does not fire spontaneously as shown in Figure 3.3.

Our data is recorded every 25 kHz, which, assuming a firing rate of 3Hz, would require $25,000$ time points to capture a train of three action potentials. To our knowledge, the largest data sets used for variational data assimilation in neuroscience have been around $100,000$ time points in total [86]. For the Morris-Lecar model, observing three action potentials may be sufficient to characterize the system with 4D-Var. For real cells, capturing the spontaneous firing is only a small portion of a much larger problem; we want to also capture the response to stimulating currents so as to have some predictive power. We therefore create a budget for our problem in terms of incorporating novel information for assimilation while creating an optimization problem which can fit into memory.

For each set of data used, we utilize a downsampling strategy. We set a threshold of -20 mV for each action potential, and within a region of 30 ms either side of when this threshold is hit, the full 25 kHz sampling is preserved. Outside of this threshold, the data used is down-sampled by some factor. For the results presented here, we use a downsampling factor of 5 so that during the action potential the resolution is 25kHz and outside the time window of the action potential it is 5kHz. The reasoning behind this that we can maintain as many points during the action potential as possible which occurs on a much faster timescale than the afterhyperpolarization or subthreshold repolarization, and we hope to better fit the spike-shape. The current-clamp data we are using is exclusively driven by steps. At the time of discontinuous change of the step, we similarly preserve the data so as to maximize the transitional information. An example of this is shown in Figure 3.12.

To reiterate, we can only use a small fraction of the data due to our computational budget. We are afforded several seconds worth of data from an array of depolarizing and hyperpolarizing step protocols, varying from -30pA to 30pA

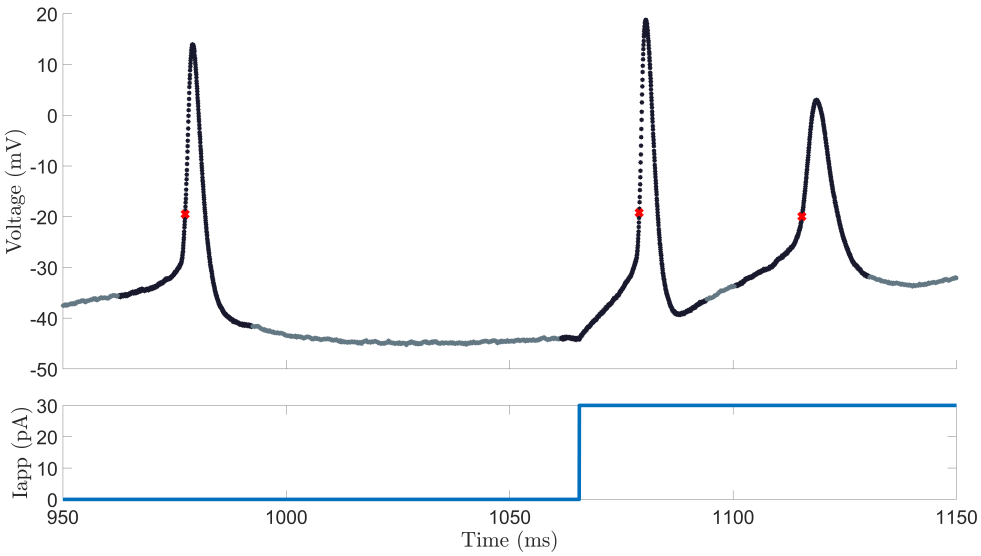


Figure 3.12 Sampling strategy for SCN data. Black corresponds to full data used, whereas gray corresponds to downsampling used. Red 'x' shows the midpoint for the high-resolution sampling.

in increments of 5pA. Not only do we want to be able to accurately capture the spontaneous firing and spike shape, but we want to capture delays, adaptations, silencing, etc. The data we used for our assimilation, in totality, is shown in Figure 3.13. Figure 3.13 A contains the response to a -30pA pulse, which may contain information about non-active channels, such as the leaks, and the capacitance. How the cell recovers from hyperpolarization is also of intrigue, as some cells immediately return to firing, a behavior we denote as "rebound", while some cell exhibits delays to firing. Figure 3.13 B-C provide similar information about the passive-properties and capacity / speed to return to spontaneous firing. Figure 3.13 D contains a longer strand of spontaneous firing activity, which we hope to aid in capturing that behavior to a greater fidelity, as spontaneous firing is a hallmark of these cells. Figure 3.13 E-F contain information regarding the response to a depolarizing pulses of varying amplitude. Figure 3.13 F specifically should reveal if the cell's firing envelope changes in amplitude, a behavior we denote as "adapting-firing", maintains similar amplitude, or ceases-firing from excess injected current.

Technically, we implement this through a natural adaptation of our 4D-Var setup, where we are solving several individual data assimilation problems simultaneously which are connected through mutually shared parameters. In the problem construction shown by Figure 3.13, 4.5 seconds of data in total are used for the assimilation, amounting to around $36,000$ time points after incorporating our downsampling strategy.

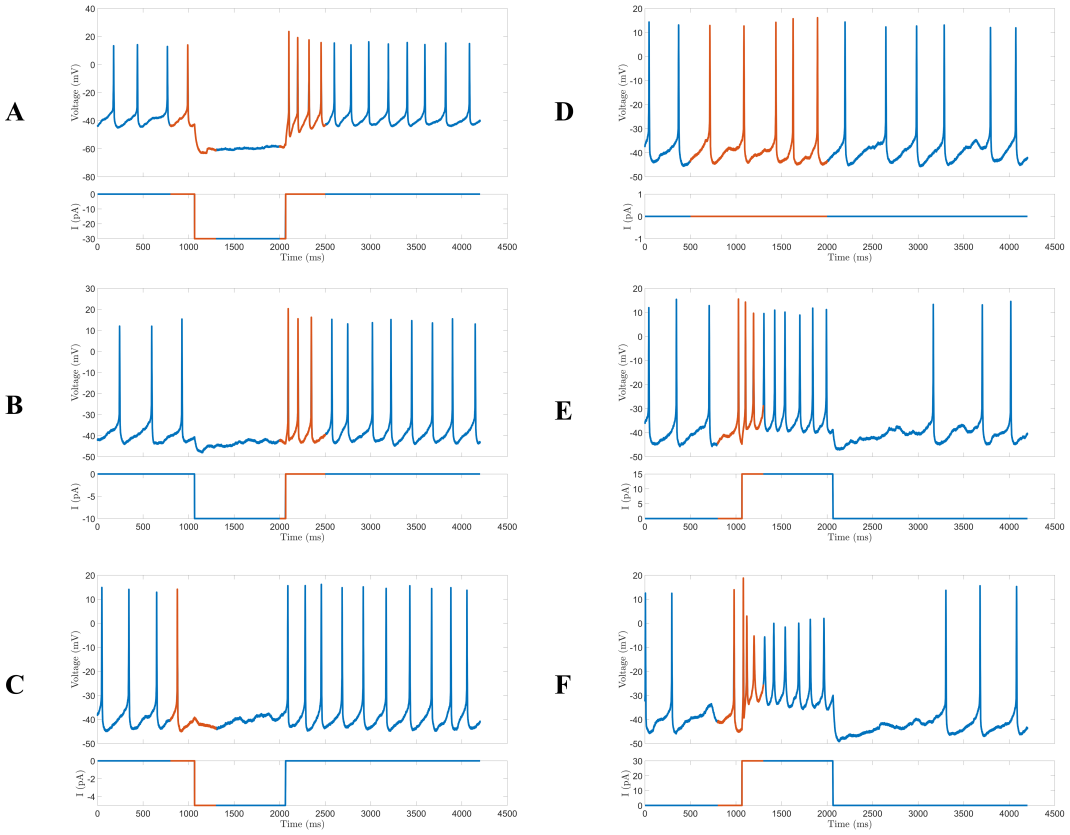


Figure 3.13 SCN data used for assimilation. Blue corresponds to full current-clamp data, whereas orange corresponds to training data used for assimilation.

The fit to the data from Figure 3.13 can be seen in Figure 3.14. Here, the green dashed line corresponds to the time in which the data was assimilated, and the red dashed line signals the end of the assimilation window. Given that the cell is assumed to be spontaneously firing at any given time $t = 0$, we set the initial conditions for the model simulation merely as the initial condition estimated from 4D-var for a

segment of data which also was firing spontaneously at the start. Deviations from the spontaneous activity are expected regardless give the irregular emergence of action potentials in the data. The purple is the estimated model for this cell.

One caveat to the estimated model parameters are that it is chosen through a developed heuristic. We recognized that none of the estimated models were successful in fully eliminating the dependence on the control term; this may mean that our model is inadequate, or this dependence marks a deeper coupling to the unobserved driving forces in the neuron e.g., external inputs, channel noise, etc. This also leads to an even greater multi-modality in the cost function, as for dozens of initial conditions, there may yet yield even dozens of final model candidates. Therefore, the "lowest extrema" in exclusion may be misleading, because this model may not produce even spontaneous activity. When we curate the optimal model, we look along a sort of Pareto front consisting of the cost function evaluation and a mismatch in firing rate between the model output and the data. To stress, we cannot claim with a high degree of certainty that this is in fact the global minimum for this set of data, but we will claim that amongst at least 50 initial guesses, it is the most optimal one.

In general, the estimated model is capturing the behavior of the cell quite well, even beyond the assimilation window. One misalignment occurs in the return to firing after depolarization, in that our estimated model re-initiates firing quite a bit earlier, as seen in Figure 3.14 F. To incorporate this information into the estimation, we might require, in this case, an additional 1500 ms of data to account for this. However, the real cell is quite close to threshold for this duration, so this delay may be attributed to a degree of noise compared to intrinsic dynamics. The bifurcation diagram and FI curve for this model are shown in Figure 3.15. Firing emerges through a subcritical Hopf bifurcation and ceases through a supercritical Hopf bifurcation, similarly to the NaKL model Figure 3.3, but for a smaller range of applied currents and lower voltage

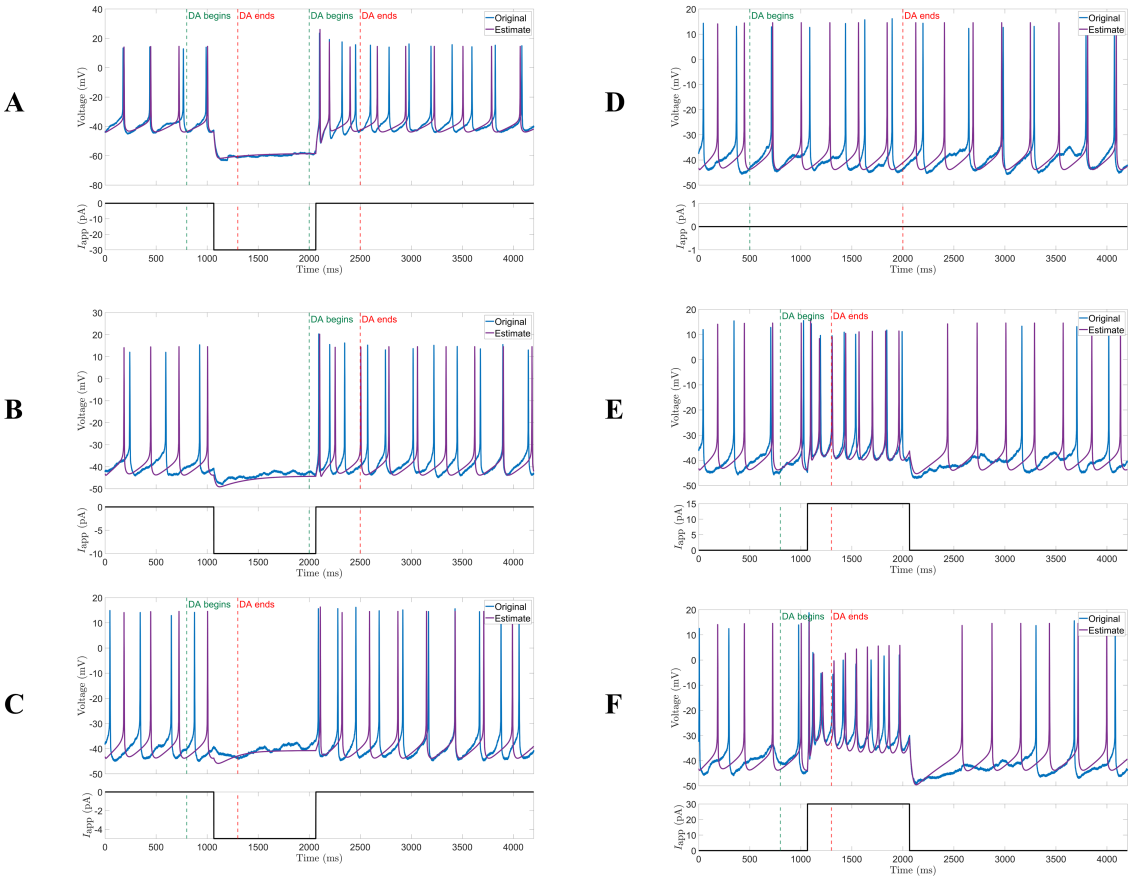


Figure 3.14 SCN data assimilation for training data for cell exhibiting rebound from hyperpolarization.

amplitude. The estimated model's firing rate, even for unobserved step protocols, matches the measurements within a couple Hz.

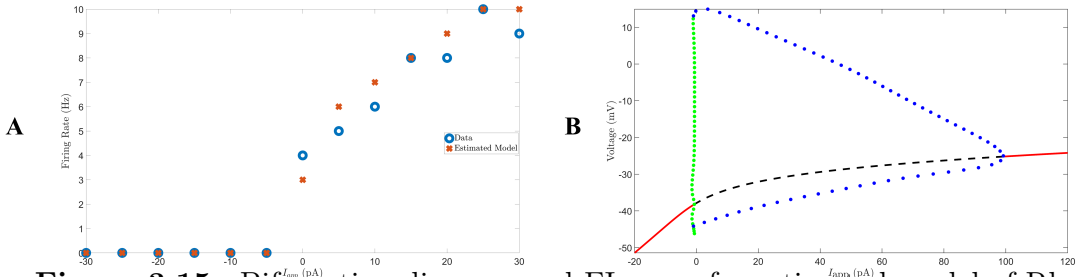


Figure 3.15 Bifurcation diagram and FI curve for estimated model of Rhabdomys cell exhibiting rebound.

3.3.1 Behavioral Responses to Hyperpolarizing Pulses

We classify this cell as exhibiting "rebound" or immediate return to spiking after a hyperpolarizing step. Figure 3.16 illustrates, for this cell model, what the generating mechanisms for this rebound are. The top panel shows the currents varying over time for the -30 pA data trace immediately after release from the hyperpolarizing pulse. The calcium channel is fully de-inactivated, as shown in the top right panel of Figure 3.16 B, and has a long time scale for its dynamics. Thus, the calcium channel acts as the pacing inward current, aided in small part by the sodium leak channel and repelled by the larger potassium leak channel, and the cell swiftly depolarizes enough to fire. Calcium slowly inactivates, and the cell returns to its spontaneous firing rate. The sodium channel is also mostly deinactivated, but the timescale for sodium is a bit quicker, leading to it returning to its steady activity within a couple APs. One hallmark of this cell's estimated model, as with many to follow, are the repulsive forces persistently created by the potassium leak channel, an outward current, and the calcium channel, an inward current. We have found that many of the cellular properties may be retained by appropriately scaling the conductances for these two channels in a symmetric fashion.

The shape of the action potential is also of paramount importance to capture, as this contains a vast amount of information relating to maximal conductances and kinetics during the AP. Figure 3.17 shows the $\frac{dV}{dt}$ vs V plot for the data versus the model for spontaneous firing. The data is noisy and achieves different amplitudes for different APs in the time series leading to its cloudy appearance. The model is quite close to approximating the mean phase plot of the data for this cell which gives us hope that our included channels are sufficient to at least characterize spontaneous firing.

The alternative response exhibited by the SCN cells when exposed to hyperpolarizing pulses are delays to return to spiking. The default model, equation 3.34, when

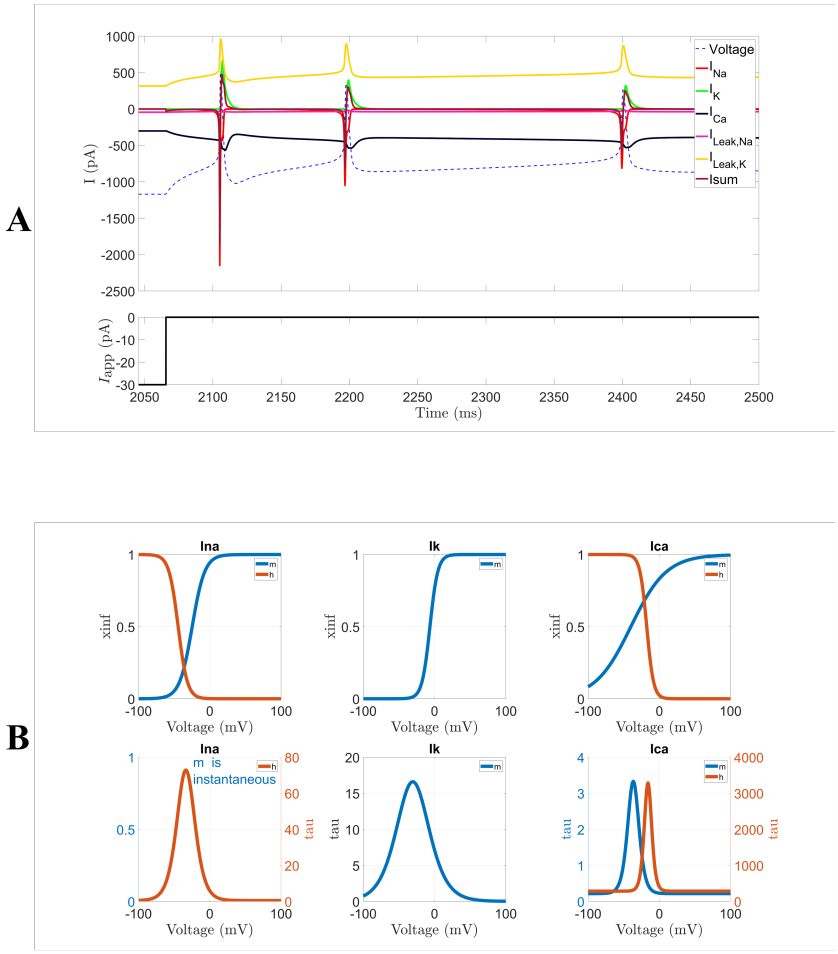


Figure 3.16 SCN rebound mechanism.

presented with data which exhibited these delays, significantly struggled, indicating that there may be insufficient currents present to explain this behavior. An additional feature, often associated with these delays, are sags during the initiation of the hyperpolarizing pulse, which, again, is a behavior which our model struggled to deal with.

To address these issues, for cells which exhibited these behaviors, we added two additional currents: a non-inactivating H-current which can generate a sag [15], and a transient outward potassium channel with slower inactivation, but we approximate the activation as instantaneous. The transient potassium channel may be representative of an A-current which has been modeled with instantaneous activation

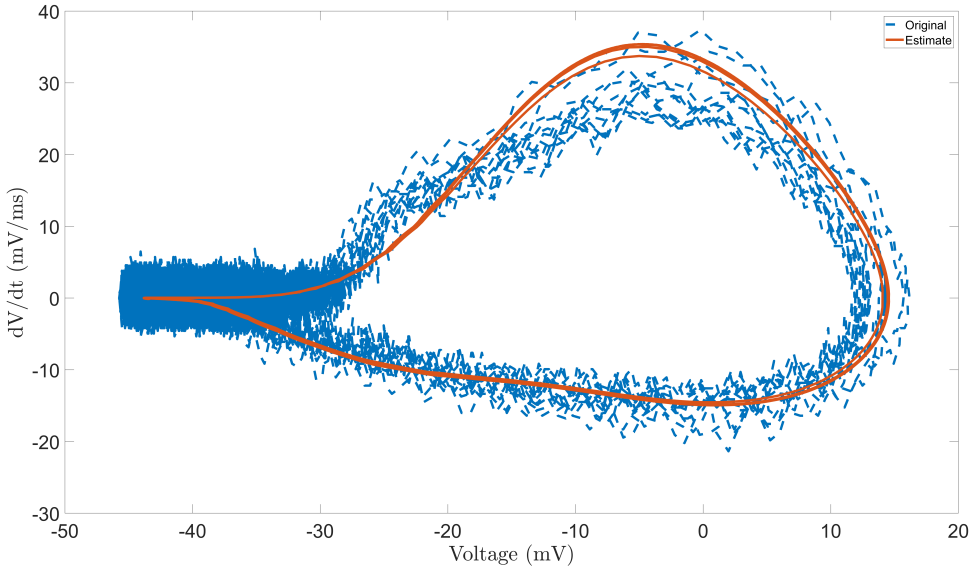


Figure 3.17 SCN rebound speed plot.

before [47]. It has been shown that blocking I_A with the drug 4-Aminopyridine eliminates the delay in mouse orexin neurons [18]. We based boundaries for the parameters of the H-current and A-current from recordings of SCN neurons [16, 34].

When including these currents, we were successful in estimating these behaviors, as shown in Figure 3.18 A. Because of the long timescales at which these behaviors operate on, we exposed the assimilation to around 3 seconds of current-clamp data with a hyperpolarizing pulse. To ensure that the model is robust, we also included responses to depolarizing pulses. Figure 3.18 B shows that this model, for certain depolarizing pulses, actually bursts as opposed to tonic firing, whereas the data is tonic firing. However, the firing rate of the burst is similar to that of the tonic firing. This assimilation problem was found to be particularly stiff; to help in regularizing the problem, we used the slack form of the constraints and penalized the slacks as in equation 3.7. We felt that this model comfortably accounted for the delay, but to illustrate the mechanism behind it, we explored the dynamics in Figure 3.19. The potassium leak and calcium channels are both open during the return step to $I_{app} = 0$, and as is the transient outward potassium channel I_{to}. Slowly, this channel inactivates,

as its time scale for inactivation is shown in the bottom right panel of Figure 3.19 B. If we alter the conductance for this current, the delay is similarly affected, as shown in Figure 3.19 D. Here, if we take the estimated conductance G_{to} and reduce it by five percent, the delay is reduced, and if we increase it the delay is increased.

3.3.2 Behavioral Responses to Depolarizing Pulses

The SCN neurons also show distinctive responses to depolarizing pulses. Figure 3.20 F is an example of cell exhibiting a non-adapting firing behavior. When exposed to a depolarizing pulse, the firing rate and amplitude of spiking is fairly consistent across the pulse duration. An explanation for this behavior can be seen in Figure 3.21. If one compares the current contributions before and after the pulse, as shown in Figure 3.21 C, the prominent features are mostly retained, as the summation of the currents during the onset of the AP closely hug the sodium channel. The inactivation time constants for sodium are fairly fast during the AP, as shown in the bottom left of Figure 3.21 B, and they quickly respond to the change in applied current.

Conversely, some cells' firing rate and firing amplitude alter during the course of the pulse, as shown in Figure 3.22 F. Here, the cell's firing rate increases and the waveform amplitude decreases. An explanation for this behavior is given in Figure 3.23. In panel D of Figure 3.23, sodium becomes inactivated with smaller window of activation and slower response time. Comparing the current contributions in Figure 3.23 A and C, the driving for firing during the 30pA pulse becomes mainly calcium, although the upstroke during spontaneous firing is driven by sodium as the sodium channel is less inactivated.

Other cells' firing becomes silenced when injected with depolarizing protocols of sufficient strength. An example of one such cell is given in Figure 3.24 F. We investigate the mechanism for this silencing in Figure 3.25. During spontaneous firing, the calcium contributes a persistent pacemaking inward current to depolarize

the membrane, and here, the sodium is the main initiator of spiking, as shown in Figure 3.25 A. However, when exposed to a depolarizing pulse of 30 pA, the cell's sodium channel almost immediately inactivates, and the calcium channel also slowly inactivates. No inward flux of positively charged ions are driving the cell to threshold, and opposing the outward potassium leak channel, so the cell maintains at rest.

