UNIVERSITY OF CALIFORNIA, SAN DIEGO

Predicting the Behavior of Dynamical and Biological Systems Using Asynchronous Data

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

in

Physics

by

Uriel Morone

Committee in charge:

Professor Henry Abarbanel, Chair
Professor Daniel Arovas
Professor Gert Cauwenberghs
Professor Michael Fogler
Professor Joshua Graff Zivin

2016

DEDICATION

To my family who made life possible and my friends who made it worthwhile

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank the other members of my group who took the time to discuss at length the ideas herein, as well as many more which were too speculative to include.

My advisor Prof. Henry Abarbanel has my sincere gratitude for the encouragement and enthusiasm with which he guided me through my graduate career.

Chapter 6 is reproduced in part from material as it appears in "Estimating the biophysical properties of neurons with intracellular calcium dynamics". Jingxin Ye, Paul J Rozdeba, Uriel I Morone, Arij Daou, and Henry DI Abarbanel. The author of this thesis was a co-author of this paper.

VITA

2011	Bachelor of Arts in Physics *magna cum laude*, Washington University in St Louis
2013	Master of Science in Physics, University of California San Diego
2016	Doctor of Philosophy in Physics, University of California San Diego

PUBLICATIONS

Ye, J., Rey, D., Kadakia, N., Eldridge, M., Morone, U. I., Rozdeba, P., ... & Quinn, J. C. (2015). Systematic variational method for statistical nonlinear state and parameter estimation. Physical Review E, 92(5), 052901.

Daniel Rey, Michael Eldridge, Uriel I. Morone, Henry D.I. Abarbanel, Jan Schumann-Bischoff, Ulrich Parlitz. "Using the Waveform of Time Series Observations to Augment Available Information and Predict with Accuracy" (Working Title). Physical Review E, In Review

Jingxin Ye, Paul J. Rozdeba, Uriel I. Morone, Arij Daou, and Henry DI Abarbanel. "Estimating the biophysical properties of neurons with intracellular calcium dynamics." Physical Review E 89, no. 6 (2014): 062714.

Anderson, Alexander G., Carl M. Bender, and Uriel I. Morone. "Periodic orbits for classical particles having complex energy." Physics Letters A 375, no. 39 (2011): 3399-3404.

ABSTRACT OF THE DISSERTATION

Predicting the Behavior of Dynamical and Biological Systems Using Asynchronous Data

by

Uriel Morone

Doctor of Philosophy in Physics

University of California, San Diego, 2016

Professor Henry Abarbanel, Chair

Physical systems often experience a complexity of behavior which requires many degrees of freedom to model accurately. In practice, it may be impossible to experimentally observe all the necessary state variables in a dynamical model. To make quantitative predictions it becomes necessary to extract information from the observable variables in order to estimate the entire state of the system. I discuss different approaches to making estimating these unobservable state variables. In particular, I explore novel ways of combining data at different times throughout the trajectory of the system to improve the estimate of the system state at a single time.

Chapter 1

Introduction

The study of the physical world is fundamentally a cycle between experimental and theoretical endeavors. An ideal version of this cycle can start with the observation of a new, previously unobserved, phenomena. A theory is developed which reproduces those new behaviors while being consistent with previous experiments. The theory is then used to predict novel behavior in the physical system, which is tested in new experiments. At each step in the process the theoretical predictions must be matched with the experimental results they are trying to reproduce to be considered valid. If the predictions do not meet our standards of accuracy then the theory is failing to incorporate important interactions which influence the physical behavior.

A physical theory commonly takes the form of a system of differential equations. These equations define how state variables, which represent dynamic properties of system, interact and change over time. Additionally, the equations include fixed parameters which quantify the strength of different interactions. In a simulation, we can choose arbitrary initial conditions for the state variables and values for the parameters. This can be interesting and gives an idea for how a typical version of the system might evolve.

Ultimately, however, to be truly be confident a theory is valid it must not just qualitatively, but quantitatively reproduce observations of the real world. This means it is not enough to use an arbitrary initial condition, because even a correct model will not reproduce a recorded time series if it starts in a different state. The only way to conclusively test a model is to start with the same variable and parameter values as the physical system, then compare how well the evolution of the simulation matches the physical observations. This is a problem if significant components of the system cannot be measured practically. Measurement techniques are constantly evolving. Often, the tools do not exist or are not sensitive enough to record a precise value for every significant variable.

Determining an accurate estimate for these unmeasured variables and parameters is the primary focus of data assimilation. We utilize the fact that all the state variables in the system are interdependent, so they exert an influence on the variables which we can observe. The problem then becomes how to most effectively extract information about these hidden components from the measurements that are experimentally possible. It is not guaranteed that the available observations will carry all the information needed by the system. In such cases, simulation can be used to determine if there's a minimum number of necessary variables or if existing experiments can be modified in a way that probes the system in more detail.

1.1 Trajectories of a Dynamical System

The term dynamical system refers to a collection of variables which evolve in concert over time. This applies to an infinite variety of real world systems: wind speeds in the atmosphere, protein synthesis in cells, and the orbits of planets are just a few examples. Such a system can generally be represented mathematically by a system of differential equations specifying the rate of change for each variable:

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}(t), \mathbf{p}, t) \tag{1.1}$$

Where $\mathbf{x}$ and $\mathbf{p}$ are vectors of arbitrary dimension. This says that in general, the rate of change for each of the variables x_i is given by an equation F_i which can depend on *any* of the other variables, as well as the parameters $\mathbf{p}$. The only distinction between the states $\mathbf{x}$ and the parameters $\mathbf{p}$ is that the latter are static quantities, and as such have no equations of motion.

Differential equations like Eq 1.1 generally have no analytic solution. Nevertheless, using numerical integration we can obtain an approximation of the time series $\mathbf{x}(t)$ from some initial conditions $\mathbf{x}(0)$. To perform a numerical integration one transforms the continuous differential equation in Eq 1.1 into a discrete difference equation, of the form:

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{x}(n), \mathbf{p}, n) \tag{1.2}$$

Previously, time was a continuously varying quantity and the equations $\mathbf{F}$ represented the rate of change of states $\mathbf{x}$ with respect to time. In the discrete formulation time is replaced with a integer index n which represents a time *step*. Accordingly the equations $\mathbf{f}$ are discrete *maps* from time n to $n+1$. Each time step corresponds to a finite quantity of time elapsed dt.

There are many possible ways to transform a continuous equation into a discrete one. Different formulations can vary in their accuracy to the continuous solution, require different amounts of computational power, and have different stability properties. Numerical integration is a entire subfield of applied mathematics with an extensive literature. It is not the intention of this work to give a complete or detailed comparison different discretization techniques, but I will list a few of the most common ones which are used in our computational routines:

The simplest and fastest, but least accurate, way is a first order Euler method:

$$f(\mathbf{x}(t_n)) = \mathbf{x}(t_n) + dt \cdot F(\mathbf{x}(t_n)) \tag{1.3}$$

The Euler method is convenient to implement, but if the system varies too quickly can introduce numerical instabilities.

The midpoint rule is a second order implicit routine:

$$f(\mathbf{x}(n+1), \mathbf{x}(n)) = \mathbf{x}(n) + \frac{dt}{2}(\mathbf{x}(n+1) + \mathbf{x}(n)) \tag{1.4}$$

This is an *implicit* method, meaning the equation for the map $\mathbf{f}$ at time n depends on the point $\mathbf{x}(n+1)$ which is being mapped to. This results in an algebraic equation for $\mathbf{x}(n+1)$ which during integration must be solved at each step. The implicit relationship adds computational work, but is typically more stable than an explicit method.

Finally, the most accurate formulation used in this paper is the fourth order Runge-

Kutta method. For clarity it is written as a sequence of equations:

$$\begin{aligned} k_1 &= \mathbf{F}(\mathbf{x}(t_n), t_n) \\ k_2 &= \mathbf{F}\left(\mathbf{x} + \frac{dt}{2}k_1, t_n + \frac{dt}{2}\right) \\ k_3 &= \mathbf{F}\left(\mathbf{x} + \frac{dt}{2}k_2, t_n + \frac{dt}{2}\right) \\ k_4 &= \mathbf{F}\left(\mathbf{x} + dt\, k_3, t_n + dt\right) \\ \mathbf{f}(x(t_n)) &= \mathbf{x}(t_n) + \frac{dt}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned} \tag{1.5}$$

1.2 Probability Distribution of a Trajectory

Mathematically, one way to state the data assimilation problem is: What is the probability distribution of possible trajectories, given the data that has been recorded. With this distribution we can find the most likely trajectories of the physical system, and the variance around them.

More precisely, thinking in the discrete time space, we have a collection of $D \cdot N_T$ points:

$$\mathbf{X} = \{x_1(t_0), \ldots, x_D(t_0), \ldots x_1(t_{N_T}), \ldots, x_D(t_{N_T})\} \tag{1.6}$$

Where D is the dimensionality of the model in Eq 1.1 and N_T is the number of time points used to discretize the time series of the system.

The data is also recorded as a time series which must be discretized with some time resolution. In principle the resolution at which the measurements are recorded does not have to be the same as the resolution at which we simulate the model. However, for the examples herein we will assume they are the same unless otherwise noted. The time series recorded from a measurement instrument can be written as:

$$\mathbf{Y} = \{y_0(t_0), \ldots, y_L(t_0), \ldots y_0(t_{N_T}), \ldots, y_L(t_{N_T})\} \tag{1.7}$$

Note that there are L dimensions at each time in the data vector, rather than D as in the model. This is because in general there is no reason every dimension in the model can be observed. In fact, there is no reason the observations need to correspond directly to a variable in the model. The relationship between the observations and the dynamic variables are given by a *measurement function*:

$$\mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t)) \tag{1.8}$$

With this notation, the distribution of interest is $P(\mathbf{x}(n)|\mathbf{Y})$. In other words, the probability distribution of the state of the system after a window of time $[0, t_n]$. To achieve some traction determining an explicit form for the distribution we can start by assuming the dynamics only depend on the state of the system at the previous time. In a discrete formulation, this means that the state at each time t_{n+1} depends only on the state at time t_n.

We can start by considering the probability of a state at time t_n, given the time series observed up to that time: $P(\mathbf{x}(t_n)|\mathbf{Y}(t_n))$. To interpret this probability it is useful to write it in terms of the probability of the state at the previous time step $n-1$ (for simplicity I will label times t_n simply by the index n):

$$\begin{aligned}
P(\mathbf{x}(n)|\mathbf{Y}(n)) &= P(\mathbf{x}(n)|\mathbf{y}(n),\mathbf{Y}(n-1)) \\
&= \frac{P(\mathbf{x}(n),\mathbf{y}(n),\mathbf{Y}(n-1))}{P(\mathbf{y}(n),\mathbf{Y}(n-1))} \\
&= \frac{P(\mathbf{x}(n),\mathbf{y}(n),\mathbf{Y}(n-1))/P(\mathbf{Y}(n-1))}{P(\mathbf{y}(n),\mathbf{Y}(n-1))/P(\mathbf{Y}(n-1))} \\
&= \frac{P(\mathbf{x}(n),\mathbf{y}(n)|\mathbf{Y}(n-1))}{P(\mathbf{y}(n)|\mathbf{Y}(n-1))} \\
&= \frac{P(\mathbf{x}(n),\mathbf{y}(n)|\mathbf{Y}(n-1))}{P(\mathbf{y}(n)|\mathbf{Y}(n-1))P(\mathbf{x}(n)|\mathbf{Y}(n-1))}P(\mathbf{x}(n)|\mathbf{Y}(n-1)) \\
&= \exp(I[\mathbf{x}(n),\mathbf{y}(n)|\mathbf{Y}(n-1)])P(\mathbf{x}(n)|\mathbf{Y}(n-1)) \qquad (1.9)
\end{aligned}$$

Where $I[x,y|Y] = \log\left[\frac{P(x,y|Y)}{P(x|Y)P(y|Y)}\right]$ is the conditional mutual information. This is a measure of how much one random variable (such as the observation $\mathbf{y}(n)$) tells us about another random variable (such as the state of the system $\mathbf{x}(n)$) given some other information (in this case, the previous observations at previous times, $\mathbf{Y}(n-1)$). If $\mathbf{x}(n)$ and $\mathbf{y}(n)$ are totally independent then the mutual information (and likewise the *conditional* mutual information) will be zero.

The final term in Eq 1.9 can be rewritten to produce a recursion relation:

$$\begin{aligned}
P(\mathbf{x}(n)|\mathbf{Y}(n-1)) &= \int d\mathbf{x}(n-1)P(\mathbf{x}(n),\mathbf{x}(n-1)|\mathbf{Y}(n-1)) \\
&= \int d\mathbf{x}(n-1)\frac{P(\mathbf{x}(n),\mathbf{x}(n-1),\mathbf{Y}(n-1))}{P(\mathbf{Y}(n-1))} \\
&= \int d\mathbf{x}(n-1)\frac{P(\mathbf{x}(n),\mathbf{x}(n-1),\mathbf{Y}(n-1))}{P(\mathbf{x}(n-1),\mathbf{Y}(n-1))}\frac{P(\mathbf{x}(n-1),\mathbf{Y}(n-1))}{P(\mathbf{Y}(n-1))} \\
&= \int d\mathbf{x}(n-1)P(\mathbf{x}(n)|\mathbf{x}(n-1),\mathbf{Y}(n-1))P(\mathbf{x}(n-1)|\mathbf{Y}(n-1))
\end{aligned} \qquad (1.10)$$

Combining Eq 1.10 with 1.9 gives us an expression for $P(\mathbf{x}(n)|\mathbf{Y}(n))$ in terms of the previous time step $P(\mathbf{x}(n-1)|\mathbf{Y}(n-1))$.

We can simplify the term $P(\mathbf{x}(n)|\mathbf{x}(n-1), \mathbf{Y}(n-1))$ even further by utilizing the fact that our system consists of differential equations which are local in time. This makes the dynamics Markovian. In other words, the state $\mathbf{x}(n)$ is completely determined by the state at the previous time $\mathbf{x}(n-1)$, up to any stochastic randomness. There is no additional information possible that can give a more accurate distribution for $\mathbf{x}(n)$. Therefore,

$$P(\mathbf{x}(n)|\mathbf{x}(n-1), \mathbf{Y}(n-1)) = P(\mathbf{x}(n)|\mathbf{x}(n-1))$$

Putting all these pieces together gives us the complete recursion relation:

$$\begin{aligned} P(\mathbf{x}(n)|\mathbf{Y}(n)) = &\exp(I[\mathbf{x}(n), \mathbf{y}(n)|\mathbf{Y}(n-1)] \\ &\times \int d\mathbf{x}(n-1)\; P(\mathbf{x}(n)|\mathbf{x}(n-1)) P(\mathbf{x}(n-1)|\mathbf{Y}(n-1)) \end{aligned} \tag{1.11}$$

A final expression is obtained by iterating this relationship from time t_0 to t_n:

$$\begin{aligned} P(\mathbf{x}(n)|\mathbf{Y}(n)) = \int \prod_{i=0}^{n-1} d\mathbf{x}(i) \exp(I[\mathbf{x}(i+1), \mathbf{y}(i+1)|\mathbf{Y}(i)]) \\ \times P(\mathbf{x}(i+1)|\mathbf{x}(i))\; P(x(i)|\mathbf{Y}(i)) \end{aligned} \tag{1.12}$$

1.3 Synchronization Theory

There are several techniques to estimate the state of a system at the end of an observation window [1–3]. One approach is to initialize a separate simulated model, starting from an arbitrary initial state. Then the observations from the system we wish to estimate (data system) will be used to adjust the trajectory of the model with the objective that over time these adjustments will synchronize the model to the observed and unobserved variables in the data system.

The problem can be framed mathematically in the following way:

$$x_i(t) = y_i(t) \qquad i = 1 \ldots L \tag{1.13}$$

$$\frac{dx_i(t)}{dt} = F_i(y_1, \ldots, y_L, x_{L+1}, \ldots x_D) \qquad i = L+1 \ldots D \tag{1.14}$$

Where $x_i(t)$ are the states of the model and $y_i(t)$ are the observed time series. The observed data is used as a replacement for the corresponding state variables, then the remaining variables are allowed to evolve according to the equations of motion, given those values.

This approach was first proposed by Pecorra and Caroll as a way of controlling the chaos in nonlinear systems [4, 5]. Researchers studying dynamical systems found that by connecting identical chaotic systems along a single degree of freedom, the uncoupled degrees of freedom would come to synchronize as well. In the context of data assimilation the two systems in question are the physical experiment from which we record data and the simulated model. If we have done our jobs well as physicists the physical system should be governed by the same (or very similar) equations as the model. If the coupling between the physical system (i.e. the measured observations) to the model is sufficiently strong then the hope is that the model will synchronize in *all* of its state variables to the same value as the data system.

An extension of this method for estimating the states and parameters $x_a(n)$ relies on systematic adjustment from some initial state $x_a^{(0)}(n)$ through an iterative process that gradually results in a more accurate estimate:

$$x_a^{(0)}(n) \rightarrow x_a^{(1)}(n) \rightarrow x_a^{(2)}(n) \rightarrow \ldots \rightarrow x_a^{(J)}(n)$$

If the procedure is successful, then after many iterations ($J \gg 1$) $x_a^{(J)}(n)$ should converge to a value consistent with the observations: $h_l(\mathbf{x}^J(n)) \approx y_l(n)$.

Applying synchronization to a data and model system has the added limitation that the coupling can only go in one direction. Because the data is typically prerecorded, and we generally want to estimate the experimental system rather than perturb it, the data is not modified by the synchronization procedure. Only the model system can be adjusted to bring it in line with the data. This means that to successfully synchronize the model the coupling may have to be stronger than when two dynamical systems are driven together simultaneously.

If we can represent the dynamics of the system exactly and record data to infinite precision, then the equations $\mathbf{F}(\mathbf{x})$ in the model provide deterministic constraints of what the state of the model can be at each time, given a previous state. In this limit of low noise, a straightforward and effective way of extracting information from the measured components of the system is to modify the equations of motion using an external forcing term based on the recorded data. The objective is over time to force the model state $\mathbf{x}(t)$ to values consistent with the recorded variables in the physical system $\mathbf{y}(t)$. The modified equations take the following form:

$$\frac{dx_l(t)}{dt} = F_l(\mathbf{x}(t)) + \sum_{l'=1}^{L} g_{l,l'}(t)\,(y_{l'}(t) - \mathbf{x}_{l'}(t)), \tag{1.15}$$

for the $l = \{1, 2, \ldots, L\}$ measured states, and

$$\frac{dx_k(t)}{dt} = F_k(\mathbf{x}(t)),$$

for the $k = \{L+1, L+2, \ldots, D\}$ unmeasured states. The control term $\vec{g}(t)$ is positive definite and has a narrow peak centered at $t = t_n$, so that it impacts the model trajectory only at times when an observation is made.

The control term $\mathbf{g}(t)$ couples the model state to the observed data. This introduces information about how the physical system is behaving. Then the equations of motion, $\mathbf{F}(\mathbf{x})$, further filter this information and adjust the unobserved variables so that the overall system can remain consistent with the time series $\mathbf{y}(t)$. This approach is commonly used the study of meteorology to simulate a very high-resolution model using recordings from a relatively sparse set of weather stations. It is often referred to as “nudging”, Newtonian relaxation, or 4DDA, and is rooted in the theory of controls and dynamical systems [6, 7].

As stated earlier, we restrict ourselves to a constant and diagonal $\vec{g}(t)$. This means each measurement y_ℓ corresponds to exactly one state variable x_ℓ. There is no interaction or transfer of information between observations and other state variables. When we extend the procedure to be include waveform information via time delayed measurements the stability becomes quite sensitive to the magnitude of the coupling term. Section 3.3.1 discuss in more detail numerical techniques to ensure an appropriate value is chosen.

1.4 Twin Experiments

The techniques described in this paper are intended for experimental data. The objective is to able to receive a time series from a measurement instrument and use that as input to the procedures described below. However, this is not an effective way to explore and refine the techniques themselves.

The problem is that real world systems are always more complicated than we can account for in our models. Moreover, we do not know which equations accurately represent the underlying dynamics (in fact, this is a major motivation for developing this procedure). If we used data from real experiments it would not be possible to distinguish between problems with the computational methods and physical errors in the model.

With this in mind we use a method we call twin experiments to produce simulated data rather than using inputs from genuine experiments. The twin experiment simply consists of simulating a model for some length of time. Then we choose a limited number of time series from that simulation and treat them as data. A second model is run with different initial conditions through a data assimilation procedure, without providing the algorithm any information about unmeasured variables of the data system. After the procedure completes, the model state variables are retroactively compared to the data state variables. Then we can judge how accurately the method was able to estimate them.

This technique is very valuable for diagnostic purposes, because it allows us to look at unmeasured states after the fact and confirm how well the assimilation procedure worked and where it broke down. It also generates data to use on the fly, rather than having to wait for the results of an experiment. The conditions of the “experiment” can be tuned to specific regimes of interest. Most importantly, we are assured that the data we have was truly produced by the model we are using so any failure in the estimate is a limitation of the technique itself, rather an unknown mistake in the physical model.

The twin experiments confirm that given an accurate model for the data, how many and which variables must be observed to provide a effective predictions of the experimental system. This is necessary to invalidate models on predictive grounds. Later, when we apply a model to real data, if the model fails to predict behavior, we can be confident that there is a mistake in the model and not the synchronization technique.

Chapter 2

Syncrhonization in Practice

Synchronization can be quite effective when a sufficient proportion of possible states can be measured. It comes at a relatively small cost, essentially just adding a linear term and external forcing (in the form of the data $y(t)$) to the equations of motion:

$$\frac{dx}{dt} = f(\mathbf{x}(t)) \quad \rightarrow \quad \frac{dx}{dt} = f(\mathbf{x}(t)) + g \cdot (y(t) - x(t)) \tag{2.1}$$

where g is an arbitrary constant which sets the strength of the coupling from the data to the model. Due to its relative simplicity, this approach provides a useful baseline for comparison to other data assimilation methods.

2.1 Lorenz 96 Model

The Lorenz 96 system provides a useful toy model for testing new techniques. It can be easily extended to an arbitrary number of dimensions, it is chaotic in the appropriate parameter regime, and is stable to perturbations during the dynamics. Additionally, dynamical properties of the model are well studied in the literature. It is an abstract model with properties relevant to atmospheric studies, namely external forcing and dissipation and quadratic terms to represent advection advection [8–10]. The dynamics are given by:

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + \nu \tag{2.2}$$

Where the index i is cyclical. This simple pattern makes it easy to extend the Lorenz 96 model to an arbitrary number of dimensions, depending how large of a system we want to work with. For parameter values of $\nu > 8$ the system is known to exhibit chaos.

Figure 2.1 shows typical time series for x_0 in two the Lorenz 96 systems. Due to the chaotic dynamics two Lorenz 96 systems with even minor differences in initial conditions will eventually decorrelate completely. System 1 and 2 in the figure were initialized with only 1% difference in initial condition. Despite the very similar initialization we see that the trajectories very quickly diverge.

2.1.1 Coupling to observations

Despite the nature of the dynamics, by using synchronization we can regulate the chaos as described in section 1.3. This means that the addition of data in the dynamics modifies the Conditional Lyapunov Exponents for the model system. When sufficient information is transferred from the data to the model, the CLEs become negative. This means that over time the state of the model converges to the state of the data the data system.

