UNIVERSITY OF CALIFORNIA SAN DIEGO

A Study in Synchrony: Forecasting Complex Systems

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

in

Physics

by

Jason Alexander Platt

Committee in charge:

Professor Henry Abarbanel, Chair
Professor Gert Cauwenberghs
Professor Michael Fogler
Professor Philip Gill
Professor Ken Intriligator

2022

The Dissertation of Jason Alexander Platt is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2022

DEDICATION

In loving memory of my grandparents Phil and Lori Platzman who continue to inspire me to follow my dreams

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

There are many people who contributed to making this thesis a reality including friends, family, collaborators and role models. Without each and every one of you I would not be where I am today and for that I am thankful.

In particular, I would like to thank Henry Abarbanel for his support as the chair of my committee and as my advisor. Throughout the last five years you have been endlessly patient, kind and thoughtful and I could not have asked for a better mentor.

Steve Penny and the team at NOAA CIRES were immensely helpful both scientifically and as my main remote point of contact during lockdown. You kept me grounded and focused on research.

I would also like to acknowledge Dan Margoliash and the Margoliash Laboratory at the University of Chicago for providing me with both data and guidance in applying numerical methods to biological systems.

Jun Wang and Gert Cauwenberghs worked with me on my first project as a graduate student, providing me with data, experimental design and support right at the beginning of my journey and for that I am grateful.

Chapter 2, in part, is adapted from the material as it appears in Anna Miller, Dawei Li, Jason Platt, Arij Daou, Daniel Margoliash, and Henry D. I. Abarbanel. "Statistical Data Assimilation: Formulation and Examples From Neurobiology". In: *Frontiers in Applied Mathematics and Statistics* 4 (2018). ISSN: 2297-4687. DOI: 10.3389/fams.2018.00053. Anna Miller was the primary inverstigator and author of the work, the dissertation author is a coauthor on the paper.

Chapter 3, in full, is a reprint of the material as it appears in Jason A. Platt, Adrian S. Wong, Randall Clark, Stephen G. Penny, and H. D. I. Abarbanel. "Robust forecasting using predictive generalized synchronization in reservoir computing". In: *Chaos* 31 (2021), p. 123118. URL: https://doi.org/10.1063/5.0066013. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in full, has been submitted for publication of the material as it may appear in Jason A. Platt, Stephen G. Penny, Timothy A. Smith, Tse-Chun Chen, and Henry D. I. Abarbanel. "A Systematic Exploration of Reservoir Computing for Forecasting Complex Spatiotemporal Dynamics". In: (Submitted to Neural Networks 2022). arXiv: 2201.08910. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in part is adapted from S. G. Penny, T. A. Smith, T.-C. Chen, J. A. Platt, H.-Y. Lin, M. Goodliff, and H. D. I. Abarbanel. "Integrating Recurrent Neural Networks With Data Assimilation for Scalable Data-Driven State Estimation". In: *Journal of Advances in Modeling Earth Systems* 14.3 (2022), e2021MS002843. DOI: https://doi.org/10.1029/2021MS002843. Stephen Penny was the primary investigator and author of this material while the disertation author was a coauthor.

VITA

2016	Bachelor of Science, Stanford University
2017	Masters of Science, Stanford University
2022	Doctor of Philosophy, University of California San Diego

PUBLICATIONS

Jason A. Platt, Stephen G. Penny, Timothy A. Smith, Tse-Chun Chen, and Henry D. I. Abarbanel. "A Systematic Exploration of Reservoir Computing for Forecasting Complex Spatiotemporal Dynamics". In: (Submitted to Neural Networks 2022). arXiv: 2201.08910

Tse-Chun Chen, Stephen G. Penny, Timothy A. Smith, and Jason A. Platt. "Learning dynamical systems and numerical integration methods". In: (Submitted to PNAS 2022). arXiv: 2201.05193

Randall Clark, Lawson Fuller, Jason Platt, and Henry D. I. Abarbanel. "Reduced Dimension, Biophysical Neuron Models Constructed From Observed Data". In: *Neural Computation* (2022 in press). DOI: 10.1101/2021.12.03.471194

Stephen G. Penny, Timothy A. Smith, Tse-Chun Chen, Jason A. Platt, Hsin-Yi Lin, Michael Goodliff, and Henry D. I. Abarbanel. "Integrating Recurrent Neural Networks with Data Assimilation for Scalable Data-Driven State Estimation". In: *Journal of Advances in Modelling Earth Systems* (2022). DOI: 10.1029/2021MS002843

A.D. Lusk, J.P. Platt, and J.A. Platt. "Natural and experimental constraints on a flow law for dislocation-dominated creep in wet quartz". In: *Journal of geophysical research* (2021). DOI: https://doi.org/10.1029/2020JB021302

Jason A. Platt, Adrian S. Wong, Randall Clark, Stephen G. Penny, and H. D. I. Abarbanel. "Robust forecasting using predictive generalized synchronization in reservoir computing". In: *Chaos* 31 (2021), p. 123118. URL: https://doi.org/10.1063/5.0066013

Jason Platt, Nicholas Moehle, John D. Fox, and William Dally. "Optimal Operation of a Plug-In Hybrid Vehicle". In: *IEEE Transactions on Vehicular Technology* 67.11 (2018), pp. 10366–10377. DOI: 10.1109/TVT.2018.2866801

Anna Miller, Dawei Li, Jason Platt, Arij Daou, Daniel Margoliash, and Henry D. I. Abarbanel. "Statistical Data Assimilation: Formulation and Examples From Neurobiology". In: *Frontiers in Applied Mathematics and Statistics* 4 (2018). ISSN: 2297-4687. DOI:

10.3389/fams.2018.00053

Jason Platt, Wolfgang Hofle, Kristin Pollock, and John Fox. "Equalizer design techniques for dispersive cables with application to the SPS wideband kicker". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 868 (2017), pp. 93–97. ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2017.06.029

ABSTRACT OF THE DISSERTATION

A Study in Synchrony: Forecasting Complex Systems

by

Jason Alexander Platt

Doctor of Philosophy in Physics

University of California San Diego, 2022

Professor Henry Abarbanel, Chair

Complex systems are found at the heart of the most pressing intellectual challenges of the 21st century. Weather and climate, ecological and neurobiological networks, many body quantum systems, and even the operation of human societies can be described as the interaction of simple components from which collectively emerge distinct phenomena. In this thesis, key concepts from dynamical systems theory and machine learning are employed to analyze complex systems in both neurobiological contexts and weather prediction. The first of these concepts is synchronization, from which data assimilation is derived in order to describe the optimal interpolation between an imperfect physical model and sparse/noisy data. Data assimilation is applied to biophysical problems such as the design of VLSI

circuits for neuromorphic computing, as well as the estimation of a biological neuron's physical characteristics from voltage and calcium fluorescence measurements. Generalized synchronization is then deployed as the framework to examine the operation of recurrent neural networks, particularly the reservoir computing (RC) architecture. RC operates through the synchronization of a high dimensional artificial network together with chaotic input data, and is able to produce state of the art forecasting performance on a number of benchmark systems in numerical weather prediction. Invariant quantities such as the Lyapunov exponent spectrum and the fractal dimension of the RC are calculated and shown to be directly related to the input dynamics of the data. The concluding remarks unify both RC and data assimilation together into a machine learning forecasting cycle that produces forecasts from sparse and noisy data with no physical model.

Chapter 1

1.1 Introduction

According to Isaac Asimov, the birth of modern physics occurred on the 28 May 585 BC when Thales of Miletus successfully predicted a solar eclipse—interrupting a battle between the Lydians and the Medes [11]. The two opposing sides, paralyzed by the perilous portent, rapidly agreed to a rapprochement. The interacting Sun, planets and moons whose motion resulted in the eclipse of 585 BC are one of the more familiar examples of a broad category of "complex systems" [12] which include the human brain, global weather and climate, human and animal ecosystems, chemical reactions and many other examples [13]. Simply put, a complex system is one composed of many, often simple, components that interact together in a way that often makes them extraordinarily difficult to predict [13].

Phenomena associated with complex systems include emergence, phase transitions, spontaneous symmetry breaking and deterministic chaos. Mathematically, most complex systems can be described by the formalism of dynamical systems theory. A dynamical system is one that obeys a set of differential equations

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = f_i(\mathbf{x}(t), t) \ ; \ i \in [1, \ldots, D] \tag{1.1}$$

with $\mathbf{x}$ giving the state of the system $\mathbf{x} = [x_1, \ldots, x_D]$, $\mathbf{f} = [f_1, \ldots, f_D]$ the equations of motion, D the dimension, and t the time. Equation (1.1) describes how the state of the system $\mathbf{x}$ changes with time. A nonlinear dynamical system is one in which the vector field $\mathbf{f}(\mathbf{x})$ is a nonlinear function of the components of $\mathbf{x}$; formally $\mathbf{f}(c_1\mathbf{x}_1 + c_2\mathbf{x}_2) \neq c_1\mathbf{f}(\mathbf{x}_1) + c_2\mathbf{f}(\mathbf{x}_2)$ if $\mathbf{f}$ is nonlinear.

The almost paradoxical concept of deterministic chaos can be characteristic of even surprisingly simple nonlinear dynamical systems; for example, the motion of three massive bodies interacting under gravity is often chaotic. Chaotic systems are distinguished by extreme sensitivity to initial conditions, to the extent that even floating point inaccuracy in numerical solvers or the software used in the integration impose a finite forecasting horizon [14]. Even under the best of circumstances—abundant data and a correct physical model—chaotic systems would be difficult to forecast. Outside of laboratory settings the circumstances are seldom ideal with sparse/noisy data combining with imperfect knowledge of the physics to make prediction extremely challenging. The applications, however, are manifold and compelling—including in numerical weather prediction [15], chemical mixing [16, 17], optics [18], robotics [19] and many other fields—and it is thus incumbent upon us to find forecasting methods.

To understand the scale of the challenge involved in forecasting these systems we can examine the case of weather prediction. Operational weather prediction systems involve the integration of models with upwards of 10^{10} degrees of freedom that must be constrained with only $\sim 10^7$ or so measurements [20]. The transfer of information from the data to the model, particularly to the unobserved states in the model, is of the utmost importance and is conducted through a method of Bayesian inference called data assimilation (DA) [15]. The models have errors and the data is sparse yet it is still possible to achieve incredible accuracy in weather forecasting.

This thesis describes developments in the application of dynamical systems theory to the prediction of (possibly chaotic) dynamics. We will start in chapter 1 with an

introduction to some of the more useful dynamical systems and data assimilation concepts, and then proceed directly in chapter 2 with the application of data assimilation and Bayesian inference for state and parameter estimation problems in neurobiology. Following that discussion, in chapters 3/4 a form of machine learning denoted reservoir computing (RC) will be investigated. Machine learning allows the forecasting of complex systems from data alone—without the development of a physical model—and thus is extremely useful when the physics is unknown or integrating the equations of motion is computationally intensive. RC is shown to operate through the theory of generalized synchronization, from which many interesting and useful properties follow. Concluding the thesis will be a discussion of the unification of data assimilation and machine learning.

1.1.1 A Simple Example

To make equation (1.1) more concrete and to introduce some fundamental quantities we will take as an illustrative example the undamped pendulum—see Fig.(1.1). The derivation of the equations of motion come from solving Newton's equation in the rotational frame of the pendulum $\tau = \mathcal{I}\alpha$; τ is the torque, $\mathcal{I}$ the moment of inertia, and α the angular acceleration. Making m the mass of the bob and ℓ the length of the rod, $\tau = -mg\ell \sin\theta$, $\mathcal{I} = m\ell^2$ and $\alpha = \ddot{\theta}$. $\cdot$ denotes a single total derivative with respect to time $\frac{\mathrm{d}}{\mathrm{d}t}$ and double dot $\frac{\mathrm{d}^2}{\mathrm{d}t^2}$. The equation of motion is therefore

$$\ddot{\theta} = -\frac{g}{\ell}\sin\theta. \tag{1.2}$$

It's possible to put this into the standard form of (1.1) by setting $x_1 = \theta$ and $x_2 = \dot{\theta}$. This gives us

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{g}{\ell}\sin x_1 \end{aligned} \tag{1.3}$$

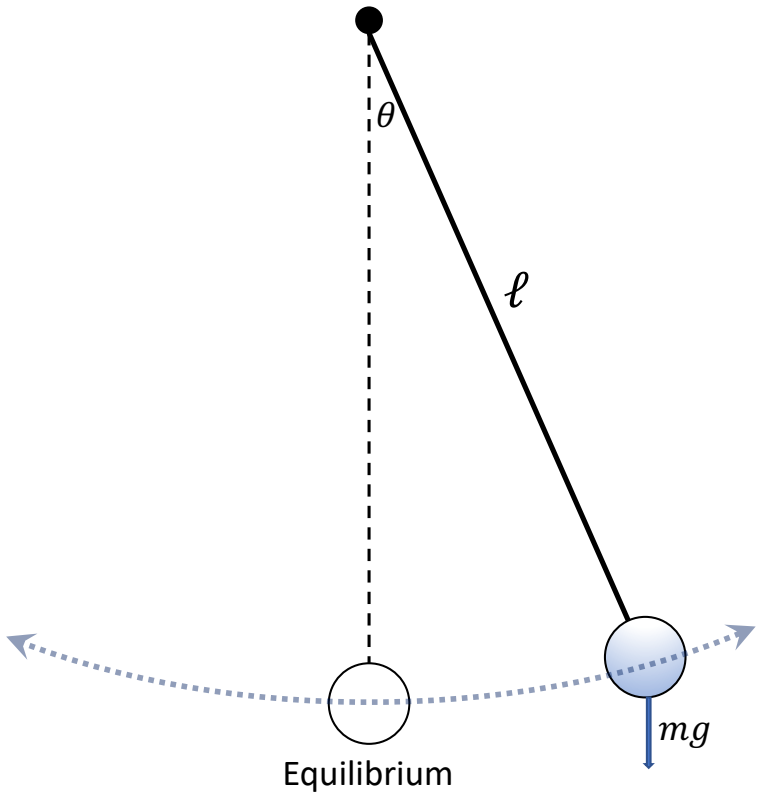

Figure 1.1. Illustration of a simple pendulum operating under gravity. The angle θ and angular velocity $\dot{\theta}$ compose the state of the dynamical system.

which is a nonlinear dynamical system because $\mathbf{f} = \left[x_2, -\frac{g}{\ell}\sin x_1\right]$ contains a nonlinear function sine. The nonlinearity means that—even for this most simple of systems—there is no closed form analytic solution in the general case. For small $\theta \to \sin\theta \sim \theta$ equation (1.2) reduces to a simple harmonic oscillator which does have an closed form solution.