To be clear, we cannot claim with certainty that these are the biological mechanisms which are producing these observed behaviors. However, we have explored the dynamics that were evoked from our data assimilation procedure, using our particular down-sampling protocol, to understand better the *possible* mechanisms for the behaviors' emergence.

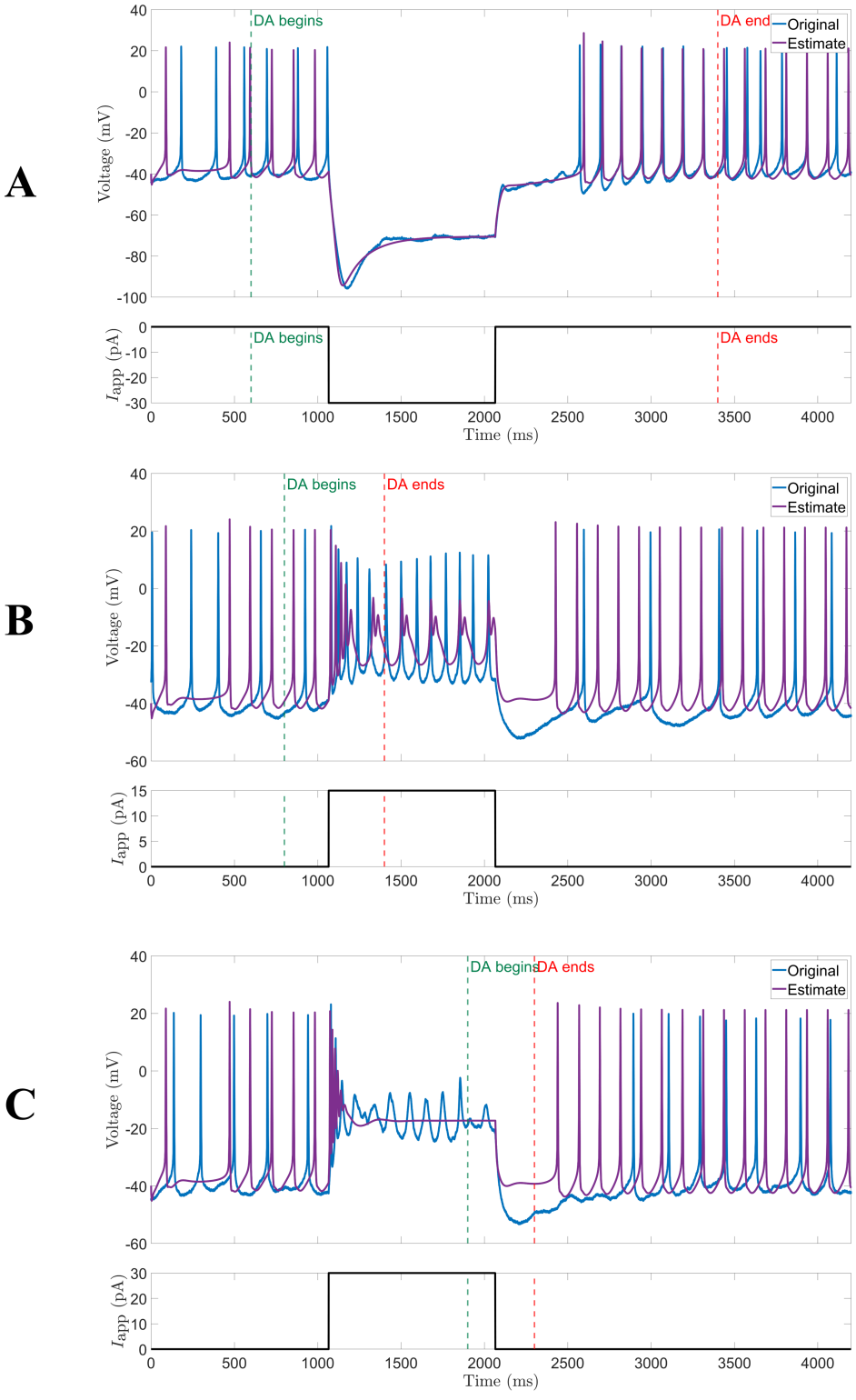


Figure 3.18 SCN data assimilation for training data for cell exhibiting delay from hyperpolarization.

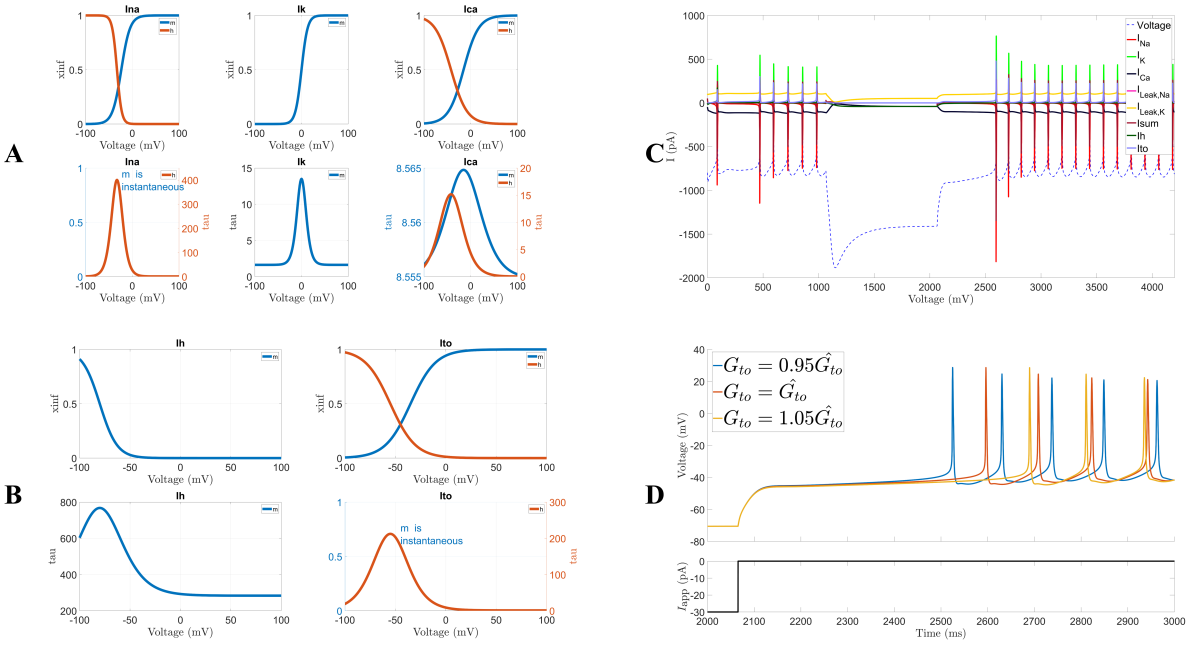


Figure 3.19 SCN delay mechanism.

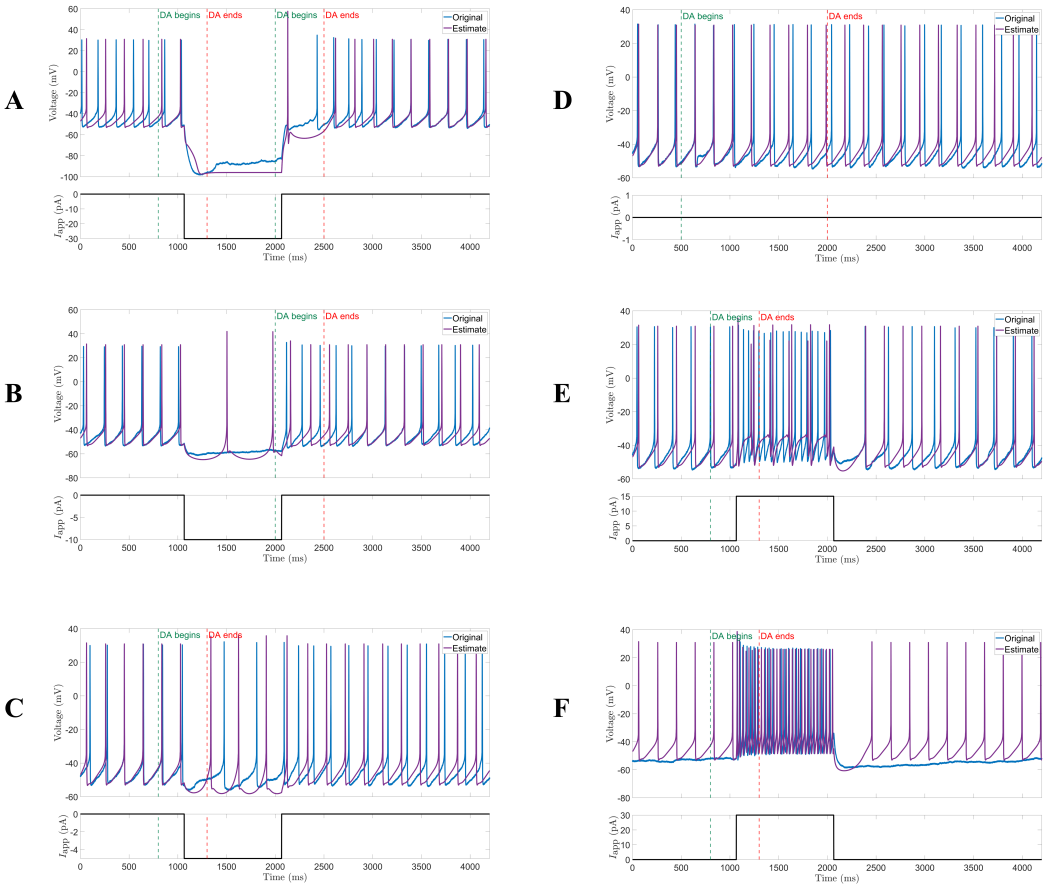


Figure 3.20 SCN data assimilation for training data for cell which is non-adapting when exposed to depolarizing pulses.

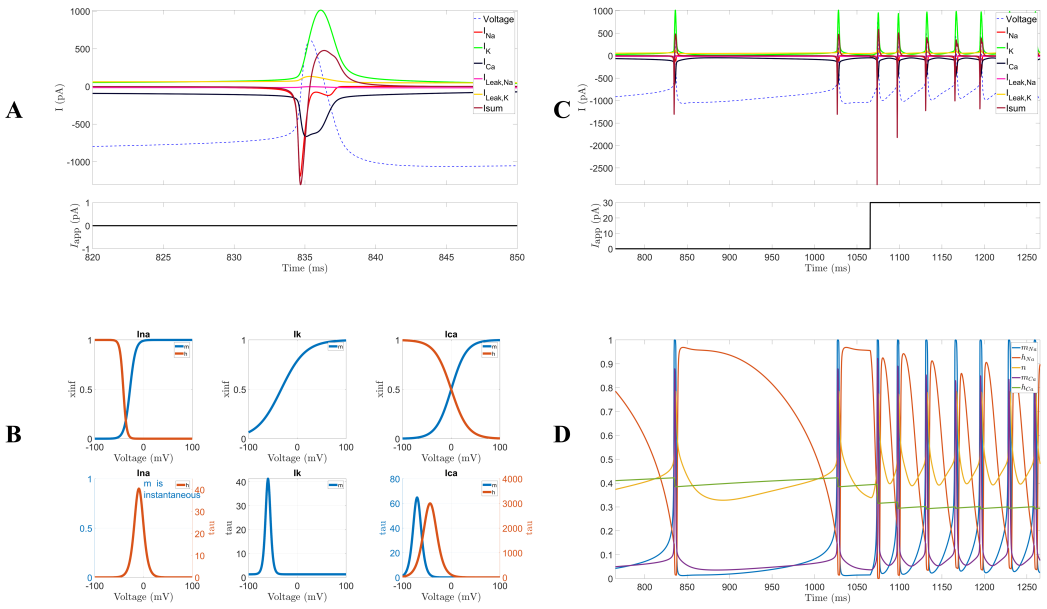


Figure 3.21 SCN non-adapting mechanism.

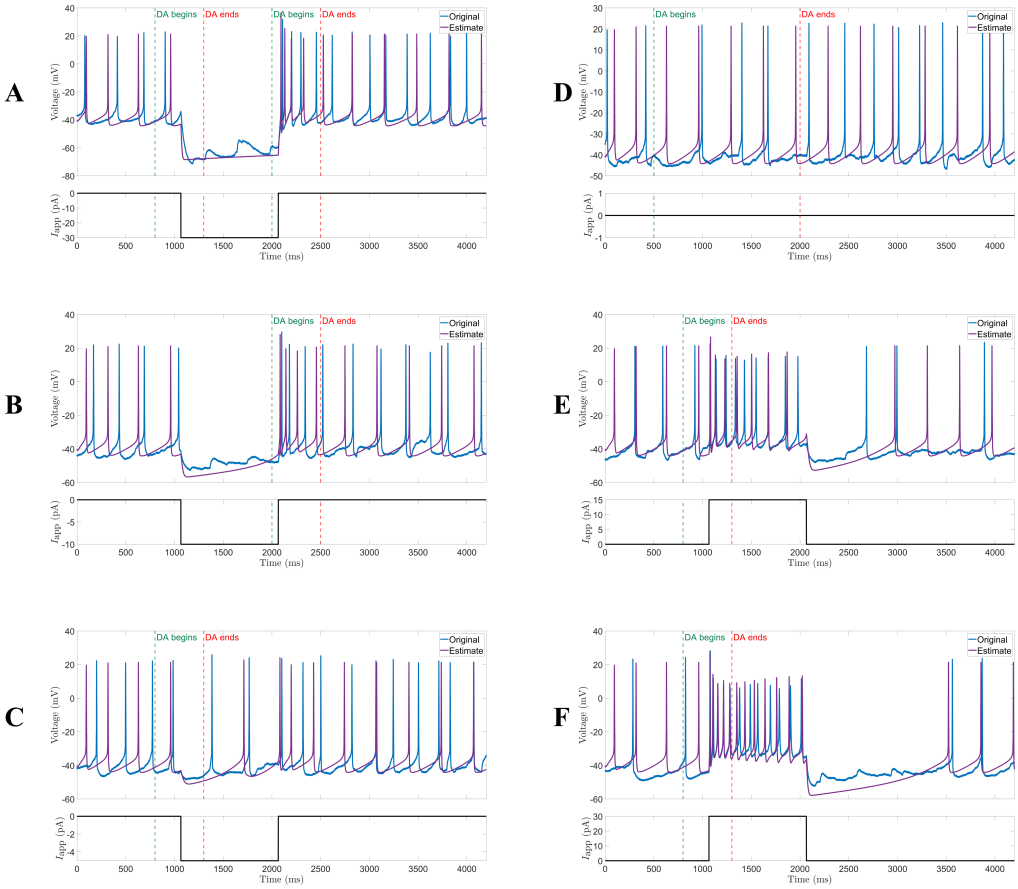


Figure 3.22 SCN data assimilation for training data for cell which is adapting-firing when exposed to depolarizing pulses.

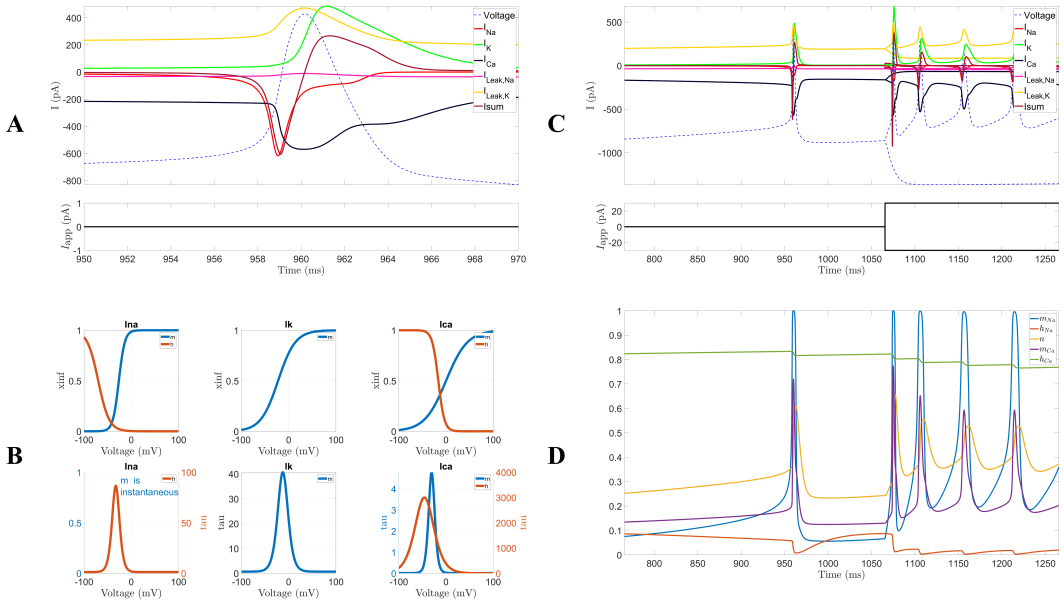


Figure 3.23 SCN adapting-firing mechanism.

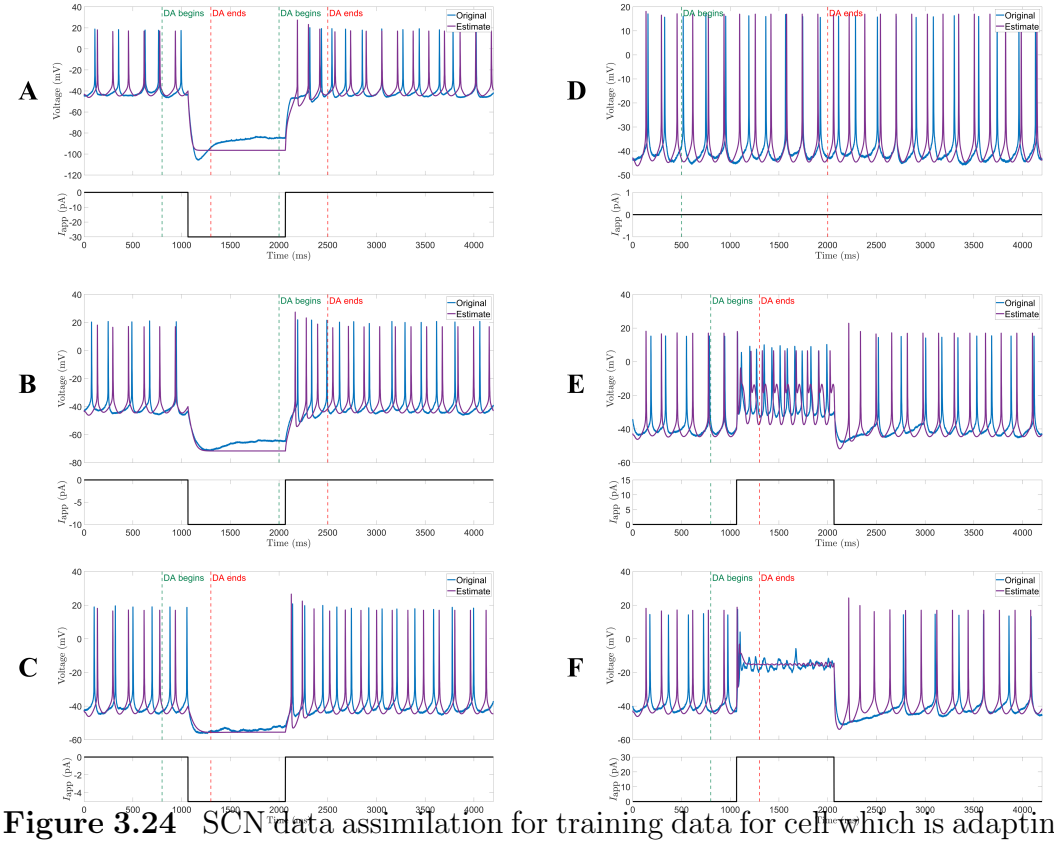


Figure 3.24 SCN data assimilation for training data for cell which is adapting-silent when exposed to depolarizing pulses.

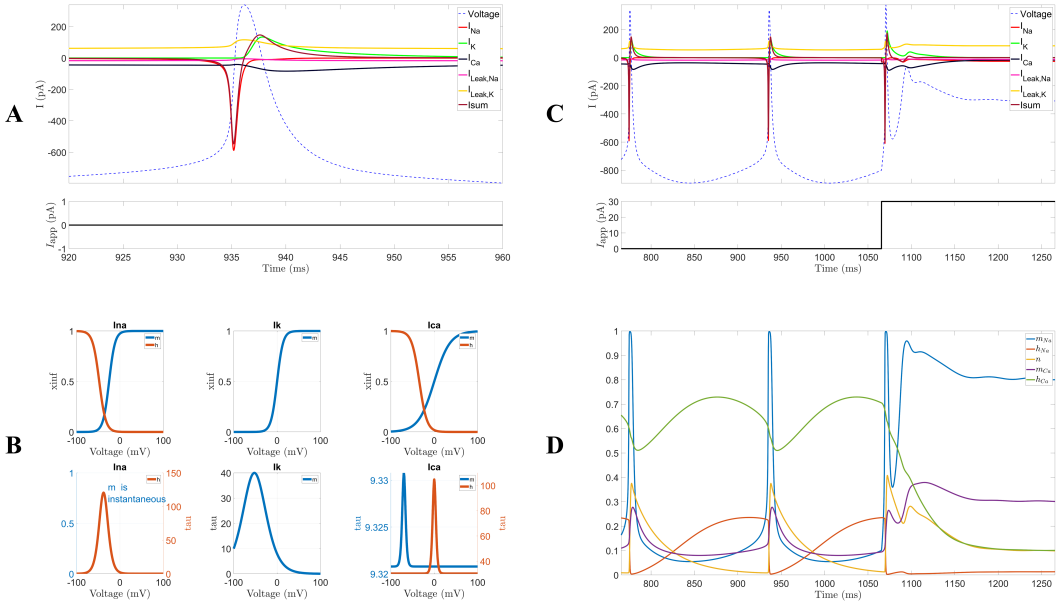


Figure 3.25 SCN adapting-silent mechanism.

CHAPTER 4

TWO STAGE ESTIMATION COMBINING 4D-VAR AND UKF

As we have shown with our experiments with the Morris-Lecar model in Chapter 2, variational data assimilation and ensemble filtering seem to have certain niches. Filtering has excellent utility in, well, *filtering* noisy data to obtain fairly accurate state estimates, and can be augmented to estimate parameters well. Unfortunately, there are few cut and dry mechanisms to ensure global optimality, and tuning the error covariances and prior distribution is a daunting task when there are many components at play. 4D-Var, to an extent, allows the tuning of optimization options to increase precision while being penalized with potentially longer optimization run times. 4D-Var requires very little in the way of prior knowledge, aside from appropriate bounds on states and parameters as well as the weights associated with each term in the Lagrangian.

From an outside perspective, if one is looking for accuracy, 4D-Var might seem to be the catch-all. Unfortunately, there are circumstances in which utilizing 4D-Var is not so straightforward. Memory limitations may prohibit formulating or solving the resulting high-dimensional optimization problem if the optimization variable size enters the hundreds of thousands or greater. Certain cells, like those of the SCN during early ZT, may have firing rates of less than 10 Hz. For example, in the 10 Hz case, we would require on average 100 ms of data before capturing an action potential. Some features of the spike shape, for certain, could be estimated by 4D-Var for this case, but to capture the dynamics underlying spiking one would likely require data from at least a few action potentials. Especially in the case of noisy data, and despite using a driving current to elicit non-stationary responses from the cell, we very well may be utilizing hundreds of milliseconds of data before we can confidently say we have captured the spike-generating mechanisms of the model. If we use a discretization of

0.02 ms, we are, at a bare minimum, looking at $100(1/.02)(L+1)+D$ where L is the number of dynamic variables. Even if we can estimate the spiking dynamics using only several hundred milliseconds of data, what if the system is undergoing some slow adaptation?