Figure 2.2 shows the effect for two cases. The left side of the figure shows the model

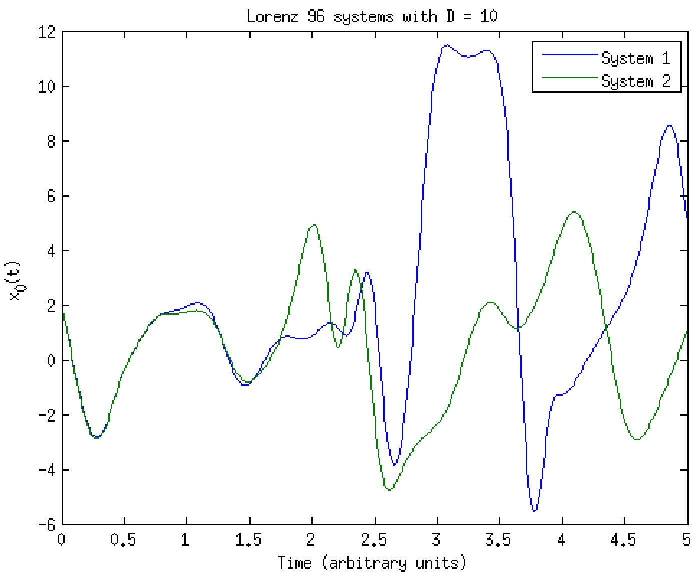


Figure 2.1: First state variable x_0 for two separate Lorenz 96 models with 10 dimensions. Initial conditions for system 2 are $\mathbf{x}^2(0) = 1.01\mathbf{x}^1(0)$. Forcing parameter $\nu = 8.17$. Due to chaos even very similar initial conditions rapidly diverge.

system with only 3 observed state variables. We see that as soon as the observation window ends the trajectories quickly diverge. Although the measured variable in the model was coupled to the data there was not enough information to synchronize the unmeasured variables. By contrast the right side of the figure shows the same initial conditions but with the first 5 states coupled to the data. The result is that enough information is transferred to reasonably estimate the unobserved variables as well, which allows model to accurately track the data for a time after the estimation window.

2.2 Limitations

Despite the convenience of the synchronization procedure, it leaves on at a loss if there are not observations to reduce all of the Conditional Lyapunov Exponents. Addi-

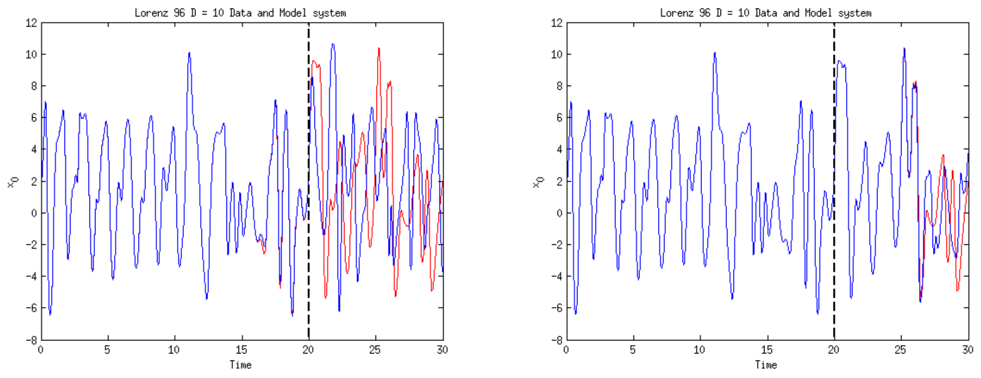


Figure 2.2: First variable in a 10 dimensional Lorenz 96 twin experiment. Both plots show the same underlying data system along with a model initialized with $\mathbf{x}^{model}(0) = 1.01\,\mathbf{x}^{data}(0)$. Left: $\{x_1, x_2, x_3\}$ in model coupled to data. Right: $\{x_1, \ldots, x_5\}$ in model coupled to data

tionally as we see on the right side of Figure 2.2, even when there appear to be enough measurements the convergence may not be strong enough or happen quickly enough to obtain a very long period of prediction.

The basic synchronization described until now only utilizes the unmeasured states' dependence on the measured states to transfer information. That is, for unmeasured state $x_{L+1}(t)$, in the model all of the information about the data system comes via the unmodified dynamics $F_{L+1}(\mathbf{x}(t))$. This leaves us dependent on the details of model to transfer sufficient data.

It is perfectly possible for example to have a set of equations where the dynamics are only one way. The measured variable dynamics may depend on the unmeasured states, but not vice versa:

$$x_i = \begin{cases} F_i(x_1, \ldots, x_D) & i = 1, \ldots, L \\ F_i(x_{L+1}, \ldots, x_{D-1}) & i = L+1, \ldots, D-1 \end{cases} \tag{2.3}$$

An example of such a system is discussed chapter 6.2. There I introduce a model for a spiking neuron which is modulated by internal Calcium dynamics. The Calcium is primarily stored in an internal repository which absorbs and emits Calcium into the cell's cytoplasm independently of voltage. Nevertheless the voltage itself depends on ions who is regulated by Calcium.

In such situations no matter how strongly we couple the observed variables to the data system, the unobserved states will never adjust. As a result, as soon as the estimation window ends and we simulate the model independently the observed components will take a totally separate trajectory in model from what happens in the data system.

An important category of problems that has this behavior is parameter estimation. As we will see later (Eq 3.12) parameters can be interpreted as state variables with trivial dynamics, but their dynamics do not have an explicit dependence on state variables.

Chapter 3

Time Delay Synchronization

There is information contained in the observed time series which is not utilized by synchronization procedure described in the previous chapter. The traditional approach compares the state of the model system to the data system at each time point independently (at least in the measured components). The approach described in this section, instead compares the *evolution* of the two systems over a short window. Looking at the waveform of the measured components allows mismatched unobserved states the time to exert their influence on the observations.

The idea of using changes in a dynamic variable as an independent source of information is not new. Time-embedding is a well-known technique in the study of dynamical systems for attractor reconstruction. Plotting the trajectory of a dynamical system in phase space can illustrate properties of that system including it's stability in different regions, the prescence of chaos, and it's fractal dimension. However, even if one does not have access to a time series for every degree of freedom it is possible to reconstruct the attractor by plotting a single variable at time delayed points: $\{x_0(t), x_0(t+\tau), \ldots, x_0(t+(D_\tau-1)\tau)\}$. When D_τ is large enough it can be shown that the trajectory in this time-embedded space will retain those properties of the attractor which are preserved under smooth transformation.

In the context of data assimilation we do not wish to reproduce qualitative properties of a dynamical system, but rather quantitatively estimate its path through phase space. The idea is to use the dynamical equations to determine how small changes in any one state variable affect the values of other variables. Then we compare the trajectory the model system to the recorded time series from the data system; these should differ because the model does not start with the same initial conditions as the data system. Finally, we utilize the Jacobian of the model equations to determine a perturbation to the entire state vector which will adjust the model's trajectory so that it coincides with the data trajectory.

Of course, this is an ideal case. In practice whenever you introduce an external forcing to a dynamical system there is the possibility of pushing the dynamics into an unstable region which diverges. The inverse problem of going from "a perturbation in variable 1 propagates a change in variable 2" to "we require variable 2 to change by so much, how can variable 1 be perturbed to accomplish this" is in general ill-posed. These issues and heuristics to deal with them are discussed after the method is described in detail.

3.1 Variational Matrix

The variational equation (Eq 3.1) quantifies how a perturbation in state variable x_i at time t_0 will propogate to state variable x_j at time t. The variational matrix, which contains this information, is initialized as the identity matrix. This reflects the fact that at t_0 a perturbation has not had any time to affect any other state variables.

$$\Phi_{ab}(t, t_n) = \frac{\partial x_a(t)}{\partial x_b(t_n)}$$

$$\frac{d\Phi_{ab}(t, t_1)}{dt} = \sum_{c=1}^{D} \frac{\partial F_a(\mathbf{x}(t))}{\partial x_c(t)} \Phi_{cb}(t, t_1) \qquad \Phi_{ab}(t_0) = \delta_{ab} \tag{3.1}$$

The equations of motion of the variational matrix $\mathbf{\Phi}$ are given by multiplying $\mathbf{\Phi}$ with the Jacobian matrix. The Jacobian of a system of equations, $\frac{\partial F_a(\mathbf{x}(t))}{\partial x_c(t)}$ represents how the equations of motion of the states change in response to changes in the states themselves. Multiplying these changes over time results in the changes of each variable to changes from earlier in the trajectory.

This can be seen with a simple derivation:

$$\begin{aligned}
\frac{d\mathbf{x}(t)}{dt} &= \mathbf{F}(x(t)) \\
\frac{\partial}{\partial \mathbf{x}(t_0)} \frac{d\mathbf{x}(t)}{dt} &= \frac{\partial \mathbf{F}(x(t))}{\partial \mathbf{x}(t_0)} \\
\frac{d}{dt} \frac{\partial \mathbf{x}(t)}{\partial \mathbf{x}(t_0)} &= \frac{\partial \mathbf{F}(x(t))}{\partial \mathbf{x}(t_0)} \\
\frac{d}{dt} \frac{\partial \mathbf{x}(t)}{\partial \mathbf{x}(t_0)} &= \frac{\partial \mathbf{F}(x(t))}{\partial \mathbf{x}(t)} \frac{\partial \mathbf{x}(t)}{\partial \mathbf{x}(t_0)}
\end{aligned}$$

3.2 Time Embedded Coordinates

To apply the variational matrix at a point in time, we need to know the precise values of the ensuing trajectory. This information is contained in what we call the *time embedding vector*, $\mathbf{S}(t)$. The values contained in $\mathbf{S}(t)$ are the values of the observed state variables at delayed times:

$$\mathbf{S}(t) = \{x_1(t), x_1(t+\tau_1), x_1(t+\tau_2), \ldots, x_L(t+\tau_K)\} \tag{3.2}$$

Here the τ's are K constants, at which we delay the observation time. For convenience we typically use a fixed time delay so that $\tau_i = i \cdot \tau$. Each of the L observed components is included in the time embedding vector.

The time delayed values are obtained by integrating $\mathbf{x}(t)$ using the model equations <u>without any coupling</u> over times $t \to t + \tau_K$

We construct a similar vector for the measured data values:

$$\mathbf{Y}(t) = \{y_1(t), y_1(t+\tau_1), y_1(t+\tau_2), \ldots, y_L(t+\tau_K)\} \tag{3.3}$$

There is no need to integrate for $\mathbf{Y}$, because all of the data has already been recorded *a priori*. Looking at the difference $(\mathbf{Y} - \mathbf{S})$ shows how the model trajectory diverges from the observed data system over the relatively short window $[t, t + \tau_K]$.

3.3 Calculating a control δx from time embedded coordinates

In the original synchronization approach we modified the equations of motion by adding a control term which nudged the measured states towards the value of the observed data:

$$\frac{dx_\ell}{dt} = F_\ell(\mathbf{x}(t)) + g \cdot (y_\ell(t) - x_\ell(t)), \qquad \ell = \{1, \ldots, L\}$$

Note that there are only L of these control terms, one for each measured component. Now we will use the time embedded coordinates to calculate a different coupling term which incorporates information from the entire time embedding window $[t, t + \tau_K]$, rather than just the time t.

The following procedure is repeated at each time point:

Starting at the current time point in the simulation t_n, we integrate both the state $\mathbf{x}$ using the unmodified equations Eq 1.1 and the variational matrix using Eq 3.1. From these values we construct the embedding vector in Eq 3.2. Additionally we have values for the D^2 entries of $\Phi(t)$ at each time in the time embedding window.

The matrix Φ, however, contains a lot of information that is not useful. Recall that the entries are $\Phi_{ij} = \frac{\partial x_i(t)}{\partial x_j(t_n)}$. We do not have knowledge of $x_i(t_n)$ for $L < i < D$, so it is not helpful to know how those states vary in response to changes in the system. Instead, we construct a new matrix, essentially the Jacobian of the time embedded coordinates, by combining rows $1 \le i \le L$ of $\Phi_{ij}(t)$ at each time embedding point in the window:

$$\frac{\partial \mathbf{S}(t_n)}{\partial \mathbf{x}} = \begin{pmatrix} \Phi_{11}(t_n) & \cdots & \Phi_{1D}(t_n) \\ \Phi_{11}(t_n + \tau) & \cdots & \Phi_{1D}(t_n + \tau) \\ \vdots & \vdots & \vdots \\ \Phi_{11}(t_n + \tau_K) & \cdots & \Phi_{1D}(t_n + \tau_K) \\ \vdots & \vdots & \vdots \\ \Phi_{L1}(t_n + \tau_K) & \cdots & \Phi_{LD}(t_n + \tau_K) \end{pmatrix}. \tag{3.4}$$

The dimensions of $\frac{\partial \mathbf{S}}{\partial \mathbf{x}}$ are: $L \cdot (K+1) \times D$

Defining $\delta\mathbf{S}(t) = \mathbf{Y}(t) - \mathbf{S}(t)$, the task of finding an appropriate control term for the dynamical equations can summarized succinctly by solving

$$\delta\mathbf{S}(t) = \frac{\partial\mathbf{S}}{\partial x}(t) \cdot \delta\mathbf{x}(t) \tag{3.5}$$

for the coupling vector $\delta\mathbf{x}$. An important point to notice is that the dimension of $\delta\mathbf{x}$ is D, rather than L. In other words, with the time delay calculation for the coupling vector we have sufficient information to adjust our estimates of *all* states simultaneously, rather than only the measured states.

To review the meaning of Eq 3.5:

- $\delta\mathbf{x}$ is a perturbation to the state $\mathbf{x}$ at time t
- $\frac{\partial\mathbf{S}}{\partial x}(t)$ is the effect a perturbation in each state variable $\mathbf{x}(t)$ will have on each of observed states at the delayed times $\{t, \ldots, t + \tau_K\}$
- Multiplying these quantities together gives the effect of perturbation $\delta\mathbf{x}$ on each of entry of $\mathbf{S}$, i.e. on each of the observed components at each time in the embedding window.

However, we wish to solve the inverse problem. We already know the perturbation we want to produce in the measured components: $\mathbf{Y} - \mathbf{S}$, which would mean the model's trajectory is aligned with the trajectory in the data. The goal is to find a perturbation $\delta\mathbf{x}$ by which we can modify the state vector to the current time t to produce such a deviation. Mathematically, we approximate the coupling term as:

$$\delta\mathbf{x} = \frac{\partial\mathbf{x}}{\partial\mathbf{S}}\delta\mathbf{S} \tag{3.6}$$

where $\frac{\partial \mathbf{x}}{\partial \mathbf{S}}$ is the pseudoinverse of $\frac{\partial \mathbf{S}}{\partial \mathbf{x}}$.

We can use this perturbation as a control term the same way the "$g(y-x)$" term in the Eq 2.1 was used. The states get nudged, but this happens during the integration, and occurs more gradually then changing the states at once. This should intuitively help prevent from being placed in an unstable regime due to a non-dynamical perturbation. However discussions with other graduate students suggest that the discrete shift may be preferable [11]. This approach is discussed in 3.4 below.

The equations of motion then become:

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}(t)) + g \cdot \delta\mathbf{x}(t) \tag{3.7}$$

The perturbation in Eq 3.7 needs to be calculated at each time point during the integration, this adds a significant computational cost. However the payoff is the ability to predict the behavior of systems which previously seemed impossible.

The pseudoinverse allows us to effectively *invert* the dynamical dependence in the equations of motion. Using the original synchronization term only the measured variables in the model are affected directly. We must rely on the natural dynamics to guide the unobserved variables to the underlying values in the data system. However, the pseudoinverse of $\frac{d\mathbf{S}}{d\mathbf{x}}$ provides the information to adjust the unmeasured variables directly, based on our observations of measured ones. This method gives us a way of reproducing behavior of a system with dynamics like those in Eq 2.3.

3.3.1 Computing the Moore-Penrose Pseudoinverse of $\frac{\partial \mathbf{S}}{\partial \mathbf{x}}$

Like many theoretical ideas inverting $\frac{\partial \mathbf{S}}{\partial \mathbf{x}}$ is easier said than done. The first question one must answer is how to solve the system of equations in Eq 3.5, if $\delta \mathbf{S}$ and $\delta \mathbf{x}$ have different dimensions. Depending on the of number of time embedded points used there may be either more or less equations than variables in $\delta \mathbf{x}$. An additional condition must be imposed in order to obtain a single solution (for under-determined systems) or an approximate solution (for over-determined systems).

We do this by computing the Moore-Penrose pseudoinverse. The M-P pseudoinverse results in a solution $\mathbf{a}$ which solves a least squares problem: $\arg\min_{\mathbf{X}} \|\mathbf{b} - \mathbf{X}\mathbf{a}\|^2$. This solution exists even if $\mathbf{a}$ and $\mathbf{b}$ have different dimensions. If $\mathbf{a}$ and $\mathbf{b}$ have the same dimensions then the pseudoinverse will be the literal inverse, which gives the unique solution for $\mathbf{a}$ in the system of equations $\mathbf{X}^{-1}\mathbf{b} = \mathbf{a}$.

A typical approach to calculating the pseudoinverse is with singular value decomposition (SVD). SVD is a way of factorizing a matrix into three distinct parts.

$$\mathbf{U}\mathbf{\Sigma}\mathbf{V}^* = \mathbf{M} \tag{3.8}$$

In general, for an $m \times n$ matrix $\mathbf{M}$ with rank ρ, SVD will produce:

- $\mathbf{U}$: $m \times \rho$ unitary matrix
- $\mathbf{\Sigma}$: $\rho \times \rho$ square diagonal matrix. The diagonal entries of this matrix are the *singular values*
- $\mathbf{V}$: $n \times \rho$ unitary matrix ($\mathbf{V}^*$ is the conjugate transpose)

This factorization is very convenient because each of the component matrices can be easily inverted. The inverse of a unitary matrix is, by definition, its conjugate transpose. Technically, a diagonal matrix in general is only invertible if all it's entries are non-zero, in which case the inverse is simply another diagonal matrix whose entries are the reciprocal of those in the original matrix:

$$\mathbf{D} = \begin{pmatrix} d_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & d_{mm} \end{pmatrix} \qquad \mathbf{D}^{-1} = \begin{pmatrix} \frac{1}{d_{11}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{d_{mm}} \end{pmatrix} \tag{3.9}$$

In general, the matrix $\mathbf{\Sigma}$ from the SVD *will* diagonal entries equal to zero. In this case the pseudoinverse is the same way as when the matrix is invertible, but with the modification that zero-valued diagonal entries are left as zero in the pseudoinverse. This will give a "reduced rank identity matrix":

$$\mathbf{\Sigma}^{-1} = \begin{pmatrix} \frac{1}{\sigma_{11}} & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & & & & \vdots \\ \vdots & & \frac{1}{\sigma_{\rho\rho}} & & & \vdots \\ \vdots & & & 0 & & \vdots \\ \vdots & & & & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 \end{pmatrix} \qquad \mathbf{\Sigma}^{-1}\mathbf{\Sigma} = \begin{pmatrix} 1 & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & & & & \vdots \\ \vdots & & 1 & & & \vdots \\ \vdots & & & 0 & & \vdots \\ \vdots & & & & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 \end{pmatrix} \tag{3.10}$$

Once we've done the SVD, the pseudoinverse of a matrix $\mathbf{M}$ is simply:

$$\mathbf{M}^{-1} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^{*} \tag{3.11}$$

Using SVD for the pseudoinverse is perhaps the primary source of instability in this method. Although it may seem like a trivial numerical consideration, it is not obvious how small a singular value should be to be regarded as "zero" for the purposes of inverting. This decision can significantly affect the success of the time delay synchronization procedure. If very small non-zero values are inverted the resulting control term may be orders of magnitude larger than the dynamics of the system and result in instabilities. On the other hand, if too many singular values are regarded as "zero" and not used in the inverse then the control terms may be too small to regulate the chaos in the system and synchronization is never achieved. This is discussed in more detail in Appendix A

3.4 A problem using $\delta\mathbf{x}$ as a dynamic control

A potential issue I have brushed over until now is the value of the coupling constant g in Eq 3.7. It was mentioned in passing that it can be chosen empirically, as a hyperparameter of the algorithm. This parameter should be adjusted so that it is large enough to transfer information from $\mathbf{y}(t)$ to $\mathbf{x}(t)$, but not so large that the resulting perturbation forces the state $\mathbf{x}(t)$ into an unstable region of phase space. In other words, tweak it til it works. In practice this represents the most rigorous method we have derived for choosing this constant.

Nevertheless, it is productive to dwell for a moment on what this constant truly represents. Placing it on sound theoretical footing could ultimately result a valuable extension which improves the method. The units of g are easily seen to be 1/[time]. Realize that the perturbation we have calculated, $\delta\mathbf{x}$, was intended to be an estimate of the difference between the state of the model system and the state of the data system. However by modifying the dynamical equations, we are adjusting the *rate of change* of the state, not the state itself. The constant g can then be interpreted as a rate, chosen for how quickly we feel the state of the model should be driven towards the (our estimate of) the state of the data.

This is in no way particular to the time delay method. In the original paper on synchronizing chaotic systems Pecorra and Caroll [4] simply replaced the observed states in $\mathbf{x}$ directly with the values of $\mathbf{y}$ in the data system like was introduced in Eq 1.14. Later this approach was generalized to act as additional driving force in the equations of motion [9, 12, 13], with variable coefficient. The original replacement scheme of $x_i = y_i$ $(i \in \{1, \ldots, L\})$ is the special case where $g \to \infty$. In our own work we have found there is often an advantage to limiting the size of the coupling constant. If the system is restrained too tightly it may be forced out of a stable chaotic region by either numerical or dynamical instability. Previous graduate students discuss the impact of using different coupling constant values on synchronization [11].

One way to frame this question is "how should we convert a spatial distance into a rate of change to arrive at that location". A simple answer is to make the whole shift in a single step, rather than nudging the equations of motion. At first glance, a reasonable choice seems to be setting $g = 1/dt$, the step size of the integrator. This should scale the control term $g\,\delta\mathbf{x}$ so that the model system makes the full shift in one time step. Unfortunately we have found that using such a large g is unstable. Even though theoretically we are taking a full step, using an explicit integrator with too large of a step can easily get out of control. The problem is primarily due to the fact that $\delta\mathbf{x}$ was calculated with the *pseudo*inverse, it is inherently imprecise. There is likely no solution which exactly solves $\frac{d\mathbf{S}}{dx}\delta\mathbf{x} = \delta\mathbf{S}$. If we are unlucky the errors in $\delta\mathbf{x}$ (by which I mean the extent to which is does correctly perturb the model to match $\mathbf{S}$ to $\mathbf{Y}$) can create instabilities.

An alternative to controlling the equations of motion was explored by others in the lab after some development had taken place on this dynamical formulation [14]. Referred to heuristically as a Newton's Method, the idea is to decouple the estimation from the integration. Rather than using the nudging as an external force the idea we tried simply using $\delta\mathbf{x}$ to shift the initial condition $\mathbf{x}(0) \to \mathbf{x}(0) + \delta\mathbf{x}$ and integrate the model from there with unmodified equations. This approach is discussed in the reference. I myself did not explore it in detail, but based on conversations I believe it is the most promising avenue to take if one wants to continue to develop and apply the time delay synchronization method more broadly. A noteworthy advantage of separating the shift in state from the dynamic equations is that the process can be repeated several times with the same segment of data. Shifting the state at a single point in time only requires an estimation window of $T = K \cdot \tau$, which is just enough to perform the time embedding. Once $\delta\mathbf{x}$ is obtained the process can be repeated starting from the new estimate $\mathbf{x} + \delta\mathbf{x}$. If the procedure converge to an estimate of $\mathbf{x}(0)$ which is not yet accurate, we can integrate the equations forward to a new time point and repeat this procedure at a somewhat later time. I think this will provide more flexibility than using $\delta\mathbf{x}$ to modify $\frac{d\mathbf{x}}{dt}$.

3.5 Putting Time Delay Synchronization to Work

3.5.1 Lorenz 96

In Fig 2.2 the Lorenz system is shown with basic synchronization. When only the observed variable is controlled directly, measurements of the first 5 of 10 state variables are necessary to achieve a reasonable amount of prediction. With a 20 second window of assimilation, the model can predict accurately for about 5 seconds.

By using the time delay information to control *all* of the states remarkably better results are achieved. Fig 3.1 shows the same system, with the same initial conditions for both the model and the data systems. In this case only $x_1(t)$ was observed from the data system. Thanks to the additional information extracted from seven time embedded coordinates, this system was able to synchronize so well that the predictions remained good for around 13 seconds. That is more than twice as long as the basic synchronization could predict with 5 observations.