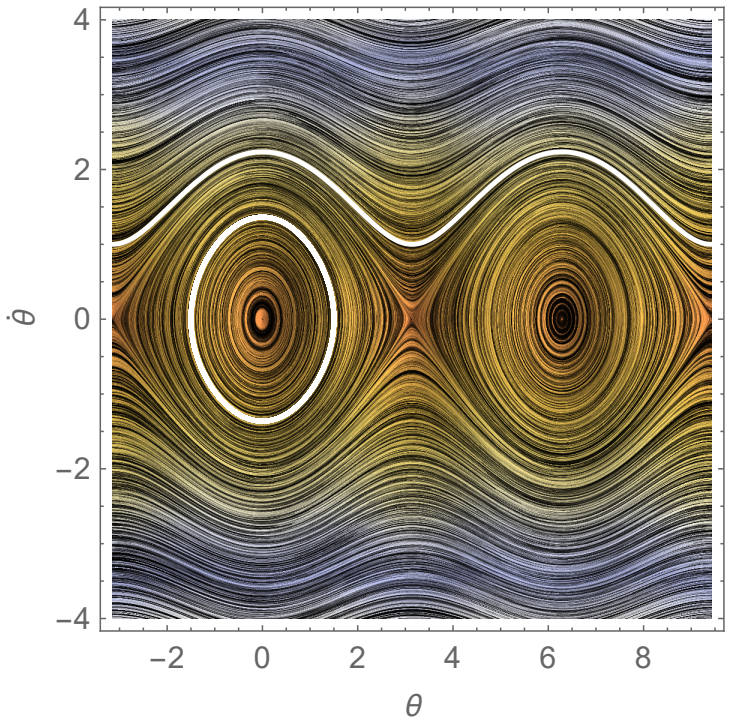

Figure 1.2. The vector field of the pendulum plotted in phase space with two particular trajectories shown in white. Dynamical systems are initial value problems and so depend on the initial conditions. For the pendulum, we see that certain initial conditions are librations with the pendulum simply swinging back and forth. For larger energy states H, specifically for $H > \frac{g}{\ell}$, we see that the solution involves rotations. The line of separation between these two kinds of solutions is called a separatrix.

The combined space of x_1 and x_2—here the angle θ and angular velocity $\dot{\theta}$—is called the phase space. It is in this space that most dynamical systems analysis will take place. Fixed points (equilibrium positions) are those for which $\dot{\theta} = 0$—no velocity implies that the system is stationary. These can be stable or unstable. The plot for the phase space of the pendulum is given in Fig.(1.2) along with two example trajectories. The trajectory forming an ellipse corresponds to the motion of the pendulum swinging back

and forth (libration) while the trajectory moving across the phase space corresponds to the pendulum in rotation. Already it is apparent that there are certain geometrical shapes forming in the phase space portrait, these will be important in the characterization of such systems.

The following sections will give a brief overview of the dynamical systems theory concepts that will underlie the forecasting methods used in the rest of this thesis.

1.1.2 Fixed Points and Stability

Fixed points $\mathbf{x}^\star$ are those for which $\dot{\mathbf{x}} = 0$. For the pendulum Eq.(1.3) this occurs in two places; $x_2^\star = \dot{\theta} = 0$ and $x_1^\star = \theta = 0, \pi$. To evaluate whether these points are linearly stable/unstable we need to see how perturbations to the fixed points evolve with time. If the perturbed system stays in the neighborhood of the fixed point then it is stable. Conversely, if the system moves rapidly away from the fixed point after perturbation then it is unstable.

For the pendulum system, the fixed point at $\theta = 0$—corresponding to the bob hanging down—is stable because a perturbation will cause small amplitude oscillatory motion around the equilibrium position. The fixed point at $\theta = \pi$ with the bob facing the sky is unstable because an infinitesimal disturbance will result in a large response.

We derive the variational equations, which describe the evolution of the tangent vectors/perturbations from $t \to t'$, by taking $\frac{\partial}{\partial x_j(t')}$ on both sides of eq (1.1). Applying the chain rule we find

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial x_i(t)}{\partial x_j(t')} = \frac{\partial f_i(\mathbf{x}(t),t)}{\partial x_k(t)}\frac{\partial x_k(t)}{\partial x_j(t')}$$

which we rewrite as

$$\dot{\boldsymbol{\phi}}(t',t) = \mathbf{J}(\mathbf{x}) \cdot \boldsymbol{\phi}(t',t) \tag{1.4}$$

with $\boldsymbol{\phi}(t',t) = \frac{\partial x_i(t)}{\partial x_j(t')}$ being the $N \times N$ variational matrix and $\mathbf{J}(x(t)) = \mathbf{J}(t) = \frac{\partial f_i(x(t),t)}{\partial x_k(t)}$ being the Jacobian. This procedure is also known as "linearization" because $\mathbf{f}(\mathbf{x} + \delta\mathbf{x}) \sim$

$\mathbf{f}(\mathbf{x}) + \delta\mathbf{x}\mathbf{J}(\mathbf{x}) + \mathcal{O}(\delta\mathbf{x}^2)$ and therefore $\dot{\delta\mathbf{x}} = \mathbf{J}(\mathbf{x})\delta\mathbf{x}$ giving a linearized evolution of the perturbations.

The Jacobian in matrix form is

$$J(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_D} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_D}{\partial x_1} & \cdots & \frac{\partial f_D}{\partial x_D} \end{bmatrix}.$$

For this analysis we add a small damping term κ to the pendulum equation so that Eq.(1.2) becomes $\ddot{\theta} = -\kappa\dot{\theta} - \frac{g}{\ell}\sin\theta \rightarrow f(\mathbf{x}) = \left[x_2, -\kappa x_2 - \frac{g}{\ell}\sin x_1\right]$. The fixed points are unchanged and the Jacobian is

$$J(\mathbf{x}) = \begin{bmatrix} 0 & 1 \\ -\frac{g}{\ell}\cos x_1 & -\kappa \end{bmatrix}.$$

Around the fixed points, the question of interest is whether the perturbations in a particular direction are growing or shrinking. The answer is given by the eigenvalues and eigenvectors of the Jacobian. If the eigenvalues all have negative real parts then the fixed point is stable.

For the damped pendulum with $g/\ell = 1$ and $\kappa = 1$ we find that the eigenvalues of the Jacobian λ around $\mathbf{x}^\star = [0, 0]$ are

$$\lambda = \left[\frac{-1-i\sqrt{3}}{2}, \frac{-1+i\sqrt{3}}{2}\right].$$

We see both eigenvalues have a negative real part as well as an imaginary part. This corresponds to an inward spiral towards the fixed point in the phase space of the dynamical system—see Fig.(1.3). Due to the fact that nearby trajectories all converge to this fixed point, it is called an attractor with an associated basin of attraction. For $\mathbf{x}^\star = [0, \pi]$ we

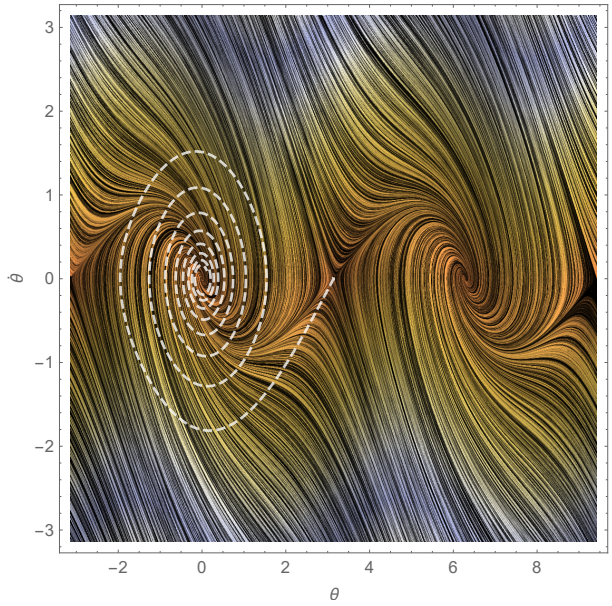

Figure 1.3. Trajectory in the phase space of the damped pendulum for initial conditions perturbed slightly from the unstable fixed point. The trajectory is immediately attracted to the stable fixed point at $\mathbf{x}^\star = [0, 0]$ and converges as an inward spiral. The fixed point at $[0, 0]$ is thus called an attractor because it attracts nearby trajectories. The region surrounding an attractor for which all trajectories converge towards it is called the basin of attraction.

find

$$\lambda = \begin{bmatrix} \frac{-1-\sqrt{5}}{2} & \frac{-1+\sqrt{5}}{2} \end{bmatrix}$$

which has one real positive and one real negative eigenvalue. It is thus an unstable saddle point.

1.2 Lorenz Attractor

Forecasting a simple pendulum is not, in general, a difficult problem. The solution for a particular set of initial conditions/parameters can be written out as an integral evaluation and from there it is not hard to predict the trajectory for the rest of time. In contrast to the pendulum, one of the main difficulties in forecasting the dynamics of complex systems is that many of them are chaotic.

Chaotic nonlinear dynamical systems are deterministic, but have the property that even small uncertainties in initial conditions lead to exponentially growing errors that limit long term forecasts. A typical linear systems analysis (*e.g.*, a fourier transform) of such signals would conclude that the signal is noise, thus missing an essential feature of chaos. An illustrative example of this is the error made by Edwin Colpitts [21] in 1918—who designed and built a nonlinear variable frequency circuit—in identifying the irregularity in the output of his circuit as "noise." 76 years later Michael Kennedy [22] recognized this feature to be deterministic chaos.

The original identification of chaotic motion in nonlinear systems was made by Poincare in the 19th century while analyzing the three body problem of classical mechanics. The present era of understanding and insight into the properties of such systems was initiated by Edward Lorenz in 1963 [23]. Lorenz's simplification of convection in the Earth's lower atmosphere introduced the idea of deterministic, nonperiodic behavior. This system is now known as the "Lorenz system."

The following follows the derivation in Hilborn [24]

The Lorenz system is an example of Rayleigh-Benard convection in an incompressible fluid. The setup is of a box of height h in the z direction and of infinite extent in x/y— Fig.(1.4). The box is heated from the bottom and cooled from the top such that the boundaries are kept at constant temperatures T_H and T_C with differential δT. We assume δT is not too large such that a constant temperature gradient $\frac{\partial T}{\partial z} = \delta T/h$ forms in the z direction.

To analyze when convection will occur we take a infinitesimal volume of fluid in the box and perturb it in the z direction. Since the fluid moving upwards is now warmer than its surroundings it will be less dense, causing it to experience a net upward force. It will also, of course, be losing heat to its surroundings. Loosely, if the movement of the fluid upward due to the buoyancy forces is faster than the drop in temperature, then the perturbation becomes self sustaining and the fluid starts convecting.

To put this intuition into equations, we need to find expressions for 1) the thermal relation time τ_T and 2) the displacement time τ_D. Starting with the τ_T, when our small volume of fluid moves a distance dz the temperature will increase by $dT = \frac{\delta T}{h} dz$. This increase will be counteracted by thermal diffusion $\frac{\partial T}{\partial t} = D_T \nabla^2 T$, where $\nabla^2 T$ can be approximated by $\frac{\delta T}{h^2}\frac{dz}{h}$ and D_T is the thermal diffusion constant. Defining τ_T as the time it takes for a temperature difference dT to dissipate $dT = \tau_T \frac{\partial T}{\partial t}$ we can put the equations together to find that

$$\tau_T \frac{\partial T}{\partial t} = \frac{\delta T}{h} dz = \frac{\tau_T D_T}{h^3} dz \delta T$$

and solving for τ_T our final expression is

$$\tau_T = \frac{h^2}{D_T}. \tag{1.5}$$

Now we have to compare this relaxation time to the movement of the fluid. The fluid density $\rho(T)$ changes as a function of temperature. We can expand $\rho(T)$ around

$\rho(T_H) = \rho_0$ as

$$\rho(T_H + (T - T_H)) = \rho_0 + \frac{\mathrm{d}\rho}{\mathrm{d}T}\Big(T - T_H\Big) + \ldots,$$

with T_H being the temperature at the bottom of the box. Defining the the thermal expansion coefficient $\alpha = -\frac{1}{\rho_0}\frac{\mathrm{d}\rho}{\mathrm{d}T}$ and plugging in for dT we can write

$$\rho(T_H + dT) = (1 - \alpha dT)\rho_0 = (1 - \alpha\frac{\delta T}{h}dz)\rho_0. \tag{1.6}$$

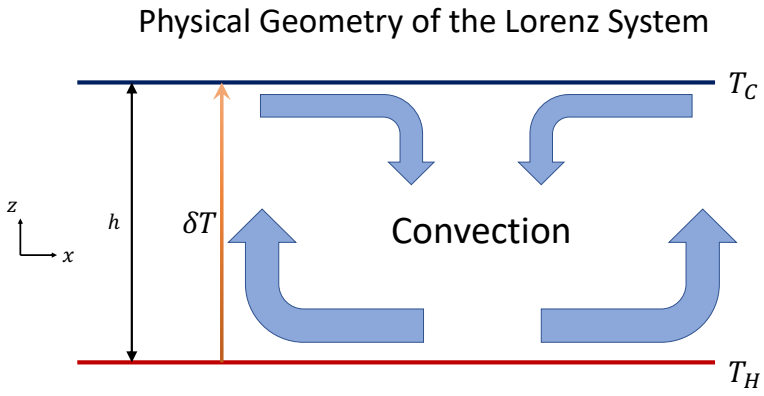

Figure 1.4. Geometry of the physical Lorenz system that roughly corresponds to atmospheric convection.

Plugging eq.(1.6) into the expression for the increased buoyancy force per unit volume as the fluid moves up the column we find

$$\Delta F_B = -\rho(dT)g = \alpha\rho_0 g\frac{\delta T}{h}dz$$

with g the acceleration due to gravity. Making the assumption that the fluid is moving at a constant upward velocity v_z— implying that the net force on the fluid is 0—the buoyancy force must be balanced by the viscous drag

$$F_v = \mu\nabla^2 v_z \sim \mu\frac{v_z}{h^2}$$

with μ being the viscosity. Equating the two forces and solving for v_z we find

$$v_z = \frac{\alpha \rho_0 g h \delta T}{\mu} dz.$$

The displacement time—the time the fluid takes to move a distance dz—is thus

$$\tau_D = \frac{dz}{v_z} = \frac{\mu}{\alpha \rho_0 g h \delta T}. \tag{1.7}$$

Clearly the ratio of τ_T eq.(1.5) to τ_D eq.(1.7) is the critical value

$$R = \frac{\tau_T}{\tau_D} = \frac{\alpha \rho_0 g h^3 \delta T}{D_T \mu} \tag{1.8}$$

where R is called the Rayleigh number. When constructing a system, R is the parameter that most controls how the system will behave. Only certain values of R cause the flow to be chaotic.

The full derivation of the Lorenz equations requires obtaining a solution to the Navier-Stokes equations for this particular geometry. After some of approximations and the nondimensionalization of the variables we end up with equations for the state variables X, Y and Z. $X \propto \psi$ is related to convection where ψ is the the stream function of the fluid; $v_x = -\frac{\partial \psi}{\partial z}$ and $v_z = \frac{\partial \psi}{\partial x}$. Y is related to the horizontal temperature variation and Z to the vertical temperature variation. The final equations are

$$\dot{X} = \sigma(Y - X)$$
$$\dot{Y} = rX - XZ - Y$$
$$\dot{Z} = XY - \beta Z$$

with $\sigma = \frac{\mu}{D_T \rho_0}$ the Prandtl number, $r = \frac{4}{27\pi^4} R$ is the reduced Rayleigh number and $\beta = 8/3$. Changing r leads to different kinds of behavior including, for some values of r,

chaos. Notice that the only nonlinearity in the equations comes from the XZ/XY terms.