Figure 4.1 shows an *in vitro* voltage recording from neurons of pre-Bötzinger complex (pre-BötC), the bursting pacemaker neurons for breathing [38]. This cell has an inter-burst interval of dozens of seconds, and it simply would be impossible to utilize an entire burst, as well as the subthreshold pacing between bursts, in one problem setup of 4D-Var. However, we may be able to capture the fast-timescale dynamics associated with the generation of spiking using a shorter recording using variational data assimilation. The creation and destruction of the burst occurs over a long-time scale, but at the onset of spiking, the system isn't displaying this slow adaption. Hence we would anticipate that the parameters associated with any slow timescale would not have a high degree of sensitivity in this recording.

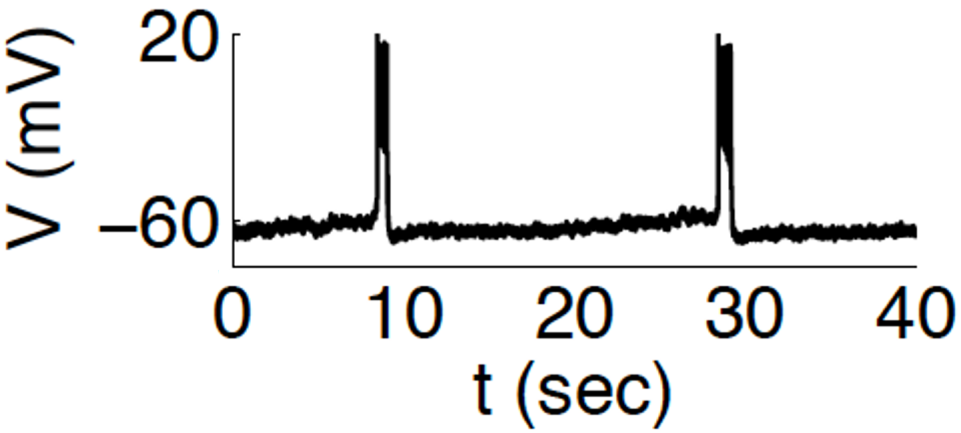


Figure 4.1 pre-Bötzinger complex neuron bursting time series.

A common practice in mathematical biology is to separate dynamics of disparate time scales, for example as in the Michaelis-Menten model of enzymatic reactions in biochemistry [65]. Partially motivated by this asymptotic technique, we propose separating the estimation into fast and slow subproblems. A similar idea associated

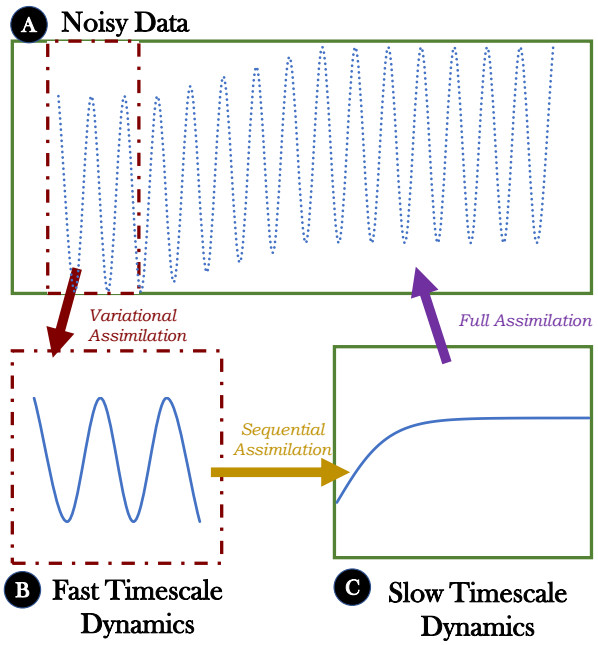


Figure 4.2 Utilizing UKF and 4D-Var for long time-series data.

with calcium dynamics was proposed in [67], wherein the authors separated channels of the model into different compartments based upon their timescales. Here, we don't intend to use time-scale separation to inform the structure of the proposed model *a priori*, but to alter the estimation strategy.

The idea for the two-stage estimation strategy is shown in Figure 4.2. For data similar to the pre-BötC data in Figure 4.1, we would try to estimate the full model using a short observation window. Afterwards, we would fix the parameters for the spike-generation channels/variables, and estimate exclusively the slow channels/variables using the Unscented Kalman Filter with the data in Figure 4.1 as the observation. Ideally, by piecing these stages together, we would have a holistic representation of the model.

Slow timescales emerge in many systems, including bursters, systems undergoing synaptic plasticity, and slowly modulating neocortical inhibitory interneurons. Two models for respiratory rhythm generation were developed by Butera, Rinzel, and Smith (BRS) in a series of papers in 1999 in which intrinsic cell properties were

sufficient to explain burst generation similar to recorded *in vitro* data from neonatal rodent [20]. Within this model exists a faster depolarizing drive for the initiation of the burst and a slow time scale associated with burst termination. BRS developed two models to explain the manifestation of the pacemaking bursting, wherein both were initiated through the activation of a persistent sodium channel. Burst termination in Model 1 occurs through the slow inactivation of persistent sodium channel, whereas Model 2 terminates though a slowly activating potassium current. We only consider Model 1 in this analysis as was used in a recently developed closed-loop respiratory control model [38].

Model 1:

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -I_{\mathrm{NaP}} - I_{\mathrm{Na}} - I_{\mathrm{K}} - I_{\mathrm{L}} - I_{\text{tonic-e}} + I_{\mathrm{app}} \tag{4.1}$$

with

$$\begin{aligned} \frac{dx}{dt} &= \frac{x_\infty(V) - x}{\tau_x(V)} \\ x_{\mathrm{inf}}(V) &= \left\{1 + \exp\left[\left(V - \theta_x\right)/\sigma_x\right]\right\}^{-1} \\ \tau_x &= \bar{\tau}_x / \cosh\left[\left(V - \theta_x\right)/(2\sigma_x)\right] \end{aligned} \tag{4.2}$$

We reformulate these equations using hyperbolic tangents which have more safely controlled derivatives when performing automatic differentiation:

$$\begin{aligned} a_{\mathrm{inf}}(V) &= \frac{1}{2}\left\{1 + \tanh\left[\left(V - vx\right)/dvx\right]\right\} \\ \tau_x &= \bar{\tau}_x \sqrt{1 - \tanh\left[\left(V - vx\right)/(dvx)\right]^2} \end{aligned} \tag{4.3}$$

where $dvx = -2\sigma_x$ and $va = \sigma_x$

Therefore, each time-dependent gating variable can be represented by merely three parameters in this model while still adhering to the typical sigmoidal form for the steady-state functions and bell-shaped curves for the time constants.

To further reduce the complexity of the model, the activation of the fast sodium I_{Na} channel is assumed to happen instantaneously, and the inactivation is approximated as function of the activation of the potassium channel, I_{K}. Thus:

$$\begin{aligned} I_{\mathrm{Na}} &= \bar{g}_{\mathrm{Na}} m_x^3(V)(1-n)\,(V - E_{\mathrm{Na}}) \\ I_{\mathrm{K}} &= \bar{g}_{\mathrm{K}} n^4\,(V - E_{\mathrm{K}}) \end{aligned} \tag{4.4}$$

The persistent sodium with slow inactivation also has instantaneous activation, with distinct parameters to the fast sodium activation, while the time constant for inactivation is orders of magnitude larger than those for the spiking dynamics. Precisely, $\bar{\tau_h} = 10,000$ milliseconds.

$$I_{\mathrm{NaP}} = \bar{g}_{\mathrm{NaP}} m_\infty(V) h\,(V - E_{\mathrm{Na}}) \tag{4.5}$$

Additionally, for our simulations, we assume that there will be no excitatory input from the system, so the maximal conductance for I_{tonic} will be zero. The dynamics of the system at the onset of the burst can be seen in Figure 4.3. Figure 4.3 A shows that the change in voltage during the spike is primarily driven by the transient sodium current, and repolarizes through the transient potassium current. Smaller contributions to the depolarization come from the persistent sodium current and repolarization from the leak current. Figure 4.3 B illustrates that the inactivation of the persistent sodium channel changes extremely slowly during the burst and can almost be perceived as constant. The model is capable of traversing from quiescence,

to bursting, to tonic spiking through shifts in the applied current, or equivalently, the leak reversal potential. For our data assimilation experiments, we assumed that the system would be reminiscent of expected *in vitro* activity, so we fixed the applied current as zero and the reversal potential for the generating model was such to elicit burst activity. Specifically, $E_L = -60$.

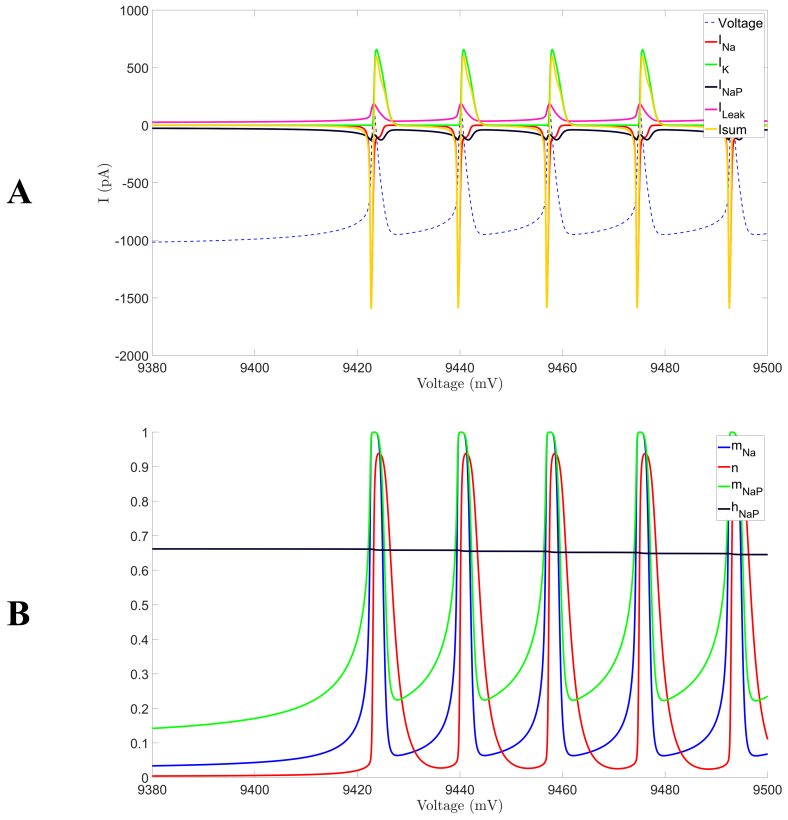


Figure 4.3 Dynamics for BRS Model 1 at the onset of the burst.

We simulate Model 1 for several seconds as the burst duration and interburst intervals are seconds long. In keeping with our established twin experiment tradition, we add some white noise atop our signal to degrade it. For this particular experiment, we will only subject our variational method to a subset of the data which contains the burst initiation. This data segment is shown in Figure 4.4, where only the first 100 milliseconds of the burst are used in the variational assimilation, with a sampling rate of 50 kHz. Because the inactivation time-constant is orders of magnitude in difference

from the rest of the system, we estimate the natural logarithm of the parameter, and feed in the exponentiation of this natural log into the dynamical equations.

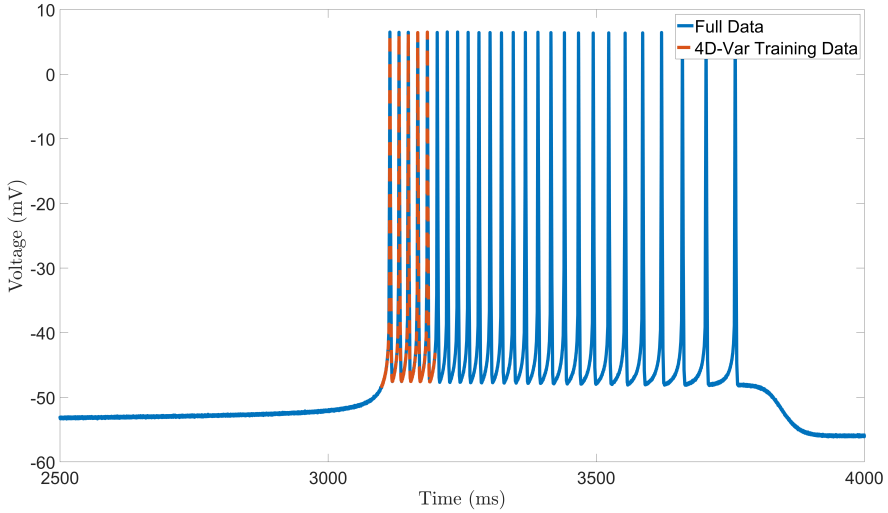


Figure 4.4 Burst for BRS Model 1.

After the 4D-Var trial is over, we then fix the parameters for everything aside from the slowly inactivating channel, I_{Nap}. In practice, we would have no knowledge that our other parameters were sufficiently close to their true values besides blind faith. We are working towards integrating sensitivity measures and other uncertainty quantifiers into our variational framework.

For the remaining parameters, we allow them to vary slowly within UKF alongside the states which are being estimated using our augmented state space model. Here, to fully ascertain the slow dynamics, we use *a lot* of data, namely, 300 seconds worth which is roughly 50 bursts. Additionally, we performed 5 incremental runs, using the initial guesses of the system at the final time point as starting conditions for the next run. A final effort we undertook to speed up the process was to down-sample the observational data between bursts significantly and down-sample a bit during the burst. Observations are recorded every 1 millisecond between bursts and every .1 millisecond during the burst phase.

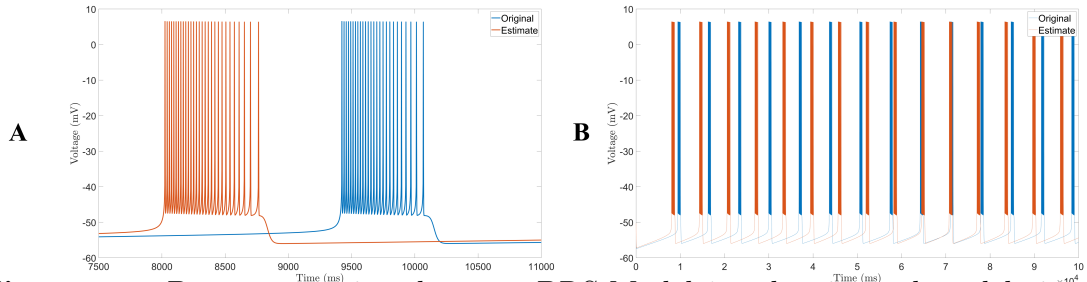


Figure 4.5 Burst comparison between BRS Model 1 and estimated model via two-stage strategy

The estimated parameters are shown in Table 4.1. Some of the parameters are still off by a bit, leading to slightly incorrect predictions of the behavior. Figure 4.5 shows the voltage versus time plot for the original model compared to the estimated model. The original model had 29 APs per burst, whereas the estimated model has 26. Figure 4.5 B shows that the bursts themselves have roughly the same interburst duration.

Table 4.1 BRS Estimated Model Parameters from Two-Stage DA Strategy.

Parameter	Original Value	4D-Var	4D-Var → UKF
g_{Na}	28	27.98	27.98
g_{K}	11.2	11.27	11.27
g_{Nap}	2.8	3.78 $*$	3.02
g_{L}	2.8	2.74	2.74
vm	-34	-33.97	-33.97
dvm	10	9.97	9.97
vn	-29	-29.07	-29.07
dvn	8	8.08	8.08
τ_n	10	9.70	9.70
vm_{Nap}	-40	-39.58	-39.58
dvm_{Nap}	12	12.07	12.07
vh_{Nap}	-48	-99.91 $*$	-48.40
dvh_{Nap}	-12	-17.46 $*$	-12.76
τ_{hNap}	1e4	8.71e4 $*$	9.15e3

The intention of this work is to apply to real neural data which contain a mix of slow and fast timescales. What would aid in this quest, and in parameter estimation in general, is perform *a priori* analysis to determine whether our data contains information relevant to the model parameters in question. We extend this thought into the following chapters. Chapter 5 defines the ideas of sensitivity and identifiability in the context of parameter estimation. Chapter 6 explores sensitivity

and its relation to information-rich data, as we endeavor to optimize the amount of information within our data by appropriately designing an injected current.

CHAPTER 5

IDENTIFIABILITY OF CONDUCTANCE-BASED MODELS

There are two complementary ideas regarding the estimation of parameters: *sensitivity analysis* and *identifiability*. Sensitivity analysis predominantly describes how sensitive one's data, with an observation operator mapping model output to the data, is to one's model parameters, and how variations in parameters introduce variation in the data. Identifiability analysis involves interactions among the model's parameters. Particularly, *structural* identifiability involves how the structure of the model may create degeneracies in the sets of unknown parameters. For instance, consider the trivial case where we can observe some quantity x_n, and the system has linear rates for two processes ax and bx:

$$x_{n+1} = ax_n + bx_n. \tag{5.1}$$

Without knowing any further information to disambiguate a and b, perhaps coming from some algebraic constraint, it is obvious that we can only hope to estimate the summation of the two. Namely, this system can be solved for by regression to obtain $\hat{c} = a + b$. However, the system may be insensitive to some parameters. For instance, in this system, if the initial condition was $x_0 = 0$, the system will remain at 0 for any parameter values given that it is a steady-state. Therefore, for our parameters, the regression problem would be underdetermined. Collectively, while the quantity $\hat{c} = a + b$ is structurally identifiable, it is not *practically* identifiable since the data is insensitive to it. Olsen et al. provide an excellent overview of these relations and the modern techniques to determine which subset of one's parameters may be practically identified [87].

To further complicate the problem, there are additional layers to these categorizations. Namely, *global* versus *local* descriptions of identifiablity and sensitivity. Hong et al. gives an excellent methodology to understand global identifiability for models whose parameters enter as rational expressions [58]. Villaverde et al. provides a digestible discussion of the differences [110].

For models in neuroscience, we are often left in an unfortunate position. The parameters for gating variables typically enter into the equations nonlinearly, and as of now, no algebraic-based technique can deal with the level of complexity in these expressions, nor the number of variables and parameters in useful models. Walch et al. has approached the problem from a differential algebra perspective, and can compute input-output equations of the dynamics through Groebner bases [113]. With voltage-clamp and under certain assumptions, they have shown the identifiability of kinetic time constants, but the non-identifiability of the combination of maximal conductances and steady-state functions. However, we are working with current-clamp data. This introduces an additionally level of complexity in that we are only seeing the effect of the summation of all the channels during an experiment as opposed to an isolated individual channel. While Walch et al. have proved that the activation functions cannot be disambiguated from the maximal conductances for voltage-clamp, we have shown that these, for at least small models in the Morris-Lecar, can be estimated. Therefore, my personal thoughts are that these analyses are important in constructing subsets of parameters to focus efforts on which have the highest sensitivity and aren't "obviously" unidentifiable, but the computational techniques need to catch up for the approach to be generally applicable.

Regardless, we endeavored to understand, as a bare minimum, the local structural identifiability of a conductance-based model beyond the performance yielded with our parameter estimates. We have shown that in Tables 2.2 and 2.4 that the appropriate parameters are estimated regardless of initial condition of the

parameters; if this was true for any subset of parameters in their permissible domains, the system would be globally identifiable. In some regard, this is analogous to attracting states in dynamical systems.

The Structural Identifiability criterion merely states that

$$y(t,\hat{\theta}) = y\left(t,\theta^*\right) \Rightarrow \hat{\theta}_i = \theta_i^* \tag{5.2}$$

If Equation (5.2) holds in some neighborhood of $\hat{\theta}$, we say that, dropping the accents, θ is locally identifiable. If multiple parameter sets have identical observations, then θ is unidentifiable. If this condition holds for any neighborhood, then θ is globally identifiable.

To compound the problem, there may be regions in parameter space where

$$\left\| y(t,\hat{\theta}) - y\left(t,\theta^*\right) \right\| < \epsilon \quad \forall \hat{\theta}, \theta^* \tag{5.3}$$

In these regions, we say that θ is *non-influential* [120]. This refers back to the example earlier, Equation (5.1), regarding the system being at steady state. When a parameter is structurally identifiable and influential, it is practically identifiable.

A major issue is that may not know where these neighborhoods are. We may have an idea; namely, between bursts of the bursting Morris-Lecar, the observed data probably is not influenced too heavily by the parameters of the spiking channels. Whether or not our parameters have some structural degeneracy, like in Equation (5.1), may not be apparent when working with possibly dozens of equations with nonlinear interactions.

Principal to this area is a seminar paper by Kalman outlining the notions of *controllability* and *observability* for linear systems [68]. This work was extended by

Hermann and Krener to nonlinear systems [55]. Effectively, controllability is the ability to use an external input, in our case an applied current, to transition the system from any initial state x_0 to any final state x_f in finite time. Observability is the dual problem; can we fully observe our system (all states) in finite time given our measurements. For the moment, we will focus explicitly on the latter question.

For linear systems, a simple check is to construct the observability matrix. For example, consider the controlled linear system:

$$\begin{aligned} \dot{x} &= Fx + Cu \\ \dot{y} &= Hx \end{aligned} \tag{5.4}$$

We can take repeated derivatives of the observation equation to arrive at a system of equations:

$$\begin{aligned} \dot{y} &= Hx \\ \ddot{y} &= H(Fx + Cu) \\ &\vdots \\ \frac{dy^{n-1}}{dt^{n-1}} &= H(F \dots (Fx + Cu)) = HF^{n-1}x + HCu \end{aligned} \tag{5.5}$$

In matrix vector form, this looks like:

$$\begin{bmatrix} y \\ \dot{y} - HCu \\ \vdots \\ \frac{dy^{n-1}}{dt^{n-1}} - HCu \end{bmatrix} = \begin{bmatrix} H \\ HF \\ \vdots \\ HF^{n-1} \end{bmatrix} x = \mathcal{O}x \tag{5.6}$$

To understand the observability, we need only to calculate the rank of $\mathcal{O}$. A system of size n is fully observable if $\text{rank}(\mathcal{O}) = n$. The extension to nonlinear systems requires use of Lie derivatives. Suppose h is a smooth scalar function and f a smooth vector field operating on some manifold $\mathcal{M} \subseteq \mathbb{R}^n$. The Lie derivative of h with respect to f is the directional derivative in f's direction:

$$\mathcal{L}_f h(x) = \nabla h(x) f(x,u) = \frac{\partial h(x)}{\partial x} f(x,u) \tag{5.7}$$

This relation can be applied recursively to obtain

$$\mathcal{L}_f^i h(x) = \frac{\partial \mathcal{L}_f^{i-1} h(x)}{\partial x} f(x,u) \tag{5.8}$$

When only observing one state variable, $h(x)$ is scalar, and $\frac{\partial h(x)}{\partial x}$ is $1 \times n$.