There is no free lunch, but in this case at least it pretty cheap. The Fig 2.2 example with 5 measurements takes about 14 seconds to run the assimilation on my laptop. On the same computer the time delayed example with 7 embedded coordinates at each time step took 108 seconds. That's about 8 times as much time to synchronize the same amount of data. In exchange, the problem can be solved with only one time series recording instead of five. In practical problems like weather prediction where each additional weather station requires funding, time, and political effort to construct it can be much cheaper to simply run extra calculations on a computer. In other fields, like neurobiology, where some variables are simply inaccesible during an experiment substituting computational power for measurements can be invaluable.

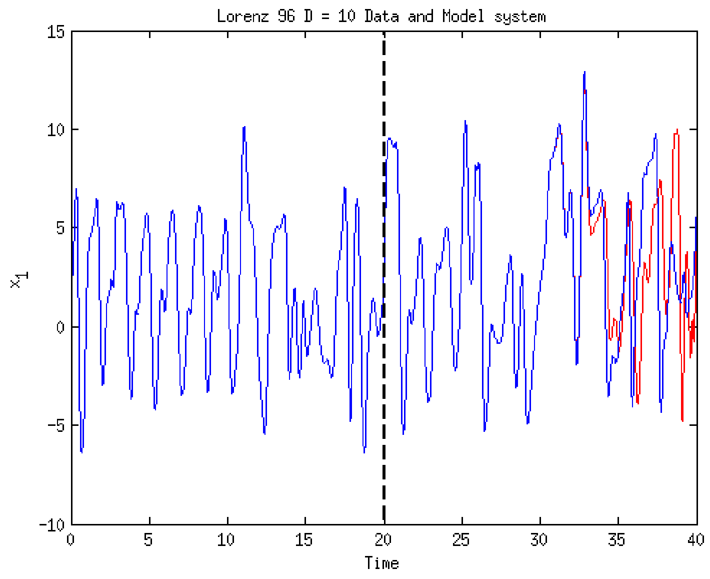


Figure 3.1: Lorenz 96 model with 10 dimensions. Only x_1 is observed. 7 time delay points are used during synchronization $\tau_k = 0.05 \cdot k$. Red: Measured data time series; Blue: Model estimation and prediction. Like Fig 2.2, $\mathbf{x}(0) = 1.01\mathbf{y}(0)$

Even this is overstating the cost, because the results turned out to be substantially better with time delays than with basic synchronization. Its conceivable that we could have used a much *shorter* estimation window with the time delay synchronization to obtain a comparable result to the standard synchronization. Using a shorter window would require proportionally less time.

To test this I ran another twin experiment on the same system, with 1 measurement and 7 time embedding dimensions. This time the data assimilation only lasted 2.59 seconds. I chose this length of window to scale down the window from the previous example by how much faster the basic synchronization ran: (20s window) * (14 sec)/(108 sec) = 2.59s window.

The results worked even better than expected. Fig 3.2 shows that even with this small window of time series data The time delay result still performs dramatically better than the basic synchronization method (this short window actually completed in 10s, almost 30% less than the basic synch. result).

In fact this approach with the shorter window works just as well as the much longer window used in Fig 3.1. This brings up at least two interesting points that are worth highlighting. First is that not only does the time delay synchronization help solve problems with fewer observed variables, it also gives much faster convergence. This is not trivial, experiments cost resources and scientists do not always have the ability to record for arbitrary lengths of time. Having a recording which is too short can be just as limiting as one which incomplete.

Another point worth mentioning is the fact that there are severe diminishing returns when estimating chaotic systems. Due to the fact that chaotic trajectories diverge exponentially one needs orders of magnitude better estimation to get obtain a linear increase in prediction time. That would explain why estimating for 2.6s or 20s gives essentially the same prediction length. In problems with noise the upper bound is really insurmountable.

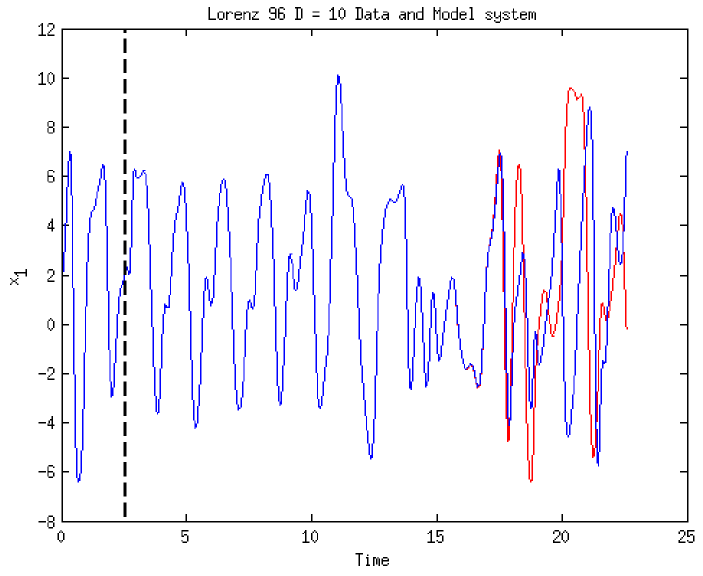


Figure 3.2: Lorenz 96 system data and model trajectories. Only observing x_1 with 7 time embedding dimensions.

The extensibility of the Lorenz 96 model allows us to really show off how powerful the time delay synchronization can be. We can include an arbitrary number of dimensions in the Lorenz 96 system. Previous work had found that with the traditional control method roughly 40% of the dimensions had to be measured to obtain successful synchronization across all states [15–17].

However by using time embedding coordinates to access information latent in the waveform of the time series, we can synchronize even very large models with only 1 variable recording.

3.5.2 Parameter estimation

The time delay synchronization method also provides a way to estimate static parameters in the model. This follows naturally from the observation that parameters can be thought of as state variables with the trivial dynamics:

$$\frac{d\mathbf{p}}{dt} = 0 \tag{3.12}$$

If these variables were appended explicitly to the state vector $\mathbf{x}$ and the basic control coupling were performed, they would never change. No matter how closely the model states $x_{1:L}$ approach the data values $y_{1:L}$, the parameters do not change dynamically.

Recall, on the other hand, that the time embedding synchronization gives control terms for all states, rather than just the observed. This means that even if there is a "variable" that doesn't change, its influence on the observed states will provide some information about how it needs to change.

If we append the parameters to the state vector $\mathbf{x}$ has dimensions $D + N_p$, where

$x_{D+1:N_p} = \mathbf{p}$. Then the of the parameters is determined entirely by the control terms:

$$\frac{dx_{D+1:N_p}}{dt} = \frac{\partial x_{D+1:N_p}}{\partial \mathbf{S}}(\mathbf{Y} - \mathbf{S}) \tag{3.13}$$

Until now the examples have all assumed the mode has access to the correct fixed values for all of the twin experiments. Naturally, without perfect knowledge of these parameters the estimation problem is harder. Nevertheless Fig 3.3 shows that time delay control compensate for this lack of information, at least to an extent.

The Lorenz 96 model in Eq 2.2 contains a forcing parameter, which determines whether the system is in a limit cycle or a chaotic regime (chaos occurs for $\nu \geq 8$). Below we show the value of the forcing parameter in the model system as the time delay control term gradually approaches the value in the data system. The model was initialized with $\nu = 9.0$, while the data system had $\nu = 8.17$.

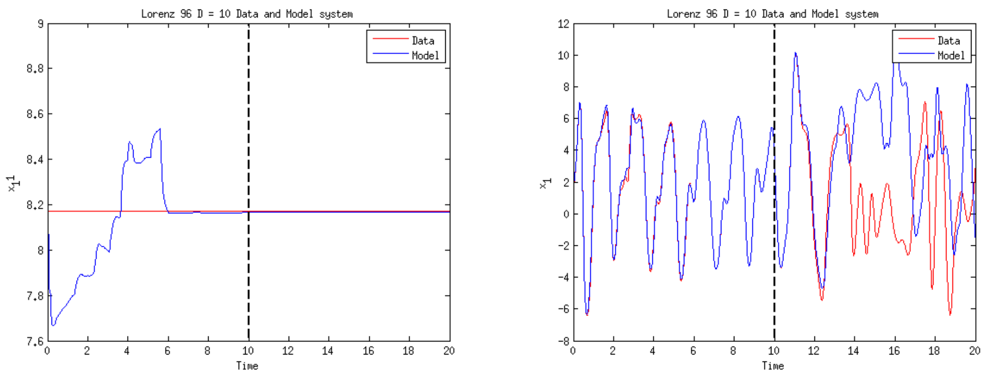


Figure 3.3: Forcing parameter ν estimated in Lorenz 96 equation. Fifty time embedding coordinates used ($\tau_k = k \cdot dt, \quad k = 1 \ldots 50$).

Fig 3.3 demonstrates that this approach works in principle. There is information about the parameters contained in the state variable behavior, which we can access with time embedding coordinates. As always though there are practical concerns. Allowing the parameter in our model to fluctuate gives the system much more freedom and makes the data assimilation correspondingly harder. To obtain the estimate in Fig 3.3 the data assimilation required 49 time embedded coordinates, whereas the previous example in Fig 3.2 made do with 7. Additionally, the prediction deteriorated significantly, with only 2-3 seconds worth of accurate prediction.

3.5.3 Accounting for Noise

Besides being limited by the number of observations, we must also consider the practical reality of noise during data assimilation. There are two main ways noise can enter the procedure. The first is through missing or incorrect terms in the dynamic equations; so that the state $\mathbf{x}(n+1)$ is not quite given by $\mathbf{f}(\mathbf{x}(n))$. This can occur in stochastic systems, where random perturbations at each time occuring at a smaller scale than the dynamics, or may simply be from missing terms in the model equations. The other form of error is the uncertainty inherent in all measurement instruments. In this case the concern is that $\mathbf{y}$ is not a precise observation of $\mathbf{x}$, even though the dynamics of the system itself were not affected. Because it does not affect the evolution of the physical, or data, system the latter type of error (measurement error) is easier to deal with and is what will be mainly discussed.

To minimize the effects of noise a longer time embedding window is required. That is, τ_K in general should be larger than what is acceptable without measurement noise. This is because the time embedding window effectively acts as a low-pass filter. To see this heuristically, assume that the measurement noise is Gaussian ($\nu(t) \sim N(0, \sigma)$) and independent at each time point. The difference between two observations is:

$$\begin{aligned}\tilde{\mathbf{y}}(t+\Delta t) - \tilde{\mathbf{y}}(t) &= \mathbf{y}(t+\Delta t) - \mathbf{y}(t) + \nu(t+\Delta t) - \nu(t) \\ &= \Delta\mathbf{y} + N(0, \sqrt{2}\sigma)\end{aligned}$$

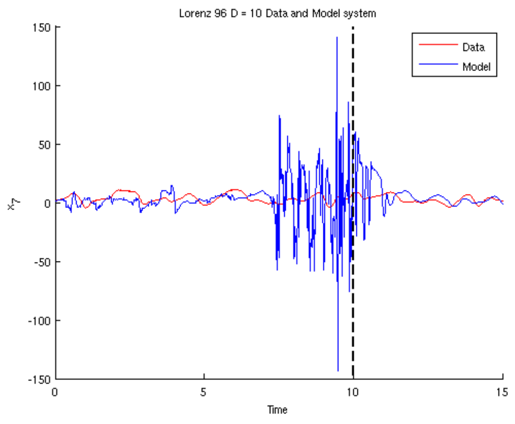
Independently of how much time elapses between two measurements, the amount of the difference due to noise is fixed at a standard deviation of $\sqrt{2}\sigma$. The difference between the underlying variables at different times, $\Delta\mathbf{y}$, is roughly proportional to the time elapsed, Δt (at least over short times; the difference $\Delta\mathbf{y}$ eventually asymptotes to the size of the attractor).

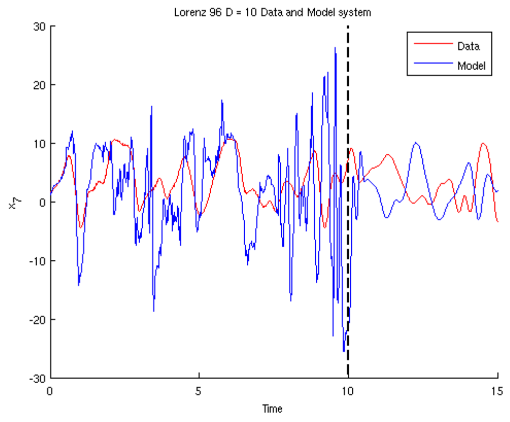
This means that when comparing values of the same variable at different times, as the time embedding control term does, the longer the time elapsed between those the less the difference is attributable to noise.

This effect is easily seen in Fig 3.4. These plots using the same conditions as Fig 3.1. In the previous figure, the measurements were basically exact, and we found that 6 time delays was sufficient to accuratley synchronize the system and predict for about 15 seconds. Adding noise makes this process much harder. The plots in Fig 3.4 below show the same procedure with Gaussian noise $N(0, 0.2)$ added to the observed component, after the data is simulated.

When the noise is added we see that even 10 embedding dimensions are not sufficient to synchronize the model, if the time delays are too short. Taking the spacing between each time delay τ_k from one time step up to fifteen, however suddenly makes synchronization possible. The issue with a short time delay is that percentage error becomes much greater the less time between the time delay terms $\delta\mathbf{S}$ which we apply in Eq 3.6.

Noise in the observations interferes with the pseudoinverse described in section 3.3.1. The $\frac{\partial s}{\partial x}$ matrix can be unstable even without noise, if there are few enough measurements. Adding rapid short time scale fluctuations to the measurements $\mathbf{y}$ means that when we try to calculate a perturbation $\delta\mathbf{x} = \frac{\partial \mathbf{s}}{\partial x}(\mathbf{Y} - \mathbf{S})$ we can huge unphysical perturbations. We are effectively looking for a perturbation which will result in the random noise fluctuations. Such a trajectory is not a real solution of the model, and so the control term $\delta\mathbf{x}$ can diverge and take on very large values. The effect will be reduced however over a longer window for $\mathbf{Y}$ and $\mathbf{S}$, because the noise will account for a smaller percentage of the changes in the variable. The time delay essentially acts as a low-pass filter, which cuts out small time scale fluctuations the longer it gets.





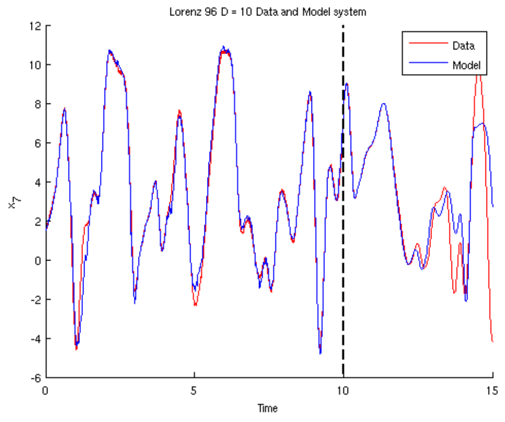


Figure 3.4: L96 10-dim system with Gaussian noise N(0,0.2) added to measured variable, $x_1(t)$. Unmeasured variable $x_7(t)$ plotted in the data and model system. Integration time step $dt = 0.01$. Top: $\tau = \{1 \dots 10\}dt$ Middle: : $\tau = \{5, 10, 15, \dots 50\}dt$ Bottom: $\tau = \{15, 30, 45, \dots, 150$

Chapter 4

Path Integral Approach

The data assimilation techniques so far have been in the form I have called "synchronization", which is often known as "filtering" in the literature. The general idea is to take recordings of the data system and, using either a single time point or a window or time, calculate a modification to the equations of motion which would drive a secondary system closer to that data. This approach is often effective, but it is not immediately obvious how to relate it to the probability distribution in Eq 1.12, which is reproduced here:

$$\begin{aligned} P(\mathbf{x}(n)|\mathbf{Y}(n)) = \int \prod_{i=0}^{n-1} d\mathbf{x}(i) \exp(I[\mathbf{x}(i+1), \mathbf{y}(i+1)|\mathbf{Y}(i)]) \\ \times P(\mathbf{x}(i+1)|\mathbf{x}(i))\, P(x(i)|\mathbf{Y}(i)) \end{aligned} \tag{4.1}$$

Where $I[x, y|Y] = \frac{P(x,y|Y)}{P(x|Y)P(y|Y)}$ is the conditional mutual information.

An alternative approach is to utilize this distribution directly to obtain an estimate of the path in phase space. We start with the observation that for any functional $G(\mathbf{X})$ which depends on the trajectory of the system, the expected value of $G(\mathbf{X})$, given the observations, is given by:

$$\langle G(\mathbf{X})|\mathbf{Y}\rangle = \frac{\int d\mathbf{X}\, G(\mathbf{X})P(\mathbf{X}|\mathbf{Y})}{\int d\mathbf{X}\, P(\mathbf{X}|\mathbf{Y})} \quad = \frac{\int d\mathbf{X}\, G(\mathbf{X})\exp(A(\mathbf{X},\mathbf{Y}))}{\int d\mathbf{X}\, \exp(A(\mathbf{X},\mathbf{Y}))} \tag{4.2}$$

Where $P(\mathbf{X}|\mathbf{Y}) = \prod_m P(\mathbf{x}(m)|\mathbf{Y})$. $A(\mathbf{X},\mathbf{Y}) = \log(P(\mathbf{X}|\mathbf{Y}))$ is referred to as the *Action*, in analogy with the physical path integral; it is equal to the log probability.

This form of the expected value is familiar to physicists. When we there the data $\mathbf{Y}$ has high accuracy (little noise) the probability distribution should become sharply peaked. This represents that there is only one path consistent with the trajectory as observed. This is analogous to the principle of least action, which states that the trajectory of a moving body must minimize the action. When the recordings become noisy, however, there may be many trajectories with a reasonable likelihood. Then the expected value becomes more like the quantum-mechanical path integral, which must integrate over several possible trajectories to give the average value.

4.0.4 Form of the Action

To write an explicit mathematical form for the action, we start by first considering the transition probability $P(\mathbf{x}(i+1)|\mathbf{x}(i))$ in Eq 1.12. This probability depends on the dynamical model we have chosen. We take the model to have the general form:

$$\mathbf{g}(\mathbf{x}n+1, \mathbf{x}(n), \mathbf{p}) = \eta(n) \tag{4.3}$$

Where η represents stochastic components in the equations.

Writing the model explicitly is slightly less general, but a bit easier to read:

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{x}(n), \mathbf{p}) + \eta(n) \tag{4.4}$$

$$P(\mathbf{x}(n+1)|\mathbf{x}(n)) = \delta^D\left(\mathbf{x}(n+1) - f(\mathbf{x}(n), \mathbf{p})\right), \quad \eta = 0 \tag{4.5}$$

When the model is fully deterministic $\eta = 0$, and the transition probability is simply:

$$P(\mathbf{x}(n+1)|\mathbf{x}(n)) = \delta^D(x(n+1) - \mathbf{f}(\mathbf{x}(n), \mathbf{p})) \tag{4.6}$$

Which states that the only possible value of $\mathbf{x}(n+1)$ is the one mapped to by the model equations g from $\mathbf{x}(n)$ has any likelihood of being state at the next time point.

If η is presumed to be a Gaussian distribution, then the transition probability broadens around $\mathbf{x}(n+1)$, because the previous state does not deterministically determine the state at the subsequent time:

$$\begin{aligned} P(\mathbf{x}(n+1)|\mathbf{x}(n)) &= P\Big(\mathbf{f}\big(\mathbf{x}(n+1)\big) + \eta\Big) \\ &= P_\eta\Big(\mathbf{f}\big(\mathbf{x}(n+1)\big) - \mathbf{x}(n+1)) \\ &= C\exp\Big[\big(x_a(n+1) - f_a(\mathbf{x}(n))\big)\, R^f_{ab}\, \big(x_b(n+1) - f_b(\mathbf{x}(n))\big)\Big] \end{aligned} \tag{4.7}$$

The diagonal elements of the matrix R^f_{ab} represent the variance in the dynamical noise η, for a particular variable. The off-diagonal elements in this matrix represent cross-correlations in the dynamics between different states. For simplicity, we typically take these off diagonal entries to be zero, however by incorporating the Jacobian in a manner similar to Kalman Filters [18, 19] there may be additional information to extract by introducing an approximation of these cross-correlations.

The relationship between the probability distributions of η and the dynamics variables $\mathbf{x}$ can be seen with the following identity for the probability distribution of a sum of random variables

$Z = X + Y$:

$$P_Z(z) = \int dY P_x(z - Y) P_y(Y) \tag{4.8}$$

Applying this to the distribution of a trajectory in path space gives:

$$\begin{aligned} P_x(\mathbf{x}^{n+1}|\mathbf{x}^n) &= P_x(\mathbf{f}(\mathbf{x}^n) + \eta|\mathbf{x}^n) \\ &= \int d\hat{\mathbf{f}}\ P_\eta(\mathbf{x}^{n+1} - \hat{\mathbf{f}}|\mathbf{x}^n) P_{f(x)}(\hat{\mathbf{f}}|\mathbf{x}^n) \\ &= \int d\hat{\mathbf{f}}\ P_\eta(\mathbf{x}^{n+1} - \hat{\mathbf{f}}|\mathbf{x}^n) \delta^D(\hat{\mathbf{f}} - \mathbf{f}(\mathbf{x}^n)) \\ &= P_\eta(\mathbf{x}^{n+1} - \mathbf{f}(\mathbf{x}^n)) \\ &= C \exp\left[\left(x_a(n+1) - f_a(\mathbf{x}(n))\right) R^f_{ab} \left(x_b(n+1) - f_b(\mathbf{x}(n))\right)\right] \end{aligned} \tag{4.9}$$

Where $P_\eta = C \exp[\eta_a R^f_{ab} \eta_b]$ is the Gaussian probability distribution of the noise term η.

The other important factor in 4.1 is conditional mutual information term, which adjusts the probability of a trajectory based on the observations.

$$I[\mathbf{x}(n), \mathbf{y}(n)|\mathbf{Y}(n-1)] = \log\left[\frac{P(\mathbf{x}(n), \mathbf{y}(n)|\mathbf{Y}(n-1))}{P(\mathbf{x}(n)|\mathbf{Y}(n-1)) P(\mathbf{y}(n)|\mathbf{Y}(n-1))}\right] \tag{4.10}$$

The CMI can be simplified when it's used to calculate the path integral in 4.2. Note that the term in the denominator depends only on the observations $\mathbf{Y}$ and as a result, can be brought out of both integrals and will cancel. If we neglect that term, the remaining fraction can be rewrtten:

$$\begin{aligned}\exp\left[CMI(\mathbf{x},\mathbf{y}|\mathbf{Y})\right] &= \frac{P(\mathbf{x}(n),\mathbf{y}(n)|\mathbf{Y}(n-1))}{P(\mathbf{x}(n)|\mathbf{Y}(n-1)P(\mathbf{y}(n)|\mathbf{Y}(n-1)))} \\ &\rightarrow \frac{P(\mathbf{x}(n),\mathbf{y}(n)|\mathbf{Y}(n-1))}{P(\mathbf{x}(n)|\mathbf{Y}(n-1))} \end{aligned} \tag{4.11}$$

$$\frac{P(\mathbf{x}(n),\mathbf{y}(n)|\mathbf{Y}(n-1))}{P(\mathbf{x}(n)|\mathbf{Y}(n-1))} = P(\mathbf{y}(n)|\mathbf{x}(n),\mathbf{Y}(n-1)) \tag{4.12}$$

$$= P(\mathbf{y}(n)|\mathbf{x}(n)) \tag{4.13}$$

Eq 4.12 is simply Bayes theorem, which inverts the conditional probability distribution. Eq 4.13 represents the fact that given the underlying state $\mathbf{x}(n)$, measurements of that state are independent of states at previous times. In layman's terms this is the statement that the probe works the same at each time point, and doesn't degrade or depending on what it has previously observed.