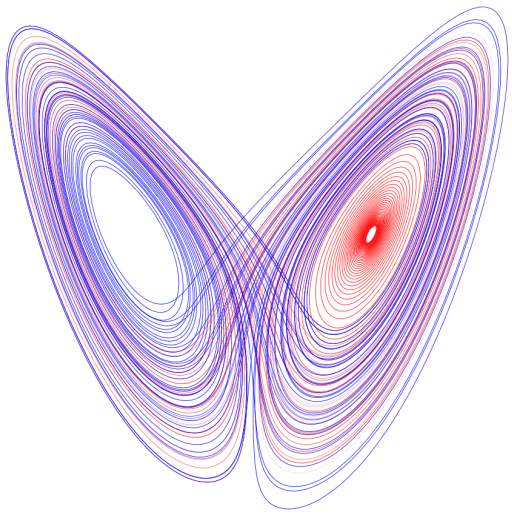

Figure 1.5. Plot of Z vs. X for the famous Lorenz attractor. The Lorenz system is a simplified model for convection in the atmosphere with X representing the convection and Z the vertical temperature variation.

We see in Fig.(1.5) that the Lorenz system traces out a particular shape/object in phase space. This is called a strange attractor. Like the fixed points we saw for the pendulum, this object attracts trajectories that start off of it so that all motion will converge asymptotically to the attractor. Unlike the pendulum example this attractor is not a fixed point but rather a lower dimensional surface in the phase space of the system. This kind of low dimensional surface on which the motion occurs is typical for these kind of systems, with even very high dimensional dynamical systems often having motion that occurs in a small subset of the phase space. These are the effective degrees of freedom and is sometimes described as self organization.

1.2.1 Lyapunov Exponents

How can we characterize the behavior of a chaotic dynamical system? We can do so by computing quantities called the Lyapunov exponents (LEs). The Lyapunov

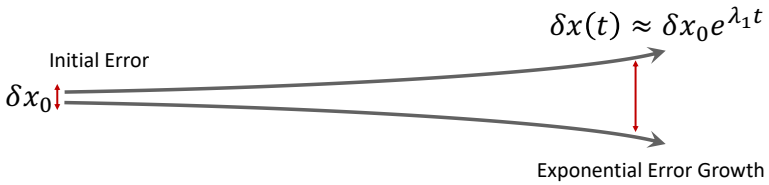

Figure 1.6. The Lyapunov exponents describe the exponential growth of perturbations due to error in the initial conditions of a dynamical system. Positive LEs are a hallmark of chaos and guarantee a finite forecasting horizon.

spectrum, composed of a system's LEs, characterizes a dynamical system [25, 26] by given a quantitative measure of how a volume of phase space stretches/shrinks over time. The LEs are an extremely useful quantity; for example, one definition of a chaotic system is that there is at least one positive LE [27]. LEs can also be used to estimate the rate of entropy production (Kolmogorov-Sinai Entropy) and the fractal or information dimension [27].

The LEs describe how two points x_1 and x_2, separated initially by a distance $\delta x_0 = |x_1 - x_2|$, evolve in time with respect to one another. More specifically,

$$\delta x(t) \approx \delta x_0 e^{\lambda_1 t}$$

with λ_1 being the largest LE[1]. Therefore if λ_1 is positive, over time the two initial conditions that at first only differ by a small amount will diverge away from one another exponentially fast in the linearized regime—the definition of chaos.

[1]it is important to note that the error only actually grows exponentially as λ_1 when δx lies exactly along the direction λ_1, however for large t the largest exponent will dominate.

Starting from the variational equation eq.(1.4)

$$\dot{\boldsymbol{\phi}}(t',t) = \mathbf{J}(x(t)) \cdot \boldsymbol{\phi}(t',t) \tag{1.9}$$

the general solution involves the matrix exponential

$$\boldsymbol{\phi}(t',t) = \mathcal{T}_+ \exp\left\{\int_t^{t'} \mathbf{J}(x(t))dt\right\}\boldsymbol{\phi}(t,t)$$

with $\boldsymbol{\phi}(t,t) = \mathbb{I}$ and $\mathcal{T}_+$ being the *time ordering operator* [28] for two operators (here matrices) $\mathbf{A}(t)$ and $\mathbf{B}(t')$ that are not commutative,

$$\mathcal{T}_+\mathbf{A}(t)\mathbf{B}(t') = \begin{cases} \mathbf{A}(t)\mathbf{B}(t') & \text{if } t > t' \\ \mathbf{B}(t')\mathbf{A}(t) & \text{if } t < t' \end{cases}.$$

$\mathcal{T}_+$ is important because the Jacobian matrices are not commutative *i.e.*, $\Big[\mathbf{J}(t), \mathbf{J}(t')\Big] \neq 0$.

The variational equation is much easier to work with in discrete time $t = t_0 + n\Delta t;\ t' = t_0 + n'\Delta t$ where it appears as

$$\boldsymbol{\phi}(n',n) = \mathbf{DF}(n)\boldsymbol{\phi}(n'-1,n),$$

and $\boldsymbol{\phi}(n,n) = \mathbb{I}$.

Then the solution for $\boldsymbol{\phi}(n+L,n)$ is the product of Jacobians

$$\begin{aligned} \boldsymbol{\phi}(n+L,n) \quad &= \mathbf{DF}(n+L) \cdot \boldsymbol{\phi}(n+L-1,n) \\ &= \mathbf{DF}(n+L) \cdot \mathbf{DF}(n+L-1) \cdot \boldsymbol{\phi}(n+L-2) \\ &= \mathbf{DF}(n+L) \cdot \mathbf{DF}(n+L-1) \cdot \ldots \mathbf{DF}(n). \end{aligned} \tag{1.10}$$

The matrix $\boldsymbol{\phi}(t',t)$ describes how small perturbations to a state $x(t)$ propagate to

the state at $x(t')$. Given equations (1.1) and (1.9) one can solve them concurrently to find Oseledec's matrix $\mathbf{\Phi} = \boldsymbol{\phi}(t', t)\boldsymbol{\phi}(t', t)^T$.

The eigenvalues of the log of $\mathbf{\Phi}(t', t)$ for large times

$$\lim_{t' \to \infty} \frac{1}{2t'} \log \boldsymbol{\phi}(t', t)\boldsymbol{\phi}(t', t)^T \tag{1.11}$$

are the **global** Lyapunov Exponents; the N eigenvalues are by definition real and the eigenvectors orthogonal since $\mathbf{\Phi}$ is a symmetric $N \times N$ matrix. We order the LEs $\lambda_1 > \lambda_2 > \ldots > \lambda_N$. The matrix $\mathbf{\Phi}$ is ill-conditioned, so accurately evaluating all of its eigenvalues requires a stable algorithm, for example as provided by [26, 27].

Oseledec's multiplicative ergodic theorem [29] states that all N LEs of a dynamical system (a) exist, (b) are independent of the initial starting point $x(t)$, and (c) are invariant under smooth coordinate transformations [2]. The finite time Lyapunov exponents (FTLE) [30], Eq.(1.11) evaluated for finite t', are not independent of $x(t)$ nor are they invariant under a smooth coordinate transformation. Furthermore, for a continuous time dynamical system one of the LEs must be 0, and $\sum_{i=1}^{N} \lambda_i \leq 0$. In Hamiltonian systems the spectrum of LEs is symmetric around 0 and thus $\sum_{i=1}^{N} \lambda_i = 0$, a natural statement of phase space volume conservation guaranteed by Liouville's theorem.

Doing such a calculation for the Lorenz system we find that the LEs are $\lambda = [0.9, 0.0, -14.3]$. We can see therefore that phase space volumes tend to shrink $\sum_i \lambda_i < 0$ so this is a dissipative system; in one direction, however, the LE is positive so there is chaos because nearby trajectories tend to diverge.

1.2.2 Fractal Dimensions

The geometrical surface in phase space that the trajectories of the Lorenz system occupy is called a "strange attractor" due to the sensitivity to initial conditions that

[2] The statement of invariance under coordinate transformation is important when attempting to recover the LEs from a time delay embedding.

trajectories on the surface exhibit. This attractor has infinite surface area but 0 volume (by definition phase space volumes in dissipative systems shrink to 0) and is thus an example of a fractal set. There are a number of methods for characterizing the attractor, not least its natural invariant density ρ which gives a notion of how often a trajectory will visit a particular point on the attractor [31]. Functions integrated with the density $\int d^D\mathbf{x}\rho(\mathbf{x})g(\mathbf{x})$ are conserved. A useful quantity for characterizing the attractor is its dimension. An intuitive definition of the dimension is often given as

$$\text{dimension} = \frac{\log \mu(\text{M})}{\log \frac{1}{2}\|\mathbf{x}_{\min} - \mathbf{x}_{\max}\|_2}$$

where $\|\mathbf{x}_{\min} - \mathbf{x}_{\max}\|$ gives the diameter of the set [31] and $\mu(M) = \int_M \rho(\mathbf{x})d\mathbf{x}$ is the invariant measure over the available state space M. This definition corresponds to the physically intuitive idea that Volume = $\text{r}^{\text{dimension}}$ where r is the radius and the volume here corresponds to the notion of volume given by the measure. While intuitive, this definition can often be difficult to calculate in practice.

The Kaplan-Yorke (KY) dimension [32, 33] or Lyapunov dimension gives an upper bound on the dimension a strange attractor through the Lyapunov exponents of that system. By definition, to calculate the KY dimension, arrange the Lyapunov exponents from largest to smallest $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_D$, where D here is the dimension of the system or reservoir.

Let α be the index for which

$$\sum_{i=1}^{\alpha} \lambda_i \geq 0$$

and

$$\sum_{i=1}^{\alpha+1} \lambda_i < 0$$

Thus the Lyapunov/KY or information dimension of the system is [34]

$$\text{Dimension} = \alpha + \frac{\sum_{i=1}^{\alpha} \lambda_i}{|\lambda_{\alpha+1}|}. \tag{1.12}$$

For the Lorenz system we find that the fractal dimension of the attractor is 2.06. Thus it is clear that the motion of the system is taking place on a low dimensional surface and that surface has a non-integer dimension. The fractal and chaotic nature of the dynamical system, even for such a simple set of equations, makes forecasting incredibly difficult. Small errors exponentially increase and the nonlinearity makes any optimization procedure nonconvex, meaning there is no efficient method to find a global minimum. The field of data assimilation (DA) is dedicated to finding the optimal method of integrating measurements of complex system into a model to produce forecasts.

1.3 Data Assimilation

Given measurements of a system we must transfer that information to a model in such a way that the model generalizes to a time period beyond the measurement window. This generalization step is called prediction or forecasting. The underlying system that generates the states is given by Eq.(1.1)

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = f_i(\mathbf{x}(t), t) \ ; \ i \in [1, \ldots, D].$$

Our measurements are given at the discrete times $t = [t_0, t_1, \ldots, t_{M-1}, t_M]$, where at each of those times we observe the state $\mathbf{u} = H(\mathbf{x}) + \epsilon_H \in \mathbb{R}^L$, where H is the observation operator with $L \leq D$ and noise ϵ_H. In general we do not know $\mathbf{f}$ exactly and therefore we must parameterize our system as

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = \tilde{f}_i(\mathbf{x}(t), t, \theta) \ ; \ i \in [1, \ldots, D].$$

where θ are some parameters to be estimated and $\tilde{\mathbf{f}}$ is an approximation to our true system $\mathbf{f}$. This model may have errors in it and the measurements could be sparse $L << D$ and noisy. The algorithms that have been designed over the years to solve this problem fall under the umbrella of data assimilation (DA).

1.3.1 Kalman Filter

The simplest solution to this problem goes back to Kalman in 1960[35] with his solution to the linear filtering problem. A filtering problem specifically refers to the sequential processing of the data such that we would like to estimate the future state given the previous state and our measurements. The Kalman filter solves the optimal linear interpolation problem between two estimations with known variance. As an example let us say we are measuring some property of a dynamical system, say true temperature T_t, with measurement T_u which is measured with error σ_u and a model prediction T_x which has a precision σ_x. The combined estimate of the system T will be the interpolation $\hat{T} = \alpha T_u + \beta T_x$ where $\alpha + \beta = 1$. Assuming the estimate is unbiased *i.e.*, $\langle \hat{T} \rangle = \langle T_t \rangle$, the error on the interpolation $\hat{\sigma}^2 = \langle [\alpha(T_u - T) + (1-\alpha)(T_x - T)]^2 \rangle$ can be minimized with respect to α. Setting $\frac{\partial \hat{\sigma}^2}{\partial \alpha} = 0$ we find

$$\alpha = \frac{\sigma_x^2}{\sigma_u^2 + \sigma_x^2} \quad \beta = \frac{\sigma_u^2}{\sigma_u^2 + \sigma_x^2}$$

which shows that the optimal interpolation of the two values is a function of their variances. Furthermore the total error

$$\hat{\sigma}^2 = \frac{\sigma_u^2 \sigma_x^2}{\sigma_u^2 + \sigma_x^2}$$

is less than that of the individual measurements σ_x^2, σ_u^2 showing that we have indeed used our measurements to reduce the error on the total $\hat{T}$ estimate. The form often used in the Kalman filter is

$$\hat{T} = T_x + K(T_u - T_x)$$

where in this case $K = \frac{\sigma_x^2}{\sigma_u^2+\sigma_x^2}$ is the gain matrix which is correcting the model estimate T_x with the measurement T_u.

Expanding our analysis to a larger system than just one variable, the assumptions for the Kalman filter are that we have a linear system $\dot{\mathbf{x}} = \mathbf{F}\mathbf{x}+\epsilon_F$, a linear observation operator $\mathbf{u} = \mathbf{H}\mathbf{x} + \epsilon_H$ and gaussian noise: $\epsilon_H \sim \mathcal{N}_L(0, \Sigma_H)$, $\epsilon_F \sim \mathcal{N}_D(0, \Sigma_F)$ with $\mathcal{N}_k(\langle \mathbf{x} \rangle, \Sigma) = \frac{1}{\sqrt{(2\pi)^k \Sigma}} \exp\left[\frac{1}{2}(\mathbf{x} - \langle \mathbf{x} \rangle)^T \Sigma^{-1} (\mathbf{x} - \langle \mathbf{x} \rangle)\right]$ the k dimensional multivariate Gaussian. Therefore the markov update of the model giving the probability of the next state given the previous one is

$$P(x_{i+1}|x_i) \sim \mathcal{N}(\mathbf{F}\mathbf{x}, \Sigma_F)$$

and the probability of the measurement given the model is

$$P(u_i|x_i) \sim \mathcal{N}(\mathbf{H}\mathbf{x}, \Sigma_H).$$

Now given these assumptions we can find our solution for maximum of the probability of the *nth* state x_n given all the previous measurements $[u_n, \ldots, u_1]$. $P(x_n|u_n, \ldots, u_1) = \mathcal{N}(\langle x_n \rangle, \Sigma_n)$ with

$$\begin{aligned} \langle x_n \rangle &= F \langle x_{n-1} \rangle + K(u_n - \mathbf{H}\mathbf{F}x_{n-1}) \\ \Sigma_n &= (\mathbb{I} - K\mathbf{H})\Sigma_n^- \end{aligned}$$

where $\langle x_{n-1} \rangle = \langle P(x_{n-1}|u_{n-1}, \ldots, u_1) \rangle$ and $\Sigma_n^- = \mathbf{F}_{n-1}\Sigma_{n-1}\mathbf{F}_{n-1}^T + \Sigma_F$. Looking at $\langle x_n \rangle$ we see that the matrix $\mathbf{K}$ is interpolating between the model update $\mathbf{F}\mathbf{x}_{n-1}$ and the difference in the measurements and the model $u_n - \mathbf{H}\mathbf{F}\mathbf{x}_{n-1}$. The matrix $\mathbf{K}$ is the Kalman gain and is given by

$$\mathbf{K} = \Sigma_n^- H^T [H\Sigma_n^- H^T + \Sigma_H]^{-1}$$

which is the weighting between the model error and the measurement error. This scheme gives an iterative update formula where we forecast from an initial time and update the state estimates as measurements come in. The solution is exact but only when the assumptions of linearity and gaussian noise hold. For complex system these are not the case and extensions and approximation must be made.