To preserve notation, our system dimensions in question are $x \in \mathbb{R}^L$, $\theta \in \mathbb{R}^D$. Thus, we form the $L \times L$ observability matrix

$$\mathcal{O}(x) = \begin{pmatrix} \frac{\partial}{\partial x} h(x) \\ \frac{\partial}{\partial x} \left(\mathcal{L}_f h(x) \right) \\ \frac{\partial}{\partial x} \left(\mathcal{L}_f^2 h(x) \right) \\ \vdots \\ \frac{\partial}{\partial x} \left(\mathcal{L}_f^{L-1} h(x) \right) \end{pmatrix}. \tag{5.9}$$

So far, we have only outlined observability of the states of our system under certain controls. Naturally, like during our augmented state-space approach with Kalman Filtering, we can augment the state space, where parameters have trivial

dynamics, and arrive at the matrix and condition we need for local observability:

$$\mathcal{O}(x) = \begin{pmatrix} \frac{\partial}{\partial x} h(x) \\ \frac{\partial}{\partial x} \left(\mathcal{L}_f h(x) \right) \\ \frac{\partial}{\partial x} \left(\mathcal{L}_f^2 h(x) \right) \\ \vdots \\ \frac{\partial}{\partial x} \left(\mathcal{L}_f^{L+D-1} h(x) \right) \end{pmatrix}. \tag{5.10}$$

Observability Rank Condition (ORC): if the observability matrix of the dynamical system has full rank, i.e. for $\mathcal{O}(x_0)$ from Equation (5.10),

$$\text{rank}\left(\mathcal{O}(x_0)\right) = L + D$$

then the system is (locally) observable and identifiable around x_0 [110]. Calculation of observability coefficients has been conducted previously for neural models in the context of trajectory estimation [4]. An alternative partial observability metric for neural systems has been developed based on the capacity of the model to reconstruct the data [101].

Beyond motivating further exploration into the scalability of identifiability tools for nonlinear systems, this chapter has a simple, and previously considered result: the Morris-Lecar model *is* locally identifiable and this can be proved numerically. For this, we used the MATLAB toolbox named STRIKE-GOLDD (STRuctural Identifiability taKen as Extended-Generalized Observability using Lie Derivatives and Decomposition) package which is freely available (a common theme in this dissertation) [110]. This package makes use of the symbolic toolbox in MATLAB.

Table 5.1 Matrix of ORC for Morris-Lecar Model Evaluated for Hopf Parameters

V	n	ϕ	G_{Ca}	$V3$	$V4$	G_{K}	G_{leak}	$V1$	$V2$
1e+00	0e+00	0e+00	0e+00	0e+00	0e+00	0e+00	0e+00	0e+00	0e+00
-1e-01	-2e+01	0e+00	1e-01	0e+00	0e+00	-4e-01	-1e+00	-5e-02	1e-01
2e-02	3e+00	3e+00	-1e-02	2e-03	-3e-03	6e-02	1e-01	4e-03	-1e-02
-3e-03	-5e-01	-6e-01	3e-03	-2e-04	3e-04	-1e-02	-2e-02	-1e-03	2e-03
4e-04	1e-01	1e-01	-4e-04	7e-05	-9e-05	2e-03	4e-03	2e-04	-4e-04
-5e-05	-2e-02	-3e-02	9e-05	-1e-05	2e-05	-3e-04	-7e-04	-4e-05	8e-05
-7e-06	2e-03	5e-03	-1e-05	3e-06	-5e-06	3e-05	6e-05	6e-06	-1e-05
1e-05	7e-04	-7e-04	2e-07	-5e-07	7e-07	2e-05	4e-05	-3e-08	-2e-06
-6e-06	-7e-04	-2e-04	2e-06	-3e-08	1e-07	-2e-05	-4e-05	-8e-07	3e-06
4e-06	5e-04	3e-04	-2e-06	1e-07	-2e-07	1e-05	2e-05	7e-07	-2e-06

STRIKE-GOLDD can try to decompose the model into submodels and has the capacity to find identifiable combinations of otherwise unidentifiable parameters.

However, we have found, and as is stressed by the developers, that systems where $L + D > 10$ are infeasible to try to compute. Thankfully, $L + D = 10$ for the Morris-Lecar parameter estimation problem we investigated in Chapter 2.

To take it a bit further, we tried to develop our own simple implementation for calculating identifiability using automatic (algorithmic) differentiation, using CasADi [5]. We were able to validate our result for Morris-Lecar from the resulting matrix rank of our expression matrix evaluated on nominal parameter values. That matrix is shown in Table 5.1 using the parameters for the Hopf regime and an intial condition for V and n taken from the limit-cycle. The rows are indexed by orders of the Lie derivatives of Equation (5.10). The columns are organized as the partial derivatives with respect to that particular state or parameter.

We were additionally able to take the system dimension up to 12 to check that the bursting Morris-Lecar model was similarly identifiable. Unfortunately, past this level, we were unable to keep the expression graphs from being too memory intensive, so any further investigation into more complex models is left for future work. Some other toolboxes for global identifiability for rational expressions, such as those emerging in models of chemical reactions, epidemiology, and pharmacokinetics, are available [28,59]. The following chapter delves further into sensitivity analysis for optimal experimental design in neuroscience.

CHAPTER 6

OPTIMAL DESIGN OF INJECTED CURRENTS

6.1 The Optimal Injected Current Problem

Thus far, we have touched on sensitivity of our parameters to data, and whether our parameters, based on their structure, are observable. An associated condition, whose conditions for nonlinear systems were summarized by Hermann and Krener, is *controllability* [55]. Namely, controllability refers to the capacity of a system to produce a trajectory which, from an arbitrary initial condition, can arrive at a final state in finite time. The dual problem for nonlinear systems has already been presented in our earlier chapter on observability, where we can consider a control "u" as our input to the system.

When collecting data, electrophysiologists are capable of controlling the real neural system to do many things and there are a number of natural protocols to choose from to extract particular information. Often, step currents of different amplitudes are used to construct FI curves. Hyperpolarizing steps may reveal the input resistance of the cell, and, thereby, the capacity of the cell to respond to external drives. Sharp transitions, or pulses, may unveil certain latent time scales in the system. Sinusoidal currents, or zap currents which vary in amplitude and frequency, can unveil resonance properties.

The above stimuli have functional implications to understanding biological phenomena. Modelers are interested in these biological properties, but are also interested in having data which can unveil aspects of their model. When the system can't be sufficiently influenced to reveal all of the parameters in a model, it is the responsibility of the modeler to fix certain parameters, or drop completely non-influential terms if they contribute little to the dynamics. We arrive then at a

motivation for a protocol: one which reveals the most information about our model, and thus, can further the understanding of existing biological phenomena.

In absence of a time varying applied current, or in complement to one, one can drive a neural system by the observed voltage; we will touch on this in Chapter 7. The issue is whether that observed voltage is dynamically rich enough to estimate all model parameters of interest. Others have chosen to drive a system with a chaotic input current for both twin experiments and real data assimilation, which seeks to push the dynamics between attracting regions and spend adequate time in the transient states. By using a chaotic current, broad frequency bands are spanned, but can be chosen so as not to be filtered out by the membrane time constant. However, depending on the framework chosen to perform the estimation, these open questions remain:

- How much data to use?
- What is the appropriate time discretization is in the estimation procedure (t_k from Equation (1.5))
- What type of current can elicit the "best" responses for both model-generated data and real-world cells?

Thankfully, some work has been done on the number of observations required [94]. Before devising an optimal control strategy, the metrics by which we consider *optimality* have to be declared. What is the best response of the dynamical system for estimation? Is it related to the CLEs of the synchronization system? Is it a measure of entropy? Or the fractal dimension? One thought process is that the resultant phase space is *dense* in some capacity. For some fixed amount of time, the system has exhausted its dynamics so much so that the inverse problem has relaxed.

Consider Figure 6.1. For any problem, these are likely the data we have at our disposal to inform any optimization problem of choice. For data riddled with noise, small deviations caused by noise will lead to large deviations in the numerical

derivative of the observations so using this as a metric directly can sometimes be ill-advised. However, as a motivation, we can see that the system stimulated by a constant applied current will only traverse its stable limit cycle (if it exists) after transient departure from an arbitrarily imposed initial condition. If we consider problems from synchronization, one could consider such stable limit cycles using a circle map which deconstructs an L dimensional phase space to a one dimensional problem of phase. Similarly here, we have reduced the data to one of infinitely many trajectories in phase space. Alternatively, when forced with the scaled chaotic output from the Lorenz '63 model, as conducted in [106], the system traverses a denser region of phase space.

Some work on optimal driving currents for neuroscience has been done [25, 26, 45]. However the goal of these applications was to initiate, or silence, spiking using terminal conditions on the voltage using a minimum energy stimulus. Namely, to impose that at the final time, the voltage matches expected values at the beginning or end of an action potential which would signal the initiation of a spike or fading to quiescence. Instead, our problem would be even less constrained as we give no considerations as to what the values at the end of the estimation window would be.

Here is the setup (following [40, 48]). Per usual, define our conductance-based model as:

$$\frac{dV}{dt} = -\sum_{\mathrm{i}} I_{\mathrm{i}} + I_{\mathrm{app}}(t) \tag{6.1}$$

$$\frac{da}{dt} = \frac{a_\infty(V) - x}{\tau_a(V)} \tag{6.2}$$

for all additional states, a. For notational convenience, define $u = I_{app}$ and our system to be

$$\frac{dx}{dt} = f(x, u), \quad x \in \mathbb{R}^L \tag{6.3}$$

where u enters only in the current balance equation. We can take initial conditions of the system to be approximately known, if we set the voltage at its resting membrane potential and the states to be their corresponding steady state values $a(t_0) = a_\infty(V_{rest})$ so $x(t_0) = x_0$ and we allow the terminal conditions to be free (even this restriction of the initial conditions can be relaxed, as will be shown).

We desire to minimize some performance index

$$J[u] = \int_{t_0}^{t_f} \mathcal{G}(u, x, t)dt \tag{6.4}$$

subject to Equation (6.3). From here, we have some choices. Two flavors exist for such optimal control problems: direct and indirect methods. Practically, the difference between these methods is when one chooses to discretize the problem. In the direct approach, we discretize Equation (6.4) directly. The collocation methods we have employed for variational data assimilation, for instance, are a product of direct transcription. For the indirect approach, we can derive the equations of motion assuming that $\mathcal{G}$ is differentiable and use an appropriate numerical scheme to try to solve for the optimal current u after discretizing the resulting boundary value problem. This requires simultaneously integrating Lagrange multiplier equations, which has been shown to, in some cases, improve accuracy of the solution [13, 14]. The performance index would then be expanded to include these terms:

$$\begin{aligned} P(u(t)) = \; & \mathcal{G}(u(t)) + \lambda_{x_1}\left(\tfrac{dx_1}{dt} - f_1(x,u)\right) \\ & + \lambda_{x_2}\left(\tfrac{dx_2}{dt} - f_2(x,u)\right) + \cdots + \lambda_{x_n}\left(\tfrac{dx_n}{dt} - f_n(x,u)\right) \end{aligned} \tag{6.5}$$

However, the introduction of additional variables in the augmented system with poorly quantified initial conditions leads to a high likelihood of divergence and a

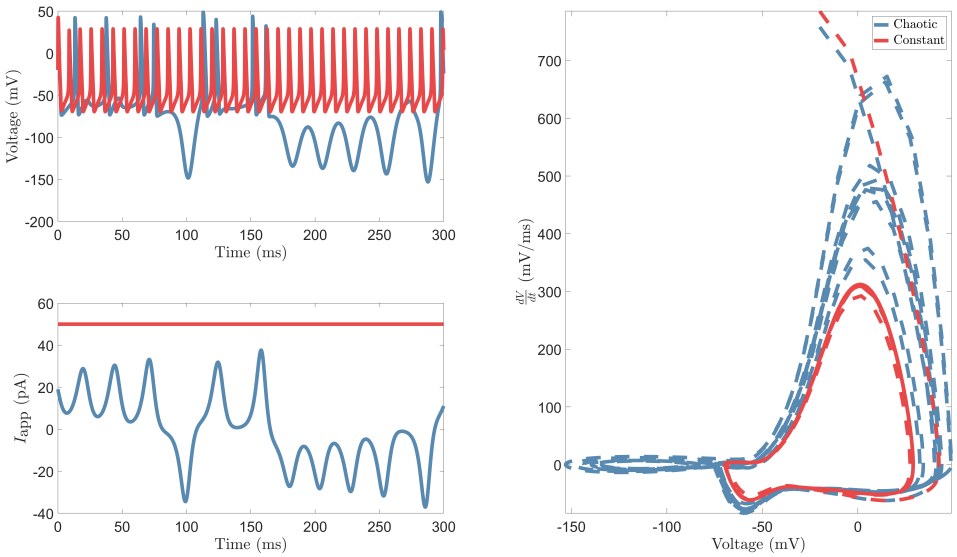


Figure 6.1 Voltage and I_{app} versus t and $\frac{dV}{dt}$ vs V for the NaKL model. The red is under constant applied forcing and the grey is forced with the scaled output of the Lorenz '63 model.

smaller basin of attraction. For the remainder of this work, we elect to consider only the direct problem.

Outside of biological applications, constructing optimal inputs for parameter estimation spans back to early work from Mehra, Nahi, and Payne, amongst others [77, 78, 84, 90]. Many of these applications sought to understand systems which could be heavily controlled and repeated experiments could be costly, such as with aircraft flight tests. Here, the optimal input might be deflections of the elevator located on the rear stabilizer and one can measure pitch rate and attack angle [77]. Optimal experimental design, as a general problem, has gained popularity across fields and across models types, as the principal ideas extend to arbitrarily parameterized models.

Recall from Chapter 1 that we related data assimilation to Bayesian estimation, as for our states and parameters,

$$P(\mathbf{X}|\mathbf{Y}) \propto P(\mathbf{Y}|\mathbf{X})P(\mathbf{X}). \tag{6.6}$$

where

$$P(\mathbf{X}|\mathbf{Y}) = P\left(\mathbf{X}\left(t_0\right), \ldots, \mathbf{X}\left(t_N\right), \boldsymbol{\theta}, |\mathbf{y}\left(t_0\right), \ldots, \mathbf{y}\left(t_N\right)\right) \tag{6.7}$$

Data assimilation can be derived from this formulation, and it can be shown that the cost functions we've been using for variational data assimilation can be reformulated as

$$-\log\left(P(\mathbf{X}|\mathbf{Y})\right) = \hat{C}(\mathbf{X};\mathbf{Y}) \tag{6.8}$$

for $\hat{C}(x) \approx C(x) + \text{Constant terms}$, for $C(x)$ from Equation (3.3) [1].

Functionally, seeking minima of this cost function relates to the expected states and parameters conditioned on the measurements [122].

$$E[\mathbf{X}|\mathbf{Y}] = \frac{\int d\mathbf{X}\left(\mathbf{X}\right) \exp\left[-\hat{C}(\mathbf{X})\right]}{\int d\mathbf{X} \exp\left[-\hat{C}(\mathbf{X})\right]} \tag{6.9}$$

From its structure, with an exponential kernel of the cost function, from variational calculus we know that by using Laplace's method we can approximate equation 6.9 by finding extremum paths such that the Jacobian is zero

$$\left.\frac{\partial \hat{C}(\mathbf{X})}{\partial \mathbf{X}}\right|_{\mathbf{X}=\mathbf{X}^k} = 0, \quad k = 0, 1, \ldots \tag{6.10}$$

and the Hessian, $\left.\frac{\partial^2 \hat{C}(\mathbf{X})}{\partial \mathbf{X}^2}\right|_{\mathbf{X}=\mathbf{X}^k}$, is positive definite [72].

When a particular minimum, $\hat{C}(\mathbf{X}_{\min})$, is uniquely smaller than any other, it exponentially dominates the integral in equation 6.9, and thus provides a good approximation to the mean of the distribution as our maximum likelihood estimate.

This statistical information is presented now to try to bridge the gap towards another statistical topic in information theory. We can separate the cost function, as we have alluded to thus far, into the "measurement term" and "background term". The measurement term

$$L(x) = \frac{1}{2}\sum_{k=0}^{N} \|y_k - h(x_k)\|^2_{R_k} = \frac{1}{2}\sum_{k=0}^{N} R^{-1}\left(y_k - V_k\right)^2 \tag{6.11}$$

equates to our log-likelihood function under a Gaussian error approximation. We want to investigate how much information our signal contains, and in particular, how changes in parameters relate to changes in our data. These dependencies are restricted to our likelihood function. We can relate these dependencies using the well-known Fisher Information theorem, which relates the variance of our likelihood function to the curvature of the likelihood function for a given parameter. The entries of the Fisher matrix are given by:

$$[\mathcal{I}(\theta)]_{i,j} = \mathrm{E}\left[\left(\frac{\partial}{\partial\theta_i}L(y|\theta)\right)\left(\frac{\partial}{\partial\theta_j}L(y|\theta)\right)|\theta\right] \tag{6.12}$$

An unbiased estimator for this expectation can come from linearizing around some nominal parameter set, $\hat{\theta}$. Thus we arrive at our linearized, Gaussian error approximation for the Fisher Information Matrix (FIM).

$$F = X^T X, \quad \text{where } X = \begin{pmatrix} \frac{\partial y(t_1)}{\partial \theta_1} & \cdots & \frac{\partial y(t_1)}{\partial \theta_D} \\ \vdots & \ddots & \vdots \\ \frac{\partial y(t_N)}{\partial \theta_1} & \cdots & \frac{\partial y(t_N)}{\partial \theta_D} \end{pmatrix} \tag{6.13}$$

Of course, we still need a way to not only construct this matrix in practice, but to use it meaningfully. For constructing and evaluating the FIM, we will unfortunately need to make a sacrifice in that we will operate from a local approximation. For the rest of the analysis here, we will assume that perhaps we have a nominal model for the system, with nominal parameter values, and we ascertain the information from perturbations of this nominal model. Of course, this is problematic, as we are then biasing our signal generation a bit to that of a set of parameters which may very well be far removed from the actual globally optimal parameters of the system. For systems with large dimensions of parameters, this is a necessary evil.

Global sensitivity measures exist, but are computationally intractable for the parameter sizes we are interested in. Variance-based global sensitivity measures, such as Sobol' indices, describe how the output variance of a model is attributed to parameters. Composition of a similar matrix to the FIM using Sobol' indices for optimal experimental design of bioreactors has been conducted [96]. An overview of recent developments for constructing local and global sensitivities is available, including derivative-based and variance-based (integral) methods [87]. There are a few candidates for how to construct these sensitivities. One can extract the partial derivative of the data with a simple numerical derivative, e.g., with forward difference

$$\frac{\partial y(t,\theta)}{\partial \theta_i} = \frac{y\left(t, \theta + he_i\right) - y(t,\theta)}{h} \tag{6.14}$$

where here e_i is a projection operator onto the ith parameter, and h may be tuned appropriately for each parameter in question. Another method draws from the dynamics themselves, to construct an augmented series of equations, known as the sensitivity equations, which are simultaneously integrated with the original ODEs.

$$\frac{\partial \left(dx_i/dt\right)}{\partial \theta_j} = \frac{d}{dt}\left(\frac{\partial x_i}{\partial \theta_j}\right) = \sum_{k=1}^{D}\left[\left(\frac{\partial f_i}{\partial x_k}\right)\left(\frac{\partial x_k}{\partial \theta_j}\right) + \frac{\partial f_i}{\partial \theta_j}\right] \tag{6.15}$$

$$\frac{\partial x_i(0)}{\partial \alpha_j} = 0 \tag{6.16}$$

A final option to constructing the sensitivities is to use automatic differentiation, and we elect to use AD via CasADi for this. For convenience, we will express our covariance matrix as the inverse of the FIM.

$$\mathbf{\Sigma} \equiv F^{-1} \tag{6.17}$$

A few interpretations of maximal information come from equivalently minimizing the variance of $\mathbf{\Sigma}$. These fall under what are known as the "alphabet": criterion, where each metric is given a different letter.

The A-criterion

$$\text{argmin}(\text{trace}(\mathbf{\Sigma})) \tag{6.18}$$

is understood to minimize the average variance of estimates.

The D-criterion:

$$\text{argmin}(\det(\mathbf{\Sigma})) \tag{6.19}$$

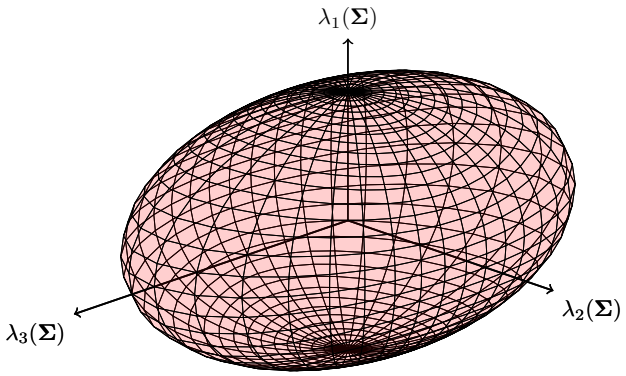


Figure 6.2 Ellipsoid representation of the covariance matrix, $\mathbf{\Sigma}$.

is understood to minimize generalized variance of estimates.

The differences between these two can be visualized in Figure 6.2. The D-criterion aims to minimize the volume of the ellipsoid formed from the principle axes of the ordered eigenvalues of $\mathbf{\Sigma}$, as the volume is proportional to $\det(\mathbf{\Sigma})$. The A-criterion aims to shorten each of the principal axes of the ellipsoid, respectively.

An implementation of efficient matrix operations for OED via the FIM for ODEs has been done in the software package casiopeia (Casadi Interface for Optimum experimental design and Parameter Estimation and Identification Applications), built on CasADi [19].

6.2 Optimal Current for Hodgkin-Huxley Type Model

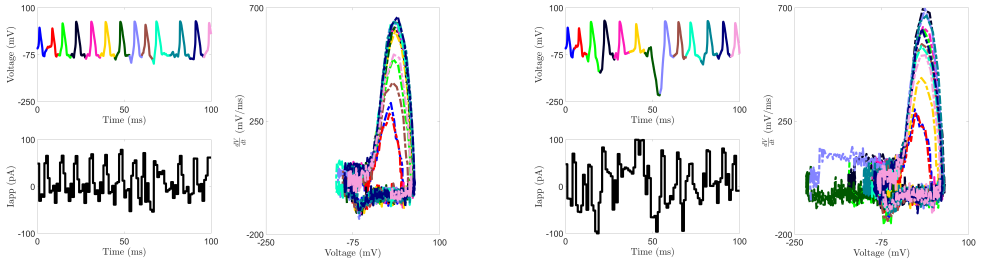
As a first pass, we decided to conduct twin experiments for a fixed time window on the standard NaKL model to see which currents emerged from satisfying the A and D criteria. We assume an observational frequency of 50kHz for 100 ms. Due to a scalability constraint via the implementation strategy, the optimal currents are only estimated on 1 ms intervals, and remain constant between forming a fine-grid series of steps. The estimated currents are shown in Figure 6.2. The optimally derived currents have different properties, as the D-optimal current traverses more hyperpolarized regions. The voltage speed plots, organized by color of the action potential train, show a dense catalog of different shapes and transitional regions.

We then wanted to conduct a formal comparison of how the currents and their generated observations would do when estimating the parameters of NaKL in a twin experiment. Their challengers were some standard currents used for estimating properties, including steps, ramps, flat currents, and a segment of the output of the chaotic system, Lorenz '63. As per usual, the observations were then diluted by additive white noise. The results for a specific random seed of the noise generation are shown in Table 6.2. If we take a relative error, $\sum_i |\frac{p_i^{est} - p_i^{true}}{p_i^{true}}|$, the computed errors are shown in Table 6.1. The "optimal" currents were more successful than their counterparts for this illustrative example, but in actuality, each injected current faired fairly well in terms of estimating the parameter values. Further exploration into optimal currents for more complex models and, ultimately, for injecting into real cells is ongoing.

One last avenue which we have yet to touch on is how we can drive a neuron through its voltage, as opposed to an injected current. Chapter 7 touches on how we can integrate data assimilation with voltage-clamp data if such data is available. The second part of this chapter details how we can use current-clamp data in a manner that's related to voltage-clamp to reduce the complexity of the parameter estimation problem.

Table 6.1 Relative Error for Various Currents for Estimating NaKL Model

	Aopt	Dopt	L63	Ramp	Steps	Constant
Relative Error	0.1277	0.1872	0.9224	0.2435	0.8060	1.3044



(a) A-optimal applied current (b) D-optimal applied current

Figure 6.3 A and D-optimal applied currents derived using OED strategies for NaKL model.