The relationship between the measurements is straightforward:

$$\mathbf{y}(n) = \mathbf{h}(\mathbf{x}(n)) + \epsilon \tag{4.14}$$

Where similar to the η in the dynamics, ϵ is the noise in the measurement instruments, which causes the observations $\mathbf{y}$ to vary from the exact value given the state, $\mathbf{h}(x)$. Note that throughout this work I assume a direct measurment of the state variables, to simplify the equations: $\mathbf{y}(n) = \mathbf{x}(n) + \epsilon$.

With the relationship between $\mathbf{y}$ and $\mathbf{x}$, we use the same argument as 4.9 to show

show that, for a Gaussian distribution of the measurment error ϵ:

$$\begin{aligned} P(\mathbf{y}(n)|\mathbf{x}(n)) &= P_\epsilon(\mathbf{y}(n) - \mathbf{h}(n)) \\ &= D \exp\left[\big(\mathbf{y}(n) - \mathbf{h}(\mathbf{x}(n))\big)_a R^m_{ab} \big(\mathbf{y}(n) - \mathbf{h}(\mathbf{x}(n))\big)_b\right] \end{aligned} \tag{4.15}$$

Where again R^m_{ab} represents the crosscorrelation matrix of observing different components of the states through $\mathbf{h}(\mathbf{x})$. D is just a normalization constant which cancels when evaluating the path integrals in the numerator and denominator of 4.2

Combining the expressions from Eq 4.7 and Eq 4.15 we get the form for the residual action (written simply A, because its the quantity we care about):

$$A(\mathbf{X}|\mathbf{Y}) = \sum_{n=1}^{N} -\log[P(\mathbf{y}(n)|\mathbf{x}(n))] - \log[P(\mathbf{x}(n+1)|\mathbf{x}(n)) \tag{4.16}$$

$$= \sum_{n=1}^{N} -\big(x_a(n+1) - f_a(\mathbf{x}(n))\big)\, R^f_{ab}\, \big(x_b(n+1) - f_b(\mathbf{x}(n))\big) \tag{4.17}$$

$$- \big(y_a(n) - h_b(\mathbf{x}(n))\big) R^m_{ab} \big(y_a(n) - h_b(\mathbf{x}(n))\big) \tag{4.18}$$

$$= \sum_{n=1}^{N} R^f \sum_{i=1}^{D} (x_i(n+1) - f_i(\mathbf{x}(n)))^2 + R^m \sum_{j=1}^{L} (y_j(n) - f_j(\mathbf{x}(n)))^2 \tag{4.19}$$

The last line Eq 4.19 is a simplifications for when the error covariance matrices R^f and R^m are diagonal and constant (effectively reducing them to a scalar number). This form is the easier to read. The most probabilistic path will be the one with the lowest action. We can see above that minimizing the Action is essentially a least squares minimization of two competing terms. One term is the difference between the model variables $\mathbf{X}$ to the data $\mathbf{Y}$. The other is between state at each time, $\mathbf{x}(n)$, and the state which is mapped to from the previous time, $\mathbf{f}(\mathbf{x}(n-1))$.

Because the model $f(x)$ is not necessarily linear, the action is not necessarily convex in the state variables $\mathbf{X}$. This is why the data is so important. The measurement term $(y_i - x_i)^2$ *is* convex, and presumably also follows the model approximately. With enough of measurement terms the action becomes convex around a few very likely paths. This makes it possible for a numerical routine to actually find a consistent trajectory in the entire space of $\mathbf{X}$, which typically numbers in the thousands of variables.

4.0.5 Surface of the Action

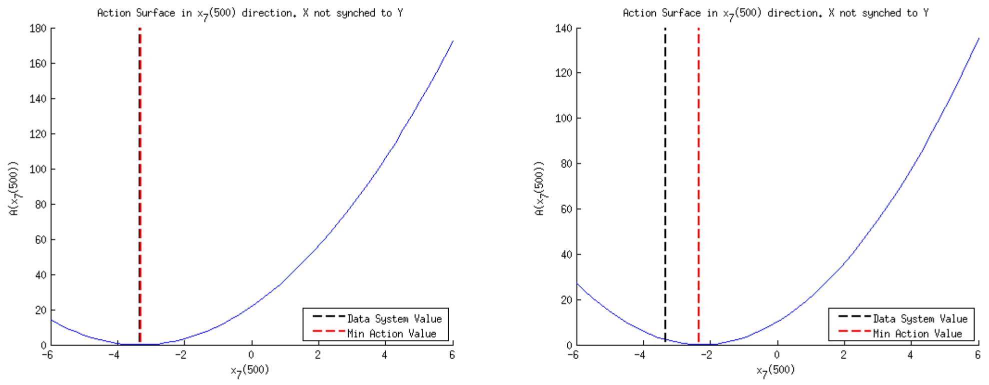


Figure 4.1: Action surface in $x_7(500ms)$ dimension. Left: Plot when other variables and time points are synchronized. Right: When path $\mathbf{X}$ is unsynchronized.

Figure 4.1 shows plots of a single variable in the path integral (one state variable at one time point) and how the value of the action changes as one variable is varied while the others are held fixed. This is an attempt to visualize the high dimensional ($N_T D$-dimensions) path space. Here I have reduced it to one dimension, a single entry $x_7(500ms)$ in the path-vector $\mathbf{X}$ (Eq 1.6). There are still many places in space where we can look along this dimension. Fig 4.1 shows plots of the action at two locations: the left is when all the other variables are near the correct values in the data system, the right is when the other variables are given some random values. Notice at when the other variables are given correct value then the minimum of the action along the $x_7(500)$ dimension is correctly located, but not when the other variables are at random values. As some of the variables approach their true values it becomes easier to optimize the ones which remain, further demonstrating the value of additional measurements.

Previous work [Kostuk, Quinn] claimed that the value of the action created a fractal surface as one variable varies. However, Fig 4.1 shows a smoothly varying, seemingly convex surface. The difference is that the plots in previous works did not really show the action in the context of an optimization problem. Instead, they plotted the synchronization error (essentially just the R_m term of the action) *after* integrating the entire trajectory with different initial conditions. Here we vary each variable independently, with no integration, and include the dynamics term. In that one variable the action looks convex.

Convexity is not the end of the story, however. Fig 4.1 shows that even though the surface is convex, the minimum of the action is not the correct underlying value (i.e. does not match the data system). This is because the action depends the state at multiple time points simultaneously. If we plot the action over state a at time t_b ($x_a(b)$), the location of the minimum depends on the other variables $x_i(n)$ ($i \neq a,\ n = \{b-1, b+1\}$). When the neighboring state variables are at their correct value (Fig 4.1, right) then indeed, the minimum occurs at the correct value as well. However, when the rest of the path is drawn from unrelated initial conditions (Fig 4.1, left) then the value of $x_a(b)$ which minimizes the action occurs far from the underlying data value.

I believe in practice, the idea that having sufficient data allows us to distinctly identify the minimum is inaccurate. The minimum is still very hard to place accurately unless the variables at *all* times are already very close to their correct values. Instead, the way an optimization routine finds the lowest action is through iteratively varying the values at every time point. I think the importance of observations is the more subtle and complex. The measurments make it mroe likely for this iterative process to *converge* to the data by repeatedly moving the variables at all time points. After moving all the variables to these incorrect minima, the interactions between them change the action surface in each dimension.

4.1 Laplace's Method

The most common way to evaluate the average of a quantity $\langle G(\mathbf{X}) \rangle$ in Eq 4.2 is to approximate it by Laplace's Method [20, 21]. The Laplace approximation utilizes the fact that integrand which includes exponential with a large coefficient has the leading-order behavior

$$I(x) = \int dx\, f(x) e^{\alpha g(x)} \tag{4.20}$$

$$\sim \int_{x_0-\epsilon}^{x_0+\epsilon} dx\, f(x_0) e^{\alpha(x_0+b(x-x_0)^2} \text{ as } \alpha \to \infty \tag{4.21}$$

$$\sim e^{\alpha x_0} f(x_0) \int_{x_0-\epsilon}^{x_0+\epsilon} dx\, e^{\alpha b(x-x_0)^2} \text{ as } \alpha \to \infty \tag{4.22}$$

Where $b = \frac{1}{2}\frac{d^2 g(x_0)}{dx^2}$. The claim is that because the term $\alpha g(x)$ is exponentially large, only points in the infinitesimal vicinity of the maximum of $g(x)$ will contribute to the integral.

In discrete time Eq 4.2 becomes $N \cdot D$ separate integrals. Nevertheless Laplace's method follows the same logic. The problem then reduces to finding a minimum of the action in the multi-dimensional path space

$$\left.\frac{\partial A(\mathbf{X})}{\partial \mathbf{X}}\right|_{\mathbf{X}=\mathbf{X}_0} = \mathbf{0} \tag{4.23}$$

Recall that in the expression of the action, Eq 4.19:

$$A(\mathbf{X}) = \sum_{n=1}^{N} R^f \sum_{i=1}^{D} (x_i(n+1) - f_i(\mathbf{x}(n)))^2 + R^m \sum_{j=1}^{L} (y_j(n) - f_j(\mathbf{x}(n)))^2$$

The coefficients R_m and R_f represent the inverse of the variance (covariances in the general case) in the observation and model errors respectively. For the Laplace approximation to be valid these coefficients must be very large. This criteria will be met to higher accuracy the smaller the noise is in both the observations and model dynamics.

This should not be surprising. The smaller the variance of the noise the more exactly we can enforce a match between the data and model. In the limit that there measurements and equations are exact we would expect only a perfect copy of the data system state to able to reproduce a the observed projection in the measured variables. This is of course true so long as there is no degeneracy, such as a limit cycle or fixed point, in phase space. If there were, an exact trajectory in one or a few variables could be reproduced even if the unobserved variables were different.

In practice, we utilize existing optimization programs to find the trajectory $\mathbf{X}_0$ which minimizes the action. We begin with some randomly chosen initial values for the state variables at different times. Then the optimization algorithm uses the gradient and Hessian of the action function to adjust the values of the state variables at each time independently until the value of entire action function converges to a local minimum. The simplest approach to simply take the gradient of the action with respect to each variable at each time and step down that gradient. Typically optimization algorithms can only guarantee a local minimum. More advanced algorithms will have different approaches for getting out of a shallow well if there is a deeper minimum nearby.

This limitation emphasizes the importance of having as many measurments as possible. Note that the measurement term in the action $R_m(y(n) - x(n))^2$ is quadratic and can be trivially minimized. More observations mean more of the model variables will be driven towards the vicininity of the corresponding data values, making it much more likely that the final answer will be the true solution. Another way to verify that the result of the minimization is a global minimum is addressed in the following section.

4.2 Annealing

One limitation with Laplace's Method is that a minimization algorithm typically returns a single path meeting the critera in Eq 4.23. However, in general there will be multiple possible paths which represent local minima in the $N_T D$-dim path space, and therefore potential solutions of the minimization routine. If the noise is too large or the observed variables do not contain sufficient information, then several of these trajectories may have comparable values of the action. In this case the procedure would be inconclusive. Perhaps we could average over the different minima to obtain an estimate. In any case we want to know if we can be confident that the minimum path found is truly the global minimum of the action. This would mean it represents the underlying trajectory that the data system takes, as accurately as possible.

Two techniques can be combined to answer this question. First we utilize an ensemble approach: we initialize the search for the minimum with multiple different initial conditions in the minimization algorithm. Because each search starts from a different region of path space if there are multiple equivalent minima we expect to find several of them.

There is still the problem that not all minima are created equal. Some may be very narrow and difficult to locate; even if they represent a lower value of the action (i.e. higher probability path). Such minima can be hard to access if a being slightly away from the minima results in a large value of the action.

To address this problem we combine the ensemble approach with an annealing technique. The name comes from the similarity to simulated annealing, where in a parameter in the cost function (traditionally, the temperature) is slowly adjusted as space is explored [22–24]. In this case the parameter we tune is R_f. We start with a very small number, making the measurement term R_m almost exclusively dominant in the action. Once a minimum is found we increase R_f by a multiplicative factor, using the previous minimum as an initial point, and minimize again. We continue increasing R_f exponentially until it is much larger than R_m and dominates the action.

Fig 4.2 and 4.3 show typical plots of the action values as R_f increases. As expected when R_f is small the value of the action increases exponentially. Even though each path is different, there does not appear to be much variation in the value of the action. This is because when R_f is small the minima are degenerate. The measurement term R_m quickly brings the observed components in line with the recorded data y, because R_f is not large enough to impose any significant constraint on the other variables.

At intermediate R_f, however, an interesting phenomena takes place. Even though the coefficient on the model error keeps increasing exponentially the value of the action at the lowest minima plateaus. This suggests that the error $(x(n+1) - f(\mathbf{x}(n)))$ is so close to zero that even when the coefficient increases exponentially the entire quantity

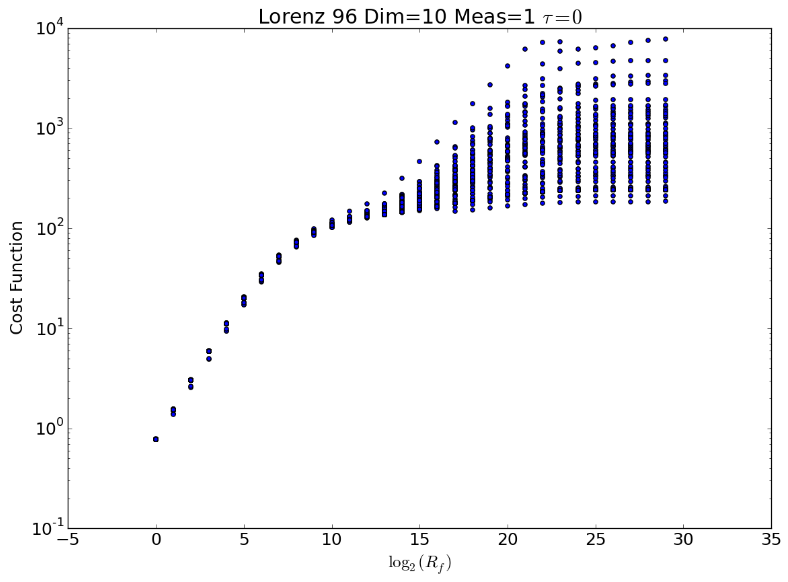


Figure 4.2: Plot of action levels for 100 randomly chosen initial paths (every $x_i(n)$ chosen randomly). Only 1 measurement available from data system $\mathbf{y} = \{y_1\}$. Measurement noise drawn from Gaussian dist: $N(\mu = 0, \sigma = 0.5)$

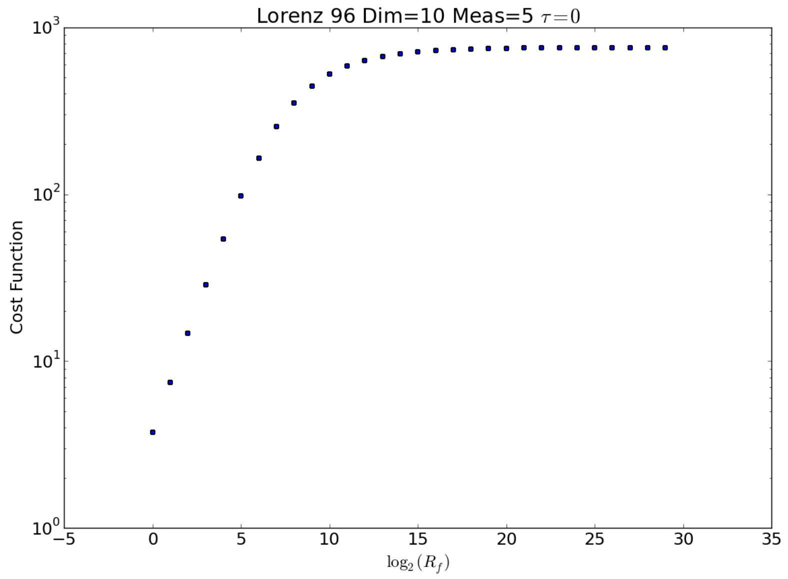


Figure 4.3: Plot of action levels for 100 randomly chosen initial paths (every $x_i(n)$ chosen randomly). 5 measurements available from data system $\mathbf{y} = \{y_1, y_3, y_5, y_7, y_9\}$. Measurement noise drawn from Gaussian dist: $N(\mu = 0, \sigma = 0.5)$

$R_f\left(x(n+1) - f(\mathbf{x}(n))\right)$ remains constant.

Fig 4.2 shows the results of the annealing procedure when a $D = 10$ Lorenz 96 system, with only 1 observed variable ($L = 1$). This is typically too little information to achieve an accurate estimate of the unobserved trajectories. Fig 4.3 was made with 5 observed variables, which should be sufficient for accurate estimation. The most obvious difference between the plots is that the latter has a clearly distinct minimum path. Regardless of the initial conditions all the paths converge to the same local minimum, suggesting it is in fact the global minumum. On the other hand, the case with insufficient observations results in a broad spread of of different action valued paths. Because there is not sufficient information in the measured components, paths that are initialized in different regions of path space get caught in whatever local well they are near.

It is sometimes possible that the lowest action level path is in fact a good estimate of the overall state, even if there are many nearby minima. However in order to find it we may need to run the optimization a hundred different times, as we did in Fig 4.2. The fact that several plausible paths were found tells us that the result is ambiguous and may not be useful with experimental data, outside the context of a twin experiment. On other hand if there is clearly one path whose action value is orders of magnitude lower than any of the other local minima, we can have some confidence that it truly represents the state of the target data system.

We can see what the lowest action level actually represents in Fig 4.4 and 4.5. Both figures show two plots: one for a measured variable $x_1(t)$ and one for an unmeasured variable $x_2(t)$. Fig 4.4 shows the lowest action trajectory from Fig 4.2, when only x_1 was observed in the data system. Fig 4.5 shows the lowest trajectory when x_1, x_3, x_5, x_7, x_9 are observed and incorporated in to the measurement term of the action. In both cases all of the observations were simulated with Gaussian “measurement noise” $N(0, 0.5)$ added to $\mathbf{y}(t)$ after the data system was fully integrated.

The main thing to note in the two trajectories is that in both cases the measured variable is well estimated. This means that just looking at the available data is not going to be very helpful. In a real experiment, x_1 is the only information we have access to and there’s no good way to distinguish from that whether the entire state is accurately estimated. Fig 4.5 shows what is happening beneath the surface, so to speak. Even though both paths represent the lowest action out 100 initial conditions, when there are sufficient measurements the optimization is able to estimate even the unmeasured components via the model term $R_f(x_i(n+1) - f_i(\mathbf{x}))^2$.

This is one of the main advantages of the annealing procedure. The qualitative difference in the distribution of the action minima is clear between Fig 4.2 and 4.3. The latter has a broad spread of different action levels, suggesting the information in the available measurements is insufficient to find an unambiguous minimum. However when the observations *are* sufficient, as in Fig 4.3, all of the initial conditions collapse to a single global minimum. Although there’s no mathematical guarantee, this indicates that we have likely found the correct trajectory taken by the data system. If all the variables have been well estimated, this latter situation will allow us to integrate the model forward to make predictions of the data system.

4.3 $\log(P)$ as Entropy, rather than Action

Throughout this section I have referred to the negative log probability distribution as the “Action”. Our group framed the problem this way for several years to draw attention to the analogy of performing a path-integral over all possible trajectories. The path

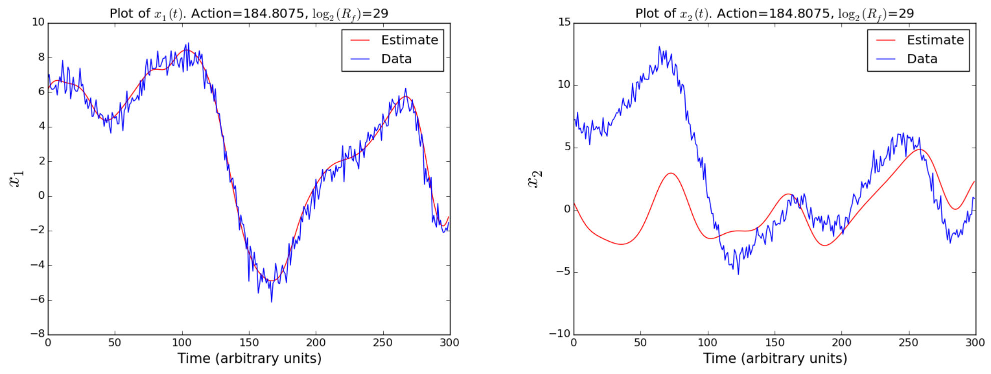


Figure 4.4: Plots of the x_1 and x_2 variables for both the data and model system in a Lorenz 96 10-dimensional system. Trajectories are taken from the largest R_f value of the annealing procedure in Fig 4.2. Only x_1 was observed, this was insufficient to accurately estimate x_2.

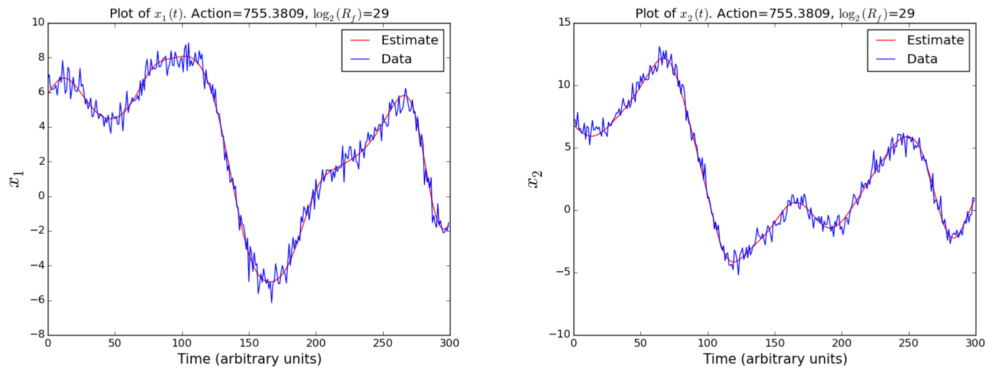


Figure 4.5: Plots of the x_1 and x_2 variables for both the data and model system in a Lorenz 96 10-D system. Taken from the largest R_f value of the annealing procedure in Fig 4.3. x_1, x_3, x_5, x_7, x_9 were observed, this was enough to estimate x_2 and other unobserved variables.

integral and Laplace's approximation can even connect to both the classical and quantum mechanical action distinctly. The measurment and observation noise play the role of $\hbar$. In the limit of small variance noise we solve the problem via Laplace's method, which only considers the lowest action value, analogous to Hamilton's Least Action Principle. When the noise is larger (R_m, R_f not small) we can attempt to estimate the probability distribution via Monte Carlo sampling, in which case we are considering contributions from all (or at least, many) possible trajectories, which must be done in quantum mechanics.

It has occurred to me that there may be an alternative analogy. I have not thoroughly explored this idea, but mention it here in case future researchers find it an inspiring or fruitful direction to pursue. Although the path integral we perform brings to mind an action, the very definition of the action $A(\mathbf{X}) = -\log(P(\mathbf{X}))$ is very similar to the traditional the definition of entropy.

I started by thinking that the thermodynamic entropy is traditionally given as $S = k_B \log(\Omega))$, where Ω is the number of microstates. If each microstate is equally probable, then

$$p = \frac{1}{\Omega} \tag{4.24}$$

$$S = -k_B \log(p) \tag{4.25}$$

While these forms look superficially similar, they depend on rather different things. The "action" we use depends on the probability of realizing a trajectory in path space, given some recordings. The thermodynamic entropy depends on the probability of being in one of many identical microstates. In the former case, we are looking for a unique path with a large probability distinct from other alternatives. In the latter, we are typically moving towards a macrostate with as many identical microstates as possible. Indeed, the goal in data assimilation is to minimize the quantity we've called "action", whereas entropy is a quantity that almost by definition must go towards a maximum.