We proceed with a fully nonlinear formulation of data assimilation and its application to various problems in neurobiology. Following that, we will look at purely data driven methods of forecasting, this requires dispensing with the physically derived model and using a more flexible approach.

Chapter 2

Data Assimilation in Biological and Neuromorphic Networks

Parts of this chapter are adapted from [1] previously published by Frontiers.

The human brain is one of the great unexplored frontiers, in the sense that the measurements we have of its function do not come close to explaining the emergent property that is human behavior. Much of our incomprehension is the result of the utter inadequacy of our measurement techniques when faced with 10^{11} neurons and up to 10^{15} synaptic connections [36]. The experimentalist's tools are often limited to

1. detailed electro-physiological recordings of a handful neurons, these can be cell-attached to explore single ion channels or whole cell recordings
2. extracellular recordings (spiking) of a few tens of neurons
3. fluorescence imaging techniques (e.g., calcium fluorescence)
4. whole brain imaging such as large scale MRI recordings
5. pharmacological blockers to block certain neurotransmitters and ion channels

with each technique having its own drawbacks [37]. For instance the electrophysiological recordings are generally the only method with enough information to fit a biophysical model, but patch clamp experiments are limited to a mere fraction of available neurons.

The other techniques can generally show aggregate behaviors, but not in the level of detail needed to determine the exact equations. Functional networks, arising from graph theory and determined through Ca imaging, have been used to gain insight into correlations between neurons but aggregate activity over time, removing information on causality within the network so crucial to the dynamics.

Nonlinear dynamics has been a powerful tool in neuroscience, applied to networks it describes activity such as sequential switching between populations of neurons. An interesting biological network whose operation can be described through a dynamical systems framework is the insect olfactory system. This system is composed of receptor neurons that react to chemical input, the antennal lobe which processes the input, and the mushroom body that classifies the various inputs. More detail can be found in Fig.(2.1).

When stimulated by distinct current inputs from a sensory network of olfactory receptor neurons, the antennal lobe—the equivalent of the olfactory bulb in insects—produces trajectories in the network phase space following distinct heteroclinic sequences among unstable regions. These trajectories move from unstable regions to other unstable regions, and continue to do so as long as the stimulus persists. When the stimulus ends, the trajectory retreats to a stable fixed point region where it responds to environmental noise [45–47]. These observed properties led to a suggestion of an AL network structure [48–50] called **winnerless competition networks** (WLC).

The idea of the AL as a WLC network was examined in experiments on locust olfactory networks by Mazor and Laurent [51]. They found that the AL responses to stimulating odors of varying duration were described by: (1) an "on-transient", when the stimulus is first received, (2) an "off-transient" as the stimulus recedes and the neural activity returns to its stable base state, and (3) movement around a 'fixed point' or stable region in AL neuron phase space. They noted that "optimal stimulus separation occurred during the transients. . . ", suggesting that the biological AL acts as a WLC network with added longer time scale activity and odor specific dynamics.

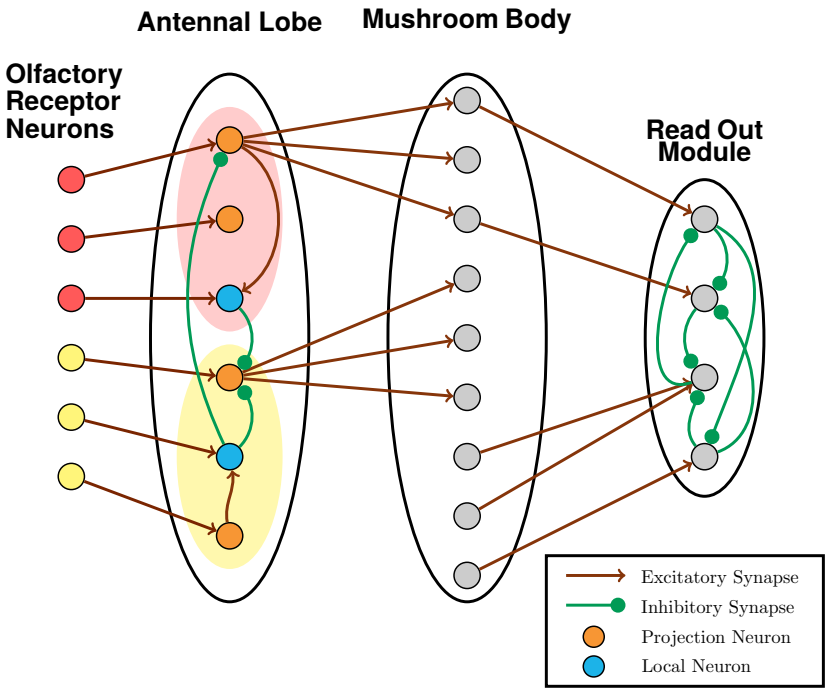

Figure 2.1. The biological neural circuits that perform the identification and classification of components in odors. The initial stage is composed of olfactory receptor neurons (ORN) that are activated by a particular chemical component of the odor. The representation of odors in ORNs is generally a complicated nonlinear function of the inputs [38, 39]. Responses of ORNs to mixtures of odors, in particular, are characterized by a large variety of receptor modulation mechanisms [40]; the form of the input greatly influences the subsequent learning process. The ORNs project to the Antennal Lobe (AL), within which are excitatory projection neurons (PNs) and inhibitory interneurons (LNs); in locusts there are approximately 850 PNs and 300 LNs. The PNs carry AL activity forward to the next stage of olfactory recognition called the mushroom body (MB) [41], which is suggested to act as a support vector machine in the biological olfactory network [42]. The AL and MB act to process the encoding given by the ORNs, first in the AL through a dense spatio-temporal encoding and then in the MB as a sparse, high dimensional representation [43, 44].

Once the WLC structure is established, the precise trajectory of the network response in network state space and time is determined by the specific stimulus. Each distinct stimulus defines a specific direction in the high dimensional space of the WLC network. As the network state space is composed of multiple regions of nonlinear, unstable behavior, the phase space trajectory is seen to be quite sensitive to the selected current stimulus. This sensitivity suggests that a WLC network could distinguish among many 'nearby' stimuli. Indeed, there is an estimation of the capacity of $e(N-1)!$ for a WLC network of N neurons [49].

Chapter 3 and 4 will delve into the dynamical operations of networks of artificial neurons as applied to forecasting problems, there are many parallels between these networks and the WLC network described above. In the rest of chapter 2 we look into the application of nonlinear dynamics in the analysis of single neuron models. Single neurons are the components of networks and are vitally important for building up networks with true biophysical components. Unfortunately, even the process of building the model and estimating its parameters is fraught with difficulties [52, 53]. The neuron models are highly nonlinear, the experiments difficult, and the variation in model complexity wide even for neurons spatially close to one another. We explore here a number of problem formulations, based on Bayesian inference, that allow us to solve these problems. After formulating the problem we apply the technique to estimating the parameters of neurons (both biological and instantiated on physical circuits) from experimental data.

2.1 Statistical Data Assimilation

Instead of the sequential filtering problem—such as in the Kalman filter described in chapter 1—for problems in neurobiology we are attempting to *jointly* estimate the state and parameters of a highly nonlinear model over an observation window. The objective is the correct estimation of the parameters in the physical model, with the forecast simply

to check the result. Therefore the quantity we are attempting to estimate is the joint probability distribution $P(X|U)$ over all the states and parameters $X = [\mathbf{x}_n, \ldots, \mathbf{x}_0]$, given the measurements $U = [\mathbf{u}_n, \ldots, \mathbf{u}_0, \mathbf{u}_b]$ and the prior over the initial condition $\mathbf{u}_b$—also known as the background.

From Bayes rule we can rewrite

$$P(X|U) = \frac{P(U|X)P(X)}{P(U)}. \tag{2.1}$$

The denominator does not depend on X so we can ignore it in when searching for the optimal X. We assume that the measurements are only dependent on the present state of the system $P(U|X) = P(\mathbf{u}_b|\mathbf{x}_0) \prod_{i=0}^{n} P(\mathbf{u}_i|\mathbf{x}_i)$ so that Eq.(2.1) becomes

$$P(X|U) = P(X)P(\mathbf{u}_b|\mathbf{x}_0) \prod_{i=0}^{n} P(u_i|x_i).$$

Now we assume that the differential equation follows a Markov process so that the next state only depends on the previous state $P(\mathbf{x}_j|\mathbf{x}_{j-1}, \ldots, \mathbf{x}_0) = P(\mathbf{x}_j|\mathbf{x}_{j-1})$ so that our final expression is

$$P(X|U) = P(\mathbf{u}_b|\mathbf{x}_0)P(\mathbf{x}_0) \prod_{i=0}^{n} P(\mathbf{u}_i|\mathbf{x}_i) \prod_{j=1}^{n} P(\mathbf{x}_j|\mathbf{x}_{j-1}). \tag{2.2}$$

Expressing the solution to Eq.(2.2) as

$$P(X|U) \propto \exp[-A(X|U)]$$

we find

$$A(X|U) = -\sum_{j=0}^{n-1} \log P(\mathbf{x}_{j+1}|\mathbf{x}_j) - \sum_{i=0}^{n} \log P(\mathbf{u}_i|\mathbf{x}_i) - \log P(\mathbf{u}_b|\mathbf{x}_0)P(\mathbf{x}_0)$$

with A called the "Action." We solve for the conditional expected values for functions

along the path of the state X as

$$E[G(X)|U] = \frac{\int dX G(X) \exp[-A(X|U)]}{\int dX \exp[-A(X|U)]}.$$

When observations U are related to their model counterparts via covariance matrix Σ_H, and model errors are associated with Σ_F then the action assumes the form

$$\begin{aligned} A(X|U) =& \frac{1}{2}\sum_{t=1}^{n}(\mathbf{x}(t) - \mathbf{f}[\mathbf{x}(t-1)])^T \Sigma_F^{-1}(\mathbf{x}(t) - \mathbf{f}[\mathbf{x}(t-1)]) + \\ & \frac{1}{2}\sum_{t=0}^{n}(\mathbf{u}(t) - H[\mathbf{x}(t)])^T \Sigma_H^{-1}(\mathbf{u}(t) - H[\mathbf{x}(t)]) + \\ & \frac{1}{2}(\mathbf{x}_0 - \mathbf{u}_b)^T \Sigma_B^{-1}(\mathbf{x}_0 - \mathbf{u}_b) \end{aligned} \tag{2.3}$$

where the matrix Σ_B represents the error on the prior belief (or background) estimate of the system; in the following case this term will be taken to be 0. This cost function is called the weak 4D-var formulation of data assimilation in the meterological community[54–56], with the background estimate usually given by the previous forecast cycle Fig.(2.2).

To locate the minima of the action $A(X) = -\log[P(X|U)]$ we must seek paths $X_j; j = 0, 1, \ldots$ such that $\frac{\partial A(X)}{\partial X_j} = 0$, and then check that the second derivative at X_j, the Hessian, is a positive definite matrix. The vanishing of the derivative is a necessary condition. Laplace's method [57–59] expands the action around the X_j seeking the path X_0 with the smallest value of $A(X)$. The contribution of X_0 to the integral is approximately $\exp[A(X_1) - A(X_0)]$ bigger than that of the path with the next smallest action.

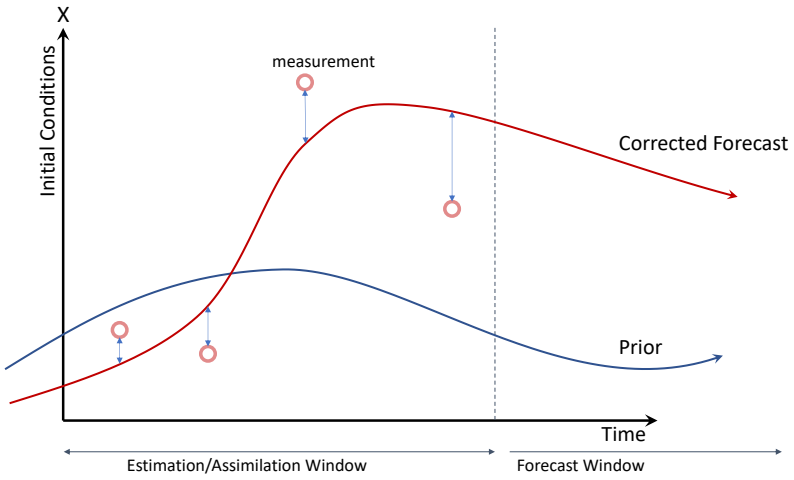

Figure 2.2. Illustration of how 4D-var combines a model, measurements and prior into a best guess estimate of the corrected forecast. Since the measurements are sparse and noisy, a physical model is needed in order to transfer information from the observed states to the unobserved states. The corrected solution is the state X that minimizes Eq.(2.3).

2.2 Euler-Lagrange Formulation

The continuous time version of the action with no prior/background is

$$A[\mathbf{x}(t)] = \frac{1}{2}\int_{t_0}^{t_f} dt[(\dot{\mathbf{x}}(t) - \mathbf{f}[\mathbf{x}(t)])^T \Sigma_F^{-1}(\dot{\mathbf{x}}(t) - \mathbf{f}[\mathbf{x}(t)]) + (\mathbf{u}(t) - H[\mathbf{x}(t)])^T \Sigma_H^{-1}(\mathbf{u}(t) - H[\mathbf{x}(t)])] \tag{2.4}$$

where f is now the continuous time model rather than a map with $\mathbf{u}$ and $\mathbf{x}$ continuous variables and the action A is a functional over the functions $\mathbf{x}(t)$ rather than variables X. The derivation of the continuous time action is given in detail in [60] using stochastic path integrals analagous to those in quantum field theory [61]. The correct equation is not, in fact, a direct application of the limit $\Delta t \to 0$ as in Eq.(2.4). On application of Ito's lemma [62] to find the integral expression for the action, the term $\frac{1}{2}\int \boldsymbol{\nabla} \cdot \mathbf{f}[\mathbf{x}(t)]dt$ must be added. We ignore this term in the subsequent analysis because we are making a suggestive rather than quantitative argument, but a direct application of the continuous

action will require this extra divergence term in the equations of motion.