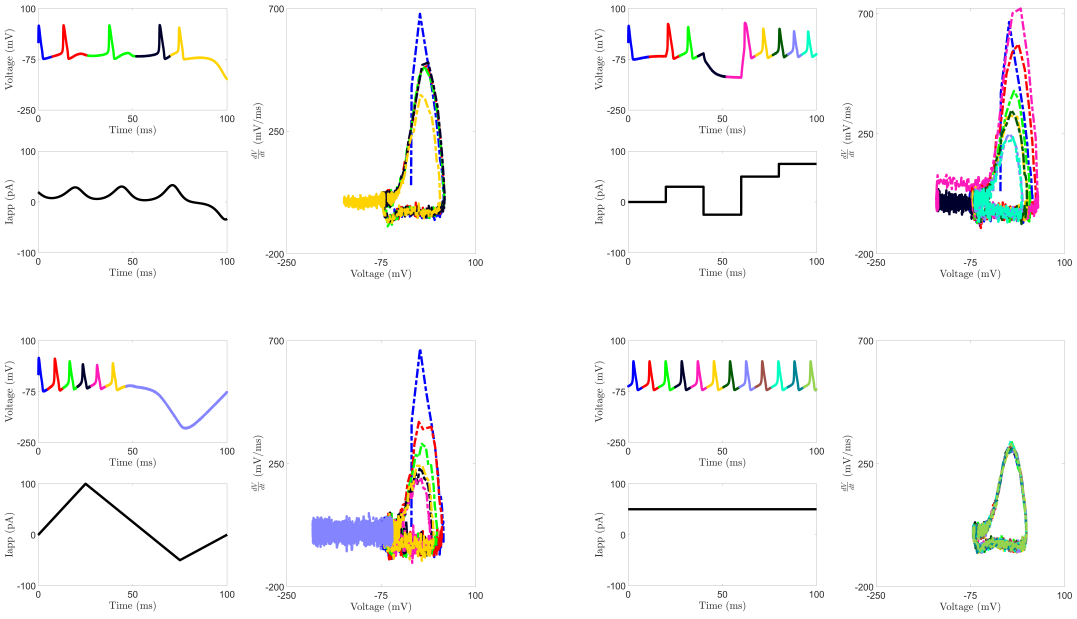


Figure 6.4 Other nominal currents used as comparisons for success at estimating parameters of NaKL model.

Table 6.2 NaKL Estimated Model Parameters from Strong 4D-Var with Nudging for Various Applied Currents

Parameter	Original Value	Aopt	Dopt	L63	Ramp	Steps	Constant
g_{Na}	120.0	119.14	121.25	158.95	116.85	116.04	124.99
g_{K}	20.0	19.69	19.72	17.50	21.06	23.98	27.38
g_{L}	0.3	0.30	0.30	0.30	0.30	0.30	0.28
vm	-40.0	-39.94	-39.75	-39.45	-39.98	-40.10	-40.22
dvm	15.0	14.93	15.25	15.18	15.00	14.67	15.16
τ_{m0}	0.1	0.10	0.10	0.11	0.10	0.10	0.09
τ_{m1}	0.4	0.40	0.41	0.40	0.40	0.43	0.39
vh	-60.0	-59.87	-59.81	-61.89	-59.39	-60.18	-61.46
dvh	-15.0	-14.83	-15.04	-15.68	-14.71	-14.33	-15.85
τ_{h0}	1.0	0.99	1.00	0.94	1.00	1.05	1.04
τ_{h1}	7.0	7.25	7.20	6.90	7.47	5.84	7.34
vn	-55.0	-55.01	-55.28	-56.02	-54.57	-53.22	-52.97
dvn	30.0	30.25	30.15	29.88	30.81	31.58	33.70
τ_{n0}	1.0	0.99	0.98	0.84	1.02	1.10	1.29
τ_{n1}	5.0	4.99	4.87	4.98	4.95	4.95	4.47

CHAPTER 7

DATA ASSIMILATION FOR VOLTAGE-CLAMP RECORDINGS

7.1 Real Time Estimation with Voltage-Clamp Data

Current-clamp experimentation provides access to the voltage time series of a cell. However, there are other data sets which could be utilized to characterize a neural system. For instance, in voltage-clamp protocols, the voltage of the cell is controlled and current response is measured. In dynamic clamp [32], the voltage of the cell is recorded, while, in real-time, fed into a model representative of a current or summation of currents to be injected back into a cell. We propose the following strategy:

1. Accurately and quickly perform parameter estimation of a neuron. The timescale will be in seconds, so that after characterization of the currents, the neuron is still alive and its biophysical properties have not degraded.

2. Use a dynamic clamp with our estimated model and identify if the voltage recorded matches the initially recorded dynamics.

While the goal would be applying this procedure to real data, the starting point, of course, is with simulated data. The following twin experiment is on a simulated voltage-clamp protocol. The model we are using (7.1) is a modified Hodgkin-Huxley model adapted from [122]. Our observed data is the sodium current I_{Na} with both activation and inactivation.

$$\begin{aligned} C\frac{dV}{dt} &= -I_{\text{Na}} - I_{\text{K}} - I_{\text{L}} + I_{app} \qquad (7.1)\\ &= -g_{\text{Na}}m^3h(V - E_{\text{Na}} - g_{\text{K}}n^4(V - E_{\text{K}}) - g_{\text{L}}(V - E_{\text{L}}) + I_{app} \\ \frac{dx}{dt} &= \frac{x_\infty(V) - x}{\tau_x(V)} \qquad (7.2) \end{aligned}$$

with

$$x_\infty = \frac{1}{2}(1 + \tanh((V - vx)/dvx)) \tag{7.3}$$

$$\tau_x = tx0 + tx1(1 - \tanh((V - vxt)/dvxt)^2) \tag{7.4}$$

for $x = m, h, n$.

The voltage-clamp data we are using is displayed in compact form in Figure 7.1. As the voltage is held fixed after a period of hyperpolarization, the current response is recorded. While overlaying the current response across different voltage steps is the typical way voltage-clamp data is presented, the time series form of the data is more useful for filtering and is displayed in Figure 7.2. The data is sampled every .05 ms.

Perhaps the most straightforward and fastest data assimilation scheme at our disposal is an Unscented Kalman Filter. For this twin experiment, the aim is to infer the parameters and estimate the states associated with I_{Na}. The state space then comprises $x = [m, h, g_{\text{Na}}, vm, dvm, tm0, tm1, vh, dvh, th0, th1]'$ as, for simplicity, we impose that $vm = vmt, dvm = dvmt, vh = vht, dvh = dvht$ and E_{Na} is taken as known. The sodium current observation is perturbed with the same level of noise as described in the twin experiments in the preceding sections. Our observational operator, H, is now truly nonlinear, as $H(x_k) = x(3)_k x(1)_k^3 x(2)_k (V_k - E_{\text{Na}})$ where the system is being forced with the voltage data V_k at time t_k. To reiterate, our forward model is (7.2) forced with known voltage V_k at every observation point. The resulting parameters are shown in Table 7.1. For this trial, the initial guesses are taken to be 70% of their true values.

Since the estimated parameters are not necessarily a perfect recovery of the truth, it is useful at least to compare the resulting I_{Na} time courses and, ultimately,

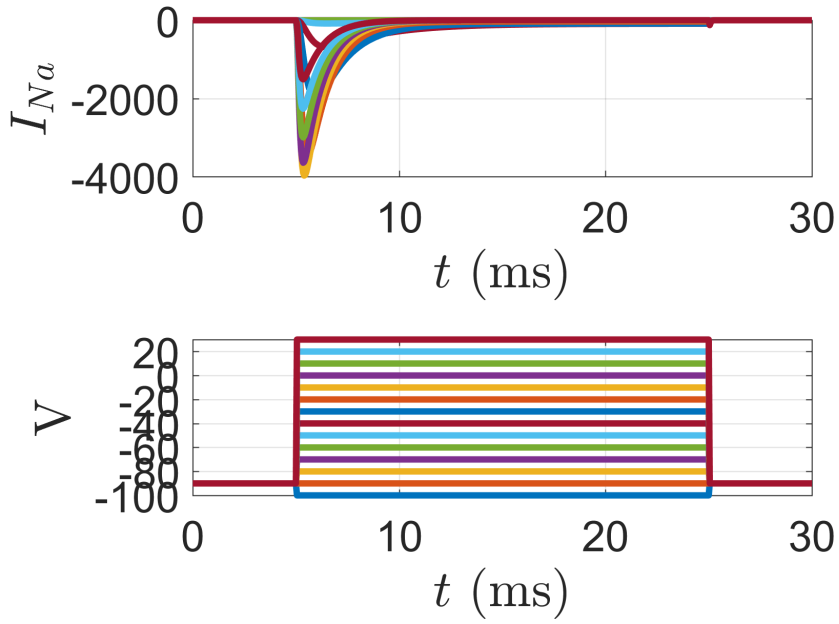


Figure 7.1 Voltage clamp data for I_{Na}.

Table 7.1 Voltage-Clamp UKF Results for I_{Na}

Parameter	True	Estimated	Initial Guess
g_{Na}	120.0	113.511	84.0
vm	-40.0	-39.779	-28.0
dvm	15.0	15.267	10.5
$tm0$	0.1	0.076	0.07
$tm1$	0.4	0.356	0.28
$vmt = vm$	-40.0	-39.779	-28.0
$dvmt = dvm$	15.0	15.267	10.5
vh	-60.0	-61.199	-42.0
dvh	-15.0	-15.936	-10.5
$th0$	1.0	1.033	0.7
$th1$	7.0	7.594	4.9
$vht = vh$	-60.0	-61.199	-42.0
$dvht = dvh$	-15.0	-15.936	-10.5

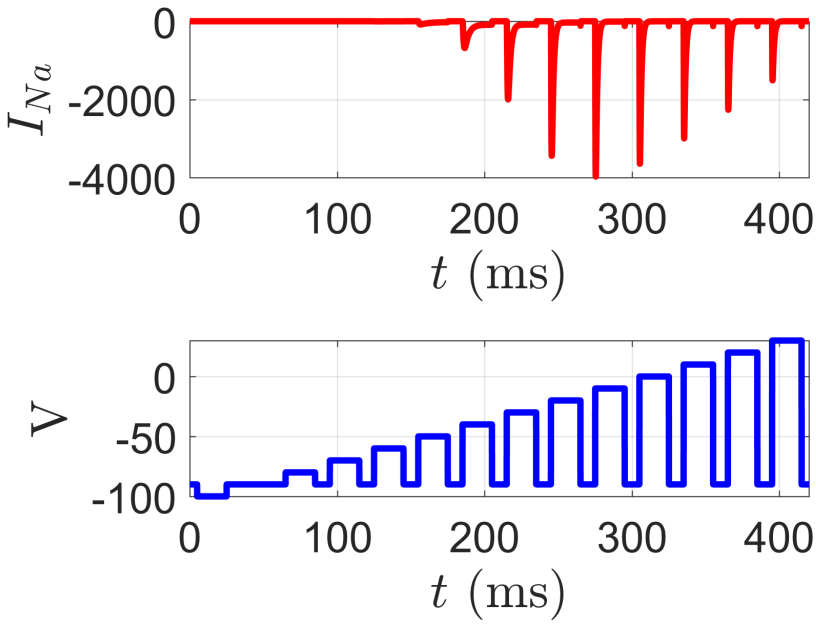


Figure 7.2 Full time course of voltage clamp data from I_{Na}.

simulations with these parameters for the full system Equations (7.1)-(7.2). These are displayed in Figure 7.3 and Figure 7.4.

This appears to be promising given that for these results we are using far fewer time points than in the UKF twin experiments in the previous sections. The total number of time points is 8414, and the run time for the UKF on my quad-core laptop (without any form of parallelization) is around 2 seconds which is of similar order to the data collection time.

7.2 Voltage-Forcing and Decoupling the System

One mechanism for trying to circumvent the irregular voltage problem is to explicitly drive the system through these observations. In DSPE, the dynamics are driven through a coupling to the observations via a control term. A slightly different way, as described by [54, 61, 74], is to insert the observed forcing in the model equations. Namely, we aim to *induce* these irregularities in some capacity or prevent

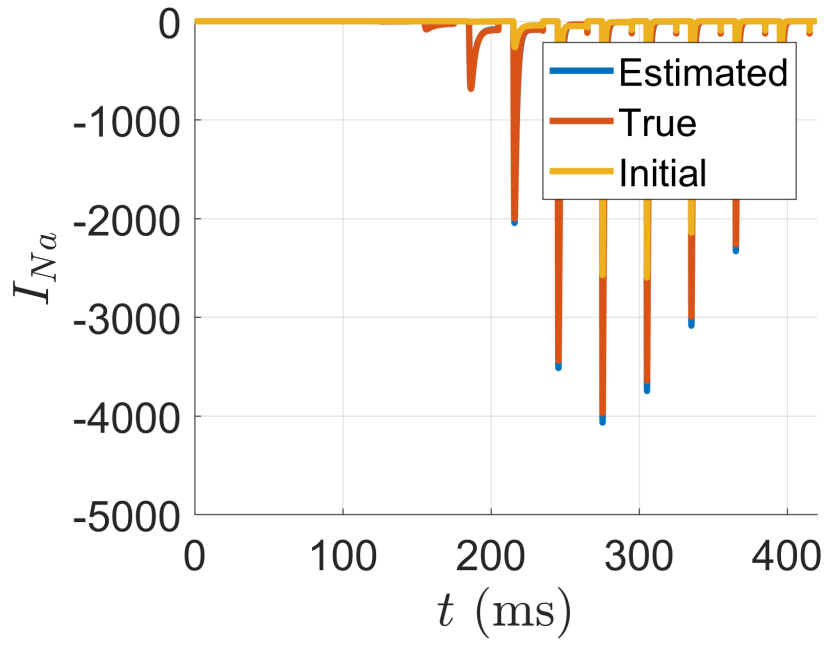


Figure 7.3 Recreation of voltage-clamp data using the estimated and initial parameter sets of Table 7.1 compared to the true data.

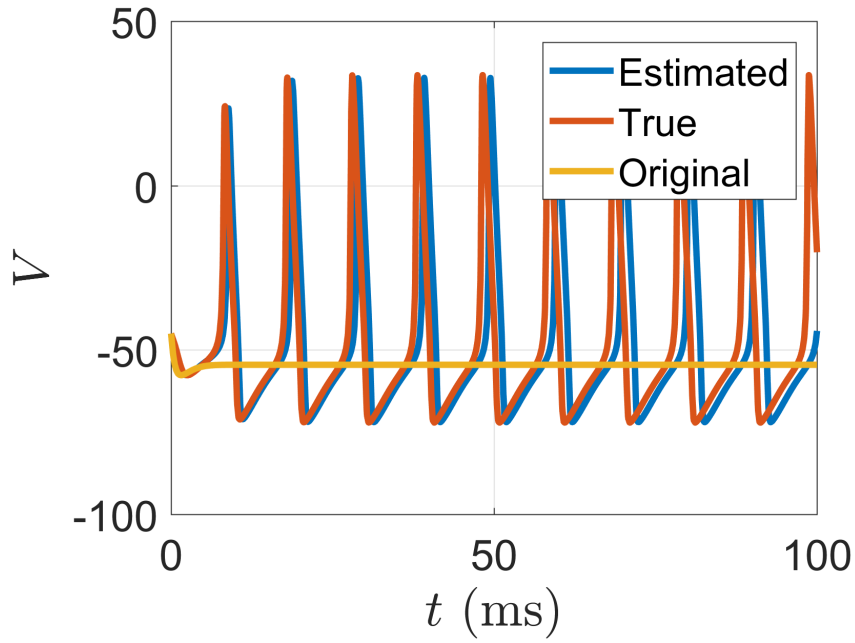


Figure 7.4 Simulation of HH model using the estimated and initial parameter sets of Table 7.1 compared to the true data.

our estimation from penalizing parameter sets which exhibit the appropriate spiking behavior aside from a slight misspecification of spike times.

The basis for this methodology is that we hope our model is an adequate enough approximation to real world dynamics, and that our observed driving voltage has a strong enough influence on the system to drive the other equations. So, we say that should an observed component, V_{obs}, be generated from a model, then it formally can be inserted into the model equations and drive the unobserved variables in the system to their attractor in solution space. An initial step is to show this using CLEs, but from my experience it has been the case for twin experiments at least.

As an example, for the time being assuming continuous observation, we can drive the the Morris-Lecar system with the voltage:

$$\begin{aligned}\frac{dV_{\text{obs}}}{dt} &= I_{app} - g_{\text{L}}(V_{\text{obs}} - E_{\text{L}}) - g_{\text{K}}n(V_{\text{obs}} - E_{\text{K}}) \\ &\quad - g_{\text{Ca}}m_{\infty}(V_{\text{obs}})(V_{\text{obs}} - E_{\text{Ca}}) = f_V(V_{\text{obs}}, n; \theta)\end{aligned} \tag{7.5}$$

$$\frac{dn}{dt} = \phi(n_{\infty}(V_{\text{obs}}) - n)/\tau_n(V_{\text{obs}}) = f_n(V_{\text{obs}}, n; \theta) \tag{7.6}$$

Now, Equation (7.6) is actually decoupled from the overall system, and can be solved analytically using integrating factors.

We arrive at a solution for $n(t)$ conditioned on V_{obs}. One can then plug this value into Equation (7.5) and at this stage consider either

$$\min C(V_{\text{obs}}; \theta) = \int \left(\frac{dV_{\text{obs}}}{dt} - f_V(V_{\text{obs}}, n; \theta) \right)^2 \tag{7.7}$$

after taking a numerical derivative of V_{obs} or

$$\min C(V_{\text{obs}}; \theta) = \int \left(V_{\text{obs}} - \int f_V(V_{\text{obs}}, n; \theta) \right)^2 \tag{7.8}$$

after numerical integration of our vector field forced by our observed voltage. The structure of Equation (7.5) can be harnessed if one is considering estimating maximal conductances. Should the parameters of the model excluding the conductance values be known exactly, the values for the maximal conductances can be solved via linear regression. Namely for $I_a = g_a x^\alpha y^\beta (V - E_a) = g_a J_a$, the problem for estimating conductances in Equation (7.7) becomes:

$$\underbrace{\begin{bmatrix} J_{\mathrm{Na}}(V_{\mathrm{obs}}(t_0)) & J_{\mathrm{K}}(V_{\mathrm{obs}}(t_0)) & J_{\mathrm{L}}(V_{\mathrm{obs}}(t_0)) \\ J_{\mathrm{Na}}(V_{\mathrm{obs}}(t_1)) & J_{\mathrm{K}}(V_{\mathrm{obs}}(t_1)) & J_{\mathrm{L}}(V_{\mathrm{obs}}(t_1)) \\ \vdots & \vdots & \vdots \\ J_{\mathrm{Na}}(V_{\mathrm{obs}}(t_N)) & J_{\mathrm{K}}(V_{\mathrm{obs}}(t_N)) & J_{\mathrm{L}}(V_{\mathrm{obs}}(t_N)) \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} g_{\mathrm{Na}} \\ g_{\mathrm{K}} \\ g_{\mathrm{L}} \end{bmatrix}}_{\mathbf{g}} \approx \underbrace{\begin{bmatrix} \frac{dV_{\mathrm{obs}}}{dt}(t_0) - I_{\mathrm{app}}(t_0) \\ \frac{dV_{\mathrm{obs}}}{dt}(t_1) - I_{\mathrm{app}}(t_1) \\ \vdots \\ \frac{dV_{\mathrm{obs}}}{dt}(t_N) - I_{\mathrm{app}}(t_N) \end{bmatrix}}_{\mathbf{y}}$$

An analogous construction for Equation (7.8) can be made.

In [54], a simplex optimization routine was run over top of Equations (7.8) and (7.7) in an effort to optimize over the remaining hidden parameters. That formulation is shown in Algorithm 1.

Algorithm 1 Synchronization-based estimation

```
procedure OptimizeHHModel(V_obs, M)   ▷ Optimize parameters of model M
    Initialize guess for θ_h
    while ε ≤ TOL do
        Integrate M(V_obs, θ_h)
        Solve for θ_l from regression
        Compute ε from C(V_obs, V_est)
```

The cost function $C(V_{\mathrm{obs}}; \theta)$ would be Equation (7.7) or (7.8). An immediate amendment to this procedure would be some manner of acceptance-rejection algorithm based upon features we may find most important. For instance, simply simulating the model with a candidate parameter set without the driving voltage and

checking if it at least exhibits the same electrical behavior as the observed voltage. Other heuristics include firing rate, known bifurcation structures, spike-width or amplitude, etc. Hopefully, this would guide our procedure towards parameter sets which not only can reconstruct the system dynamics when slaved to the voltage, but can exhibit intrinsic dynamics reminiscent of the observations although perhaps with spike-time matching removed.

If we desired to use an acceptance-rejection method from the branch of Markov Chain Monte Carlo methods, we need to be cautious of the statistics at play. First we would need to construct the necessary likelihood function (or a function proportional to it). One could make an argument that this is a form of smoothing, where any evolution of the dynamic variables at play is due to their initial conditions and the stationary parameter values.

Then after discretization

$$x_{k+1} = \Phi(\theta, n), \quad x_{k+1} \in \mathbb{R}^{L-1} \tag{7.9}$$

$$y_{k+1} = G(\Phi(\theta, n)) + \eta_{k+1}, \quad \eta_{k+1} \in \mathbb{R}^{M} \tag{7.10}$$

For θ being the parameters of interest and possibly the initial conditions of the system, Φ being the mapping of our reduced system, y being V_{obs} or $\frac{V_obs}{dt}$, G representing the observational operator corresponding to $\int f_V(V_{\text{obs}}, n; \theta)$ or $f_V(V_{\text{obs}}, n; \theta)$ respectively, and with $\eta_k \sim \mathcal{N}(0,\, R)$. For the Morris-Lecar system, $L - 1 = 1$ reflecting that the dynamics of the voltage are taken out of the estimation, but used as a forcing and incorporated into the observation function. If using only the voltage or its derivative, $M = 1$, but this formulation considers any integral or derivative of the data and the respective vector fields. Thereby, one could use say both the voltage and its derivative as "observables". The posterior density that we are trying to estimate would be [6]:

$$P(\theta|V_{\text{obs}}) \propto P(\theta) \exp\left\{-\frac{1}{2}\sum_{k=0}^{N} R^{-1}\left(y_k - G(x_{k+1})\right)^2\right\} \tag{7.11}$$

In the strictest sense, this is not our problem, because Equation (7.9) implies that the dynamics are deterministic, and Equation (7.10) implies the observations have noise. But our system is conditioned on the observations, i.e. $x_{k+1} = F(x_k, y_k)$. We can lie to ourselves for convenience, since we lie about lack of correlations between observation points and noise distributions in practice, but there may be a better mathematical way to set up this problem. Additonally, smoothers are primarily concerned with the recovery of initial conditions, but the synchronization of Pecora and Caroll concerns long term behavior. It may be necessary to truncate the time window which we estimate over after some initial transient behavior of the response variables for better performance. Regardless, we would also desire to augment Equation (7.11) with additional information about the heuristics we are primarily concerned with.

However, as a baseline, under these assumptions, one can use a Metropolis Hastings algorithm as shown in Algorithm 2. $\mu \in \mathbb{R}^D \sim \mathcal{U}(-\alpha/2, \alpha/2)$ and $b \in \mathbb{R}^D$ is

Algorithm 2 Synchronization-based Metropolis Hastings

procedure OPTIMIZEHHMODEL(V_{obs}, M) ▷ Optimize parameters of model M
    Initialize guess for θ_0
    for $i = 0$ to N_{ens} **do**
        if $i > 0$, propose: $\theta_i^p = \theta_i + \mu \cdot b$
        Integrate $M(V_{\text{obs}}, \theta_h)$
        Solve for θ_l from regression
        Evaluate $C(\theta_i^p)$
        if $\mathcal{U}(0,1) < \min\left(1, \exp\{-(C(\theta_i^p) - C(\theta_i)\}\right)$ **then**
            $\theta_{i+1} = \theta_i^p$
        else
            $\theta_{i+1} = \theta_i$

a scaling factor determined by the bounds. Statistics can then be calculated from the collection of samples θ_i to find the best estimate and its variance. A version of the

above algorithm for application to 4D-Var has been applied to neural systems [70]. The lower dimensionality of our search space (D, the dimension of the parameters) of our problem may provide great opportunity for different cost functions incorporating appropriate heuristics while having the overall computational complexity being low.