A better way to relate these quantities instead is to view the likelihood of a given path $P(\mathbf{X})$ as being like the probability of the macrostates of a thermodynamic system, which is exactly proportional to the number of microstates Ω. If $P(\mathbf{X}) \sim \Omega$, then *minimizing* $A(\mathbf{X}) = -\log(P(\mathbf{X}))$ is effectively the same as *maximizing* $S = k_B \log \Omega$.

My interpretation of this analogy is that each trajectory of the model system in path space is like a distinct macrostate. Different instantiations of noise in either the observations or the dynamics result in a variety of possible microstates that are all related to the fundamental path. When the model system path is closer to the data systems trajectory, then more of these different instantiations of noise will be consistent with the recordings. The path that's consistent for widest variety of noise is the most likely to be an accurate reflection of the data system.

Chapter 5

Incorporating Time Delays into Action

A typical scenario in data assimilation involves a model of D differential equations, which can be discretized into a map from one time point to another:

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{x}(n))$$

along with L observations of these variables. In general the observations can arbitrary functions of the variables, but for simplicity we will assume we have direct observations, $\mathbf{y}(n)$, which except for noise represent the exact state variable:

$$\mathbf{y}(n) \approx \mathbf{x}(n)$$

The Path Integral approach typically consists of optimizing cost function which is a function of the entire trajectory

$$\mathbf{X} = \{x_1(n=1), x_2(1), \ldots, x_D(1), x_1(2), \ldots x_D(n=N)\}$$

as well as the measured observations, $\mathbf{y}$.

A standard form for the cost function in these procedures is

$$A(\mathbf{X}) = \sum_{n=1}^{N} \left(\frac{1}{2} R_m \sum_{\ell=1}^{L} (x_\ell(n) - y_\ell(n))^2 + \frac{1}{2} R_f \sum_{i=1}^{D} (x_i(n+1) - f_i(\mathbf{x}(n)))^2 \right)$$

In previous works, we have investigated the relation between this cost function and the probability distribution of all the possible paths. We formulated probability of the path in a form of a path integral and from this context we referred to the cost function as the *action*, in a sense analogous to the action in Hamiltonian and Quantum Mechanics. Minimizing this function will ideally result in a path which matches observations in the measured components and which obeys the model throughout the entire trajectory.

However, if the number of measurements L is too small the manifold of the action as a function of the path will be highly irregular and difficult, if not impossible to search over. By modifying the action, my intention is to regularize the action and make it more conducive to an optimization search. Equivalently, one could say that the goal is to extract more information from the measurements, or use the information more efficiently, to make it possible to find the minimum of the action.

The modification I propose is to add on to the previous form of the action additional terms using time-delay measurements:

$$A(\mathbf{X}) = \ldots + \frac{1}{2} R_\tau \sum_{\ell=1}^{L} \sum_{k=1}^{D_M} \left(f_\ell(\mathbf{f}^{\tau_k - 1}(\mathbf{x}(n))) - y_\ell(n + \tau_k) \right)^2$$

Where D_M is the total number of time delays one wishes to use, and τ_k is the total number of time steps in one delay. $\mathbf{f}^{\tau_k - 1}(\mathbf{x}(n))$ is shorthand for $\mathbf{f}(\mathbf{f}(\mathbf{f}(\ldots \mathbf{f}(\mathbf{x}(n)))))$ which is to say, τ_k recursive applications of the map.

The general idea is add a cost if the unmeasured variables (the term $f_\ell(\mathbf{f}^{\tau_k-1}(\mathbf{x}(n)))$ depends on the entire state vector at $t = n$) are inconsistent with the measurements at a later time.

5.1 Theory

To learn the impact of the time delay term in the action, we find it instructive to look at the gradient. Most optimization algorithms will utilize the gradient to inform which direction the algorithm should search in to find a minimum.

With traditional cost function the derivative with respect to a state variable at a given time can take one of two forms, depending on whether there is a measurement of that state.

For measured states:

$$\begin{aligned}\frac{dA(\mathbf{X})}{dx_i(n)} &= R_m\left(x_i(n) - y_i(n)\right) + R_f\left(x_i(n) - f_i(\mathbf{x}(n-1))\right)\\ &\quad + R_f \sum_j \left(x_j(n+1) - f_j(\mathbf{x}(n))\right)\frac{df_j(x(n))}{dx_i}\end{aligned}$$

For unmeasured states there is no R_m term, and as a result the observations y have no direct influence on the gradient:

$$\frac{dA(\mathbf{X})}{dx_i(n)} = R_f\left(x_i(n) - f_i(\mathbf{x}(n-1))\right) + R_f \sum_j \left(x_j(n+1) - f_j(\mathbf{x}(n))\right)\frac{df_j(n)}{dx_i}$$

The addition of the time delay term will change the derivative with respect to an unmeasured state to:

$$\begin{aligned}\frac{dA(\mathbf{X})}{dx_i(n)} &= R_f\left(x_i(n) - f_i(\mathbf{x}(n-1))\right) + R_f\left(x_j(n+1) - f_j(\mathbf{x}(n))\right)\frac{df_j(n)}{dx_i} \\ &\quad + R_\tau \sum_{\ell=1}^{L}\sum_{k=1}^{D_M}\big(f_\ell(\mathbf{f}^{\tau_k-1}(\mathbf{x}(n))) - y_\ell(n+\tau_k)\big) \\ &\quad \times \frac{df_\ell(\mathbf{f}^{\tau_k-1}(n))}{d\mathbf{x}}\frac{d\mathbf{f}(\mathbf{f}^{\tau_k-2}(n))}{d\mathbf{x}}\cdots\frac{d\mathbf{f}(\mathbf{x}(n))}{dx_i(n)}\end{aligned}$$

All the terms like $\frac{d\mathbf{f}}{d\mathbf{x}}$ are matrices, and the products follow the rules of matrix multiplication. It is also understood that $\mathbf{f}^\tau(n)$ really means $\mathbf{f}^\tau(\mathbf{x}(n))$.

The first thing to notice is that we have added a lot of extra terms to the gradient. Previously we had two or three terms, whereas now we have added an additional $L \cdot D_M$ terms, where remember L is the number of measured variables and D_M is the number of time delays we choose to use. Additionally these terms all involve significantly more calculation because at each time step we have to apply the chain rules and multiply another instance of the Jacobian of the map $\frac{df}{dx}$.

However, in exchange for all this computation we are able to adjust the direction of the gradient in the unmeasured variables using information from the observations. My hope is that this additional information produces a smoother surface, particularly along the unobserved directions. As a result, it may be possible to optimize on a surface that was otherwise too irregular.

5.2 Annealing with Time Delay terms in Action

To explore the effects of using the proposed time delay action vs the traditional 4DVar action I have primarily utilized the continuous optimization technique, also referred to as annealing described in section 4.2. This technique consists of adjusting R_f term so that it is initially negligible compared to R_m. With this cost function, we take several initial paths (randomly distributed in path space), and for each one find an optimum using some optimization algorithm. For now, all of these results have been optimized with a L-BFGS quasi-newton algorithm. The precise results may vary if one uses different optimization algorithms, such as IPOpt which was utilized elsewhere.

Once the minimum paths are found for each initial condition the cost function is modified by increasing R_f, typically by a factor of 2x. Then using the previous minimum path as an initial condition the optimization routine is run again and a new minimum path is found for this new cost function. This is repeated several times until R_f is eventually much bigger than R_m (which remains constant the entire time).

We have found that using this procedure it is possible to find global minima in cases where it not be possible the optimization started with R_f comparable in magnitude to R_m and a random initial condition. The easier it is for the optimization to find the global minimum, the larger the fraction of paths which end up there will be. In this way, trying several initial conditions can give us an idea of the likelihood of finding the true minimum path.

5.2.1 Value of R_τ

One question that arises when extending the continuous optimization procedure to the time delayed action is what we should do with a value of R_τ, the coefficient on the time-delay term in the action. As it stands, R_m remains constant and R_f varies over many order of magnitude. Because the time delay term relies on both the model equations, $\mathbf{f}(\mathbf{x}(n))$, and the measured observations, $\mathbf{y}(t)$ errors in either piece will appear in the term.

As a result, my thinking was that R_τ should therefore be bounded by whichever coefficient is smaller (representing less confidence or more error in that part cost function).

A intuitive functional form with this behavior is what we've called the "parallel resistors" form, named after the familiar equation for combining these. R_τ is calculated as follows:

$$R_\tau = \frac{1}{1/R_m + 1/R_f} \tag{5.1}$$

If we recall that the action represents the logarithm of the probability distribution: $P(\mathbf{X}) = \exp\left(-A(\mathbf{X})\right)$, then it is apparent that the R coefficients are just the reciprocal of the variance of a Gaussian error, $R = 1/\sigma^2$. Then the parallel resistor form is simply saying that the variance due to the model error and due to the observations simply add independently to give the variance of the time-delay error:

$$\frac{1}{R_\tau} = \frac{1}{R_m} + \frac{1}{R_f} \tag{5.2}$$

$$\sigma_\tau^2 = \sigma_m^2 + \sigma_f^2 \tag{5.3}$$

5.2.2 Lorenz 96 D = 5

The best place to start with a new technique is with the simplest problem available. In the case of data assimilation this is the Lorenz 96 model, with 5 dimensions. As mentioned in previous sections, this system is chaotic for the chosen parameter value ($\nu = 8.17$) and typically cannot be well estimated without at least 2 measurements. To evaluate the performance of the time delayed action I performed the annealing procedure by gradually increasing R_f from 0.01 to $0.01 * 2^{30}$, with R_τ changing concurrently according to Eq 5.1

above.

Figures 5.1 and 5.2 show the action level plots for a system with no time delay terms and with a single time delay $\tau = 10dt$. Qualitatively, there is not a dramatic difference. Both figures appear to show a clear break between the lowest action level and the others. Fig 5.2 shows some more fragmentation in the larger action levels. However, given that the higher trajectories all represent inaccurate estimates this is not too interesting. Fine-grain separation in the minima is perhaps to be expected due to the added complexity of the function we're evaluating when we include time delay terms.

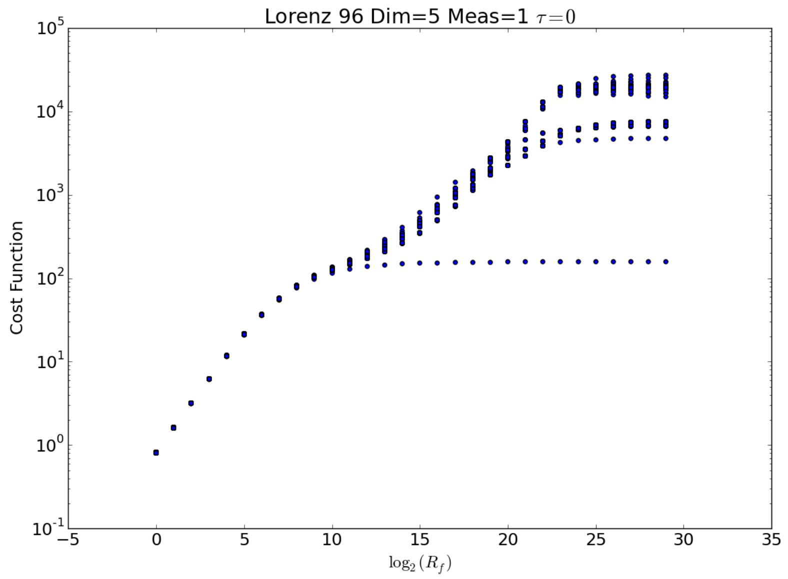


Figure 5.1: Lorenz 96 action levels with no time delay terms, when $D = 5$ and $L = 1$.

The structure of the action levels, is not the whole story however. These plots were made by taking one hundred random initial conditions and seeing which local minima they settled into as we increased R_f. An important question then is, *how many* of the initial conditions actually end at the apparent global minimum, once R_f is large? Table 5.1 summarizes the results for different lengths and quantities of time delays.

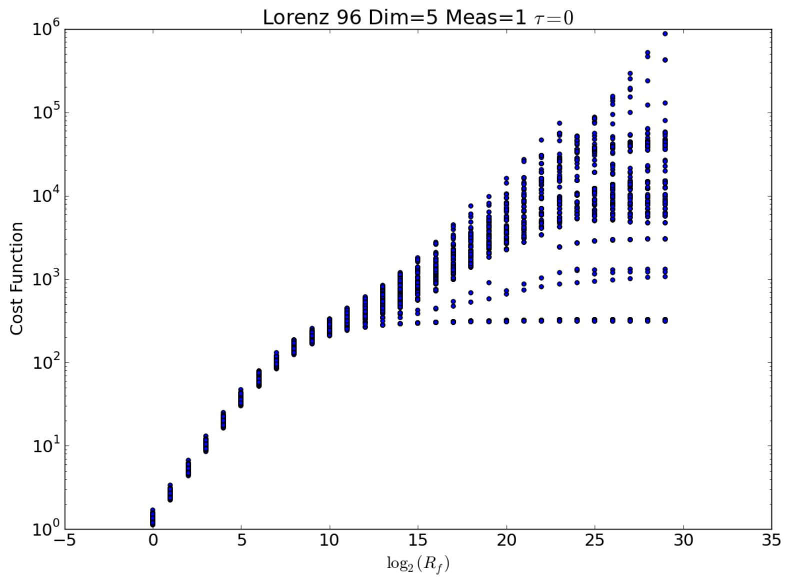


Figure 5.2: Lorenz 96 action levels with a time delay $\tau_k = 10 \cdot dt$, when $D = 5$ and $L = 1$.

The first thing to note is that although the action plot in Fig 5.1 looks reasonable, in fact only 1 of the 100 possible paths actually ended up at the global minimum. This 1% success rate is not feasible for any kind of real estimation. With a slightly different initial condition it is possible none of the paths would have determined good estimates.

However with an appropriate choice of time delays, the success rate can be increased to over 10%. This is a substantial improvement considering no new information was introduced to the procedure. Evaluating the time delayed action and its derivatives for the purposes of optimization certainly requires more computational work, however this is small cost when experimental limitations make obtaining information directly from the system impossible.

Table 5.1: Table showing how many random initial paths (out of 100 initializations) which ultimately arrived at the lowest action value (i.e. the best path estimate)

Time Delays (in units of dt)	Paths at lowest Action level
0	1
5	8
10	11
20	6
30	6
5, 10	12
10, 20	12
5, 10, 15	10
5, 10, 20	10
10, 20, 30	12
5, 7, 9, 11	7

5.2.3 Lorenz 96 D = 10

With a 10 dimensional model, the task of finding a the global minimum becomes much harder. Even with annealing and time delays, I was not able to find an accurate estimate of the entire path. This fail case is reflected the action level plots in Fig 5.3 and 5.4. Unlike the previous section, with the 5 dimensional model, here we see there is no significant separation between the lowest minimum and the other local minima. This suggests that the lowest action path found during this procedure is not qualitatively different than the others. Indeed, Fig 5.5 and 5.6 show the estimated trajectories of an unmeasured variable in the lowest path. Fig 5.5 was estimated with 1 observation and no time delays whereas Fig 5.6 was estimated with 1 observation and 5 time delays. However in both cases the estimates are visibly quite different than the underlying variable in the data system.

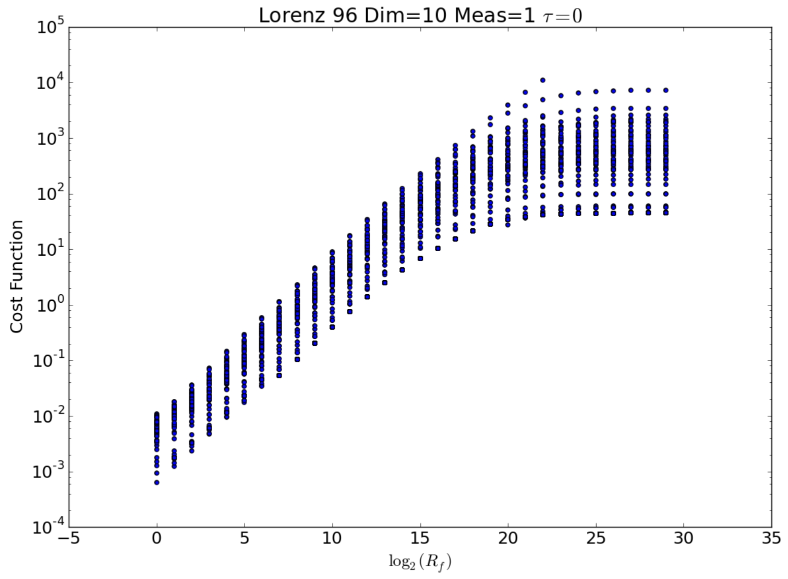


Figure 5.3: Action levels for a 10-D Lorenz 96 system with no time delay terms. No noise was added to data.

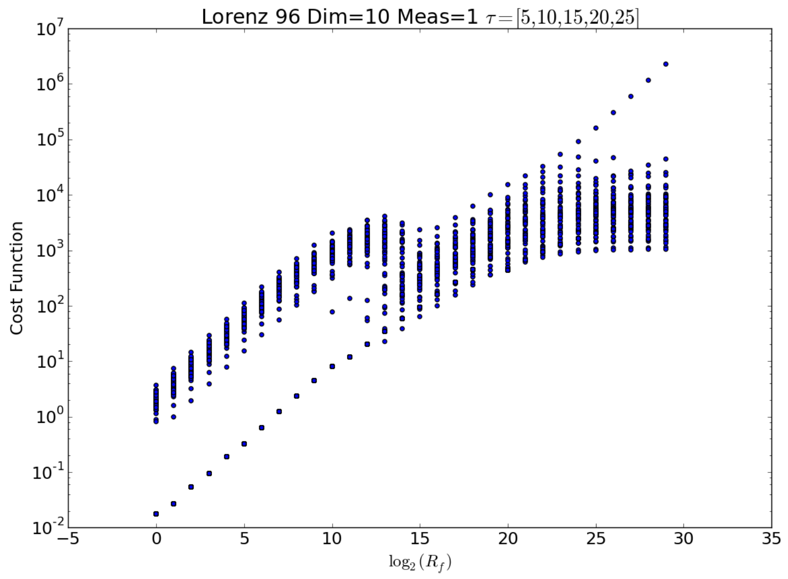


Figure 5.4: Action levels for a 10-D Lorenz 96 system with $\tau = \{5, 10, 15, 20, 25\}$. No noise was added to data

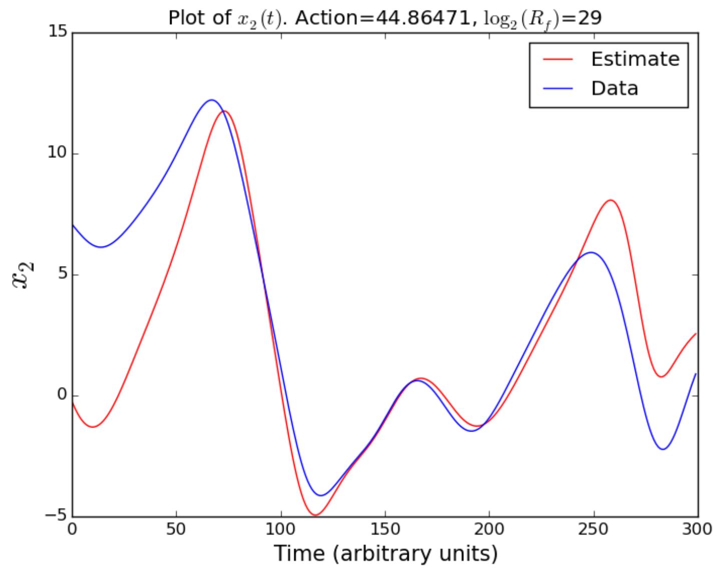


Figure 5.5: Plot of $x_2(t)$, which was not observed, for one of the paths with lowest action level. 10-D Lorenz 96 system, minimized with no time delays.

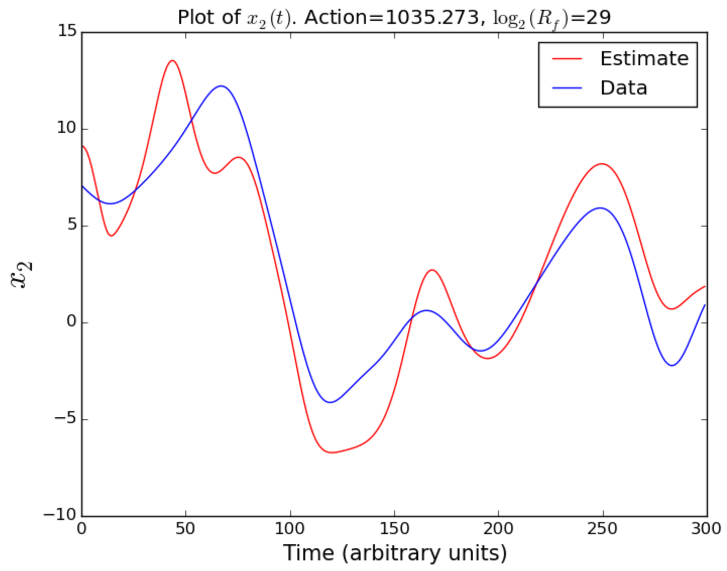


Figure 5.6: Plot of $x_2(t)$, which was not observed, for one of the paths with lowest action level. 10-D Lorenz 96 system, minimized with time delay terms $\tau = \{5, 10, 15, 20, 25\}$.

5.2.4 Lorenz 96 D=10 with two observed variables

In the five dimensional model there is a modest but clear improvement with the inclusion of time delays. Although by no means 100% successful, using a cost function with time delays results in a substantially higher percentage of successful estimation given random initial conditions.

Unfortunately, as the systems become more complicated the benefits of the time delayed action becomes ambiguous. With only 1 observed component even the inclusion of time delays was insufficient to accurately identify a correct action minimum, as shown by Fig 5.5 and 5.6.

The next natural place to look for a difference between the traditional and time delayed cost function is to consider the same 10 dimensional system, but with 2 of the observations recorded. However, here again we fail to distinguish a difference. Fig 5.7 and 5.8 show that with and without time delays 2 observed variables provide enough information that all paths find the correctly estimated global minimum. In this case the additional computational cost of including the time-delay terms becomes unnecessary.

It has been difficult to find a problem sufficiently challenging that annealing the original action function is insufficient, but which is improved by the adding time-delay terms. It is possible that the approach suggested here is incomplete or can be improved further, to broaden the scope of problems where this technique would help. Nevertheless, the results with the Lorenz 96 $D = 5$ system show that there is in fact residual information which is not accessed via the previous action minimization approach.

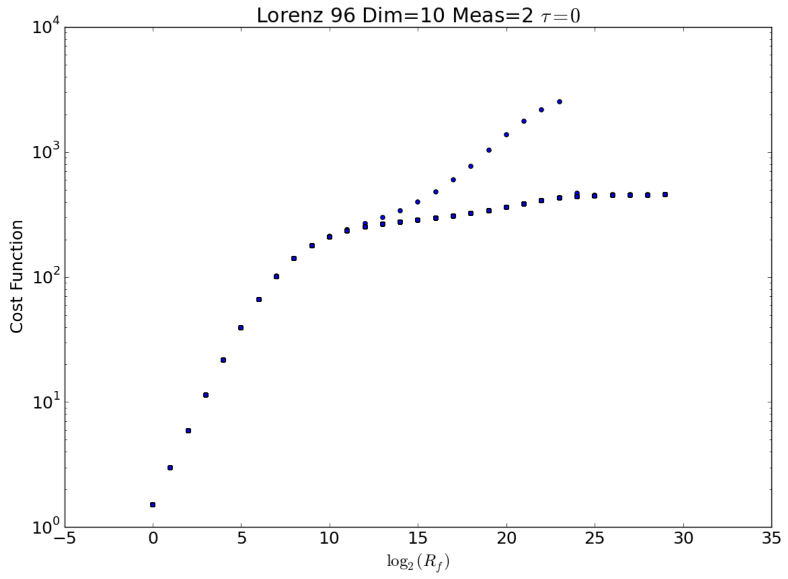


Figure 5.7: Annealing plot for D=10 with L=2, with no time delay terms. All paths end at the global minimum.