Eq.(2.4) has an associated Lagrangian $A = \int \mathcal{L}(x, \dot{x}, t)dt$ such that

$$\mathcal{L}(x, \dot{x}, t) = \frac{1}{2}[(\dot{\mathbf{x}}(t) - \mathbf{f}[\mathbf{x}(t)])^T \Sigma_F^{-1} (\dot{\mathbf{x}}(t) - \mathbf{f}[\mathbf{x}(t)]) + (\mathbf{u}(t) - H[\mathbf{x}(t)])^T \Sigma_H^{-1} (\mathbf{u}(t) - H[\mathbf{x}(t)])].$$

The extremum of this action is given by the Euler-Lagrange equations for the variational problem [63]

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\mathrm{d}}{\mathrm{d}t} \frac{\partial \mathcal{L}}{\partial \dot{x}}$$

with the boundary conditions [60]

$$\frac{\partial \mathcal{L}}{\partial \dot{x}} = 0.$$

Plugging in for our Lagrangian and after some manipulation we end with the Euler-Lagrange equations for the action Eq.(2.4) [64]

$$\left[\delta_{ij} \tfrac{\mathrm{d}}{\mathrm{d}t} + \tfrac{\partial f_j(\mathbf{X})}{\partial x_i}\right] \left[\dot{x}_j - f_i(\mathbf{x})\right] = \frac{\Sigma_F}{\Sigma_H} \delta_{ik} (u_k - H[x_k]).$$

The right hand side of this equation is suggestive of a particular interpretation of the equations of motion. Namely, the equations of the system—which are solving a two point boundary value problem—are being forced at the measurement locations [64]. This forcing is reminiscent of the synchronization of a dynamical system through the introduction of a forcing term $k(u - H[x])$—synchronizing the dynamical system to the measurements with a coupling constant k.

With this suggestion in mind we rewrite the equations of motion as

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}) + k(t)(\mathbf{u} - H[\mathbf{x}]) \tag{2.5}$$

where k is a time dependent parameter expressing the forcing of the model equations by the measurements. This is sometimes called nudging [65] and gives the equations of

motion for the DA problem. This form will be used directly to find the optimal state and parameter estimate for the system.

2.3 Nudging

The minimization of $A(X|U)$ is in general a non-convex, NP complete problem with large numbers of local minima. We can, however, make use of the nudging formulation [66] to pose the minimization of $A(X|U)$ a generalized synchronization problem. We introduce the non-physical time dependent parameter $k(t)$ into the dynamical system in order to synchronize the model with the observations [67]. The minimization of the action of Eq.(2.3) is rewritten as a constrained optimization problem with minimization over the measurements with the constraints being the DA equations of motion Eq.(2.5). The minimization problem is

$$\begin{aligned} &\min_{X,k}. && A_{\text{nudge}}(X|U) \\ &\text{subject to} && \dot{x} = f(\mathbf{x}(t)) + k(t)(H[\mathbf{x}(t)] - \mathbf{u}(t)) \end{aligned} \tag{2.6}$$

with our modified action as

$$A_{\text{nudge}} = \sum_{t=0}^{n} \|H[\mathbf{x}(t)] - \mathbf{u}(t)\|^2 + \|k(t)\|^2 \tag{2.7}$$

and $k(t) = 0$ when we have no observation, we can now solve this problem with an interior point optimization routine [68]. While the problem is not convex in general, we are guaranteed convexity in the limit as $k \to \infty$ as long as the measurement operator is linear. To see this we can take the limit and see that the optimization problem becomes

$$\begin{aligned} &\min_{X,k}. && \textstyle\sum_{t=0}^{n} \|k(t)\|^2 \\ &\text{subject to} && \dot{\mathbf{x}} = k(t)(\mathbf{H}\mathbf{x}(t) - \mathbf{u}(t)) \end{aligned}$$

which is a convex problem with a solution at $k = 0$ and $\mathbf{Hx} = \mathbf{u}$.

To provide some intuition into the nudging operation we can start by analyzing Eqs.(2.6, 2.7) at the initialization of the optimization routine when the euclidean distance between the measurements and the model $\|H[\mathbf{x}(t)] - \mathbf{u}(t)\|^2$ is large and the coupling constant $k \sim 0$. Since the initial distance between $\mathbf{u}$ and H[$\mathbf{x}$] is large, the initial iterations of the routine will minimize the cost function by increasing the coupling constant k, driving the model towards the measurements and finding the solution to the convex version of the problem. Once $\|H[\mathbf{x}(t)] - \mathbf{u}(t)\|^2 \sim 0$ the subsequent minimization routine will minimize k^2 thus returning the solution to the original minimization problem, but with $H[\mathbf{x}]$ still close to $\mathbf{u}$ and with $k \sim 0$. Thus we have recovered the solution to the DA problem with the non-physical synchronization term k small and our parameters estimated. While not guaranteed to converge to the optimum, in practice this technique has been found to give good estimates of the parameters for small systems [69].

2.4 Hodgkin-Huxley Model

The Hodgkin-Huxley equations (HH) [70] describe the non-linear dynamical system governing the action potential of an neuron with sodium (Na) and potassium (K) ion channels, as well as some leakage. Dynamics are governed by the voltage and three parameters m, h and n which are the sodium current activation, inactivation and potassium current activation respectively. The HH equations have been held up to be one of the great triumphs of biophysics since they predicted the existence and use of ion channels in the neural system from extensive voltage experiments on the giant squid axon. At the time it was not feasible to look for the movement of ions directly so their existence had to be inferred. The HH equations can also be extended by adding extra ion channels into the equations of motion, making the extended HH model the bread and butter of computational neuroscience.

The Hodgkin-Huxley equations are given by

$$C_m\dot{V}(t) = \bar{g}_{Na}m^3h(E_{Na} - V) + \bar{g}_K n^4(E_k - V) + g_L(E_L - V) + I_{inj}(t) \tag{2.8}$$

$$\dot{q}(t) = \frac{q_\infty(V) - q}{\tau_q(V)};\ q \in [m, h, n] \tag{2.9}$$

$$\tag{2.10}$$

where

$$q_\infty(V) = \frac{1}{2}(1 + \tanh[\frac{V - \sigma_q}{\Delta_q}])$$
$$\tau_q(V) = t_q^0 + t_q^1(1 - \tanh^2[\frac{V - \sigma_q}{\Delta_q}]).$$

The model has $16 - 19$ configurable parameters (depending on the exact formulation) and is fundamentally a relaxation oscillator in that the motion can be described as a spike, followed by a period of quiescence and then often continued by a new spike.

Physically, a (simplified) neuron is composed of the

1. soma, the body of the neuron containing the nucleus and the other accoutrements of a typical cell

2. dendrites, receptor branches for the input chemical signals into the neuron

3. axon, the single—often extremely long—branch from the soma connecting to other neurons.

The HH equations describe the reaction of the voltage across the cell membrane to an external stimulus. In biological settings the stimulus is given by the interaction of synaptic receptors in the dendrites to inbound chemical stimulus such as the neurotransmitters GABA, acetocholine, dopenmine, etc. In patch clamp experiments the input stimulus is a direct current injection directly into the neuron.

2.5 NeuroDyn

An ambitious effort in neuroscience is the creation of low power consumption analog neural-emulating very large-scale integration (VLSI) circuitry [71]. The goals for this effort are the development of fast, reconfigurable circuitry on which to incorporate information revealed in biological experiments for use in

1. spiking neural networks for "edge computation" applications where low power consumption is key [72]

2. hardware for brain-computer interfaces which can interact with the nervous system on both synaptic and network scales.[73]

One of the roadblocks in achieving critical steps towards these goals is the error prone nature of the manufactured circuitry. Given the sensitivity of the models being instantiated on these VLSI chips, the tolerances on the components are too high to guarantee reasonable behavior. To overcome this barrier in using the VLSI devices in networks, one can specify an algorithmic tool to determine the factory components without having to manually measure the constituent parts. As each chip is an electronic device built on a model design, and the flaws in manufacturing are imperfections in the realization of design parameters, the data from the physical chip and data assimilation can be used to estimate the parameter offsets on the chip.

If we present to the chip input signals, we can measure everything about each output from the chip and use DA to estimate the actual parameters produced in manufacturing. Of course, we do not know those parameters a priori so after estimating the parameters, thus "calibrating" the chip, we must use those estimated parameters to predict the response of the chip to a new stimulating currents. That will validate the completion of the model of the actual circuitry on the chip and permit confidence in using it in building interesting networks.

The NeuroDyn chip [74, 75] simulates the behavior of a HH Neuron on an integrated circuit. While the behavior of NeuroDyn is similar to a neuron, the model and the parameters of a physical neuron are translated into various currents and constants on the chip itself. The dynamics in the model are given by a Na ion current, a K current and a leak current

$$\begin{aligned} C_{mem}\frac{dV}{dt} &= I_{inj} - I_{Na} - I_K - I_L \\ I_{Na} &= \frac{k}{V_T} I_1 (V - E_{Na}) \\ I_K &= \frac{k}{V_T} I_2 (V - E_K) \\ I_L &= \frac{k}{V_T} I_3 (V - E_L) \end{aligned}$$

where the currents I are functions of the gating variables n, m and h.
As an example we can take I_1:

$$I_1 = I_{g_{Na}} (\frac{I_m}{I_{ref}})^3 \frac{I_h}{I_{ref}}$$

where $I_{g_{Na}} = I_{master}\frac{g_{Na}}{1024}$. We can think of this equation as matching the usual $I_{Na} = g_{Na} m^3 h$. I_{ref}, g_{Na} and I_{master} are all configurable constants with g_{Na} an integer between 0 and 1023.

The gating variables are I_m/I_{ref}, I_n/I_{ref}, I_h/I_{ref}. Making $q = I_q/I_{ref}$ we can write the differential equation

$$\frac{dq}{dt} = \alpha(1-q) - \beta q$$

α and β are summations over sigmoid functions.

$$\alpha = \frac{1}{C_T V_T} \sum_{j=1}^{7} \frac{I_{master}\alpha_j/1024}{1 + \exp\{(sign * k(V_{bj} - V)/V_T\})}$$

where α_j is a constant between 0-1023.

V_{bj} is defined recursively as follows:

$$V_{bn} = V_{b(n-1)} + I_{factor} * 100e3$$

where $V_{b1} = V_{low} + I_{factor} * 50e3$ and $I_{factor} = \frac{V_{high} - V_{Low}}{700e-3}$.

Summary of Configurable Parameters

Parameter	Description	Range
g_x	Conductance	0-1024
α_j	Gating Variables	0-1024
β_j	Gating Variables	0-1024
e_x	Reversal Potential	0-1024
V_{ref}	Constant Reference Voltage	∼1V
R	Resistance	∼1.63e6 Ω
$I_{Voltage}$	current constant	∼270nA
I_{master}	master current	∼200nA
I_{ref}	Reference current	∼100nA

Summary Constants

Constant	Description	Approx Value
C_{mem}	Membrane Capacitance	4pf
C_T	Gating Capacitance	5pf
V_T	Gating voltage	26mV
k	Sigmoid Parameter	∼0.7
V_{high}	Highest Voltage	∼1.1V
V_{low}	Lowest Voltage	∼0.65V

2.5.1 Designing a Current

A HH neuron is a driven dynamical system that tends to a fixed point with no input and a limit cycle for a constant input. The value and design of the input current is therefore important in determining the identifiability of the parameters in the model. If there was a constant stimulus into the neuron it would be impossible to characterize it well because the measurements only cover a limited subsection of the total behaviors of the neuron.

To maximize the information gained from the experiment the neuron should be stimulated across a wide number of its possible behaviors including both subthreshold and spiking operating regions. In other words, we must well sample all feasible states in the phase space of the dynamical system. A high amplitude current was chosen in the experimental procedure, driving the "neuron" throughout its different operating modes and maximizing the number of spikes. A large number of spikes increases the amount of information available on the kinetic parameters that govern the shape and timing of the spikes. Additionally the current was chaotic and thus contained a broad frequency spectrum, enabling it to stimulate the neuron across multiple time scales.

In addition to the amplitude of the injected current, the frequency spectrum is also of the utmost importance. The membrane of a neuron acts as a low pass filter with a cutoff time constant—using standard values of $C = 1\mu F/cm^2$ and $R = 10,000\Omega cm^2$—on the order $10ms$. Thus any signal over $100Hz$ is filtered out by the cell membrane. Accordingly, the power spectrum of the input should be mostly below $100Hz$.

2.5.2 NeuroDyn Results

We estimate the parameters for a NeuroDyn chip using the nudging method and measuring V, m, h and n—this builds on the results of [75] which has similar results using a different DA procedure. In a biophysical neuron only the voltage can be measured,

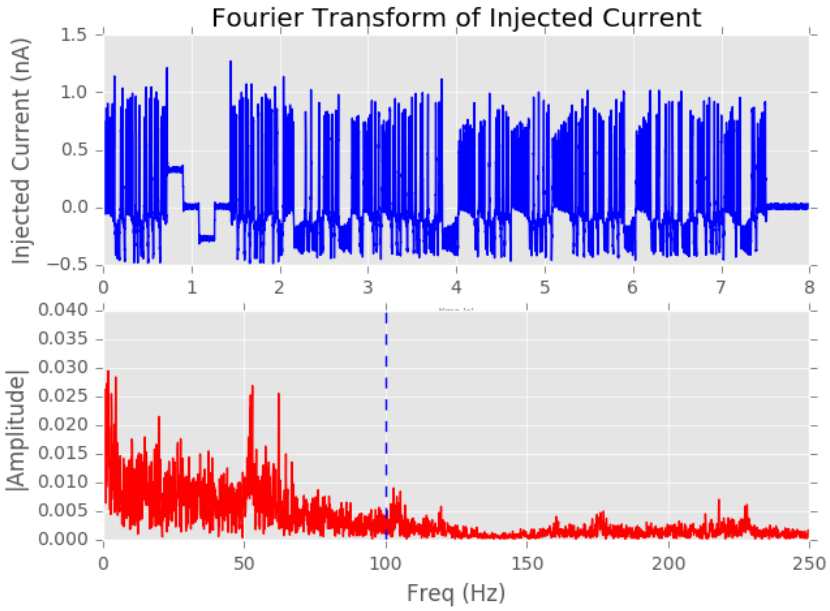

Figure 2.3. This is a sample chaotic current for parameter estimation in neural systems along with its fourier transform; the low pass cutoff frequency for a neuron is shown at 100 Hz. It is important to choose a current that excites the neuron throughout its entire dynamical range and contains the maximum amount of information. Most of the power of the frequency spectrum should thus be below 100 Hz.

but on the chip all four state variables are physical currents and can thus be measured. The parameters were fit to the model during the observation window and the model was validated during the prediction period. As shown in Fig.(2.4) the parameters estimated give almost perfect predictions, thus validating the model. The results show that DA can be a powerful tool to identify the parameter offsets due to manufacturing defects in NeuroDyn thus allowing the correct programming for a particular behavior in these neuromorphic systems.