7.2.1 Pushing the Synchronization

Alternatively, one can consider algebraic manipulations of the vector field. For Equation (7.5), for $V_{\text{obs}} \neq E_{\text{K}}$,

$$n_{\text{obs}} \equiv \frac{\frac{dV_{\text{obs}}}{dt} - I_{app} - g_{\text{L}}(V_{\text{obs}} - E_{\text{L}}) - g_{\text{Ca}} m_{\infty}(V_{\text{obs}})(V_{\text{obs}} - E_{\text{Ca}})}{g_{\text{K}}(V_{\text{obs}} - E_{\text{K}})} \tag{7.12}$$

for the fictitious definition of n_{obs}. Certainly now we could try to perhaps generate a cost functional incorporating a minimization of n_{obs} and an n from numerical integration with V_{obs} as a forcing. However, it is not immediately apparent what usefulness this produces.

One manner would be to write another vector field which would be self-consistent in accordance with Equation (7.12):

$$\frac{dn}{dt} = f_n(n, V; \theta) + u_n(n_{\text{obs}} - n) \tag{7.13}$$

Thus, we have reduced our system dimension by 1. We have introduced dependence now on performing a formal numerical derivative, but [27] may provide suitable approximations. For a larger conductance-based model, by incorporating Equation (7.13) ideals in DSPE, we trade estimating the voltage and its control for controls on each of the state components. If Equation (7.12) is used in a filtering framework, we may actually overcome the observability criterion of the system, and

could use an adaptive algorithm to try to recover the process noise associated with the system. Unfortunately, even in that limit, there is no guarantee for a nonlinear system to converge to the global optima of the system [12].

The formulations discussed in this section have yet to be implemented into code; this will be left as future work. We will conclude with some final thoughts in the following chapter.

CHAPTER 8

CONCLUDING THOUGHTS

There are some overarching themes that I am trying to address with this work. *Driving forces* in dynamics, in estimation, and in the behaviors of circadian neurons. *Optimality*, how to characterize it, and how to traverse the complex landscape in a high-dimensional optimization manifold. *Observability*, the incorporation of known geometric features of phase-space, motivating experiments for informed data collection, and the incorporation of the data we have in different ways.

I hope I didn't give off a false impression; even with the amazing capacity afforded to our efforts by data assimilation algorithms, solving these problems, particularly for real data, remains exceedingly hard. If our parameters of interest are not practically identifiable, our hard work can ultimately yield an incomplete or misleading result. Therefore, if there is any takeaway I want to impart the reader of this dissertation, it is to critically analyze one's data, model, estimation algorithm, and own understanding of the underlying mechanisms, and try to extract out the meaningful picture. The automation of data assimilation routines seriously aid, but cannot replace, the insights that the modeling and biological community can provide. Looking forward, data assimilation can complement the growth of new recording technologies for collecting observational data from the brain. The joint collaboration of these automated algorithms with the painstaking work of experimentalists and model developers may help answer many remaining questions about neuronal dynamics and applications beyond.

APPENDIX A

CH2: MATLAB CODE

The following MATLAB code files are provided to implement the Unscented Kalman Filter and weak 4-D Var methods for state and parameter estimation of the Morris-Lecar model in Chapter 2. Thanks to Franz Hamilton, Tyrus Berry, and Tim Sauer for sharing their UKF code which has been modified and presented here.

1. UKFML.m
2. w4DvarML.m
3. w4DvarOutputFun.m
4. generateMLME.m
5. fXaug.m
6. fV.m
7. fn.m
8. ninf.m
9. taun.m
10. minf.m

```
function UKFML()
%% UKFML
% Adapted from code by Franz Hamilton, Tyrus Berry, and Tim Sauer (George Mason University).
% Produces example figure for initializing with HOPF regime parameters for
% estimating parameters associated with SNIC regime.
%%% Preliminary setup
obsNoise=.01; s=4; rng(s);
T = 200000; dt=.1; dT=.1; nn=fix(dT/dt);
```

```
%%% Define parameters
truep = [.067,4,12,17.4,120, 84, 60,8,2, 1.2,18,20,100];
guessp= [.04,4,2,30,120, 84, 60,8,2, 1.2,18,20,100];
SimName='tSNICgHOPF';
EstSet={'V','n','phi','gCa','V3','V4','gK','gL','V1','V2'};
dx = length(EstSet);
icovmat=.001*eye(dx); % prior, P_{{xx}_0}
lambda=5;
%%% Generate Data
x00=[4 .14]';
parameters=ones(T+1,1)*truep; time=0:dT:dT*T;
x0=generateMLME(x00,T,parameters,dT); x0=generateMLME(x0(:,end),T,parameters,dT);
signalStdev=std(x0(1,:));
ML_Noisy=x0(1,:)+obsNoise*(signalStdev.*randn(size(x0(1,:))));
obs = ML_Noisy';
%%% Define initial guesses for state components
istate = zeros(dx,1)*10;
istate(2)=0; istate(1)=obs(1); istate(3:end)=[guessp(1:4) guessp(8:11)];
%%% Define best guesses for Q,R
Vrange=max(obs) min(obs);
Qscale=1e 7; Q = Qscale*diag([Vrange, 1, abs(istate(3:end))']);
R = (obsNoise*signalStdev)^2; if R==0, R=1e 4; end
redistEnsemble=1; % Can make 0 and compare performance
%%% Run Filter
[stateEstimate,variance]= ...
    UKF_Estimate(time,istate,obs',@MLstatefctModEuler,@MLstateobsfct,Q,R,...
    parameters(1:end,:),lambda,icovmat,dt,nn,redistEnsemble);
%%% Make parameter estimation plot
for i =3:dx
    figure(11)
    plot(time,stateEstimate(i,:),'linewidth',3)
    hold on
end
paramLegend={'\phi', 'g_{Ca}', 'V_3', 'V_4', 'g_{K}',...
    'g_l', 'V_1','V_2'};
xlim([0 time(end)])
legend(paramLegend)
set(gca,'FontSize',16)
xlabel('Time (ms)','FontSize',25)
end
function [fstate,fcovmat,innovation] = UKF_STEP(itime,ftime,istate,...
    icovmat,fobs,F,H,Q,R,p,lambda,t,dt,nn,redistEnsemble)
%%%Propagate from one observation time step to the next
% itime   :     the current time
% ftime   :     the final time for this step of the filter
% istate  :     an Nx1 vector containing the state at itime
```

```
55 % icovmat :      an NxN matrix containing the covariance matrix at itime
56 % fobs     :      an Mx1 vector which contains the observation at ftime
57 % F        :      a function handle which computes the vector field as a
58 %                  function of the state and time
59 % H        :      a function handle which computes the observation as a
60 %                  function of the state and time
61 % Q        :      the NxN covariance matrix for the dynamical noise
62 % R        :       the MxM covariance matrix for the observation noise
63 % p        :      fixed parameter set for the model. initial guesses for
64 %                  parameters to be estimated are included in istate.
65 % lambda   :      algorithmic parameter to spread or shrink ensemble
66 %
67 N = length(istate);
68 M = length(fobs);
69 %%% Fix up the covariance matrices
70 icovmat = (icovmat+icovmat')/2;
71 [u,l]=svd(Q); Q = u*l*u';
72 [u,l]=svd(R); R = u*l*u';
73 %%% Initialize Filter
74 xhat_a = istate;
75 Pxx = icovmat;
76 %%% Build the matrix square root and ensemble
77 [U,S,V] = svd(Pxx);
78 squareRoot = U*diag(sqrt(diag(S)))*U';
79 xEnsemble = zeros(N,2*N+1);
80 xEnsemble(:,1) = xhat_a;
81 xEnsemble(:,2:N+1) = repmat(xhat_a,1,N)+sqrt(N+lambda)*squareRoot;
82 xEnsemble(:,N+2:2*N+1) = repmat(xhat_a,1,N) sqrt(N+lambda)*squareRoot;
83 W = [lambda/(N+lambda) 1/(2*(N+lambda))*ones(1,2*N)]; %weights
84 xUpdatedEnsemble = F(xEnsemble,p,t,dt,nn); % forward mapping
85 yEnsemble = H(xUpdatedEnsemble); % observational mapping
86 yhat_b = sum(repmat(W,M,1).*yEnsemble,2);
87 observationDifference = yEnsemble repmat(yhat_b,1,2*N+1);
88 Pb = (W.*observationDifference)*observationDifference';
89
90 %%% Calculate the background state estimate
91 xhat_b = sum(repmat(W,N,1).*xUpdatedEnsemble,2);
92 %%% Calculate the background covariance estimate
93 ensembleDifference = xUpdatedEnsemble repmat(xhat_b,1,2*N+1);
94 FPa = (W.*ensembleDifference)*ensembleDifference';
95 Pxx_f = FPa+Q;
96 Pxx_f = min(Pxx_f,2); %Restrict uncertainty, can remove
97
98 if redistEnsemble
99     %%% Redistribute sigma points around background mean
100    %%% Build the matrix square root and ensemble
```

```
101         [U,S,V] = svd(Pxx_f);
102         squareRoot = U*diag(sqrt(diag(S)))*U';
103         xUpdatedEnsemble = zeros(N,2*N+1);
104         xUpdatedEnsemble(:,1) = xhat_b;
105         xUpdatedEnsemble(:,2:N+1) = repmat(xhat_b,1,N)+sqrt(N+lambda)*squareRoot;
106         xUpdatedEnsemble(:,N+2:2*N+1) = repmat(xhat_b,1,N) sqrt(N+lambda)*squareRoot;
107         %%% Realculate the background state estimate
108         xhat_b = sum(repmat(W,N,1).*xUpdatedEnsemble,2);
109         %%% Recalculate the background covariance estimate
110         ensembleDifference = xUpdatedEnsemble repmat(xhat_b,1,2*N+1);
111         Pxx_f = (W.*ensembleDifference)*ensembleDifference';
112         %%% Calculate predicted measurements
113         yEnsemble = H(xUpdatedEnsemble);
114         yhat_b = sum(repmat(W,M,1).*yEnsemble,2);
115     observationDifference = yEnsemble repmat(yhat_b,1,2*N+1);
116     Pb = ((W.*observationDifference)*observationDifference');
117     end
118     %%% Construct the Kalman gain
119     Pyy = Pb+R;
120     Pxy = (W.*ensembleDifference)*observationDifference';
121     K = Pxy/Pyy;
122     innovation = (fobs yhat_b);
123     %%% Analysis (estimation) step
124     xhat_a = xhat_b+K*innovation;
125     Pxx = Pxx_f   K*Pyy*K'; % same as Pxx_f  KPxy'
126     fcovmat = Pxx;
127     fstate = xhat_a;
128     end
129
130     function r=MLstatefctModEuler(x,p,j,dt,nn)
131     xnl=x;
132     for n=1:nn
133         k1=dt*fXaug(0,xnl,p(j,:));
134         k2=dt*fXaug(0,xnl+k1,p(j+1,:));
135         xnl=xnl+k1./2+k2./2;
136     end
137     r=xnl;
138     end
139
140     function Hx=MLstateobsfct(x)
141     Hx=x(1,:);
142     end
143     function [stateEstimate,varEstimate] = UKF_Estimate(time,istate,obs,F,H,...
144         Qguess,Rguess,p,lambda,icovmat,dt,nn,redistEnsemble)
145     N = length(istate); M = size(obs,1); T = size(obs,2);
146     Q = Qguess; R = Rguess;
```

```
stateEstimate = zeros(N,T); stateEstimate(:,1)=istate;
varEstimate = zeros(N,T); varEstimate(:,1)=diag(icovmat);
innovseq = zeros(M,T);
for t = 1:T 1
    %%% One step of UKF (t_k > t_{k+1})
    [istate,icovmat,innovation] = UKF_STEP(time(t),time(t+1),istate,...
        icovmat,obs(:,t+1),F,H,Q,R,p,lambda,t,dt,nn,redistEnsemble);
    stateEstimate(:,t+1) = istate;
    varEstimate(:,t+1) = diag(icovmat);
    innovseq(:,t+1) = innovation;
end
end
```

```
function w4DvarML()
%% Weak 4d Variational for Morris Lecar
% Produces example figure for initializing with HOPF regime parameters for
% estimating parameters associated with SNIC regime.
%%% Preliminary setup
obsNoise=.01; sd=4;rng(sd);
T=2000; dT=.1; dt=.1; nn = fix(dT/dt);
truep = [.067,4,12,17.4,120,84,60,8,2,1.2,18,20,100];
guessp= [.04,4,2,30,120,84,60,8,2,1.2,18,20,100];
SimName='tSNICgHOPF';
EstSet={'V','n','phi','gCa','V3','V4','gK','gL','V1','V2'};
dx = length(EstSet); Ntheta=8;
%%% Define weightings for cost function (inv(Q), inv(R))
measStdDev=1;% observational noise variance is StDev^2
Rinv=inv(measStdDev^2);
% The dynamic range of v is larger than n by orders of magnitude. Need to
% weight deviations appropriately. Also need to choose ratio of measurement
% to model noise.
weightScales=10.^[4:5]; weightIndex = 7;
weightScale=weightScales(weightIndex);
Qinv=weightScale*Rinv*diag([1 10000]);
% Presumes each time series component of the system has equal weighting
% For simplicity of implementation, Q,R taken as diagonal matrix
%%% Generate Data
x00=[4 .14]';
Ttotal=5*T;
parameters=[ones(Ttotal+1,1)*truep];
x0=generateMLME(x00,Ttotal,parameters,dT);
x0=generateMLME(x0(:,end),Ttotal,parameters,dT);
parametersGuess=[ones(Ttotal+1,1)*guessp];
%%% guesses
% Set initial guesses.
Vtruth=x0(1,:); ntruth=x0(2,:);
signalStdev=std(Vtruth);
```

```
y=Vtruth+ obsNoise*(signalStdev.*randn(size(Vtruth)));
% Guess n to be integration of guessed model forced with observations
x02=generateMLMEVforced(x00(2),Ttotal,parametersGuess,dT,y);
nguess=x02(1,:);
% Used more data than needed, pick subset to estimate over
windowInds=2*T+1:3*T+1;
Vobswindow=y(windowInds);
nguesswindow=nguess(windowInds);
InitGuess=zeros(2,T+1);
InitGuess(1,:)=Vobswindow;
InitGuess(2,:)=nguesswindow;
InitGuess=[InitGuess(1,:)'; InitGuess(2,:)'];
pEstInds=[1 2 3 4 8 9 10 11];
InitGuess =[InitGuess; guessp(pEstInds)'];
%%%
% InitGuess is a stack of guesses for system components
% InitGuess = [V_1 V_2... V_{J+1} n_1 n_2 ... n_{J+1}, phi, gCa, V3, V4, gK , gL , V1, V2]';
% Next set the bounds.
LB = zeros(size(InitGuess)); RB=LB;
LB(1:T+1)= 100; LB(T+2:end Ntheta)=0;
RB(1:T+1)=100;  RB(T+2:end Ntheta)=1;
LB(end Ntheta+1:end)= [0 0  20 .1 0 0  10  0.1]';
RB(end Ntheta+1:end)= [1 10 20 35 10 5 20 35]';
%%% Construct an output file/function to save data stream
ClockTime=fix(clock);
ClockString=[num2str(ClockTime(1)),'_',num2str(ClockTime(2)),'_',num2str(ClockTime(3))];
FileName=['w4dvar',SimName,'obs',num2str(obsNoise),...
    'wscale',num2str(weightIndex),'T',ClockString];
FileName=strrep(FileName,'.','pt'); % same output during iteration cycle
myOutputFun =@(x,optimValues,state)w4dvarOutputFun(x,optimValues,state,[FileName,'.txt']);
%%% Options can be changed based on performance, preferences, etc.
options1 = optimoptions('fmincon','Display','iter',...
    'MaxFunctionEvaluations',4e8,'MaxIterations',1e5,...
    'HessianApproximation','lbfgs','InitBarrierParam',1e 3 ,...% limited memory hessian approx
    'OutputFcn',@(x,optimValues,state)myOutputFun(x,optimValues,state));
%%% Run optimization
[xest,fval,exitflag,output] = fmincon(@(u)w4dvarobjfun(u,Vobswindow,Qinv,...
    Rinv,@MLstatefctw4dvarModEuler,T,truep,Ntheta,dt,nn,dx),InitGuess,...
    [],[],[],[],LB,RB,[],options1);
%save(FileName) %(uncomment to save all data at end in .mat)
end
function [fout]=w4dvarobjfun(xbold,y,Qinv,Rinv,F,T,truep,Ntheta,dt,nn,dx)
% The following code accepts as arguments:
%
% xbold : vector of N*(T+1) + P dimensions, for N being total number of
% states in dynamical model (2 for Morris Lecar)
```

```
81  %
82  % y : observation. Currently set to operate on an observation operator
83  % which is a simple projection onto the first system component i.e.
84  % H_{1,1} = 1, H_{j,k} = 0 otherwise.
85  %
86  % Qinv : inverse covariance matrix associated with model error.
87  % inv(smaller values of covariance) imply larger weights towards those
88  % terms in cost functional. Taken to be a diagonal matrix.
89  %
90  % Rinv : inverse covariance matrix associated with measurement
91  % noise
92  %
93  % F : discrete map representation of model. for our purposes, F is a
94  % numerical integration scheme applied to a model in question e.g.
95  % Morris Lecar. Currently, using modified Euler set up for autonomous system.
96  %
97  % T : T+1 points in time
98  %
99  % truep : passing parameters to model. for those which will be
100 % determined from the scheme, their truep counterparts will not be
101 % used, but their component in xbold will. These are at the end of xbold.
102 theta=xbold(end Ntheta+1:end);
103 v=xbold(1:T+1);
104 n=xbold(T+2:end Ntheta);
105 u = [reshape(v,1,T+1); reshape(n,1,T+1)];
106 MeasurementMismatch=0; ModelMismatch=0;
107 %%% Compute F(x_k; \theta) for all k
108 fwdmapj=F(u,theta,truep,dt,nn,dx);
109 %%% Model term
110 for l=1:2
111     ModelMismatch=ModelMismatch+...
112         1/2*sum(Qinv(l,l)*((u(l,2:end) fwdmapj(l,1:end 1))).^2);
113 end
114 %%% Measurement term
115 MeasurementMismatch=MeasurementMismatch+1/2*sum(Rinv*((y(1,:) u(1,:))).^2);
116 fout=MeasurementMismatch+ModelMismatch;
117 end
118 function r=MLstatefctw4dvarModEuler(x,theta,p,dt,nn,dx)
119 Ntheta=length(theta);
120 xnl=zeros(dx,size(x,2));
121 xnl(1:2,:)=x;
122 xnl(3:end,:)=repmat(reshape(theta,Ntheta,1),1,size(x,2));
123 for n=1:nn
124     k1=dt*fXaug(0,xnl,p);
125     k2=dt*fXaug(0,xnl+k1,p);
126     r=xnl+k1./2 + k2./2;
```

```
end
end
function x=generateMLMEVforced(x0,T,p,dt,V)
x(:,1)=x0;
for j=1:T
k1=dt*fXVforced(0,x(:,j),p(j,:),V(j));
k2=dt*fXVforced(0,x(:,j)+k1,p(j+1,:),V(j+1));
x(:,j+1)=x(:,j)+k1./2+k2./2;
end
end
function dy = fXVforced(t,y,p,V)
phi=p(1);
V3=p(3);
V4=p(4);
n = y(1,:);
dy=zeros(size(y));
dy(1,:)=fn(V,n,phi,V3,V4);
end
```

```
function stop =w4dvarOutputFun(x,optimValues,state,FileName)
% Writes out text file which, after  each iteration,
% will display :
% [IterationNumber CostFunctionEval ParameterEstimates]
funEval=optimValues.fval;
iterNumber=optimValues.iteration;
    importantDetails=x(end-7:end);
    importantDetails = [iterNumber funEval reshape(importantDetails,1,8)];
    dlmwrite(FileName,importantDetails,'-append',...
        'delimiter',' ','roffset',1)
stop=false;
end
```

```
function x=generateMLME(x0,T,p,dt)
%%% Modified Euler for Morris Lecar with fixed time step
x(:,1)=reshape(x0,1,2);
for j=1:T
    k1=dt*fX(0,x(:,j),p(j,:));
    k2=dt*fX(0,x(:,j)+k1,p(j+1,:));
    x(:,j+1)=x(:,j)+k1./2+k2./2;
end
    function dy = fX(t,y,p)
        phi=p(1);
        Gca=p(2);
        V3=p(3);
        V4=p(4);
        Eca=p(5);
        Ek=p(6);
```

```
            El=p(7);
            Gk=p(8);
            Gl=p(9);
            V1=p(10);
            V2=p(11);
            Cm=p(12);
            Iapp=p(13);
            V = y(1,:);
            n = y(2,:);
            dy=zeros(size(y));
            dy(1,:)= fV(V,n,Iapp,Gca,Gk,Gl,Eca,Ek,El,Cm,V1,V2);
            dy(2,:)=fn(V,n,phi,V3,V4);
        end
end
```

```
function dy = fXaug(t,y,p)
% Vector field for augmented state (states + parameters)
Eca=p(5);
Ek=p(6);
El=p(7);
Cm=p(12);
Iapp=p(13);
V = y(1,:);
n = y(2,:);
phi=y(3,:);
Gca=y(4,:);
Gk=y(7,:);
Gl=y(8,:);
V3=y(5,:);
V4=y(6,:);
V1=y(9,:);
V2=y(10,:);
dy=zeros(size(y)); % parameters' derivatives are 0
dy(1,:)= fV(V,n,Iapp,Gca,Gk,Gl,Eca,Ek,El,Cm,V1,V2);
dy(2,:)=fn(V,n,phi,V3,V4);
end
```

```
function dVdt= fV(V,n,Iapp,Gca,Gk,Gl,Eca,Ek,El,Cm,V1,V2)
dVdt= (1/Cm)*(Iapp  Gl.*(V El) Gk.*n.*(V Ek) Gca.*minf(V,V1,V2).*(V Eca));
end
```

```
function dndt = fn(V,n,phi,V3,V4)
dndt = phi.*(ninf(V,V3,V4) n)./taun(V,V3,V4);
end
```

```
function nout = ninf(V,V3,V4)
nout = .5*(1+tanh((V V3)./V4));
end
```

```
function tau = taun(V,V3,V4)
tau = 1./(cosh((V V3)./(2*V4)));
end
```

```
function mout = minf(V,V1,V2)
mout = .5*(1+tanh((V V1)./V2));
end
```

APPENDIX B

CH5: MATLAB CODE

The following MATLAB code files are provided to implement identifiability analysis of the Morris-Lecar model in Chapter 5. For STRIKE-GOLDD, options_ml1.m are the original options. After running the analysis once, several parameters are recognized to be identifiable. After rerunning, fixing these as identifiable in options_ml2.m, the identifiability is achieved.