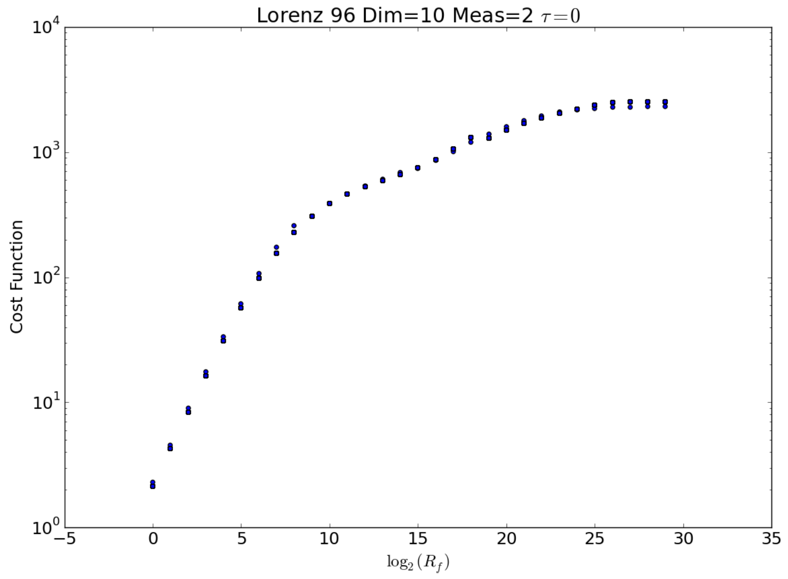


Figure 5.8: Annealing plot for D=10 with L=2, with no time delay terms. All paths end at the global minimum.

Chapter 6

Neuron Models

Neuroscience lends itself naturally to the application of data assimilation. Neurons are cells which communicate with one other by fluctuating the voltage in the interior of the cell relative the exterior. The typical behavior is that voltage will quickly spike from a resting negative potential to high positive value, then back down. This spike will propagate like a wave down along the cell until it reaches a synapse at the end. A synapse is a gap between two adjacent neurons. The voltage spike initiates a release of chemicals into the synapses between neurons and these chemicals prompt a spike in the next neuron's internal voltage. These interactions continue throughout a network of cells, which convey information about everything from sensation and motor function thoughts and emotion [25, 26].

Neuron dynamics depend on the interactions of chemical concentrations and microscopic membrane properties which continually fluctuate. Additionally, they are living cells which must be kept mostly intact to continue functioning. This creates significant limitation on what kind of sensors can be developed. Many of the physiological changes which occur during spiking are at a molecular scale and we have not developed a way of recording these changes on the same time scale as the voltage spiking.

6.1 Neurons in DA

In developing and using methods of statistical data assimilation to characterize the biophysical properties of functional networks of neurons, we have previously built a Hodgkin-Huxley type model (HH model) of the dynamics of individual neurons [27–29] . This model, as all such models, has numerous unknown fixed parameters that must be determined for each class of neuron under consideration. We used well designed stimulating currents for individual neurons in the avian song system nucleus HVC and measured the response of the membrane voltage to estimate the many biophysical parameters in the voltage and kinetic equations of such a model. This estimation 'completes' the model in the sense that once the fixed parameters are established, then given initial conditions for the state variables, observed and unobserved, we can predict the response of the model neuron to a new stimulus [29].

We tested/validated the biophysical HH model by showing that with the estimated parameters it could reliably predict the observed response to new stimulating currents. The required initial conditions for prediction in the completed model were established by using a very short (100 ms) segment of the data set for times beyond the observation window.

The methods we utilize here were quite instrumental in designing the large collection of data sets analyzed in [29], and we expect that to be the case again when we move from simulations of neurons with important Ca dynamics to the design of experiments to explore those dynamics.

Our stimulation/response protocols presented data comprising an applied stimulating current $I_{\text{app}}(t)$ and the observed response membrane voltage $V_{\text{data}}(t)$. These time series alone allow estimation of the fixed parameters and unobserved state variables of the neuron. The latter are the voltage dependent gating variables associated with ionic currents of Na$^+$, K$^+$, and Ca^{2+} ions.

The cost function or objective function representing the error between the data and the model output is taken as

$$\sum_{n=0}^{m}(y_1(t_n) - V(t_n))^2, \tag{6.1}$$

where observations are made at times $t_n;\ n = 0, 1, 2, ..., t_m = T$. This objective function was minimized subject to the deterministic HH equations of motion, given below, as equality constraints dynamically taking the states between time t_n and time t_{n+1}. This cost function is also seen as the error in the synchronization of the model output with the data. It is not always clear that measurement of a single state variable, here membrane voltage, even when very well sampled in time, will suffice for this estimation procedure. Many examples are known of the necessity for more measurements at each measurement time to remove impediments associated with the instability of the manifold in state space where the data and the model output are synchronized [30]. When synchronization fails, the synchronization error Eq. (6.1) has multiple local minima as a function of the parameter or state value sought through the minimization. This impediment to estimation must be regulated to provide a smooth surface on which one implements a search procedure for the minimum of the cost function Eq. (6.1).

The systems where more measurements are required appear to be those where chaotic solutions to the dynamical equations are possible for some biophysically plausible set of stimuli and model parameters. We did not encounter models of this form among those in our analysis of the HVC data [29]. p There is now substantial evidence of the role of Ca^{2+} dynamics in the neurons of HVC [31–35]. This adds a set of rather slow dynamical processes to the much faster voltage gated processes involving Na^{+} and K^{+} ions. When we extended these voltage dynamics models to include the important biophysical processes of Ca^{2+} uptake and release from internal stores, the question of how many measurements are required changes. Mixing slow and fast dynamical processes, coupled nonlinearly to each other, can be a setting for the appearance of chaotic behavior. In the case of voltage plus calcium dynamics, chaos does appear, as we shall discuss, and this leads to a situation where more than voltage measurements alone are required to estimate the system. Our objectives are:

- to synchronize data with model output,
- accurately estimate fixed parameters and unobserved state variables, and
- provide accurate predictions as validation of the consistency of the model with the observations.

The goal of this chapter is to discuss the properties of an HH neuron model with both voltage and Ca^{2+} dynamics to demonstrate that the estimation of model parameters and unobserved state variables when such models are considered will require more than just voltage measurements to complete the model through the use of experimental observations. Afterwards, I will explore the use of information in the waveform of the measurements, via the time embedded coordinates, as a way to succeed in providing sufficient information to the HH model, and thus permit accurate estimation of parameters and states.

6.1.1 Hodgkin-Huxley (1952) model

A common framework for neuron models is to think of them as a capacitor with the cell membrane separating the internal and external voltage. This approach was first used by Hodgkin and Huxley in 1952 [36] A semi-permeable membrane allows ion currents to enter and exit the cell. The voltage dynamics are then:

$$C\frac{dV}{dt} = \sum_{\alpha} I_\alpha(V, \alpha_1, \alpha_2, \ldots) \tag{6.2}$$

The currents are derived from a balance of diffusion and electric repulsion called the GHK equation. They take a general form:

$$I_\alpha(V, \alpha_1, \alpha_2) = g_\alpha \alpha_1^{p_1} \alpha_2^{p_2} (E_\alpha - V) \tag{6.3}$$

The α_i are called gating variables. They are dynamic state variables in the system, representing the proportion of channels for ion α that are open or blocked at a given time. As a result, they vary between 0 and 1. The parameters p_i in the current equations determine how sensitive the current is to ion channels. The parameter E_α is known as the reversal potential, it is the voltage at which the current changes from flowing into the cell to out of the cell.

The gating variables make up most of the non-voltage state variables in the system. The equations are derived by averaging over the stochastic dynamics of individual channels to express the macroscopic behavior in terms of an opening and closing rate. The dynamic equations for these variables can also be written in terms of an equilibrium value and time constant, which in general can both be functions of voltage:

$$\frac{d\alpha_i}{dt} = \frac{\alpha_{i,\infty}(V) - \alpha}{\tau_{\alpha_i}(V)}$$

One of the first models to use these representations of neuron dynamics is the Hodgkin-Huxley neuron. With an appropriate stimulus ($I_{ext}(t)$) this model produces voltage spiking as a limit cycle behavior:

$$\begin{aligned} C\frac{dV(t)}{dt} &= I_{Na}(t) + I_K(t) + I_L(t) + I_{ext}(t) \\ I_{Na}(t) &= g_{Na} m(V(t))^3 h(t)(E_{Na} - V(t)) \\ I_K(t) &= g_K n(t)^4 (E_K - V(t)), \\ I_l(t) &= g_L(E_L - V(t)) \end{aligned} \tag{6.4}$$

A variety of data assimilation techniques have been applied to the Hodgkin Huxley neuron [37–40]. One feature of these equations is that the gating variables all depend very strongly on the voltage. Indeed the dynamics of each gating variable depends only on itself and voltage. This makes a technique like synchronization very effective because if we restrict the voltage in a model to observed time series from the data, then the gating variables are almost guaranteed to line up without any more measurements. The HH neuron does not typically exhibit chaotic behavior, which makes the estimation much easier but may not be reflect real neuron behavior.

6.1.2 The Biophysical Role of Ca^{2+} Dynamics

Calcium ions play an important role in regulating a great variety of neuronal processes. Intracellular calcium signals regulate processes that operate over a wide time range, from neurotransmitter release at the microsecond scale to gene transcription, which lasts for minutes and hours [41]. In the HVC of the songbird, the contributions of calcium channels and calcium mediated events to spiking and bursting have been observed. In vivo, L-type Ca^{2+} bursting activity in HVC_{RA} neurons [35], and calcium transients show strong preference for bird's own song (BOS) in identified HVC neurons where a strong correspondence between calcium signals and juxtacellular electrical activity is exhibited [32]. In vitro, T-type low voltage activated Ca channels are expressed in HVC neurons contributing to their postinhibitory rebound firing [34]. Moreover, the spike frequency adaptation seen in HVC projection neurons is largely due to a Ca^{2+} induced K^+ channel [34, 42]. In addition multiple calcium binding proteins that determine the dynamics of free calcium inside neurons [43] are enriched in the HVC of songbirds [31, 33]. All these studies motivate an investigation of the roles that Ca^{2+} play in the electrical activity of HVC neuronal subpopulations in vitro and during singing.

Calcium is a crucial intracellular messenger in mammalian neurons where the final transduction of any neuronal signal involves the movement of calcium ions. At rest, the intracellular calcium concentration of most neurons is about 50$\sim$100 nM, and that can rise to levels that are ten to 100 times higher during electrical activity [44]. The cytosolic calcium concentration is determined by the balance between calcium influx and efflux as well as by the exchange of calcium with internal stores. In addition, calcium-binding proteins such as parvalbumin or calretinin, acting as calcium buffers, determine the dynamics of free calcium inside neurons. However, and most importantly, only the free calcium ions inside the cytosol are biologically active. Calcium influx from the extracellular space is controlled by various mechanisms including voltage gated calcium channels, ionotropic glutamate receptors, nicotinic acetylcholine receptors (nAChR), and transient receptor potential type C (TRPC) channels [45–48]. The extrusion of calcium ions from the cytosol is done via the plasma membrane calcium ATPase (PMCA) and the sodium-calcium exchanger (NCX) [41]. The release of messenger calcium ions from internal stores, mostly the endoplasmic reticulum (ER), is controlled by the inositol trisphosphate receptors and ryanodine receptors [49]. The high calcium level inside the ER is controlled by the sarco-endoplasmic reticulum calcium ATPase (SERCA) pump that transports calcium ions from the cytosol to the ER. In addition to the ER, mitochondria can also play the role of calcium buffers by absorbing calcium ions during cytosolic calcium elevations via the calcium uniporter and then releasing the calcium ions back to the cytosol slowly via the sodium-calcium exchange [50].

Variation in the intracellular concentration of Ca^{2+} ions, $[\mathrm{Ca}^{2+}]_i(t) \equiv Ca(t)$, is governed by the flow of these ions through the cell membrane via voltage gated channels, as well as by uptake and release by the endoplasmic reticulum (ER) as a storage device. These properties and more are discussed in many research papers [41,51], including a very informative review and summary by Falcke and colleagues [52].

Earlier work on coupling voltage and calcium dynamics [53, 54], as well as the dynamics of Ca^{2+} uptake and release [55] independent of its connection to voltage dynamics of Na^{+} and K^{+} ion flow, provide the foundation for the model we discuss here. The presence of Ca^{2+} channels and their interaction with voltage dynamics in HVC neuron cells has been established by [34]. These results have strongly motivated us to explore the inclusion of Ca^{2+} dynamics in the model utilized in conjunction with our analysis of experiments on

HVC neurons [29].

Ca^{2+} ions are released and stored in the ER predominately through the mediation of inositol 1,4,5-triphosphate (IP_3) in many cell types [52], and we will incorporate a model of these processes in our overall model of the cellular dynamics. This uptake and release via IP_3 mediation has been suggested as a source of Ca^{2+} oscillations in the work of Houart, et al [55], and we adopt their model, with a faster time scale, compared to their original idea of its role in more global rhythms of animals. This is not the only candidate for incorporating calcium dynamics into neuronal processes; we have analyzed two other models of Ca dynamics with some care [53, 54], but we do not report on them here.

Examples of the important role of calcium in intracellular dynamics abound. For instance, the permeability of some potassium channels in the cellular membrance are affected by the presence of calcium, and membrane calcium channels themselves may be voltage-gated [28, 56]. The interplay between these two mechanisms acts to regulate firing patterns [57]. Additionally, calcium is known to be a major determinant in the potentiation and depression of excitatory synaptic strength, and thus is expected to play an important role in memory and learning. This is widely thought to underly how networks of neurons "rewire" and learn. While these processes are certainly important, they act on a much longer time scale than the interacting voltage and calcium processes we include in our model.

The main issues we address in this chapter are those raised when the relatively slow dynamics associated with Ca^{2+} ion uptake and release interact with the much faster voltage dynamics to produce chaotic behavior. These are issues in using methods of data assimilation to estimate parameters and unobserved state variables using time series of observed quantities from experiments.

We do not yet use any of these models in the analysis of experiments in this chapter, but we perform “twin experiments”. In these we generate data by solving the model with known parameters, and then by presenting observables such as membrane voltage $V(t)$ or intracellular Ca^{2+} concentration $Ca(t)$ to the model, we are able to examine our estimation methods in a controlled context.

When additional complex dynamics is introduced into a neuron model, such as the intracellular calcium dynamics we present here, the synchronization manifold of the model may become unstable [12]. This can make it impossible to estimate unknown parameters and unobserved states with voltage measurements alone, raising the following interesting questions:

- How do calcium dynamics affect neuron behavior?
- How many measurements are needed to synchronize such a neuron model with observed data after the introduction of calcium dynamics?
- If other measurements are required in addition to membrane voltage, what can play that role?

We do not have full answers to these questions. However, below we construct a conductance-based neuron model, into which we couple a detailed model of intracellular calcium dynamics. Using this model, we investigate the estimation problems through numerical experiments to give a clear image of the biophysical issues raised via calcium dynamics. We will show that voltage measurements are not enough to ‘cure’ the instabilities just noted, and we will address a solution to this. When we move from simulations to experimental data, we expect to encounter the same issues in a more cluttered context.

6.2 Chaotic Calcium and Voltage model

To try a more difficult challenge we extended the HH neuron to include Calcium dynamics. In addition to new currents in the $\frac{dV}{dt}$ equation, we coupled a chaotic 3-state model of Calcium concentrations between different reservoirs in the cell. The final model consists of 9 state variables: Voltage, 5 gating variables, and 3 distinct Calcium concentrations.

$$\begin{aligned}
C\frac{dV}{dt} &= I_{Na} + I_K + I_l + I_L + I_T + I_{K/Ca} + I_h + I_{app} \\
\frac{d\alpha_i}{dt} &= \frac{\alpha_{i,\infty}(V) - \alpha}{\tau_{\alpha_i}(V)} \qquad \alpha_i \in \{h, n, r_T, r_f, r_s\} \\
\frac{d[Ca]}{dt} &= \nu_0 + \beta_{input}(I_L + I_T) - \gamma_{leak}[Ca] + \nu_{CICR}([Ca], [Ca]_{ER}, [IP_3]) \\
&\quad + \gamma_{ERleak}[Ca]_{ER} - \nu_{pump}([Ca]), \\
\frac{d[Ca]_{ER}}{dt} &= -\nu_{CICR}([Ca], [Ca]_{ER}, [IP_3]) - \gamma_{ERleak}[Ca]_{ER} + \nu_{pump}([Ca]) \\
\frac{d[IP_3]}{dt} &= \nu_{synthesis} - \gamma_{[IP_3]leak}[IP_3] - \nu_{degradation}([Ca], [IP_3])
\end{aligned} \tag{6.5}$$

In the Voltage dynamics $I_{app}(t)$ is an external applied current selected by us, and the other currents are given as

$$
\begin{aligned}
I_{Na}(t) &= g_{Na} m_0(V(t))^3 h(t)(E_{Na} - V(t)) \\
I_K(t) &= g_K n(t)^4 (E_K - V(t)), \\
I_l(t) &= g_L(E_L - V(t)) \\
I_L(t) &= g_{CaL} s_0(V(t))(E_{Ca} - V(t)) \\
I_T(t) &= g_{CaT}[a_{T0}(r_T(V(t)))\, b_{T0}(r_T(V(t)))]^3 (E_{Ca} - V(t)) \\
I_{K/Ca}(t) &= g_{K/Ca} \frac{[Ca](t)^4}{[Ca](t)^4 + \kappa_{K/Ca}^4} \\
I_h(t) &= g_h(0.3 r_f(t) + 0.7 r_s(t))(E_h - V(t)).
\end{aligned}
\tag{6.6}
$$

and for the Calcium model:

$$
\begin{aligned}
\nu_{pump} &= \gamma_{pump} \frac{[Ca]^2}{[Ca]^2 + \kappa_p{}^2} \\
\nu_{CICR} &= \gamma_{CICR} \frac{[Ca]^2}{[Ca]^2 + \kappa_{cCa}{}^2} \cdot \frac{[Ca]_{ER}{}^2}{[Ca]_{ER}{}^2 + \kappa_{cER}{}^2} \cdot \frac{[IP_3]^4}{[IP_3]^4 + \kappa_{cIP_3}{}^4} \\
\nu_{degradation} &= \gamma_{degradation} \frac{[IP_3]}{[IP_3] + \kappa_{dIP_3}} \cdot \frac{[Ca]^4}{[Ca]^4 + \kappa_{pCa}{}^4}
\end{aligned}
\tag{6.7}
$$

This model has the following features:

- Calcium variables ($[Ca]$, $[Ca]_{ER}$, $[IP_3]$) collectively make a chaotic three-state system.
- Calcium influences voltage dynamics primarily through $I_{K/Ca}$, Calcium-gated potassium current.

- Voltage affects $[Ca]$ through Calcium currents, I_T and I_L.
- Voltage and Calcium dynamics occur on different time scales (1-10 ms vs 100-1000 ms). This results in a pattern of "bursting", where the voltage will spike frequently over a period while the Calcium concentration is low, then shut off when it rises above a certain threshold.

The model we used in [29] for describing the experimental data on stimulus/voltage response experiments we have conducted on HVC neurons comprised these currents, absent the $I_{K/Ca}(t)$ current, along with additional Na and K currents. The manner in which some voltage dependent conductances, especially $I_h(t)$, was represented there is different in that model, and the Ca currents used GHK [28] voltage current relations reflective of the 10,000:1 ratio of extracellular to intracellular CA concentrations. As we are concentrating here on the role of the added slow Ca dynamics to be described in a moment, we adopted a subset of the full model used earlier.

When we utilize the lessons from the "twin experiment" analysis of this V+Ca model and select models with which to analyze the observed laboratory data, we will examine several variants, all with the same core issue as explored here, but with somewhat different realizations of the Calcium dynamics. To proceed we select one version of Calcium dynamics: the model of Houart, et al [55].

6.2.1 Coupling Ca^{2+} to voltage dynamics

We introduce a calcium-dependent potassium current $I_{K/Ca}$ as the first ingredient in coupling the voltage and calcium dynamics. The conductance of this channel depends

Table 6.1: Contributions to the kinetics of the voltage gated channels.
$\sigma(x,y,z) \equiv 0.5\left[1+\tanh\frac{x-y}{2z}\right], \quad E(x,y,z) \equiv \exp\left[\frac{x-y}{z}\right]$.

$I_{channel}(t)(\mu A)$	X(t)	$X_0(V)$	$\tau_X(V)(ms)$
$I_{Na}(t)$	m(t)	$\sigma(V(t),-35,-5)$	—
$I_{Na}(t)$	h(t)	$\sigma(V(t),-37.4,4.3)$	1
$I_K(t)$	n(t)	$\sigma(V(t),-30,-5)$	$10[\cosh(\frac{V(t)+30}{10})]^{-1}$
$I_{CaT}(t)$	$a_T(V(t))$	$\sigma(V(t),-65,-7.8)$	—
$I_{CaT}(t)$	$b_T(r_T(t))$	$\sigma(r_T(t),0.4,-0.1)-\sigma(0,0.4,-0.1)$	—
$I_{CaT}(t)$	$r_T(t)$	$\sigma(V(t),-67,-2)$	$200+87.5\,\sigma(V(t),68,2.2)$
$I_{CaL}(t)$	s(t)	$\sigma(V(t),-40,-5)$	—
$I_h(t)$	$r_f(t)$	$\sigma(V(t),-105,5)$	$100/\{\frac{-7.4[V(t)+70]}{[E(V(t),70,-0.8)-1]}+65\,E(V(t),56,23)\}$
$I_h(t)$	$r_s(t)$	$\sigma(V(t),-105,25)$	1500

on $Ca(t)$ through a Hill function, which has the generic form

$$\mathbf{Hill}(x, \kappa_x, n) = \frac{x^n}{x^n + \kappa_x^n} \tag{6.8}$$

where the Hill coefficient n is a positive integer. It is a sigmoid curve which represents the opening and saturation of a channel as the concentration of x becomes much larger than the half-activation κ_x. This type of $I_{K/Ca}$ is also referred to as an SK current, distinct from a so-called BK current, whose conductance has dependence on both voltage and intracellular calcium [56]. This gives rise to a K/Ca current of the form

$$I_{K/Ca}(t) = g_{K/Ca} \frac{Ca(t)^4}{Ca(t)^4 + \kappa_{K/Ca}^4}(E_K - V(t)). \tag{6.9}$$

The choice $n = 4$ gives it a high sensitivity to $Ca(t)$, which is commonly used in similar studies [53, 58]. Additionally, we selected $\kappa_{K/Ca} = 0.35\,\mu$M since intracellular calcium levels are normally about 0.1 μM. This means only a slight rise in internal calcium levels are required to activate $I_{K/Ca}$, which suppresses spiking behavior and "turns off" bursts. Since calcium levels are generally about 10^4 times larger in the extracellular medium than in the cytoplasm, this yields a relatively small influx of Ca^{2+} ions.

With an appropriate stimulus, a model containing only I_{Na} and I_K would produce a continuously repeating spike train. The introduction of $I_{K/Ca}$ means the neuron model intermittently activates another K current which drives the voltage response of the neuron towards $E_K = -90\,mV$ and turns off spiking behavior of the neuron. Combined with the hyperpolarization-activated I_h and I_{CaL}, the neuron model's subthreshold behavior is greatly enriched, as will be seen in the simulations to follow.