To summarize the whole procedure, we

1. Design a NeuroDyn VLSI circuit with variable parameters to emulate a physical neuron

2. Send the designs to be manufactured

3. Run the DA procedure on the manufactured chip to estimate parameter offsets due

to manufacturing defects

4. Use the recovered error in the parameters to recover the desired behavior.

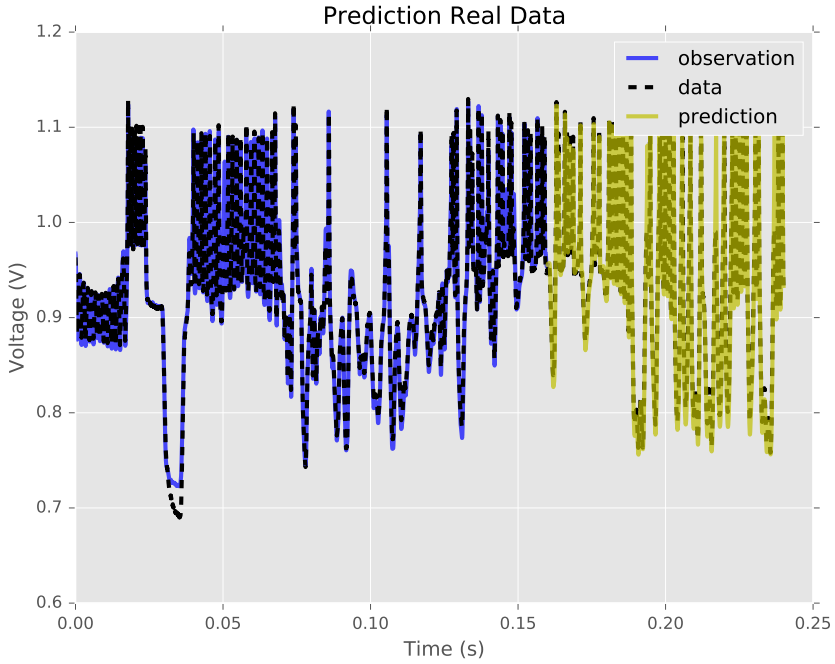

Figure 2.4. Predictions for NeuroDyn chip with conductances, reversal potentials and kinetic parameters estimated from V,m ,h, n. We see that the parameters estimated give near perfect predictions. Data collected by Jun Wang in the laboratory of Gert Cauwenberghs at UCSD [75]

2.6 Data Assimilation with Neuron Data

Now we pivot to using DA on measurements from biological rather than neuromorphic neurons. The neurons in question are physically taken from the high vocal center (HVC) song nucleus of the zebra finch [76, 77]. HVC is a central player in song production, participating in generating the timing that controls song output [78]. Study of HVC affords numerous advantages. Firstly, the ionic currents of all the major HVC cells types are described, permitting HH descriptions [79, 80]. These single cell models can be explored in terms of networks, giving considerable insight into patterns of connections between

cells. Furthermore, the output of HVC and its functional role in generating song timing is intensely studied [81], with the complexity of the network operation constrained by the stereotypy of song. Analysis of HVC presents an outstanding opportunity to extrapolate from the behavior of individual neurons to the emergent property of networks through the study of sequential pattern generation, a central problem in motor systems research. A further technical advantage is that HVC sits close to the dorsal surface of the brain, greatly facilitating two photon miscroscopy [82] for imaging Ca fluorescence signals arising from the activity of neurons in active networks.

For the experiment, the activity of basal ganglia Area X projecting (HVCX) neurons, projecting to the basal ganglia component of the zebra finch song system, are measured through patch clamp recordings. The data is shown in Fig.(2.5) and was recorded by Dan Meliza in the laboratory of Dan Margoliash at the University of Chicago.

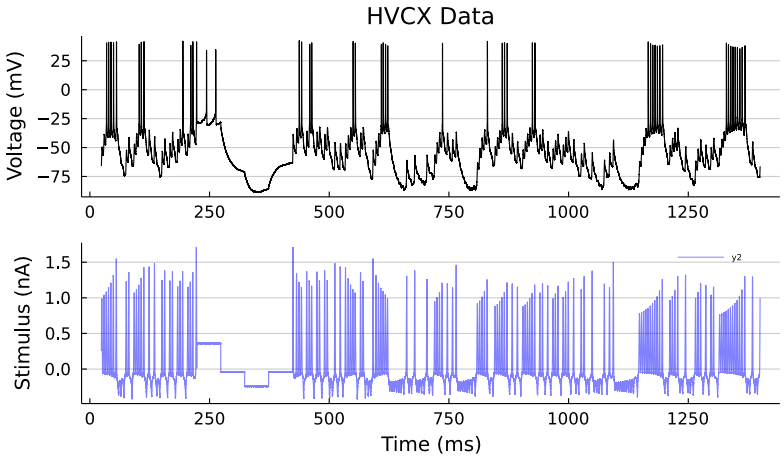

Figure 2.5. Voltage trace data and stimulating current from patch clamp recording on a HVC X-projecting neuron.

We would like to fit a model to the data using data assimilation. The model details are given in the next section. The data assimilation action differs from the nudging routine used for the VLSI problem in that we don't use the minimization suggested by the Euler-Lagrange equations but instead minimize the action directly. Starting from Eq.(2.3) we make a number of assumptions:

1. The prior is uniform

2. The model is perfect

3. The fit of the model to the data is determined not just by the mean square error (MSE) but also the spike timings, the spike shape and the fit of the subthreshold behavior.

Expanding on the last point, small variations in parameters can have a big impact on the exact timing of spikes. A mistimed spike can lead to a dramatic increase in MSE, but the model may be very close from a biological perspective. Therefore instead of the usual least squares action with no prior and a perfect model $A_0(X|U) = \frac{1}{2}\sum_{t=0}^{n}(\mathbf{u}(t) - H[\mathbf{x}(t)])^T\Sigma_H^{-1}(\mathbf{u}(t) - H[\mathbf{x}(t)])$, we use the normalized root mean square error

$$\mathrm{NRMSE}(\mathbf{x}) = \frac{1}{\max \mathrm{y} - \min \mathrm{y}}\sqrt{\frac{1}{\mathrm{n} \times \mathrm{D}}\sum_{\mathrm{t}=0}^{\mathrm{n}} \|\mathbf{u}(\mathrm{t}) - \mathrm{H}[\mathbf{x}(\mathrm{t})]\|^2} \tag{2.11}$$

combined with a measure of the accuracy of spike timings called the bivariate spike distance (BSD)—invented by Kreuze et.al.[83, 84].

BSD gives a value between 0 (identical spike trains) and 1 (completely dissimilar spike trains). It is calculated by identifying each instant in time t with the previous and next spikes of each of the two spike trains. Using these these spike timings they define a spike dissimilarity metric $S(t)$ which can be integrated to find BSD [85]

$$BSD = \frac{1}{T}\int_0^T S(t)dt.$$

Therefore our action is

$$A_0(\mathbf{X}) = (1 - \epsilon)\ \mathrm{NRMSE}(\mathbf{X}) + \epsilon\ \mathrm{BSD}(\mathbf{X}) \tag{2.12}$$

which is mediated by the variable ϵ which allows us to give more or less weight to the spike

times vs the subthreshold activity and the spike shape. The model is enforced directly by applying the equations of motion and therefore the action is only a function of the initial condition X_0 and the parameters of the model.

2.6.1 Model

The model consists of a Hodgkin-Huxley like model with 9 ion channels/74 parameters and is described in detail in the paper by Nogaret et.al. [86].

Table 2.1. The ion channels of the HH like model of a HVC X projecting neuron, reproduced from [86]

ID	Channel	Current Density
NaT	Fast and transient Na^+ current	$I_{NaT} = g_{NaT}m^3h(E_{Na} - V)$
NaP	Persistent Na^+ current	$I_{NaP} = g_{NaP}m(E_{Na} - V)$
K1	Transient depolarization activated K^+ current	$I_{K1} = g_{K1}m^4(E_k - V)$
K2	Rapidly inactivating K^+ current (A current)	$I_{K2} = g_{K2}m^4h(E_K - V)$
K3	Ca^{2+} activated K^+ current	$I_{K3} = g_{K3}m(E_k - V)$
CaL	High threshold Ca^{2+} current	$I_{CaL} = \rho m^2 I_{Ca}$
CaT	Low threshold Ca^{2+} current	$I_{CaT} = m^2 h I_{Ca}$
HCN	Hyperpolarization-activated cation current	$I_{HCN} = g_{HCN}h(E_{HCN} - V)$
Leak	Leakage channels	$I_L = g_L(E_L - V)$

2.6.2 Results and Discussion

The action Eq.(2.12) is nonlinear and thus requires a global optimization routine to solve for the minima. The routine used in this study is the covariance matrix adaption evolutionary strategy (CMAES) [87].

CMAES is a stochastic method that attempts to minimize a nonlinear, high dimensional function with respect to an input vector. It is part of the evolutionary class of algorithms that have achieved remarkable success in gradient free global optimization. Other examples of this class of algorithm include differential evolution [88]. CMAES is known to work well for highly nonlinear problems up to $\sim 100 - 200$ parameters and is thus well suited for use in data assimilation where we are minimizing the action over a set

of parameters.

The results of the estimation procedure are shown in Fig.(2.6). We see that both the subthreshold behavior and the spike timings match up well both in the estimation and the prediction window. The goal was to reproduce the results in [86] but with the new optimization technique rather than nudging which was used in that study. The advantages of the new technique include the speed and scaling of the procedure, as well as the ability to weight the spike timings vs spike shape. Having proved that the technique works well with voltage data, we now extend this study to the case where we have Ca fluorescence data.

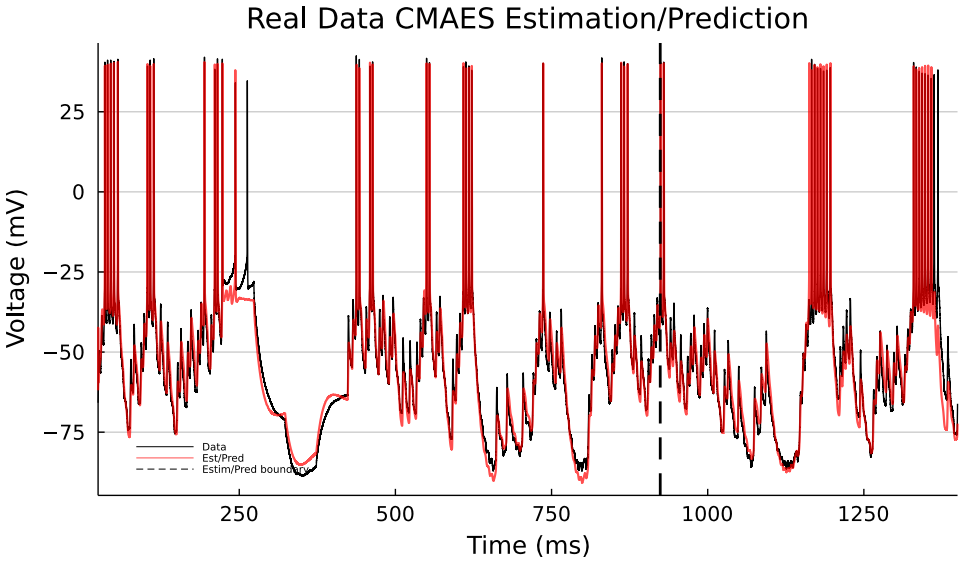

Figure 2.6. Result from the DA procedure. $\epsilon = 0.5$ in this case which evenly balances the spike timing cost with the normalize root mean square error which tends to emphasize the subthreshold activity. The red curve is the estimate while the black curve is the measured data. The vertical dashed line seperates the estimation from the prediction window.

2.7 Twin Experiments with Ca Fluorescence Data

We describe a twin experiment using Ca Fluorescence data with a known I_{ext} to estimate voltage as well as the maximal conductances. A twin experiment is one in which artificial data with added noise is generated from a computational model, then we see if

parameters in the model can be inferred from the generated data. This kind of experiment gives a proof of concept that the experiment is possible and gives us a simulation platform to study experimental protocol.

The main reason we would want to be able to use Ca data in a DA context is that the development of two photon microscopy [82, 89–93] has made it possible to image the activity of networks of neurons near the surface of the brain in vivo. This gives it a distinct advantage in the study of network connectivity and function when compared to electrophysiology. While previous studies have focused on spike timing inference [94], here we look at the possibility of fitting a HH like model to Ca data given a known stimulus.

2.7.1 Neuron Model

The neuron model is a slightly modified version of the model in Ye et. al [95]. There are 5 state variables: V, h, n, Ca, r_T. The voltage equation

$$\dot{V}(t) = -\frac{1}{C_m}(I_{Na} + I_K + I_{CaL} + I_{CaT} + I_{SK} + I_L - I_{\text{ext}}(t))$$

contains 6 currents. Na, K and L are sodium, potassium and leakalong with T and L type Ca currents and a K/Ca current denoted SK.

The sodium current is assumed to be fast and therefore does not include voltage gated ion channels

$$I_{Na} = g_{Na} m_\infty^3 h(V - E_{Na}).$$

I_K/I_L are in the standard HH form

$$I_K = g_K n^4 (V - E_K)$$
$$I_L = g_L (V - E_L).$$

The Ca^{2+} currents are separated in L type and T type. The Ca concentration is

governed by

$$\frac{\mathrm{d}[Ca]}{\mathrm{d}t} = -\epsilon(I_{CaL} + I_{CaT}) - k_{ca}([Ca] - 0.1))$$

which is simply mass conservation of Ca. The L type Ca current is

$$I_{CaL} = g_{CaL} s_\infty^2 (V - E_{Ca})$$

where $s_\infty = \sigma(V, \theta_S, \sigma_S)$ with $\sigma(x, y, z) = \frac{1}{2}(1 + \tanh \frac{y-x}{2z})$. T type Ca is

$$I_{CaT} = g_{CaT} aT_\infty(V, \theta_{aT}, \sigma_{aT})^3 bT_\infty(rT, \theta_{bT}, \sigma_{bT})^3 (V - E_{Ca})$$

with $aT_\infty = \sigma(V, \theta_{aT}, \sigma_{aT})$ and $bT_\infty = \sigma(rT, \theta_{bT}, \sigma_{bT}) - \sigma(0, \theta_{bT}, \sigma_{bT})$. The K/Ca current depends on the Ca concentration through the Hill equation

$$I_{SK} = gSK \frac{Ca^4}{Ca^4 + ks^4}(V - E_K).$$

The stimulating current is taken from the x variable of the Lorenz 63 equations.

2.7.2 Fluorescence Data

The fluorescence signal is recorded as $F(t)/F_0$ with F_0 a constant. Vogelstein et.al.[94] suggests that this can be related to the change in intracellular calcium as

$$\frac{F(t)}{F_0} = \frac{Ca^N}{Ca^N + K_D^N} + \text{noise}.$$

Solving for Ca we find

$$Ca(t) = K_D \left(\frac{F(t)}{F_0 - F(t)} \right)^{1/N}.$$

This model in experimental data will have to be calibrated to find K_D and N. Here we set them to $K_D = 1$, $N = 1$.