An example of a symbolic construction of the observability rank criterion in CasADi, but evaluated numerically, is given by run_morris_lecar_identifiability_casadi.m with helper function morris_lecar_identifiability_casadi.m

1. STRIKE-GOLDD Output
2. z_create_MorrisLecar_model.m
3. options_ml1.m
4. options_ml2.m
5. morris_lecar_identifiability_casadi.m
6. run_morris_lecar_identifiability_casadi.m

```
>> edit z_create_MorrisLecar_model
>> options

ans =

'MorrisLecar'

>> STRIKE_GOLDD


```

```
>>> StrIkE GOLDD TOOLBOX


 Analyzing identifiability of MorrisLecar ...

>>> The model contains:
 2 state(s):
 V
 n

 1 output(s):
 V

 1 input(s):
 Iapp

 8 parameter(s):
 Gl
 Gk
 Gca
 phi
 V1
 V2
 V3
 V4

>>> Building the observability identifiability matrix requires at least 9 Lie derivatives
 Calculating derivatives: 1 2 3 4 5 6 7 8
 The model will be decomposed.


>>> Finding an optimal submodel including state number 1 ...
 Init. point: Bestf:  0.125       CPUTime: 0.156250
 NFunEvals: 7   Bestf:  0.166667       CPUTime: 21.015625
 NFunEvals: 145   Bestf:  0.166667       CPUTime: 303.531250
 *************************
 END OF THE OPTIMIZATION
 Best solution value              0.166667
 Decision vector
 1
 0
 CPU time                 303.531
 Number of function evaluations          145
```

```
>>> Analysing identifiability of a submodel containing:
1 states:
V
1 outputs:
V
1 inputs:
Iapp
6 parameters:
matrix([[Gl], [Gk], [Gca], [V1], [V2], [n]])

>>> Building the observability identifiability matrix requires at least 6 Lie derivatives
Calculating derivatives: 1 2 3 4 5 6
>>> Observability Identifiability matrix calculated with 6 Lie derivatives
(calculated in 6.676544e 01 seconds)
>>> Rank = 6 (calculated in 2.331481e+00 seconds)
>>> Observability Identifiability matrix calculated with 7 Lie derivatives
(calculated in 1.953944e+00 seconds)
>>> Rank = 6 (calculated in 1.202557e+01 seconds)
The submodel is structurally unidentifiable
=> Parameter Gl is structurally identifiable
=> We cannot decide about identifiability of parameter Gk at the moment
=> Parameter Gca is structurally identifiable
=> Parameter V1 is structurally identifiable
=> Parameter V2 is structurally identifiable

>>> Finding an optimal submodel including state number 2 ...
Init. point: Bestf: 0.125       CPUTime: 0.093750
NFunEvals: 135  Bestf: 0.125       CPUTime: 303.781250
*************************
END OF THE OPTIMIZATION
Best solution value              0.125
Decision vector
1
1
CPU time                 303.781
Number of function evaluations          135



>>> Analysing identifiability of a submodel containing:
2 states:
matrix([[V], [n]])
1 outputs:
V
1 inputs:
Iapp
```

```
8 parameters:
matrix([[Gl], [Gk], [Gca], [phi], [V1], [V2], [V3], [V4]])

>>> Building the observability identifiability matrix requires at least 9 Lie derivatives
Calculating derivatives: 1 2 3 4 5 6 7 8
Rank = 9 (calculated in 1.625555e+03 seconds)
Parameter Gl has already been classified as identifiable.
=> We cannot decide about identifiability of parameter Gk at the moment
Parameter Gca has already been classified as identifiable.
=> We cannot decide about identifiability of parameter phi at the moment
Parameter V1 has already been classified as identifiable.
Parameter V2 has already been classified as identifiable.
=> We cannot decide about identifiability of parameter V3 at the moment
=> We cannot decide about identifiability of parameter V4 at the moment
=> We cannot decide about state n at the moment


>>> RESULTS SUMMARY:


>>> These parameters are identifiable:
matrix([[Gca, Gl, V1, V2]])
>>> These states are directly measured:
V
Total execution time: 8.898812e+03

STRIKE_GOLDD



>>> StrIkE GOLDD TOOLBOX


Analyzing identifiability of MorrisLecar ...

>>> The model contains:
2 state(s):
V
n

1 output(s):
V

1 input(s):
Iapp

```

```
4 parameter(s):
Gk
phi
V3
V4

>>> Building the observability identifiability matrix requires at least 5 Lie derivatives
Calculating derivatives: 1 2 3 4 5
>>> Observability Identifiability matrix built with 5 Lie derivatives
(calculated in 1.270037e+00 seconds)
>>> Calculating rank...
Rank = 6 (calculated in 1.181469e+00 seconds)


>>> RESULTS SUMMARY:


>>> These parameters are identifiable:
matrix([[Gk, V3, V4, phi]])
>>> The model is observable:
All its states are observable.

Total execution time: 4.813718e+00
```

```
function [] = z_create_MorrisLecar_model()

%clear all;

syms V n ...
      Gl Gk Gca Iapp...
      phi V1 V2 V3 V4
% states:
x     = [V; n];

% outputs:
h     = [V];

%  input:
u     = Iapp;
El =  60;
Eca = 120;
Ek =  84;
Cm = 20;
% parameters:
%p     = [a; b; AA; sigma; alpha; beta; gamma; delta];
p = [Gl Gk Gca...
      phi V1 V2 V3 V4].';
```

```
% dynamic equations:
f     = [(1/Cm)*(Iapp   Gl*(V El) Gk*n*(V Ek) Gca*minf(V,V1,V2)*(V Eca));
           phi*(ninf(V,V3,V4) n)/taun(V,V3,V4)];

% initial conditions:
ics   =[4 .14].';

% which initial conditions are known:
known_ics = [1,1];

save('MorrisLecar','x','h','u','p','f','ics','known_ics');


end
function mout = minf(V,V1,V2)
         mout = .5*(1+tanh((V V1)/V2));
end

function tau = taun(V,V3,V4)
         tau = 1/(cosh((V V3)/(2*V4)));
end
function nout = ninf(V,V3,V4)
         nout = .5*(1+tanh((V V3)/V4));
end
```

```
%=========================================================================
% THE USER CAN DEFINE THE PROBLEM AND SET OPTIONS IN THE FOLLOWING LINES:
%=========================================================================

function [modelname,paths,opts,submodels,prev_ident_pars] = options()
%%% (1) MODEL:
%modelname = 'MAPK';
%modelname = 'Goodwin';
%modelname= 'MorrisLecarReduced'
%modelname='simpleBCVA';
%modelname = 'MorrisLecarReduced';
% modelname='NaKLred';
%modelname='NaKLext';
%modelname='NaKL_conductances';
%modelname='NaKL';
modelname='MorrisLecar';
%%% (2) PATHS:
% laptop
paths.meigo      = 'C:\Users\Matt\Documents\meigo64\MEIGO';
paths.models     = strcat(pwd,filesep,'models');
paths.results    = strcat(pwd,filesep,'results');
paths.functions = strcat(pwd,filesep,'functions');
```

```
23
24 %%% (3) IDENTIFIABILITY OPTIONS:
25 opts.numeric    = 0;       % calculate rank numerically (= 1) or symbolically (= 0)
26 opts.replaceICs = 0;       % replace states with known initial conditions (= 1) or use generic
       values (= 0) when calculating rank
27 opts.checkObser = 1;       % check observability of states / identifiability of initial conditions
       (1 = yes; 0 = no).
28 opts.findcombos = 0;       % try to find identifiable combinations? (1 = yes; 0 = no).
29 opts.unidentif  = 0;       % use method to try to establish unidentifiability instead of
       identifiability, when using decomposition.
30 opts.forcedecomp= 0;       % always decompose model (1 = yes; 0 = no).
31 opts.decomp     = 1;       % decompose model if the whole model is too large (1 = yes; 0 = no:
       instead, calculate rank with few Lie derivatives).
32 opts.decomp_user= 0;       % when decomposing model, use submodels specified by the user (= 1) or
       found by optimization (= 0).
33 opts.maxLietime = 30;      % max. time allowed for calculating 1 Lie derivative.
34 opts.maxOpttime = 300;     % max. time allowed for every optimization (if optimization based
       decomposition is used).
35 opts.maxstates  = 3;       % max. number of states in the submodels (if optimization based
       decomposition is used).
36 opts.nnzDerIn   = [10];    % number of nonzero derivatives of the inputs (specify them in one
       column per input).
37
38 %%% (4) User defined submodels for decomposition (may be left = []):
39 submodels = [];
40 %%% Submodels are specified as a vector of states, as e.g.:
41 %          submodels     = [];
42 %          submodels{1}  = [1 2];
43 %          submodels{2}  = [1 3];
44
45 %%% (5) Parameters already classified as identifiable may be entered below.
46
47 % syms Gk vq dvq tq0 tq1
48 % prev_ident_pars = [Gk vq dvq tq0 tq1];
49 prev_ident_pars='';
50 %syms G1
51 %prev_ident_pars=[G1];
52 %%% They must first be declared as symbolic variables. For example:
53 %         syms mRNA0
54 %         prev_ident_pars = mRNA0;
55 end
```

```
1 %=========================================================================
2 % THE USER CAN DEFINE THE PROBLEM AND SET OPTIONS IN THE FOLLOWING LINES:
3 %=========================================================================
4
5 function [modelname,paths,opts,submodels,prev_ident_pars] = options()
```

```
%%% (1) MODEL:
%modelname = 'MAPK';
%modelname = 'Goodwin';
%modelname= 'MorrisLecarReduced'
%modelname='simpleBCVA';
%modelname = 'MorrisLecarReduced';
% modelname='NaKLred';
%modelname='NaKLext';
%modelname='NaKL_conductances';
%modelname='NaKL';
modelname='MorrisLecar';
%%% (2) PATHS:
% laptop
paths.meigo      = 'C:\Users\Matt\Documents\meigo64\MEIGO';
paths.models     = strcat(pwd,filesep,'models');
paths.results    = strcat(pwd,filesep,'results');
paths.functions = strcat(pwd,filesep,'functions');

%%% (3) IDENTIFIABILITY OPTIONS:
opts.numeric     = 0;       % calculate rank numerically (= 1) or symbolically (= 0)
opts.replaceICs = 0;       % replace states with known initial conditions (= 1) or use generic
    values (= 0) when calculating rank
opts.checkObser = 1;       % check observability of states / identifiability of initial conditions
     (1 = yes; 0 = no).
opts.findcombos = 0;       % try to find identifiable combinations? (1 = yes; 0 = no).
opts.unidentif  = 0;       % use method to try to establish unidentifiability instead of
    identifiability, when using decomposition.
opts.forcedecomp= 0;       % always decompose model (1 = yes; 0 = no).
opts.decomp      = 1;       % decompose model if the whole model is too large (1 = yes; 0 = no:
    instead, calculate rank with few Lie derivatives).
opts.decomp_user= 0;       % when decomposing model, use submodels specified by the user (= 1) or
    found by optimization (= 0).
opts.maxLietime = 30;      % max. time allowed for calculating 1 Lie derivative.
opts.maxOpttime = 300;     % max. time allowed for every optimization (if optimization based
    decomposition is used).
opts.maxstates  = 3;       % max. number of states in the submodels (if optimization based
    decomposition is used).
opts.nnzDerIn    = [10];    % number of nonzero derivatives of the inputs (specify them in one
    column per input).

%%% (4) User defined submodels for decomposition (may be left = []):
submodels = [];
%%% Submodels are specified as a vector of states, as e.g.:
%          submodels     = [];
%          submodels{1}  = [1 2];
%          submodels{2}  = [1 3];
```

```
44
45 %%% (5) Parameters already classified as identifiable may be entered below.
46
47 % syms Gk vq dvq tq0 tq1
48 % prev_ident_pars = [Gk vq dvq tq0 tq1];
49 syms Gca Gl V1 V2
50 prev_ident_pars=[Gca, Gl, V1, V2];
51 %syms Gl
52 %prev_ident_pars=[Gl];
53 %%% They must first be declared as symbolic variables. For example:
54 %         syms mRNA0
55 %         prev_ident_pars = mRNA0;
56 end
```

```
1  function [f,fVout,fnout, xaug] = morris_lecar_identifiability_casadi()
2  %import casadi.*
3  truep = [.04,4,2,30,120,84,60,8,2,1.2,18,20,100]; %hopf
4  % truep = [.067,4,12,17.4,120,84,60,8,2,1.2,18,20,100]; % snlc , Iapp 100 fine for this
5  % %truep = [.23,4,12,17.4,120,84,60,8,2,1.2,18,20,36]; % homoclinic, choose y0 inside b.o.a e.g
       . y0=[4 .14] for Iapp= 36
6  import casadi.*
7  V= MX.sym('V');
8  n = MX.sym('n');
9  %x = [V,n]';
10 %C = MX.sym('C');
11 %Ena = MX.sym('Ena');
12 %Ek =   MX.sym('Ek');
13 %Eca= MX.sym('Eca');
14 %Eleak    = MX.sym('Eleak');
15 %Gna= MX.sym('Gna');
16 Gk= MX.sym('Gk');
17 Gca = MX.sym('Gca');
18 Gleak =  MX.sym('Gleak');
19 phi = MX.sym('phi');
20 V1= MX.sym('V1');
21 V2 = MX.sym('V2');
22 V3 = MX.sym('V3');
23 V4 = MX.sym('V4');
24        %  phi=p(1);
25      %    Gca=p(2);
26      %    V3=p(3);
27       %   V4=p(4);
28           Eca=truep(5);
29           Ek=truep(6);
30           El=truep(7);
31           %Gk=truep(8);
32           %Gl=p(9);
```

```
33          % V1=p(10);
34          % V2=p(11);
35           Cm=truep(12);
36           Iapp=truep(13);
37  fVout = fV(V,n,Iapp,Gca,Gk,Gleak,Eca,Ek,El,Cm,V1,V2);
38  fnout =fn(V,n,phi,V3,V4);
39           xdot = [ fVout ...
40               fnout];
41           xaug= [V,n,phi Gca V3 V4 Gk Gleak V1 V2];
42  f = Function('f', {xaug}, {xdot});
43  end
44  function dVdt= fV(V,n,Iapp,Gca,Gk,Gl,Eca,Ek,El,Cm,V1,V2)
45  dVdt= (1/Cm)*(Iapp  Gl*(V El) Gk*n*(V Ek) Gca*minf(V,V1,V2)*(V Eca));
46  end
47  function nout = ninf(V,V3,V4)
48  nout = .5*(1+tanh((V V3)./V4));
49  end
50  function tau = taun(V,V3,V4)
51  tau = 1./(cosh((V V3)./(2*V4)));
52  end
53  function mout = minf(V,V1,V2)
54  mout = .5*(1+tanh((V V1)/V2));
55  end
56  function dndt = fn(V,n,phi,V3,V4)
57  dndt = phi.*(ninf(V,V3,V4) n)./taun(V,V3,V4);
58  end
```

```
1  import casadi.*
2  [f,fVout,fnout, xaug] = morris_lecar_identifiability_casadi();
3  g=xaug(1);
4  %O = MX.sym('O',length(xaug),length(xaug));
5  %O = DM(length(xaug),length(xaug));
6  O = MX(length(xaug),length(xaug));
7  Lfg = MX.sym('Lfg',length(xaug),1);
8  % f(x) = [fv(x) fn(x) 0 0 0 0 0 0 0 0]';
9  lfgtemp = fVout;
10 Lfg(1) = lfgtemp; % dg/dx = [1  0...];, dg/dx f(x) = fv(x);
11 otemp = jacobian(xaug,xaug(1));
12 O(1,:) = otemp;
13 otemp=jacobian(lfgtemp,xaug);
14 for i =2:length(xaug)
15     O(i,:) = otemp;
16     lfgtemp = otemp(1)*fVout + otemp(2)*fnout;
17     otemp = jacobian(lfgtemp,xaug);
18 end
19
20 fO =  Function('fO', {xaug}, {O});
```

```
21
22  truep = [.04,4,2,30,120,84,60,8,2,1.2,18,20,100]; %hopf
23  true_xpaug=[40 .175 truep(1:4) truep(8:11)]';
24  % primes
25  %plist=primes(100);
26  %true_xpaug=[40 .175 plist(1:8)];
27  fO_true_xp=fO(true_xpaug);
28  full_fO_true_xp=full(fO_true_xp)
29  rankO=rank(full_fO_true_xp,1e12)
30
31  %% make table
32  input.data=full_fO_true_xp;
33  input.makeCompleteLatexDocument=0;
34  input.tableCaption = 'Matrix of ORC for Morris Lecar Model Evaluated for Hopf Parameters';
35  input.dataFormat = {'%.0e'};
36  % p = {'C'; 'Ena'; 'Ek';'Eca';  'Gna'; 'Gk'; 'Gca';'Gleak_na';'Gleak_k';
37  %      'vm_na'; 'dvm_na';
38  %      'vh_na'; 'dvh_na'; 'th0_na'; 'th1_na'; 'vht_na'; 'dvht_na'; ...
39  %      'vn'; 'dvn'; 'tn0'; 'tn1'; 'vnt'; 'dvnt'; ...
40  %      'vm_ca'; 'dvm_ca'; 'tm0_ca'; 'tm1_ca'; 'vmt_ca'; 'dvmt_ca';
41  %      'vh_ca'; 'dvh_ca'; 'th0_ca'; 'th1_ca'; 'vht_ca'; 'dvht_ca';};
42  % p=[Csoma, Caxon, Ena, Ek, Enaleak, Gna, Gk, Gnaleak, Gaxon, vm, dvm, tm0, tm1, vh, dvh, th0, th1
        , vn, dvn, tn0, tn1];
43  platexformat = ...
44  {'$V$','$n$','$\phi$', '$G_{\textrm{Ca}}$', '$V3$','$V4$', '$G_{\textrm{K}}$',...
45  '$G_{\textrm{leak}}$' '$V1$' '$V2$'};
46    % Set column labels (use empty string or no label):
47    %input.tableColLabels = ;
48    % Set row labels (use empty string for no label):
49     input.tableColLabels = platexformat';
50
51       latex = latexTable(input);
52
53
54  %%
55  %% test against known failure
56  import casadi.*
57
58  [f,fVout,fnout, xaug] = morris_lecar_identifiability_casadi_bad();
59  g=xaug(1);
60  %O = MX.sym('O',length(xaug),length(xaug));
61  %O = DM(length(xaug),length(xaug));
62  O = MX(length(xaug),length(xaug));
63  Lfg = MX.sym('Lfg',length(xaug),1);
64  % f(x) = [fv(x) fn(x) 0 0 0 0 0 0 0 0]';
65  lfgtemp = fVout;
```

```
Lfg(1) = lfgtemp; % dg/dx = [1  0...];, dg/dx f(x) = fv(x);
otemp = jacobian(xaug,xaug(1));
O(1,:) = otemp;
otemp=jacobian(lfgtemp,xaug);
for i =2:length(xaug)
    O(i,:) = otemp;
    lfgtemp = otemp(1)*fVout + otemp(2)*fnout;
    otemp = jacobian(lfgtemp,xaug);
end

fO =  Function('fO', {xaug}, {O});

truep = [.04,4,2,30,120, 84, 60,8,2, 1.2,18,20,100]; %hopf
%true_xpaug=[40 .175 truep(1:4) truep(8:11) truep(11)+1]';
true_xpaug=[40 .175 plist(1:9)];

fO_true_xp=fO(true_xpaug);
full_fO_true_xp=full(fO_true_xp);
rankObad=rank(full_fO_true_xp,1e 12)
% remove last column, check if rank changed.
full_f0_true_xp_red =full(fO_true_xp(:,1:end 1))
rankObad_red = rank(full_f0_true_xp_red,1e 12)

full_f0_true_xp_red2 =full(fO_true_xp(:,1:end 2))
rankObad_red2 = rank(full_f0_true_xp_red2,1e 12)

%% KNOWN failure in state variables

import casadi.*
[f,fVout,fnout, xaug] = morris_lecar_identifiability_casadi_bad2();
g=xaug(1);
%O = MX.sym('O',length(xaug),length(xaug));
%O = DM(length(xaug),length(xaug));
O = MX(length(xaug),length(xaug));
Lfg = MX.sym('Lfg',length(xaug),1);
% f(x) = [fv(x) fn(x) 0 0 0 0 0 0 0 0]';
lfgtemp = fVout;
Lfg(1) = lfgtemp; % dg/dx = [1  0...];, dg/dx f(x) = fv(x);
otemp = jacobian(xaug,xaug(1));
O(1,:) = otemp;
otemp=jacobian(lfgtemp,xaug);
for i =2:length(xaug)
    O(i,:) = otemp;
    lfgtemp = otemp(1)*fVout + otemp(2)*fnout;
    otemp = jacobian(lfgtemp,xaug);
end
```

```
112
113 fO =  Function('fO', {xaug}, {O});
114
115 truep = [.04,4,2,30,120, 84, 60,8,2, 1.2,18,20,100]; %hopf
116 true_xpaug=[40 0 truep(1:4) truep(8:11)]';
117
118 fO_true_xp=fO(true_xpaug);
119 full_fO_true_xp=full(fO_true_xp)
120 rankOn0=rank(full_fO_true_xp,1e 12)
121
122 %%
123 %% test against known failure, gleak = gl*V5
124 import casadi.*
125
126 [f,fVout,fnout, xaug] = morris_lecar_identifiability_casadi_bad3();
127 g=xaug(1);
128 %O = MX.sym('O',length(xaug),length(xaug));
129 %O = DM(length(xaug),length(xaug));
130 O = MX(length(xaug),length(xaug));
131 Lfg = MX.sym('Lfg',length(xaug),1);
132 % f(x) = [fv(x) fn(x) 0 0 0 0 0 0 0 0]';
133 lfgtemp = fVout;
134 Lfg(1) = lfgtemp; % dg/dx = [1 0...];, dg/dx f(x) = fv(x);
135 otemp = jacobian(xaug,xaug(1));
136 O(1,:) = otemp;
137 otemp=jacobian(lfgtemp,xaug);
138 for i =2:length(xaug)
139     O(i,:) = otemp;
140     lfgtemp = otemp(1)*fVout + otemp(2)*fnout;
141     otemp = jacobian(lfgtemp,xaug);
142 end
143
144 fO =  Function('fO', {xaug}, {O});
145
146 truep = [.04,4,2,30,120, 84, 60,8,2, 1.2,18,20,100]; %hopf
147 true_xpaug=[40 .175 truep(1:4) truep(8:11) 2]';
148
149 fO_true_xp=fO(true_xpaug);
150 full_fO_true_xp=full(fO_true_xp);
151 rankObad=rank(full_fO_true_xp)
152 % remove last column, check if rank changed.
153 full_f0_true_xp_red =full(fO_true_xp(:,1:end 1))
154 rankObad_red = rank(full_f0_true_xp_red,1e 12)
155
156
157
```

```
158 %% Bursting
159 import casadi.*
160 CSE=1;
161 [f,xdot, xaug] = morris_lecar_bursting_identifiability_casadi(CSE);
162 %truep=[.04,4.4,2,30,120, 84, 60,8,2, 1.2,18,20,.75,.005,.02,120];
163 truep=[.23,4,12,17.4,120, 84, 60,8,2, 1.2,18,20,.25,.005,.02,45];
164
165 Nstates=3;
166 g=xaug(1);
167 %O = MX.sym('O',length(xaug),length(xaug));
168 %O = DM(length(xaug),length(xaug));
169 O = SX(length(xaug),length(xaug));
170 Lfg = SX.sym('Lfg',length(xaug),1);
171 % f(x) = [fv(x) fn(x) 0 0 0 0 0 0 0 0]';
172 lfgtemp = xdot(1);
173 Lfg(1) = lfgtemp; % dg/dx = [1 0...];, dg/dx f(x) = fv(x);
174 %otemp = jacobian(xaug,xaug(1));
175 otemp = zeros(length(xaug),1);
176 otemp(1)=1;
177 O(1,:) = otemp;
178 otemp=jacobian(lfgtemp,xaug);
179 for i =2:length(xaug)
180     O(i,:) = otemp;
181     %lfgtemp = otemp(1)*fVout + otemp(2)*fnout;
182     lfgtemp = dot(otemp(1:Nstates),xdot);
183     otemp = jacobian(lfgtemp,xaug);
184     %otemp = jtimes(otemp(1:4),xaug,xout');
185     i
186 end
187
188 fO =  Function('fO', {xaug}, {O});
189
190 %truep = [.04,4,2,30,120, 84, 60,8,2, 1.2,18,20,100]; %hopf
191 %true_xpaug=[40 .175 truep(1:4) truep(8:11)]';
192 %truep=paramboundsSpecsdotText(:,3);
193 %true_xpaug=[   38.9338     0.3083     0.2684     0.5845 truep']; % good for Iapp=50;
194 truex=[   4.951947538278363   0.062735819057212   0.694917947518991];
195 switch CSE
196     case 1
197         true_xpaug=[truex truep(1:4) truep(8:11) truep(13)];
198     case 2
199         true_xpaug=[truex truep(1:4) truep(8:11) truep(13:14)];
200     case 3
201         true_xpaug=[truex truep(1:4) truep(8:11) truep([13 15])];
202
203     case 4
```

```
204           true_xpaug=[truex truep(1:4) truep(8:11) truep([13 14 15])];
205
206  end
207  %truep_xaug = [
208  fO_true_xp=fO(true_xpaug);
209  full_fO_true_xp=full(fO_true_xp)
210  rankO=rank(full_fO_true_xp,1e 12)
```

REFERENCES

[1] Henry D.I. Abarbanel. *Predicting the Future: Completing Models of Observed Complex Systems*. Understanding Complex Systems. Springer New York, New York, NY, 2013.