6.2.2 Modeling intracellular Ca^{2+} uptake and release

To fully model the complexity of calcium dynamics, we are looking for a model that exhibits various complex behaviors. [59] proposed a model to study complex Ca^{2+} oscillations [59], later studied in detail by [55,60]. They demonstrated that the model shows complex oscillatory phenomena such as limit cycle oscillations, bursting, quasiperiodic oscillations, and deterministic chaos [55]. In many cell types, the uptake and release of Ca^{2+} ions by the ER is mediated predominantly by inositol 1,4,5-triphosphace (IP$_3$) [52]. This uptake and release via IP$_3$ mediation has been suggested as a source of intracellular Ca^{2+} oscillations. The complex $Ca(t)$ oscillations arising in this model are due to the release of Ca^{2+} from internal stores, with dynamics based mainly on mechanisms of Ca^{2+}-induced Ca^{2+} release (CICR) that take into account the Ca^{2+}-stimulated degradation of IP$_3$ by a 3-kinase [55].

CICR was originally found to occur in muscle and cardiac cells, and was later found in a variety of other cells including neurons [61]. [62] have observed CICR in cultured rat dorsal root ganglia neurons [62]. However, they also observed CICR triggered by Ca^{2+} entry through voltage-gated Ca^{2+} channels. The sum of the two voltage-gated calcium currents I_{CaT} and I_{CaL} may therefore act as a calcium-release stimulus. We thus used the intracellular calcium dynamics model of [55] [55], but with the replacement $\beta_{input} \rightarrow \beta_{input}\left(I_L(t) + I_T(t)\right)$ for the (previously constant) external stimulus. This change now acts to couple the calcium dynamics to the membrane voltage.

In this model we assume free calcium ions are uniformly distributed across the cytosol. This avoids the additional complication of introducing a partial differential equation to model the spatial dependence of the $Ca(t)$ dynamics. This simplification is consistent with experimental results [62].

6.2.3 Calcium in Time Delay Synchronization

Unlike the gating variables, Calcium variables are only lightly affected by the voltage. This means that synchronizing the model by controlling just this one variable is unlikely to correctly synchronize the others. Even if the voltage estimates look accurate during estimation, the model will diverge from the data if the Calcium variables are not synchronized as soon as the estimation window ends.

However, by using the time delay control terms we utilize the Calcium's effect on voltage to adjust the Calcium variables directly. It becomes possible to synchronize Calcium and, as a result, the entire model. This demonstrates the real power of the time embedded coordinates. Even though we have a chaotic model with asymmetric relationships between different variables the time embedding coordinates allow us to use the influence of the unmeasured states on the voltage *in addition* to the influence of the voltage on the unmeasured states.

Figure 6.1 shows synchronization performed with the standard nudging approach and with time delay synchronization. While there is no detectable difference between the data and model voltage trace in the estimation window, the prediction is much more accurate with the time delays. Figure 6.2 demonstrates why. Although the voltage is estimated accurately, with the traditional synchronization the intracellular Calcium concentration differs between the data and the model system.

It is worth noting in Fig 6.2 that both situations use the same initial conditions. In the time delay case the control term initially pushes Calcium variable in the model way off of the trajectory before correcting. That is the reason the Calcium spikes suddenly near the beginning.

This highlights one of the pitfalls of the time embedding method. The pseudoin-

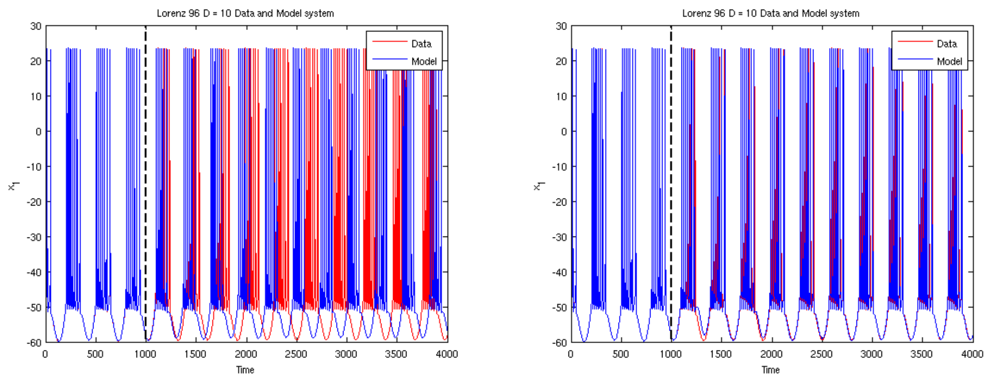


Figure 6.1: Voltage estimation and prediction, with only voltage observed. Left: No time embedding coordinates. Right: With time-embedding $\tau_k = k \cdot dt \quad \{1, \dots, 9\}$

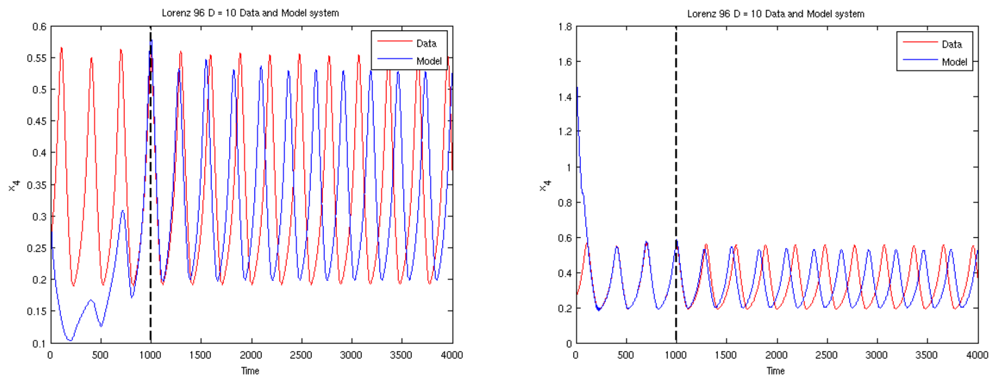


Figure 6.2: Calcium estimation and prediction, with voltage observed. Left: No time embedding coordinates. Right: With time-embedding $\tau_k = k \cdot dt \quad \{1, \dots, 9\}$

verse is not perfect, and when it goes wrong it can go significantly wrong. In this situation it self-corrects, however if the equations were more sensitive or unstable, perturbing a state variable as dramatically as in the right side of Fig 6.2 could cause the whole simulation to crash.

6.3 Noise in Voltage measurement

It is especially interesting to consider the effect of noise when dealing with a biophysical model. Biological systems are extremely complex. As a result, any model trying to represent such a system is likely in complete and will be subjected to stochastic noise from external influences. This inherent complexity also means that recordings are often limited in their accuracy. With so many chemical reactions occurring inside a 1 micron cell, it is not hard to believe that a probe would have trouble measuring voltage to arbitrary precision.

Like much of data assimilation, there is no hard and fast rule for compensating for noise with time embedding synchronization. The trend however, is clear enough. The more noise that exists in the measurements, the longer the time delay window needs to be. This can be accomplished either through using a longer spacing between the τ's or by taking more time delays. To some extent these two approaches are equivalent. Like the noiseless case, it can be effective to have more time delays for larger dimensionality systems.

Figure 6.3 shows the model estimates for two twin experiments. Both twin experiments performed a regular RK4 numerical integration of the data system. After the trajectory was determined in a twin experiment, uniform random noise of $\pm 0.1V$ was added to the voltage measurement. This is effectively a representation of an inaccurate voltage probe, which might pick up noise from poor contact or other electronics in the room. In other words, we are artifically simulating observation noise: the dynamics are unaffected but our values of $\mathbf{y}(t)$ are perturbed.

Noise in an observation makes the estimation of unobserved variables much harder. This is the same phenomena as in Fig 3.4. Notice that when τ is too short, the synchronization fails completely in the presence of noise (left half of Fig 6.3). As the length of the time delay gets longer the results improve dramatically. In fact too short of a delay ($\tau = 0.1$) caused the routine to crash, most likely by making the pseudoinverse diverge.

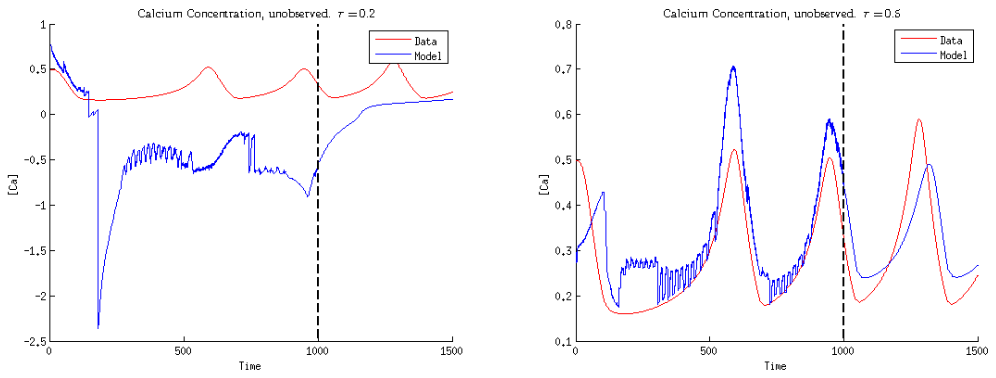


Figure 6.3: Plots of $[Ca]$ variable in twin experiment. The observed variable (Voltage) had uniform noise $\pm 0.1V$. 10 time embedding dimensions were used to synchronize. Left: Short time delays $\tau_k = 0.2k$. Right: $\tau_k = 0.5k$.

6.4 Parameter Estimation

Yet another advantage of the time embedding procedure over conventional synchronization is that it allows us to treat unknown constant parameters within the same framework as state variables. We simply use the dynamics $\frac{dp}{dt} = 0$. Doing so increases the dimensionality of the problem from $D \to D + D_p$, where D_p is equal to the number of unknown parameters. We can do this because time embedding synchronization provides a coupling term to every one of the dynamical equations. So although the parameters do not naturally have dynamical equations the coupling term can still change them over time until they agree with the data.

In many cases solving for parameters may be even more important than the state variables. Because they are constants of a physical system, understanding them can tell us about the structure of the system beyond the instantaneous dynamics. The same parameter can also appear in different types of models. In a neuron for example, the membrane capacitance is relevant for voltage dynamics as well as modeling ion channels. By comparing values obtained from different models we can test their consistency. Estimates can also be compared to direct experimental evidence to evaluate the validity of a given model.

Previous attempts at data assimilation in neurons [11,63] searched in path space for an optimal trajectory. In those trials, the researchers did something similar, by including parameters as state variables with trivial dynamics. However, what we are developing now is a little different; rather than searching over different parameter values and calculating a new trajectory for each one, we are instead introducing virtual dynamics. These virtual dynamics move a parameter *during* the model simulation, adjusting it as the model integrates forward in time.

Using time be can effective for synchronizing parameters, but there are caveats that make this an important area of further research. As we better understand the challenges with estimating parameters, we hope to merge these results with previous methods to make predictions which were previously impossible without more data.

6.4.1 Synchronizing Sodium Conductance

I start by only synchronizing the Maximum Sodium Conductance. This is a particularly convenient parameter to start with because it enters directly in the equation of the measured state variable (Voltage). In addition the $\frac{dV}{dt}$ equation is linear in g_{Na} and the Sodium current is typically the largest in the model. It stands to reason that the parameters with most influence on the trajectory and with the simplest functional form would synchronize the best. Nevertheless, this serves as an important proof of concept for a new method to determine the parameters of a physical model.

When this parameter is synchronized by itself (other parameters are all fixed at the correct values) I can obtain an accurate value even when the starting point of the model simulation is very different. In Figure 6.4 we see that when the conductance begins at 0.5 mS in the model and the actual value of the data simulation is 4.5 mS, the time embedding coupling term quickly pulls the model value closer to the actual value.

An important phenomenon that occurred during this coupling is that the unknown parameter synchronizes better with a smaller time-delay and less embedding dimensions. Normally we would expect a system with more degrees of freedom to require more embedded dimensions to synchronize. However when the additional dimension is a parameter (i.e. it has dynamics $\frac{dx_i}{dt} = 0$), additional embedding dimensions hinder the synchronization process.

Similarly, if the time-delay (τ) is too large the SE suddenly jumps up to a substantially larger value. The smaller the embedding dimension is the larger τ can be before the synchronization becomes very inaccurate.

Both of these effects are exactly opposite of what was observed with noise. When I had introduced additional embedding dimensions and longer time delays became necessary

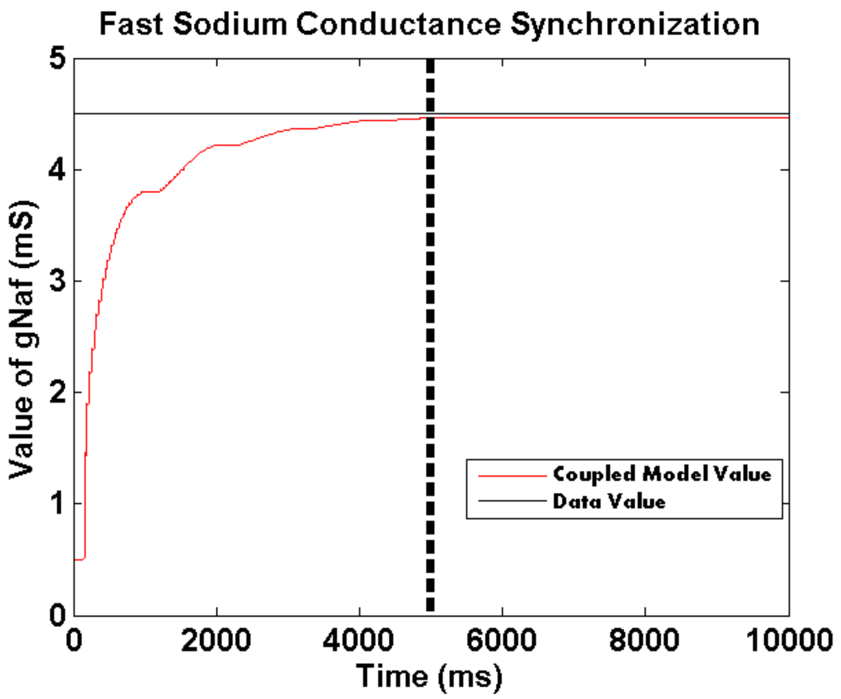


Figure 6.4: Fast Sodium Conductance of model over time. In the data simulation a $g_{Na} = 4.5$ was used. The model was started with $g_{Na}(0) = 0.5$. Sodium Conductance rises over time to a value of 4.4799 at t = 5000ms, at which point the coupling is turned off.

to obtain accurate synchronization results. We might have expected that inaccurate parameter values would have an effect comparable to measurement noise, and indeed the trajectories look qualitatively similar (compare the red model trajectories in Fig 6.5 in the coupling period before 5000ms).

This is the first hint that parameters are not as simple to determine as I had hoped. At least for some parameters, not knowing the exact value has a very different effect than simply adding noise. Because the effects are opposite, I expect that when both effects are present (noise and an unknown parameter) there is an optimum τ somewhere in the middle. Indeed, Figure 6.6 shows that when noise is introduced the SE at low values of τ goes up, so that a larger τ becomes optimal. The larger the embedding dimension used, the smaller the optimal tau is when noise is present.

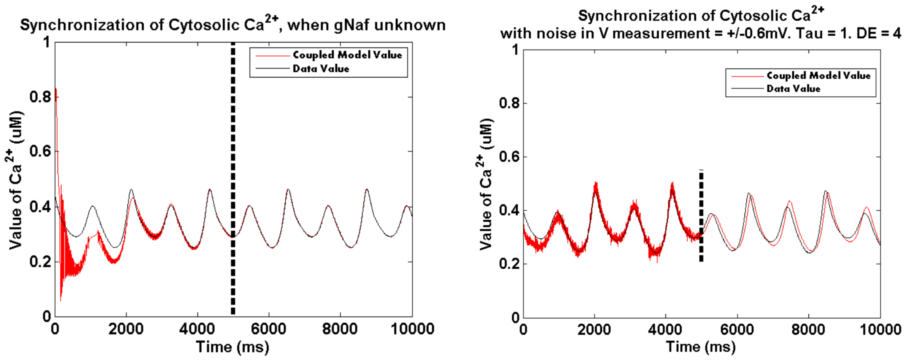


Figure 6.5: Left: Cytosolic Calcium when Sodium Conductance is unknown and synchronized, no noise. DE = 2, Tau = 0.4. Right: Cytosolic Calcium when there is measurement noise = 0.5mV. DE = 4, Tau = 1.0 ms.

Despite this issue, we have successfully demonstrated a new technique that can simultaneously determine both a parameter and all state variables in a complex chaotic model of a neuron. Further work is necessary to determine what can be done if the synchronization at the optimal time-delay is insufficient.

6.4.2 Other Parameters

Additional challenges are revealed by looking at other parameters. These parameters were chosen from the total of 60 present in the model for the variety of different equations they appeared in and how they appeared (linearly, exponentially, or some other more complicated form). The results in this section were obtained by varying each parameter separately, so only one was undetermined at a time.

The parameters I have looked at are:

- $xhV1$ - The half-activation voltage of the h gating variable sigmoid function. Appears in the $\frac{dh}{dt}$ equation only. h produces slow sodium deactivation. Actual Value = -37.4 mV

- t_n - The time constant of the n gating variable. Appears in $\frac{dn}{dt}$ only. n produces slow

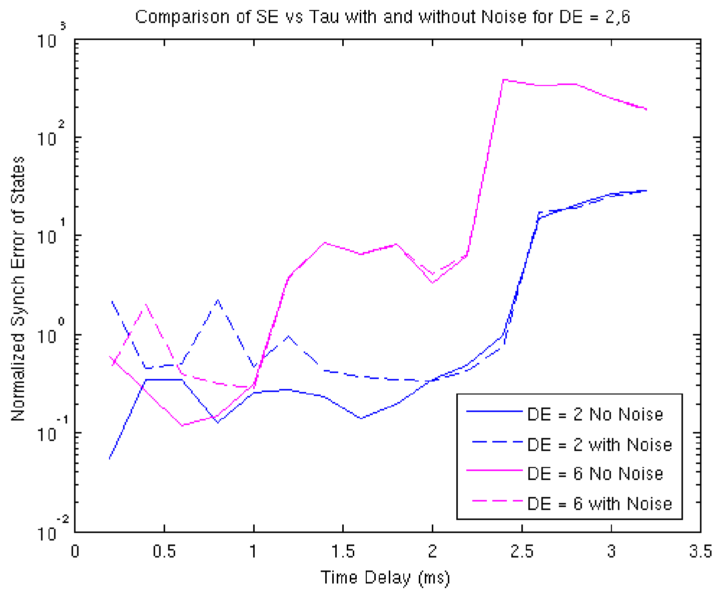


Figure 6.6: Spike Timing differences vs Time-Delay when g_{Na} undetermined AND δV = +/-0.5 mV of noise is present in Voltage measurement. The synchronization becomes very inaccurate at large Time delay values. For DE = 10 a much smaller time-delay is optimal than when DE = 2.

potassium activation. Actual Value = 10 ms.

- K_p - The Half-Concentration of the ν_{pump} Hill Function. Appears in $\frac{dCa_{cyt}}{dt}$ and $\frac{dCa_{ER}}{dt}$.

- β - The coefficient of the Calcium currents (I_{CaT} and I_{CaL}) which drives the Cytosolic Calcium concentration.

The first problem is that not all parameters synchronize as quickly as g_{Na}. This should not be too surprising because all of the parameter are further removed, in a functional sense, from the voltage dynamical equation. As a result we expect the voltage trace to carry less information about these parameters than the current conductance, for example.

Fig 6.7, 6.8, and 6.9 show how little these parameters actually vary. Over 5000 ms of synchronization a typical change is less than 5%. The sodium conductance on the other

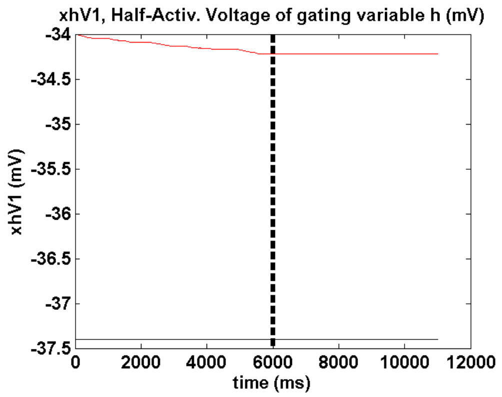


Figure 6.7: Parameter xhV1 when attempted to synchronize with DE = 2, $\tau = 0.1ms$. Initial Model value was $xhV1_0 = -34mS/cm^2$. True data value was $xhV1 = -37.4mS/cm^2$. The perturbation $\delta\mathbf{x}$ is not large enough to sufficiently adjust the parameter

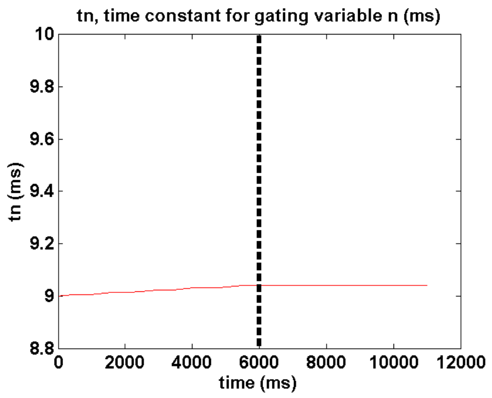


Figure 6.8: Parameter tn when attempted to synchronize with DE = 2, $\tau = 0.1ms$. Initial Model value was $tn_0 = 9ms$. True data value was $tn_0 = 10ms$. The perturbation $\delta\mathbf{x}$ is not large enough to sufficiently adjust the parameter

hand, could move from 0.5 mS to 4.5 mS with no problem.

To correct this, a somewhat simple, albeit inelegant, solution often proves effective. What I have found is that many times the sign of the coupling term is correct, even though the magnitude is too small to significantly change the value in the time allowed. This can be compensated by adjusting the control coefficient g to be a much larger number. For most states we use $g = 1$, but for parameters we found that a coefficient as large as $10,000 - 100,000$ allows the time-delay procedure to adjust insensitive parameters

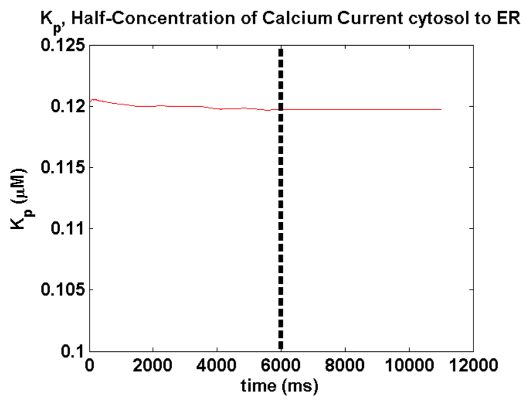


Figure 6.9: Parameter K_p when attempted to synchronize with DE = 2, $\tau = 0.1ms$. Initial Model value was $K_{p,0} = 0.12\mu M$. True data value was $K_{p,0} = 0.1\mu M$. The perturbation $\delta\mathbf{x}$ is not large enough to sufficiently adjust the parameter

effectively.

Although most parameters vary very slowly, the opposite can happen as well. This seems to be at least partially based on the magnitude of the parameter. When I did the procedure with the external Calcium current β, it fluctuated wildly. In this case, I had to use a $g = 0.01$ to obtain synchronization.

We are currently developing the theoretical understanding of the control coefficients g_p to understand what exactly they influence the model. Our hope is that as we understand the dynamical significance of this coefficient on the rate of convergence and stability of the dynamics we will be able to develop better, more systematic, ways of choosing it appropriately.

6.4.3 Multiple Parameters

We can use this procedure to estimate multiple parameters simultaneously. This is certainly a more difficult problem. When multiple parameters are included they will often compensate for each other by moving in the wrong direction. This means that several of the parameters will actually be moved *away* from their correct values.