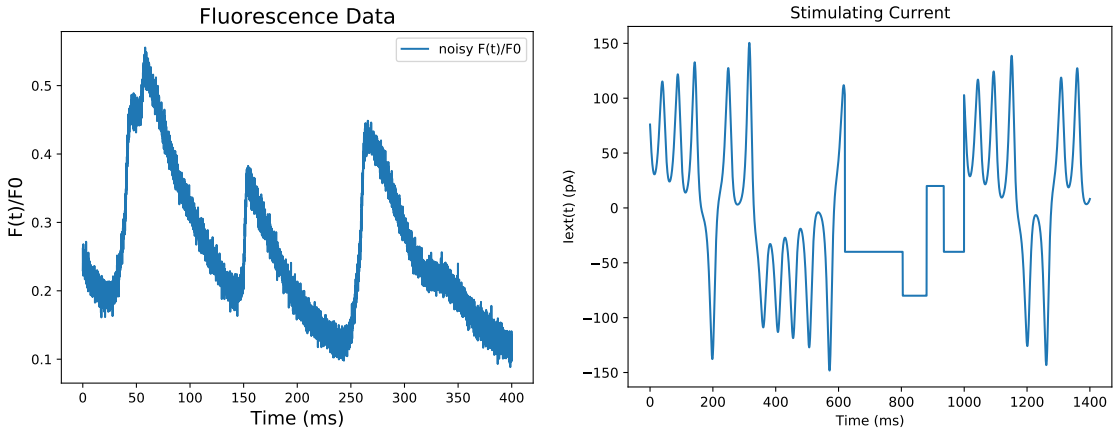

Figure 2.7. Fluorescence data and stimulating current for the twin experiment. These are the two measurements we have from which to estimate the model.

2.7.3 Twin Experiment

We present the CMAES method with the fluorescence data, the injected current Fig.(2.7) and the model. We estimate $V(t)$, $h(t)$, $n(t)$, $Ca(t)$, $r_T(t)$ as well as g_{Na}, g_K, g_{SK}, g_{CaT}, g_L and g_{CaL}. The bounds, truth and estimated parameters are shown in Table.2.7.3 and the initial guess was selected randomly between the lower and upper bounds of the search. Results are shown in the Figure 2.8.

The results show that given a correct model with missing parameters as well as noisy fluorescence data one can reconstruct the state variables and the missing parameters. With experimental data, of course, there are many additional complications. For a start, two photon microscopy can tell the activity of the neuron but the input signal is unknown in vivo. Additionally the data may have a higher signal to noise ratio or there may be other factors influencing the fluorescence. Finally, the model may be incorrect and this will need to be accounted for in the procedure. Thus, this is a preliminary study that will need some development before being fully ready to deploy experimentally.

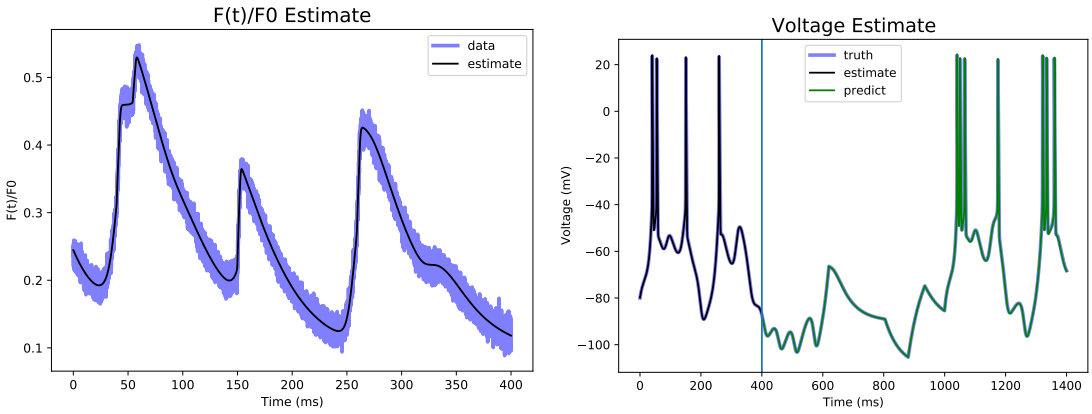

Figure 2.8. Estimate of the Fluorescence data through the noise as well as the voltage estimate and prediction. Since only $F(t)/F_0$ was measured, the voltage estimate and prediction has to be inferred along with h, n and r_T. Given that the model is correct it is not entirely surprising that we were able to recover the correct state. In true experiments the noise may not be gaussian and the model may be incorrect.

name	low	high	true	estim
g_{Na}	0	2500	450	463.4
g_K	0	100	50	51.1
g_{SK}	0	10	2	2.01
g_{CaT}	0	10	3	2.98
g_{CaL}	0	20	10	10.09
g_L	0	10	2	1.9995

2.8 Summary of Chapter 2

Fitting physical models of neurons to data is an extremely challenging task due to their high degree of dynamical nonlinearity, as well as the large number of possible parameters to specify in any computation. A gradient descent based curve fitting approach would fail due to the innumerable local minima in the cost function Eq.(2.3). In this chapter we have developed an optimization formulation, based on Bayesian inference and synchronization, that allows us to fit models of individual neurons to data.

The examples chosen to illustrate the power of this technique include

1. the estimation of parameter errors due to manufacturing in neuromorphic VLSI circuits through the measurement of all state variables in response to a driving current

2. the estimation of all state variables and parameters in a complex neuron model from voltage data taken from a zebra finch HVCX projecting neuron

3. simulation study based on Ca data.

The ability to fit an arbitrarily complex neuron model to data allows the asking of a number of interesting questions. Firstly, it constrains the physical characteristic of the neuron, telling us the number and kind of ion channels, the voltage sensitivity of the membrane proteins, the capacitance of the cell membrane and other quantities. One can group and identify types of neurons within a cluster based on their function and physical qualities. Furthermore, one can compare neurons within species to see whether the same type of neurons have similar properties and what can cause these properties to change. In Miller et.al. [1] the neuron ion channel conductance of zebra finch siblings was compared, finding that the physical characteristics of sibling's neurons are very similar compared to birds not in the family.

The next steps will be to connect the microscopic details of individual neurons and types of neurons to the aggregate activity of neuron networks. While there has been much study on human pyschology and behavior, as well as on individual and small networks of neurons, it has been extraordinarily difficult to connect the two. Complex systems produce intricate behavior that can not easily be extrapolated from the interaction laws of its various components. Therefore these techniques must be made scalable both experimentally as well as computationally. The study of calcium is an important first step in this regard, as it extends the use of DA to an experimental technique that can

measure hundreds of neurons rather than the handful one has access to using detailed voltage recordings.

In the rest of this thesis we move from using Bayesian inference to estimate parameters in a physical model to replacing that physical model with a recurrent neural network (RNN). This has some advantages in that the RNN learns a surrogate model from the data alone, thus removing the issue of introducing errors into the model through incorrect parameterization. Of course the advantages of the approach come with drawbacks including reduced interpretability and the inability to estimate unmeasured state variables.

Chapter 2, in part, is adapted from the material as it appears in Anna Miller, Dawei Li, Jason Platt, Arij Daou, Daniel Margoliash, and Henry D. I. Abarbanel. "Statistical Data Assimilation: Formulation and Examples From Neurobiology". In: Frontiers in Applied Mathematics and Statistics *4 (2018). ISSN: 2297-4687. DOI: 10.3389/fams.2018.00053. Anna Miller was the primary inverstigator and author of the work, the dissertation author is a coauthor on the paper.*

Chapter 3

Forecasting Chaos through Recurrent Neural Networks

This chapter is adapted from [2] previously published by Chaos.

3.1 Introduction

Machine Learning (ML) is a computing paradigm for data-driven prediction in which an algorithm is presented with input data in a network training phase, and then asked to predict the future behavior of new data in a generalization/forecast phase. No detailed physical model is needed for this procedure.

RNNs are a family of artificial neural networks where the internal state of the network depends explicitly both on the previous internal or 'hidden' state and an external driving signal. Thus, there is a natural ordering to the data—we call the order label time—which explicitly allows treatment of an RNN as a dynamical system. In contrast, other forms of artificial neural networks impose no ordering on the data and thus their natural mathematical description is as a function [96].

An RNN takes a D-dimensional input sequence $\mathbf{u}(t) \in \mathbb{R}^D$, evolves as the N-dimensional state $\mathbf{r}(t) \in \mathbb{R}^N$, and is asked to predict a value $\mathbf{y}(t) \in \mathbb{R}^Y$. Equation 10.5 in *Deep Learning* by Goodfellow, et al. [97], gives the general discrete time RNN evolution

equation

$$\mathbf{r}(t) = F(\mathbf{r}(t-1), \mathbf{u}(t-1); \theta) \tag{3.1}$$

where θ represents the RNN hyperparameters. This shows the dependence of $\mathbf{r}(t)$ on the previous RNN state $\mathbf{r}(t-1)$ and external driving signal $\mathbf{u}(t-1)$. The output/readout layer of the network then 'reads' $\mathbf{r}(t)$ as containing the memory of previous inputs, through the recursive relation, to predict $\mathbf{y}(t)$. In other words, the output of the network $\hat{\mathbf{u}}(t) \equiv \mathbf{W}_{\text{out}}(\mathbf{r}(t))$ is a function $\mathbf{W}_{\text{out}}$ of the RNN hidden state $\mathbf{r}(t)$. The operator $\mathbf{W}_{\text{out}}$ defined here is a generic function, and could even be implemented as a neural network. Later we will take $\mathbf{W}_{\text{out}}$ to be a linear operator (a matrix). We can thus use $\hat{\mathbf{u}}(t) \sim \mathbf{y}(t) = \mathbf{u}(t)$ as a forecast.

Long short-term memory (LSTM) networks, introduced by Hochreiter and Schmidhuber (1997) [98] and Gers et al. (1999) [99] have been one of the most useful varieties of RNN, as they are capable of maintaining a representation of long-term dependencies. This design helped LSTMs to circumvent a number of difficulties that plagued RNNs regarding exploding and vanishing gradients during optimization using backpropagation. A simplified form of the LSTM was later developed by Kyunghyun Cho et al. in 2014 [100] called the Gated Recurrent Unit (GRU), which has a reduced set of trainable model parameters.

A class of RNN architectures with demonstrated capability for forecasting dynamical systems is reservoir computing (RC) [101–110]. In RC, a large fixed random network is selected and only the final output layer is trained—typically through linear regression. The network is straightforward to train because the architecture and weights in the reservoir layer are fixed during training and operation, bypassing the vanishing/exploding gradients that may appear to trouble other RNN models [111–113].

The training input signal to the network may be generated from a known dynamical system [114, 115], or it may result from observations where the underlying dynamical rules

are unknown. RC's ease of training and demonstrated prediction capabilities make them a serious contender for time series forecasting tasks [116].

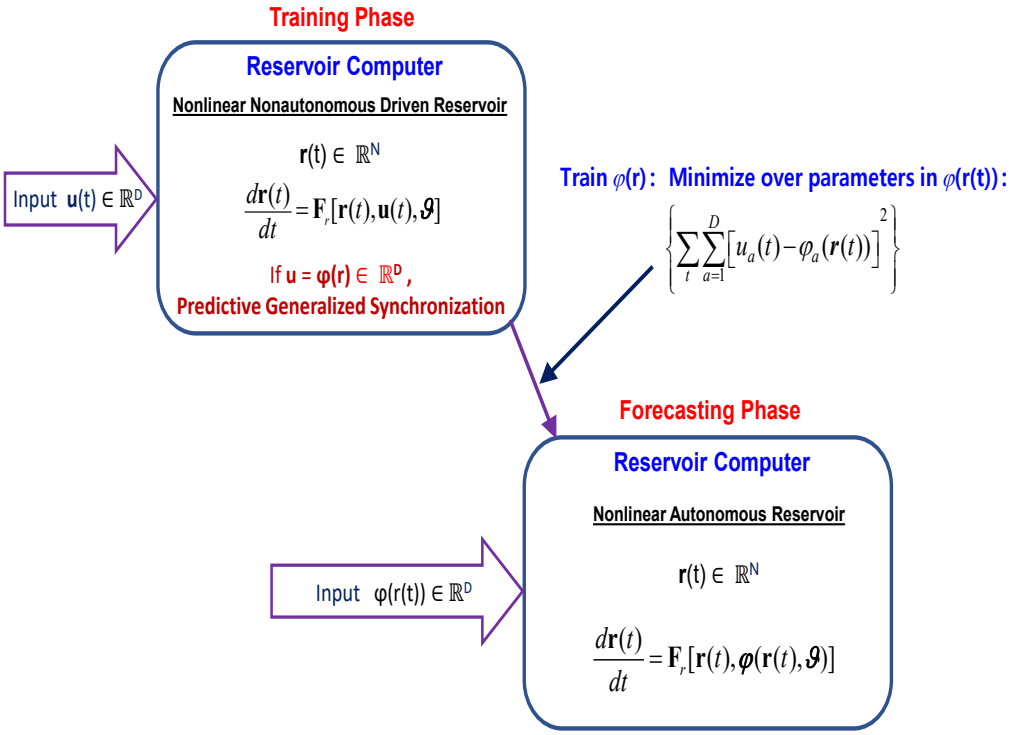

Figure 3.1. Flow of operations for implementing a Reservoir Computation (RC) strategy to perform forecasting/prediction of an input $\mathbf{u}(t) \in \mathbb{R}^D$ presented to a RC with dynamical degrees-of-freedom $\mathbf{r}(t) \in \mathbb{R}^N$. The RC dynamics are given as $\dot{\mathbf{r}}(t) = \mathbf{F}_r(\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta})$; $\boldsymbol{\theta}$ are fixed parameters in the RC. When the input and the reservoir exhibit *predictive generalized synchronization* (PGS), $u_a = \varphi_a(\mathbf{r}); a = 1, 2, ...D$; training consists of estimating any parameters in a representation of $\varphi(\mathbf{r})$. After the regions of PGS are established for a given $\mathbf{u}(t)$ and a selected $\mathbf{F}_r(\mathbf{r}, \mathbf{u}, \boldsymbol{\theta})$, one may wish to change the values of $\boldsymbol{\theta}$ within the PGS region to optimize the predictive performance of the RC.

The ability to develop a data-driven model using a method such as RC is attractive for a number of practical reasons. RC allows us to construct predictive models of unknown or poorly understood observed dynamics. Should the input signal $\mathbf{u}(t)$ arise from measurements of high dimensional geophysical or laboratory flows [117, 118], for example, the striking speedup in computing with a reservoir network realized in hardware [119, 120] may permit the exploration of detailed statistical questions about the observations that

might be difficult or impossible otherwise. RC has the potential to provide significant computational cost savings in prediction applications, since the RC dynamics typically comprise a network with computationally simple active dynamics at its nodes.

3.1.1 Background

RC [101, 105] is a general term that encompasses multiple research tracks that developed in different fields in the early 2000's, in particular Echo State Networks (ESNs) [104, 106, 121] and Liquid State Machines (LSMs) [108]. It is a kind of RNN with a structure consisting of a fixed high-dimensional "reservoir" combined with a single trained output layer. The fixed nature of the reservoir reduces the size of the trained parameter space to a handful of 'macro-scale' parameters governing global properties, as well as 'micro-scale' parameters that comprise the matrix and vector elements. A key advantage of RC is that the final output or 'readout' layer can be formulated as a linear layer and then trained using linear regression. This reduction in the size of the searchable parameter space allows one to easily bypass the exploding/vanishing gradient problems that arise due to the use of backpropagation schemes [111–113].