[2] Henry D.I. Abarbanel, Daniel R. Creveling, Reza Farsian, and Mark Kostuk. Dynamical state and parameter estimation. *SIAM Journal on Applied Dynamical Systems*, 8(4):1341–1381, January 2009.

[3] Henry D.I. Abarbanel, Sasha Shirman, Daniel Breen, Nirag Kadakia, Daniel Rey, Eve Armstrong, and Daniel Margoliash. A unifying view of synchronization for data assimilation in complex nonlinear networks. *Chaos*, 27(12), 2017.

[4] Luis A. Aguirre, Leonardo L. Portes, and Christophe Letellier. Observability and synchronization of neuron models. *Chaos*, 27(10), 2017.

[5] Joel A.E. Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, March 2019.

[6] Amit Apte. An introduction to data assimilation. In Susmita Sarkar, Uma Basu, and Soumen De, editors, *Springer Proceedings in Mathematics & Statistics*, number iii, pages 31–42. Springer India, Kolkata, India, 2014.

[7] Mark Asch, Marc Bocquet, and Maelle Nodet. *Data Assimilation: Methods, Algorithms, and Applications*. Fundamentals of Algorithms. SIAM, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2016.

[8] Sara J. Aton, Christopher S. Colwell, Anthony J. Harmar, James Waschek, and Erik D. Herzog. Vasoactive intestinal polypeptide mediates circadian rhythmicity and synchrony in mammalian clock neurons. *Nature Neuroscience*, 8(4):476–483, April 2005.

[9] Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.

[10] Stephen Becker. L-bfgs-b-c, https://github.com/stephenbeckr/l-bfgs-b-c, 2018.

[11] Mino D.C. Belle, Casey O. Diekman, Daniel B. Forger, and Hugh D. Piggins. Daily electrical silencing in the mammalian circadian clock. *Science*, 326(5950):281–284, 2009.

[12] Tyrus Berry and Timothy Sauer. Adaptive ensemble kalman filtering of non-linear systems. *Tellus A: Dynamic Meteorology and Oceanography*, 65(1):20331, December 2013.

[13] John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, March 1998.

[14] John T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. SIAM, Philadelphia, PA, second edition, 2010.

[15] Clair A. Booth, Jonathan Witton, Jakub Nowacki, Krasimira Tsaneva-Atanasova, Matthew W. Jones, Andrew D. Randall, and Jonathan T. Brown. Altered intrinsic pyramidal neuron properties and pathway- specific synaptic dysfunction underlie aberrant hippocampal network function in a mouse model of tauopathy. *Journal of Neuroscience*, 36(2):350–363, 2016.

[16] Yona Bouskila and F. Edward Dudek. A rapidly activating type of outward rectifier k+ current and a-current in rat suprachiasmatic nucleus neurones. *The Journal of Physiology*, 488(2):339–350, 1995.

[17] Ted Brookings, Marie. L. Goeritz, and Eve Marder. Automatic parameter estimation of multicompartmental neuron models via minimization of trace error with control adjustment. *Journal of Neurophysiology*, 112(9):2332–2348, 2014.

[18] Denis Burdakov, Haris Alexopoulos, Angela Vincent, and Frances M. Ashcroft. Low-voltage-activated a-current controls the firing dynamics of mouse hypothalamic orexin neurons. *European Journal of Neuroscience*, 20(12):3281–3285, 2004.

[19] Adrian Bürger, Dimitris Kouzoupis, Angelika Altmann-Dieses, and Moritz Diehl. A schur complement method for optimum experimental design in the presence of process noise. *IFAC-PapersOnLine*, 50(1):14118–14124, 2017.

[20] Robert J. Butera, John Rinzel, and Jeffrey C. Smith. Models of respiratory rhythm generation in the pre-botzinger complex. i. bursting pacemaker neurons. *Journal of Neurophysiology*, 82(1):382–397, 1999.

[21] Richard H. Byrd, Jean Charles Gilbert, and Jorge Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming, Series B*, 89(1):149–185, 2000.

[22] Richard H. Byrd, Mary E. Hribar, and Jorge Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, January 1999.

[23] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, September 1995.

[24] James A. Carton and Benjamin S. Giese. A reanalysis of ocean climate using simple ocean data assimilation (soda). *Monthly Weather Review*, 136(8):2999–3017, 2008.

[25] Joshua Chang. *Flipping Biological Switches: Solving for Optimal Control*. Dissertation, University of Massachusetts Medical School, Worcester, MA, 2015.

[26] Joshua Chang and David Paydarfar. Optimizing stimulus waveforms for electroceuticals. *Biological Cybernetics*, 2018.

[27] Rick Chartrand. Numerical differentiation of noisy, nonsmooth data. *International Scholarly Research Notices Applied Mathematics*, 2011:1–11, 2011.

[28] Oana Teodora Chis, Julio R. Banga, and Eva Balsa-Canto. Structural identifiability of systems biology models: A critical comparison of methods. *PLoS ONE*, 6(11), 2011.

[29] Choon Kiat Sim and Daniel B. Forger. Modeling the electrophysiology of suprachiasmatic nucleus neurons. *Journal of Biological Rhythms*, 22(5):445–453, October 2007.

[30] Christopher S. Colwell. Linking neural activity and molecular oscillations in the scn. *Nature Reviews Neuroscience*, 12(10):553–569, September 2011.

[31] Daniel R. Creveling. *Parameter and state estimation in nonlinear dynamical system*. PhD thesis, University of California, San Diego, CA, 2008.

[32] Astrid Cudmore. Dynamic clamp. *Scholarpedia*, 6(5):1470, 2011.

[33] Jonathan Currie and David I Wilson. Opti: Lowering the barrier between open source optimizers and the industrial matlab user. In Nick Sahinidis and Jose Pinto, editors, *Foundations of Computer-Aided Process Operations*, Savannah, Georgia, USA, 2012.

[34] Marcel T.G. De Jeu and Cyriel M.A. Pennartz. Functional characterization of the h-current in scn neurons in subjective day and night: A whole-cell patch-clamp study in acutely prepared brain slices. *Brain Research*, 767(1):72–80, 1997.

[35] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[36] Timothy Delsole and Xiaosong Yang. State and parameter estimation in stochastic dynamical models. *Physica D: Nonlinear Phenomena*, 239(18):1781–1788, 2010.

[37] Casey O. Diekman, Mino D C Belle, Robert P. Irwin, Charles N. Allen, Hugh D. Piggins, and Daniel B. Forger. Causes and consequences of hyperexcitation in central clock neurons. *PLoS Computational Biology*, 9(8):e1003196, August 2013.

[38] Casey O. Diekman, Peter J. Thomas, and Christopher G. Wilson. Experimental validation of a closed-loop respiratory control model using dynamic clamp. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, volume 2018-July, pages 5273–5276. IEEE, July 2018.

[39] Susanne Ditlevsen, Samson, and Adeline. Estimation in the partially observed stochastic morris-lecar neuronal model with particle filter and stochastic approximation methods. *Annals of Applied Statistics*, 8(2):674–702, June 2014.

[40] Huagui Duan and Yiming Long. *Calculus of Variations*. Dover Books on Mathematics. Dover Publications, 2008.

[41] Heinz W. Engl, Christoph Flamm, Philipp Kügler, James Lu, Stefan Müller, and Peter Schuster. Inverse problems in systems biology. *Inverse Problems*, 25(12), 2009.

[42] G. Bard Ermentrout and David H. Terman. *Mathematical Foundations of Neuroscience*, volume 35 of *Interdisciplinary Applied Mathematics*. Springer New York, New York, NY, 2010.

[43] Geir Evensen. *Data Assimilation*. Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, 2009.

[44] Geir Evensen. The ensemble kalman filter for combined state and parameter estimation. *IEEE Control Systems Magazine*, 29(3):83–104, June 2009.

[45] Daniel B. Forger, David Paydarfar, and John R. Clay. Optimal stimulus shapes for neuronal excitation. *PLoS Computational Biology*, 7(7):1–9, 2011.

[46] David M. Fox, Hua An Tseng, Tomasz G. Smolinski, Horacio G. Rotstein, and Farzan Nadim. Mechanisms of generation of membrane potential resonance in a neuron with multiple resonant ionic currents. *PLoS Computational Biology*, 13(6):1–30, 2017.

[47] Joshua H. Goldwyn, Bradley R. Slabe, Joseph B. Travers, and David Terman. Gain control with a-type potassium current: Iaas a switch between divisive and subtractive inhibition. *PLoS Computational Biology*, 14(7):1–23, 2018.

[48] Wander Almodovar Golfetto and Sandro da Silva Fernandes. A review of gradient algorithms for numerical computation of optimal trajectories. *Journal of Aerospace Technology and Management*, 4(2):131–143, 2012.

[49] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F Radar and Signal Processing*, 140(2):107, 1993.

[50] Steven J. Greybush, R. John Wilson, Ross N. Hoffman, Matthew J. Hoffman, Takemasa Miyoshi, Kayo Ide, Timothy McConnochie, and Eugenia Kalnay. Ensemble kalman filter data assimilation of thermal emission spectrometer temperature retrievals into a mars gcm. *Journal of Geophysical Research E: Planets*, 117(11):1–17, 2012.

[51] Toshiyuki Hamada, Michael C. Antle, and Rae Silver. Temporal and spatial expression patterns of canonical clock genes and clock-controlled genes in the suprachiasmatic nucleus. *European Journal of Neuroscience*, 19(7):1741–1748, April 2004.

[52] Franz Hamilton, Tyrus Berry, Nathalia Peixoto, and Timothy Sauer. Real-time tracking of neuronal network structure using data assimilation. *Physical Review E*, 88(5):052715, 2013.

[53] Jenna R.M. Harvey, Amber E Plante, and Andrea L. Meredith. Ion channels controlling circadian rhythms in suprachiasmatic nucleus excitability. *Physiological Reviews*, page physrev.00027.2019, March 2020.

[54] D. Haufler, F. Morin, J. C. Lacaille, and F. K. Skinner. Parameter estimation in single-compartment neuron models using a synchronization-based method. *Neurocomputing*, 70(10-12):1605–1610, 2007.

[55] Robert Hermann and Arthur J Krener. Nonlinear controllability and observability that the quotient inherits a system which has the. *Transactions on Automatic Control*, (5), 1977.

[56] Alan L. Hodgkin and Andrew F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Bulletin of Mathematical Biology*, 52(1-2):25–71, January 1990.

[57] M. J. Hoffman, N. S. LaVigne, S. T. Scorse, F. H. Fenton, and E. M. Cherry. Reconstructing three-dimensional reentrant cardiac electrical wave dynamics using data assimilation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 26(1):013107, 2016.

[58] Hoon Hong, Alexey Ovchinnikov, Gleb Pogudin, and Chee Yap. Global identifiability of differential models. pages 1–28, 2018.

[59] Hoon Hong, Alexey Ovchinnikov, Gleb Pogudin, and Chee Yap. Sian: software for structural identifiability analysis of ode models. *Bioinformatics*, 35(16):2873–2874, August 2019.

[60] Brian R. Hunt, Eric J. Kostelich, and Istvan Szunyogh. Efficient data assimilation for spatiotemporal chaos: A local ensemble transform kalman filter. *Physica D: Nonlinear Phenomena*, 230(1-2):112–126, 2007.

[61] Q. J. M. Huys. Efficient estimation of detailed single-neuron models. *Journal of Neurophysiology*, 96(2):872–890, 2006.

[62] Quentin J.M. Huys and Liam Paninski. Smoothing of, and parameter estimation from, noisy biophysical recordings. *PLoS Computational Biology*, 5(5), 2009.

[63] Eugene M Izhikevich. *Dynamical Systems in Neuroscience*. MIT Press, Cambridge, MA, 2007.

[64] Alexander C. Jackson, Gui Lan Yao, and Bruce P. Bean. Mechanism of spontaneous firing in dorsomedial suprachiasmatic nucleus neurons. *Journal of Neuroscience*, 24(37):7985–7998, 2004.

[65] Kenneth A. Johnson and Roger S. Goody. The original michaelis constant: Translation of the 1913 michaelis–menten paper. *Biochemistry*, 50(39):8264–8269, October 2011.

[66] Simon J. Julier and Jeffrey K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, March 2004.

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

[68] Rudolf Emil Kalman. Contributions to the theory of optimal control. *Boletín de la Sociedad Matemática Mexicana*, 5(2):102–119, 1960.

[69] Matthew Kelly. An introduction to trajectory optimization: How to do your own direct collocation. *SIAM Review*, 59(4):849–904, 2017.

[70] Mark Kostuk, Bryan A. Toth, C. Daniel Meliza, Daniel Margoliash, and Henry D.I. I Abarbanel. Dynamical estimation of neuron and network properties ii: Path integral monte carlo methods. *Biological Cybernetics*, 106(3):155–167, 2012.

[71] Milad Lankarany, W. P. Zhu, and M. N.S. Swamy. Joint estimation of states and parameters of hodgkin-huxley neuronal model using kalman filtering. *Neurocomputing*, 136:289–299, 2014.

[72] Pierre Simon Laplace. Memoir on the probability of the causes of events. *Statistical Science*, 1(3):364–378, August 1986.

[73] Kody Law, Andrew Stuart, and Konstantinos Zygalakis. *Data Assimilation: A Mathematical Introduction*, volume 62 of *Texts in Applied Mathematics*. Springer International Publishing, Cham, Switzerland, 2015.

[74] Nathan F. Lepora, Paul G. Overton, and Kevin Gurney. Efficient fitting of conductance-based model neurons from somatic current clamp. *Journal of Computational Neuroscience*, 32(1):1–24, 2012.

[75] Jane Liu and Mike West. *Combined Parameter and State Estimation in Simulation-Based Filtering*, pages 197–223. Springer New York, New York, NY, 2001.

[76] James Lu, Heinz W. Engl, and Peter Schuster. Inverse bifurcation analysis: Application to simple gene systems. *Algorithms for Molecular Biology*, 1(1):1–16, 2006.

[77] Raman K. Mehra. Optimal input signals for parameter estimation in dynamic systems - survey and new results. *IEEE Transactions on Automatic Control*, 19(6):753–768, 1974.

[78] Raman K. Mehra. Choice of input signals. In *Trends and progress in system identification*, pages 305–366. 1981.

[79] 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, 2014.

[80] Liang Meng, Mark A. Kramer, Steven J. Middleton, Miles A. Whittington, and Uri T. Eden. A unified approach to linking experimental, statistical and computational analysis of spike train data. *PLoS ONE*, 9(1), 2014.

[81] Larry P. Morin and C.N. Allen. The circadian visual system, 2005. *Brain Research Reviews*, 51(1):1–60, June 2006.

[82] Catherine Morris and Harold Lecar. Voltage oscillations in the barnacle giant muscle fiber. *Biophysical Journal*, 35(1):193–213, 1981.

[83] Matthew J. Moye and Casey O. Diekman. Data assimilation methods for neuronal state and parameter estimation. *The Journal of Mathematical Neuroscience*, 8(1):11, December 2018.

[84] N. Nahi and G. Napjus. Design of optimal probing signals for vector parameter estimation. In *IEEE Conference on Decision and Control*, pages 162–168. IEEE, December 1971.

[85] Wataru Nakamura. Differential response of period 1 expression within the suprachiasmatic nucleus. *Journal of Neuroscience*, 25(23):5481–5487, June 2005.

[86] Alain Nogaret, C. Daniel Meliza, Daniel Margoliash, and Henry D.I. Abarbanel. Automatic construction of predictive neuron models through large scale assimilation of electrophysiological data. *Scientific Reports*, 6(August):1–14, 2016.

[87] Christian Haargaard Olsen, Johnny T. Ottesen, Ralph C. Smith, and Mette S. Olufsen. Parameter subset selection techniques for problems in mathematical biology. *Biological Cybernetics*, 113(1-2):121–138, 2019.

[88] Mette S. Olufsen and Johnny T. Ottesen. A practical approach to parameter estimation applied to model predicting heart rate regulation. *Journal of Mathematical Biology*, 67(1):39–68, July 2013.

[89] Seon Ki Park and Liang Xu. *Data assimilation for atmospheric, oceanic and hydrologic applications (Vol. II)*, volume II. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[90] R.L. Payne. *Optimal Experiment Design for Dynamic System Identification*. PhD thesis, Imperial College of Science and Technology, University of London, London, UK, 1974.

[91] Louis M. Pecora and Thomas L. Carroll. Synchronization in chaotic systems. *Physical Review Letters*, 64(8):821–824, 1990.

[92] Louis M. Pecora and Thomas L. Carroll. Synchronization of chaotic systems. *Chaos*, 25(9):097611, September 2015.

[93] Rolf H. Reichle. Data assimilation methods in the earth sciences. *Advances in Water Resources*, 31(11):1411–1418, 2008.

[94] Daniel Rey. *Chaos, observability and symplectic structure in optimal estimation*. PhD thesis, University of California, San Diego, CA, 2017.

[95] John Rinzel and G. Bard Ermentrout. Analysis of neural excitability and oscillations. In C Koch and I Segev, editors, *Methods in neuronal modeling*, pages 251–292. MIT Press, Cambridge, MA, 2 edition, 1988.

[96] Maria Rodriguez-Fernandez. Optimal experimental design based on global sensitivity analysis. *Decision and Control*, pages 1–6, 2007.

[97] Cyrille Rossant. Automatic fitting of spiking neuron models to electrophysiological recordings. *Frontiers in Neuroinformatics*, 4(March):1–10, 2010.

[98] Horacio G. Rotstein, Motolani Olarinre, and Jorge Golowasch. Dynamic compensation mechanism gives rise to period and duty-cycle level sets in oscillatory neuronal models. *Journal of Neurophysiology*, 116(5):2431–2452, 2016.

[99] Steven J. Schiff. *Neural Control Engineering: The Emerging Intersection Between Control Theory and Neuroscience*. Computational neuroscience. MIT Press, 2012.

[100] Desiree M. Schumann, Howard M. Cooper, Margaretha D. Hofmeyr, and Nigel C. Bennett. Circadian rhythm of locomotor activity in the four-striped field mouse, rhabdomys pumilio: A diurnal african rodent. *Physiology and Behavior*, 85(3):231–239, 2005.

[101] Madineh Sedigh-Sarvestani, Steven J. Schiff, and Bruce J. Gluckman. Reconstructing mammalian sleep dynamics with data assimilation. *PLoS Computational Biology*, 8(11), 2012.

[102] Shelby Stanhope, Jon E. Rubin, and David Swigon. Identifiability of linear and linear-in-parameters dynamical systems from a single trajectory. *SIAM Journal on Applied Dynamical Systems*, 13(4):1792–1815, 2014.

[103] Shelby Stanhope, Jon E. Rubin, and David Swigon. Robustness of solutions of the inverse problem for linear dynamical systems with uncertain data. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):572–597, January 2017.

[104] Joseph H. Tien and John Guckenheimer. Parameter estimation for bursting neural models. *Journal of Computational Neuroscience*, 24(3):358–373, 2008.

[105] Bryan A. Toth. *Computational Methods for Parameter Estimation in Nonlinear Models*. PhD thesis, University of California, San Diego, CA, 2011.

[106] Bryan A. Toth, Mark Kostuk, C. Daniel Meliza, Daniel Margoliash, and Henry D.I. I Abarbanel. Dynamical estimation of neuron and network properties i: Variational methods. *Biological Cybernetics*, 105(3-4):217–237, October 2011.

[107] Ghanim Ullah and Steven J. Schiff. Assimilating seizure dynamics. *PLoS Computational Biology*, 6(5):1–12, May 2010.

[108] Rudolph Van der Merwe and Eric A. Wan. The square-root unscented kalman filter for state and parameter-estimation. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 6, pages 3461–3464. IEEE, 2001.

[109] Werner Van Geit, Erik De Schutter, and Pablo Achard. Automated neuron model optimization techniques: A review. *Biological Cybernetics*, 99(4-5):241–251, 2008.

[110] Alejandro F Villaverde, Antonio Barreiro, and Antonis Papachristodoulou. Structural identifiability of dynamic systems biology models. pages 1–22, 2016.

[111] Henning U. Voss, Jens Timmer, and Jürgen Kurths. Nonlinear dynamical system identification from uncertain and indirect measurements. *International Journal of Bifurcation and Chaos*, 14(06):1905–1933, June 2004.

[112] Andreas Wächter and Lorenz T. Biegler. *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming*, volume 106. Springer-Verlag, Berlin, Germany, 2006.

[113] Olivia J. Walch and Marisa C. Eisenberg. Parameter identifiability and identifiable combinations in generalized hodgkin-huxley models. *Neurocomputing*, 199:137–143, 2016.

[114] Richard A. Waltz, Jose L. Morales, Jorge Nocedal, and Dominique Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming*, 107(3):391–408, 2006.

[115] Eric A. Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. *IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium, AS-SPCC 2000*, pages 153–158, 2000.

[116] Jun Wang, Daniel Breen, Abraham Akinin, Frederic Broccard, Henry D.I. Abarbanel, and Gert Cauwenberghs. Assimilation of biophysical neuronal dynamics in neuromorphic vlsi. *IEEE Transactions on Biomedical Circuits and Systems*, 11(6):1258–1270, 2017.

[117] Alexis B Webb, Nikhil Angelo, James E Huettner, and Erik D Herzog. Intrinsic, nondeterministic circadian rhythm generation in identified mammalian neurons. *Proceedings of the National Academy of Sciences of the United States of America*, 106(38):16493–8, September 2009.

[118] Yina Wei, Ghanim Ullah, Ruchi Parekh, Jokubas Ziburkus, and Steven J. Schiff. Kalman filter tracking of intracellular neuronal voltage and current. *Proceedings of the IEEE Conference on Decision and Control*, pages 5844–5849, 2011.

[119] Matthew J. Weinstein and Anil V. Rao. Algorithm 984. *ACM Transactions on Mathematical Software*, 44(2):1–25, August 2017.

[120] Mami T. Wentworth, Ralph C. Smith, and H. T. Banks. Parameter selection and verification techniques based on global sensitivity analysis illustrated for an hiv model. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):266–297, January 2016.

[121] Lily Yan and Hitoshi Okamura. Gradients in the circadian expression of per1 and per2 genes in the rat suprachiasmatic nucleus. *European Journal of Neuroscience*, 15(7):1153–1162, April 2002.

[122] Jingxin Ye, Daniel Rey, Nirag Kadakia, Michael Eldridge, Uriel I. Morone, Paul Rozdeba, Henry D.I. I Abarbanel, and John C. Quinn. Systematic variational method for statistical nonlinear state and parameter estimation. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 92(5):1–17, November 2015.

[123] Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software*, 23(4):550–560, 1997.