If the parameters are independently straightforward to estimate, then there is a good chance they can be solved for simultaneously as well. The impact can be seen in Figure 6.10 when multiple parameters are estimated simultaneously. As usual, the voltage is estimated very well in the observation window, however prediction is much less accurate than previous results. This is because the parameters in Fig 6.11 and 6.12 are not quite synched to the value they're supposed to have. Comparing Fig 6.11 to the situation with *only* g_{Na} estimated, in Fig 6.4, we see that it is noticeably less accurate. This inaccuracy results in inaccurate estimates of other state variables as well.

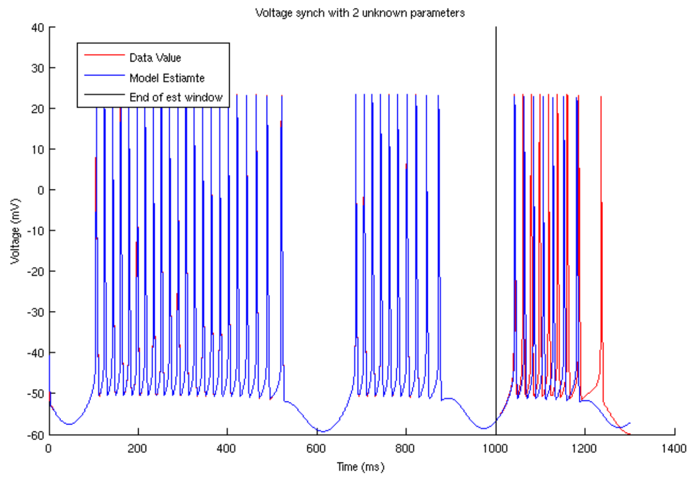


Figure 6.10: Voltage with 2 params estimated simultaneously

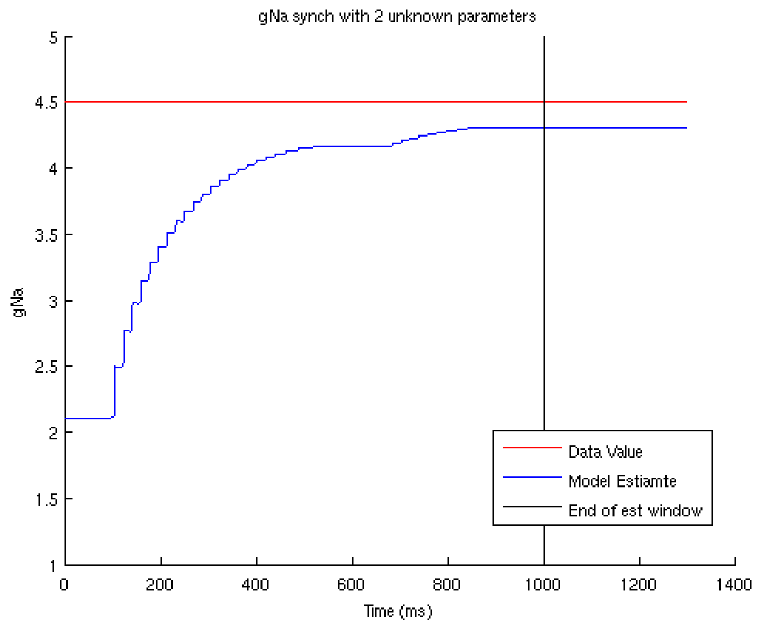


Figure 6.11: Sodium conductance gNa with another param (gK) also estimated

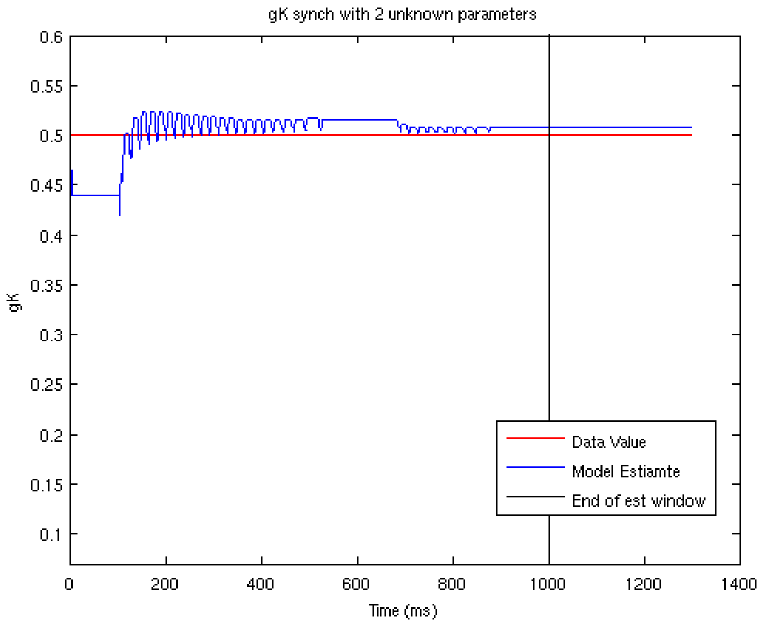


Figure 6.12: Potassium conductance gK with another param (gNa) also estimated

Chapter 6 is reproduced in part from material as it appears in “Estimating the biophysical properties of neurons with intracellular calcium dynamics”. Jingxin Ye, Paul J Rozdeba, Uriel I Morone, Arij Daou, and Henry DI Abarbanel. The author of this thesis was a co-author of this paper.

Appendix A

Techniques for choosing the threshold of M-P Psuedoinverse

In theory, the pseudoinverse inverts any singular value that is not equal to zero. However, for numerical calculations the numbers are never exactly zero. Most pre-programmed algorithms use machine precision (typically $\sim 10^{-14}$) as the cutoff. Any number larger than this will be inverted. This can be a problem in the time delay synchronization procedure, because the smaller the singular value the larger of a perturbation it will result in. Usually terms of order 10^{13} will be a such a large magnitude that they push the trajectory off the attractor into a very different region of phase space. The psuedoinverse is not exact, and a large perturbation only needs to happen once to knock the system into an unstable regime, at which point the model will quickly diverge. In many systems, such as neuron models shown later, many of the states are variables which are highly constrained on physical grounds. Gating variables, for example, express a proportion so can only ever take on values between 0 and 1. If they artificially exceed their requisite range by a sufficient amount the model becomes nonsensical.

Different heuristics or procedures can be used to determine an optimal threshold. The more singular values one includes (treats as non-zero), the larger the coupling term in the equations will eventually be. The fundamental balancing is finding a cutoff that produces a large enough coupling term to synchronize the model in a reasonable length of time, but not so big that it moves the state variables in the model outside of stable regimes.

A.1 Absolute Threshold

The simplest way to set the threshold between zero and non-zero is to choose a constant small number. Most SVD routines default to machine precision, $\sim 10^{-14}$. This will often be far too small. After inverting, we could end up with an entry in $\mathbf{S}^{-1}$ which is $\sim 10^{12}$. Assuming our variables are order 1, this perturbation will usually be vastly too big. On the other hand, selecting a much larger threshold (say, 10^{-5}) may be sufficient to keep the perturbations small enough to be in a stable range everywhere along the attractor. However, if we increase the threshold for "zero", then in more stable regions the resulting $\delta\mathbf{x}$ might be too small to make any meaningful difference in the dynamics.

Some systems, like Lorenz 96, do not change qualitatively as they evolve very much. The time scales of the fluctuations stay around the same, the magnitude of the state variables stays within a constant range, and the local Lyapunov exponents do not change dramatically throughout the trajectory. In situations like these setting a fixed a threshold for inverting singular values can work alright. As long as the threshold is only slightly larger than we need to guarantee the stability of the perturbations, the perturbations are likely to still be relevant throughout the trajectory.

Other systems may resemble neuron models, the distinguishing feature of which is very rapid large magnitude spikes, followed by long periods of slowly varying behavior. In this situation choosing a threshold which keeps the system stable during spiking could easily be so large that the system cannot synchronize during any subthreshold behavior. The problem, is that a system can have very different properties as it explores different regions of the attractor.

It is worth noting, that we have observed that the most synchronization happens during periods of of large Lyapunov exponents, such as during spiking. This makes sense if we think that the variational equation essentially quantifies how the trajectory of the model system diverges from the data. During stable periods, such as subthreshold neuron behavior, the model and data systems can display very similar trajectories regardless of the particular values of the unobserved state variables. With this in mind, it is possible in some systems using an absolute threshold might be acceptable, if the coupling during only the quickly varying periods is sufficient to synchronize the system overall.

The most obvious advantage of this approach is its simplicity. The threshold cutoff is implemented into every SVD routine can be quickly changed as a parameter in the algorithm.

A.2 Fixed Rank Inverse

A useful alternative to setting constant numerical threshold is to consider the effect it has on the matrix being inverted. Choosing a threshold for the inversion effectively sets the rank of the matrix. Instead of thinking what is the value above which we will include singular values, we can ask *how many* singular values we want to include. A particularly useful number of singular to include is the humble 1.

If we have R nonzero singular values, the inverse matrix is given by (recall the $\mathbf{S}$ is diagonal):

$$\frac{dx_i}{ds_k} = \sum_j V_{ij} S_{jj}^{-1} U_{jk}^* \qquad j = 1, \ldots, R \tag{A.1}$$

So each additional singular value that is included just adds additional terms the pseudoinverse matrix. This does not necessarily make the entries of $\frac{dx}{ds}$ larger, because some of the terms can be negative. However, assuming we sort the singular values from greatest to smallest and invert only the largest R of them, each additional value included will be larger in magnitude and more likely to result in instability.

Using a rank of 1 for the inverse guarantees we have the most conservative possible perturbation at any point along the trajectory. This can be very helpful for chaotic systems which seem to inevitably hit a difficult part of the dynamics and diverge if they're perturbed a little too much. It is also much more convenient effectively set the coupling term to its minimum nonzero value, rather than simulating the entire model repeatedly and tweaking the absolute threshold to see it is large enough to finish the simulation.

Similar to the absolute threshold, this method is also very easy to implement. One simply specifies how many columns and rows to use for the matrix multiplication in Eq A.1. Each additional rank results in a significant addition to the coupling terms over the course of the integration. This means that if 1 singular value is found to be totally ineffectual, just going up to 2 or 3 can produce a totally different data assimilation result.

A.3 Continuity Constraint

The most effective method we found is a continuity constraint. The intention is to limit the size of the coupling term based on unmodified rates of change in the model.

We choose the singular values in such a way that all the coupling terms in the dynamic equation do not exceed some percentage of the original derivative $f_i(\mathbf{x})$.

So long as the condition

$$\min_{1 \le i \le D} \frac{F_i(\vec{x}(t))}{g\, \delta x_i(t)} \le \epsilon. \tag{A.2}$$

holds, we increase the rank and try again.

An appealing aspect of this method is that the effective threshold is in some way normalized to account for the different state variables. Assuming the inherent dynamics themselves are always stable, keeping the additional coupling on the same scale should make it very unlikely to drive the model outside of its basin of attraction.

The continuity condition works as follows: Starting with $r = 1$, make all singular values of $\frac{\partial \mathbf{s}}{\partial \mathbf{x}}$ equal to 0, except the largest. Calculate the time derivatives from both the original dynamical equations and the equations with the added coupling. If the ratio $\dot{x}_{original}/\dot{x}_{coupling}$ is smaller than some predetermined ratio ρ increase r by one, and repeat the procedure with more singular values. Use the largest rank so long as the ratios of the derivatives are all less than the tolerance ρ. We typically used $\rho = 0.1$, so the largest control term can be no more than 10 times the unmodified derivative.

Bibliography

[1] P. J. Brockwell and R. A. Davis, *Introduction to time series and forecasting*. Springer Science & Business Media, 2006.

[2] P. L. Houtekamer, H. L. Mitchell, G. Pellerin, M. Buehner, M. Charron, L. Spacek, and B. Hansen, "Atmospheric data assimilation with an ensemble kalman filter: Results with real observations," *Monthly weather review*, vol. 133, no. 3, pp. 604–620, 2005.

[3] G. Evensen, "Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics," *Journal of Geophysical Research: Oceans*, vol. 99, no. C5, pp. 10143–10162, 1994.

[4] L. M. Pecora and T. L. Carroll, "Synchronization in chaotic systems," *Physical review letters*, vol. 64, no. 8, p. 821, 1990.

[5] L. M. Pecora, T. L. Carroll, G. A. Johnson, D. J. Mar, and J. F. Heagy, "Fundamentals of synchronization in chaotic systems, concepts, and applications," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 7, no. 4, pp. 520–543, 1997.

[6] J. E. Hoke and R. A. Anthes, "The initialization of numerical models by a dynamic-initialization technique," *Monthly Weather Review*, vol. 104, no. 12, pp. 1551–1556, 1976.

[7] S. Lakshmivarahan and J. M. Lewis, "Nudging methods: a critical overview," in *Data Assimilation for Atmospheric, Oceanic and Hydrologic Applications (Vol. II)*, pp. 27–57, Springer, 2013.

[8] E. N. Lorenz, "Predictability: A problem partly solved," in *Proc. Seminar on predictability*, 1996.

[9] L. Kocarev and U. Parlitz, "General approach for chaotic synchronization with applications to communication," *Phys. Rev. Lett.*, vol. 74, pp. 5028–5031, Jun 1995.

[10] A. Rauh, L. Hannibal, and N. Abraham, "Global stability properties of the complex lorenz model," *Physica D: Nonlinear Phenomena*, vol. 99, no. 1, pp. 45–58, 1996.

[11] M. Kostuk, *Thesis: Synchronization and Statistical Methods for the Data Assimilation of HVc Neuron Models*. PhD thesis, University of California San Diego, http://escholarship.org/uc/item/2fh4d086, 2012.

[12] H. D. Abarbanel, D. R. Creveling, R. Farsian, and M. Kostuk, "Dynamical state and parameter estimation," *SIAM Journal on Applied Dynamical Systems*, vol. 8, no. 4, pp. 1341–1381, 2009.

[13] N. F. Rulkov, M. M. Sushchik, L. S. Tsimring, and H. D. Abarbanel, "Generalized synchronization of chaos in directionally coupled chaotic systems," *Physical Review E*, vol. 51, no. 2, p. 980, 1995.

[14] M. Eldridge, *Thesis: Use of Data Assimilation to Determine Features of Neuron Structure and Connectivity*. PhD thesis, University of California San Diego, http://eprints.cdlib.org/uc/item/7st8j1r8, 2016.

[15] J. C. Quinn and H. D. Abarbanel, "State and parameter estimation using monte carlo evaluation of path integrals," *Quarterly Journal of the Royal Meteorological Society*, vol. 136, no. 652, pp. 1855–1867, 2010.

[16] D. R. Creveling, P. E. Gill, and H. D. Abarbanel, "State and parameter estimation in nonlinear systems as an optimal tracking problem," *Physics Letters A*, vol. 372, no. 15, pp. 2640–2644, 2008.

[17] H. D. Abarbanel, M. Kostuk, and W. Whartenby, "Data assimilation with regularized nonlinear instabilities," *Quarterly Journal of the Royal Meteorological Society*, vol. 136, no. 648, pp. 769–783, 2010.

[18] A. C. Harvey, *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[19] G. Evensen, "The ensemble kalman filter: Theoretical formulation and practical implementation," *Ocean dynamics*, vol. 53, no. 4, pp. 343–367, 2003.

[20] P.-S. Laplace and P. Simon, "Théorie analytique des probabiletés. introduction," *Oeuvres de Laplace*, vol. 7, no. 6, 1847.

[21] C. M. Bender and S. A. Orszag, *Advanced mathematical methods for scientists and engineers I*. Springer Science & Business Media, 1999.

[22] R. Eglese, "Simulated annealing: a tool for operational research," *European journal of operational research*, vol. 46, no. 3, pp. 271–281, 1990.

[23] C.-R. Hwang, "Simulated annealing: theory and applications," *Acta Applicandae Mathematicae*, vol. 12, no. 1, pp. 108–111, 1988.

[24] S. Kirkpatrick, "Optimization by simulated annealing: Quantitative studies," *Journal of statistical physics*, vol. 34, no. 5-6, pp. 975–986, 1984.

[25] D. Johnston and S. M.-S. Wu, *Foundations of cellular neurophysiology*. MIT press, 1994.

[26] B. Hille *et al.*, *Ion channels of excitable membranes*, vol. 507. Sinauer Sunderland, MA, 2001.

[27] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, p. 500, 1952.

[28] D. Johnston, S. M.-S. Wu, and R. Gray, *Foundations of cellular neurophysiology*. MIT press Cambridge, 1995.

[29] C. D. Meliza, M. Kostuk, H. Huang, A. Nogaret, H. D. I. Abarbanel, and D. Margoliash, "Dynamical state and parameter estimation validated by prediction of experimental membrane voltages for conductance-based models of individual neurons," *Journal of Neurophysiology*, vol. Submitted, 2013.

[30] H. D. Abarbanel, *Predicting the Future: Completing Models of Observed Complex Systems, Chapter 2*. Springer, 2013.

[31] J. M. Wild, M. N. Williams, G. J. Howie, and R. Mooney, "Calcium-binding proteins define interneurons in hvc of the zebra finch (taeniopygia guttata)," *Journal of Comparative Neurology*, vol. 483, no. 1, pp. 76–90, 2005.

[32] M. H. Graber, F. Helmchen, and R. H. Hahnloser, "Activity in a premotor cortical nucleus of zebra finches is locally organized and exhibits auditory selectivity in neurons but not in glia," *PloS one*, vol. 8, no. 12, p. e81177, 2013.

[33] M. Kato and K. Okanoya, "Molecular characterization of the song control nucleus hvc in bengalese finch brain," *Brain research*, vol. 1360, pp. 56–76, 2010.

[34] A. Daou, M. Ross, F. Johnson, R. L. Hyson, and R. Bertram, "Electrophysiological characterization and computational models of hvc neurons in the zebra finch," *Journal of Neurophysiology*, 2013.

[35] M. A. Long, D. Z. Jin, and M. S. Fee, "Support for a synaptic chain model of neuronal sequence generation," *Nature*, vol. 468, no. 7322, pp. 394–399, 2010.

[36] A. L. Hodgkin and A. F. Huxley, "Propagation of electrical signals along giant nerve fibres," *Proceedings of the Royal Society of London. Series B, Biological Sciences*, pp. 177–183, 1952.

[37] G. Ullah and S. J. Schiff, "Tracking and control of neuronal hodgkin-huxley dynamics," *Physical Review E*, vol. 79, no. 4, p. 040901, 2009.

[38] B. A. Toth, M. Kostuk, C. D. Meliza, D. Margoliash, and H. D. Abarbanel, "Dynamical estimation of neuron and network properties i: variational methods," *Biological cybernetics*, vol. 105, no. 3-4, pp. 217–237, 2011.

[39] M. Kostuk, B. A. Toth, C. D. Meliza, D. Margoliash, and H. D. Abarbanel, "Dynamical estimation of neuron and network properties ii: path integral monte carlo methods," *Biological cybernetics*, vol. 106, no. 3, pp. 155–167, 2012.

[40] C. Knowlton, C. D. Meliza, D. Margoliash, and H. D. Abarbanel, "Dynamical estimation of neuron and network properties iii: network analysis using neuron spike times," *Biological cybernetics*, vol. 108, no. 3, pp. 261–273, 2014.

[41] M. J. Berridge, M. D. Bootman, and H. L. Roderick, "Calcium signalling: dynamics, homeostasis and remodelling," *Nature Reviews Molecular Cell Biology*, vol. 4, no. 7, pp. 517–529, 2003.

[42] M. Kubota and N. Saito, "Sodium-and calcium-dependent conductances of neurones in the zebra finch hyperstriatum ventrale pars caudale in vitro.," *The Journal of physiology*, vol. 440, no. 1, pp. 131–142, 1991.

[43] B. Schwaller, "Cytosolic ca2+ buffers," *Cold Spring Harbor perspectives in biology*, vol. 2, no. 11, 2010.

[44] M. J. Berridge, P. Lipp, and M. D. Bootman, "The versatility and universality of calcium signalling," *Nature reviews Molecular cell biology*, vol. 1, no. 1, pp. 11–21, 2000.

[45] C. Grienberger and A. Konnerth, "Imaging calcium in neurons," *Neuron*, vol. 73, no. 5, pp. 862–885, 2012.

[46] S. Fucile, "Ca^{2+} permeability of nicotinic acetylcholine receptors," *Cell calcium*, vol. 35, no. 1, pp. 1–8, 2004.

[47] M. J. Higley and B. L. Sabatini, "Calcium signaling in dendrites and spines: practical and functional considerations," *Neuron*, vol. 59, no. 6, pp. 902–913, 2008.

[48] I. S. Ramsey, M. Delling, and D. E. Clapham, "An introduction to trp channels," *Annu. Rev. Physiol.*, vol. 68, pp. 619–647, 2006.

[49] M. J. Berridge, "Neuronal calcium signaling," *Neuron*, vol. 21, no. 1, pp. 13–26, 1998.

[50] M. R. Duchen, "Contributions of mitochondria to animal physiology: from homeostatic sensor to calcium signalling and cell death," *The Journal of physiology*, vol. 516, no. 1, pp. 1–17, 1999.

[51] D. D. Friel, "Calcium oscillations in neurons," *Calcium waves, gradients and oscillations*, vol. 188, p. 210, 1995.

[52] K. Thurley, A. Skupin, R. Thul, and M. Falcke, "Fundamental properties of Ca 2+ signals," *Biochimica et Biophysica Acta (BBA)-General Subjects*, vol. 1820, no. 8, pp. 1185–1194, 2012.

[53] M. Falcke, R. Huerta, M. I. Rabinovich, H. D. Abarbanel, R. C. Elson, and A. I. Selverston, "Modeling observed chaotic oscillations in bursting neurons: the role of calcium dynamics and IP3," *Biological Cybernetics*, vol. 82, no. 6, pp. 517–527, 2000.

[54] T. Nowotny, R. Levi, and A. I. Selverston, "Probing the dynamics of identified neurons with a data-driven modeling approach," *PloS one*, vol. 3, no. 7, p. e2627, 2008.

[55] G. Houart, G. Dupont, and A. Goldbeter, "Bursting, chaos and birhythmicity originating from self-modulation of the inositol 1, 4, 5-trisphosphate signal in a model for intracellular Ca^{2+} oscillations," *Bulletin of mathematical biology*, vol. 61, no. 3, pp. 507–530, 1999.

[56] P. Sah, "Ca^{2+}-activated K^{+} currents in neurones: types, physiological roles and modulation," *Trends in neurosciences*, vol. 19, no. 4, pp. 150–154, 1996.

[57] T. R. Chay and J. Rinzel, "Bursting, beating, and chaos in an excitable membrane model," *Biophysical Journal*, vol. 47, no. 3, pp. 357–366, 1985.

[58] Y.-X. Li, S. Stojilković, J. Keizer, and J. Rinzel, "Sensing and refilling calcium stores in an excitable cell," *Biophysical journal*, vol. 72, no. 3, pp. 1080–1091, 1997.

[59] G. Dupont and A. Goldbeter, "One-pool model for ca^{2+} oscillations involving ca^{2+} and inositol 1, 4, 5-trisphosphate as co-agonists for ca^{2+} release," *Cell calcium*, vol. 14, no. 4, pp. 311–322, 1993.

[60] J. M. Borghans, G. Dupont, and A. Goldbeter, "Complex intracellular calcium oscillations a theoretical exploration of possible mechanisms," *Biophysical chemistry*, vol. 66, no. 1, pp. 25–41, 1997.

[61] T. R. Chay, "Modelling for nonlinear dynamical processes in biology," *Patterns, Information and Chaos in Neuronal Systems*, pp. 73–122, 1993.

[62] N. Solovyova, N. Veselovsky, E. Toescu, and A. Verkhratsky, "Ca^{2+} dynamics in the lumen of the endoplasmic reticulum in sensory neurons: direct visualization of Ca^{2+}-induced Ca^{2+} release triggered by physiological Ca^{2+} entry," *The EMBO journal*, vol. 21, no. 4, pp. 622–630, 2002.

[63] H. D. Abarbanel, *Predicting the Future: Completing Models of Observed Complex Systems, Chaper 6.* Springer, 2013.