In 2007, Schrauwen, Verstraeten, and Campenhout [107] drew the connection between methods proposed by [104] and [108] and the state-of-the-art learning rule for RNNs described by [122]. For example, the recurrent learning rule of Atiya and Parlos (2000) [123] trains the output weights while the internal weights are only globally scaled up or down.

In addition to being easy to use and train, RCs have been shown to be very successful for chaotic time series prediction [110, 116, 124, 125]. For numerical weather prediction (NWP) specifically, a major use case of multivariate chaotic time series prediction, Arcomano et al. (2020) [126] applied RC to a global atmospheric forecast model. Penny et al. (2021) [4] integrated RC with state-of-the-art data assimilation methods to estimate the forecast error covariance matrix, tangent linear model representation, and other

components of the data assimilation process that are typically costly to either formulate or compute. Further computational advantage are afforded by implementing RCs on GPUs or even dedicated hardware [120].

We note that other ML methods have been explored for modeling dynamical systems, with varying degrees of success. While we cannot name them all, we highlight a few examples. Bocquet et al. [127, 128] used NNs and Convolutional NNs [97] to learn ODEs that could then be integrated using traditional numerical methods. As with RC, they have similarly been able to reconstruct the Lyapunov spectrum of simple systems. Convolutional LSTMs have been used in MetNet [129] to predict the evolution of precipitation. Eric Bollt [130] and Gauthier et al. [131] introduced a vector autoregression scheme that has shown remarkable success in the prediction of small systems; see, for example, Clark et al. [132] for an application in neurobiology.

3.1.2 Discerning how RC Works

The success of RNNs and their increased adoption in research applications has perhaps out paced the understanding of how these data driven processes are successful. It is not known how best to design a network for a particular problem, nor how much or what kind of data is most useful for training. General guidelines are established and widely discussed [103, 105, 107, 133]. These tend to be justified with empirical rather than theoretical considerations. In probing this question, the very interesting idea arose [114, 134] that the explanation might be a form of synchronization known as 'generalized synchronization' (GS) [135–137].

Unidirectional driven systems such as these have long been studied, especially in the analysis of nonlinear dynamical systems [34, 138]. There exists a kind of synchronization between systems with different numbers of state variables, such as the D-dimensional source of the $\mathbf{u}(t)$ and the N-dimensional reservoir $\mathbf{r}(t)$, and this is called generalized synchronization (GS) [135, 136]; this is in contrast with the synchronization introduced by

Pecora and Carroll [139], which analyzes how two identical systems may synchronize. GS is a statement that there is a relation $\mathbf{r}(t) = \psi(\mathbf{u}(t))$ meaning the $\mathbf{r}(t)$ dynamics tracks the driving dynamics $\mathbf{u}(t)$ even though they can be quite different dynamical systems.

GS was established by experiments as early as 1998 [140] and has a role in areas as diverse as synchronization of neurons in a brain circuit to cryptographic code breaking[141]. In the synchronization of dynamical systems in the presence of noise, the two systems being synchronized are never precisely the same so GS is essential in explaining how synchronization works in practice.

In the framework of reservoir computing—to achieve the goal of the reservoir learning the dynamical information in the signal—one uses the idea that the signal $\mathbf{u} = \mathbf{W}_{out}\mathbf{r}$ turns the driven system Eq. (3.1) into an autonomous system. The reservoir may be analyzed as an initial value problem while yielding a forecasting model for the learning $\mathbf{u}(t)$. In the language of GS this suggests that the *inverse* of the GS relationship $\mathbf{r}(t) = \psi(\mathbf{u}(t))$, namely $\mathbf{u}(t) = \varphi(\mathbf{r}(t))$ would generalize the linear $\mathbf{u} \to \mathbf{r}$ feedback relation $\mathbf{u} = \mathbf{W}_{out}\mathbf{r}$.

The existence of the inverse GS relationship is associated with the requirements of the inverse function theorem and requires the functions $\psi(\mathbf{u})$ and $\varphi(\mathbf{r})$ be differentiable [142]. This is a local requirement for the inverse of $\psi(\mathbf{u})$ to exist.

Dynamical systems often have many basins of attraction in $\mathbf{r}$ space, and the basin one is in depends on the choice of $\mathbf{r}(0)$ in the initial value solution of the reservoir equation. The boundaries between basins of attraction are known to be fractal in many examples [143], and maintaining differentiability across such boundaries would appear to be unlikely. The idea that $\psi(\mathbf{u})$ is invertible, expressed as invertible GS [144], cannot be a global property of all reservoirs. The inverse in one basin of attraction may be quite different in another.

We are interested in investigating the role of GS in the formulation of the attractive idea of RC, since a relation $\mathbf{u} = \varphi(\mathbf{r})$ would turn the reservoir into an autonomous system and achieve the training needed to make the reservoir an excellent forecasting machine.

GS can be identified for a given driving or teaching signal $\mathbf{u}(t)$ in a most straightforward manner using the auxiliary system approach [136] which for choices of parameters of the reservoir selection identifies when $r(t) = \psi(\mathbf{u}(t))$ holds. The condition is succinctly stated as requiring that the Lyapunov exponents of the reservoir, conditioned on the teaching signal, be negative [139]. This feature of GS appears equivalent to the fading memory feature of useful reservoirs.

In reservoir computing a GS relation is not immediately useful for moving the driven reservoir into an autonomous dynamical system for $\mathbf{r}(t)$. We introduce the term predictive generalized synchronization (PGS) to remind us that invertibility of the GS relation is not globally true in general, and to give us a reservoir parameter region where it could hold. In such a region we may assume that $\mathbf{u} = \varphi(\mathbf{r})$ happens and proceed with the framework of reservoir computing with confidence.

3.1.3 Goals

The goal of this paper is to provide a useful path for finding reservoir parameters where good generalization/forecasting is possible.

The training of an RC, as indicated in Fig.(3.1), consists of minimizing

$$\sum_t \|\mathbf{u}(t) - \varphi(\mathbf{r}(t))\|^2, \tag{3.2}$$

with respect to parameters that enter in the representation of the vector valued function $\varphi(\mathbf{r})$ of the reservoir variables $\mathbf{r}(t)$. This training of the PGS function $\varphi(\mathbf{r})$ to match the data stream $\mathbf{u}(t)$ is the essential step of transferring the information content in the data $\mathbf{u}(t)$ to the reservoir dynamics.

We pursue the interesting suggestion given in [114, 134] that GS is instrumental in RC, using, however, PGS rather than GS, to move from a somewhat *ad hoc* training approach to a systematic strategy. In training, we ensure that the input $\mathbf{u}(t)$ and the

reservoir degrees-of freedom $\mathbf{r}(t)$ satisfy PGS, $\mathbf{u}(t) = \varphi(\mathbf{r}(t))$, and point out that it is parameters in the representation of the function $\varphi(\mathbf{r})$ that we need to estimate.

We demonstrate a computationally efficient way to choose regions of RC hyperparameters where PGS occurs as well as regions where PGS does **not** occur. This is then employed in guiding hyperparameter choices for skillful forecasting of the input training data $\mathbf{u}(t)$, given an accurate enough approximation to $\varphi(\mathbf{r})$. These hyperparameters may include some properties of the $N \times N$ adjacency matrix $\mathbf{A}$ such as the spectral radius and the density of connections among the $\mathbf{r} \in \mathbb{R}^N$ active units ρ_A. These quantities are collected together in Table 1 as they appear in an RC with tanh dynamics at its nodes.

A related issue is that the traditional method of evaluating the effectiveness of an RNN, with training and testing data sets, is awkward when performing dynamical systems forecasting. This method gives no indication of the stability of the forecast. Another approach to evaluation, showing a prediction of a single time series, also gives no indication of the stability of the predictions over the entire range of inputs. In this paper we attempt to rectify these deficiencies by using dynamical properties of the reservoir to design and evaluate a trained network.

3.1.4 Presentation Strategy

1. We introduce a computationally efficient numerical test, based on PGS, and using the 'auxiliary method', to guide hyperparameter selections in RCs resulting in very good forecasting.

2. We portray the ideas for the use of PGS with some simple illustrative models [145–147], then discuss an important geophysical model, the Shallow Water Equations (SWE) [148–150], and finish with a discussion of a biophysical model of neuron dynamics [151, 152]. The last item comprises data from a driven dynamical system

(the neuron), and these data depend on an injection of current to stimulate the neuron into interesting oscillations. The RC must obtain information about the driving force as it is trained.

3. We explore a metric for a "well trained" RC network using the reproduction of the input system's Lyapunov exponent spectrum. We introduce a criterion for excellent forecasting connected to the conditional Lyapunov exponents [139] of the reservoir, and

4. We briefly address, and speculate about, the long-term value of RC for problems encountered in physical systems.

3.2 PGS in Reservoir Computing

RCs are often applied to forecasting problems where a familiar task is to learn from an input sequence $\mathbf{u}(t) \in \mathbb{R}^D$ generated from observed data produced by a likely unknown autonomous dynamical system

$$\frac{d\mathbf{u}(t)}{dt} = \mathbf{F}_u(\mathbf{u}(t)), \tag{3.3}$$

and then forecast the future of $\mathbf{u}(t)$.

In this paper we will utilize data from three autonomous dynamical systems:

1. The Lorenz 1963 three dimensional model [145]. This a familiar test bed for ML forecasting of chaotic dynamics.

2. The Lorenz 1996 [146] D-dimensional model. Often used in geophysics as a platform for examining ideas in a context in which the dimension D of the produced data can be easily made large.

3. The shallow water equations (SWE). These describe the flow of a thin layer of fluid (say the ocean/atmosphere system which has a depth approximately 10-15 km) on

a sphere (the earth with a radius of about 6400 km). The SWE are a set of three partial differential equations in three state variables $\{u(x,y,t), v(x,y,t), h(x,y,t)\}$ on a mid-latitude tangent plane to the earth.

3.3 RC Definition

In the context of chaotic time series prediction we apply RC to the temporal ML problem where the task is to predict a time series $\mathbf{u}(t)$ generated from an autonomous dynamical system

$$\dot{\mathbf{u}}(t) = f_u(\mathbf{u}(t)), \tag{3.4}$$

where the dot denotes a time derivative and f_u denotes the equations of the dynamical system. The dimension of the input system is D. In general f_u is not known, nor does it need to be known for RC to be used. We do assume that f_u exists and that time dependent data can therefore be generated deterministically. The system f_u can describe any physical process and therefore these techniques can be applied in fields such as biology, hydrology, meteorology, oceanography, economics, chemistry, and many others.

Here we use an RC model with the specific form of a simplified "Elman" style[153] RNN. The RC consists of three layers: an input layer $\mathbf{W}_{\text{in}}$, the reservoir itself, and an output layer $\mathbf{W}_{\text{out}}$. The reservoir is composed of N nodes that are generally acted upon with simple nonlinear elements *e.g.*, tanh activation functions. The nodes in the network are connected through an $N \times N$ adjacency matrix $\mathbf{A}$, chosen randomly to have a connection density ρ_A and non-zero elements uniformly chosen between $[-1, 1]$, scaled such that the maximal eigenvalue of $\mathbf{A}$ is a number denoted the spectral radius (ρ_{SR}).

The input layer $\mathbf{W}_{\text{in}}$ is an $N \times D$ dimensional matrix that maps the input signal $\mathbf{u}(t)$ from D dimensions into the N dimensional reservoir space. The elements of $\mathbf{W}_{\text{in}}$ are chosen uniformly between $[-\sigma, \sigma]$. The matrix $\mathbf{W}_{\text{out}}$ provides the mapping from the RC reservoir state to the system state as $\hat{\mathbf{u}}(t) = \mathbf{W}_{\text{out}}\mathbf{r}(t)$. The reservoir state $\mathbf{r}(t)$ can be

viewed as embedding the information given in the time series $\mathbf{u}(t), \mathbf{u}(t-1), \ldots, \mathbf{u}(0)$ in a higher $N > D$ dimensional space, consistent with Takens theorem of time-delay embedding [154]. Takens theorem implies that information from unobserved states of the dynamics is contained in the time delay signal, thus allowing an RC to operate even when not all the information is measured.

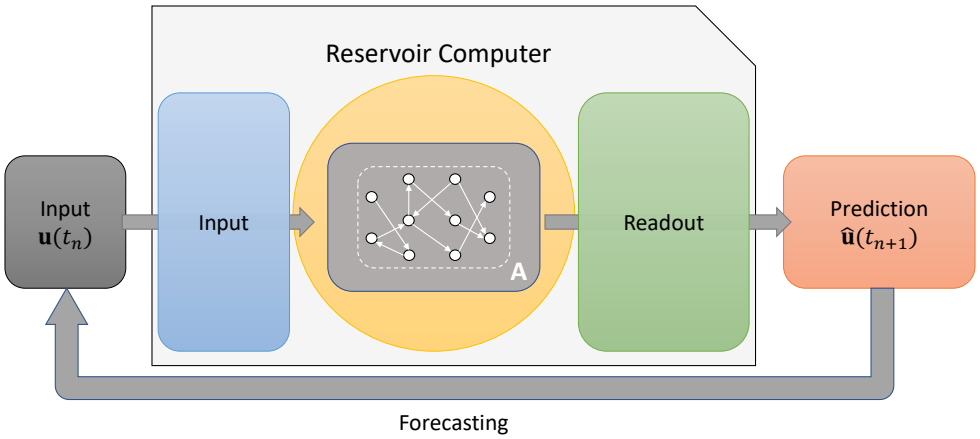

Figure 3.2. Schematic of information flow in the RC. The input data $\mathbf{u}(t_n) \in \mathbb{R}^D$ is mapped through the input layer $\mathbf{W}_{\text{in}} \in \mathbb{R}^{N \times D}$ into the reservoir $\mathbf{r} \in \mathbb{R}^N$ where the nonlinear activation function (e.g. *tanh*) is applied and mixed through a fixed (i.e. not trained) adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$. The readout layer $\mathbf{W}_{\text{out}} \in \mathbb{R}^{D \times N}$ is typically trained using linear regression and gives the one-step-ahead prediction. Longer-term forecasts are achieved by cycling a feedback of the output back to the input.

Training $\mathbf{W}_{\text{out}}$

The output layer is a function such that $\varphi_a(\mathbf{r}) = u_a(t)$, chosen during the **training phase** during which we estimate any parameters in $\varphi(\mathbf{r})$. It is common practice, followed in this paper, to choose $\varphi(\mathbf{r})$ as a linear (or at most quadratic) function of $\mathbf{r}$, but this is by no means the only, or best, representation [155–161].

We take the output layer $\mathbf{W}_{\text{out}} \in \mathbb{R}^{D \times N}$ to be a matrix such that $\mathbf{W}_{\text{out}}\mathbf{r}(t) \equiv \hat{\mathbf{u}}(t) \sim \mathbf{u}(t)$ chosen during the training phase. The elements of $\mathbf{W}_{\text{out}}$ are what we call

"micro-scale" parameters—in contrast to the "macro-scale" parameters (σ, ρ_{SR}, ρ_A, N,...) that set overall properties of the RC by adjusting multiple matrix elements at once. An analogy would be that setting the macro-scale parameters is similar to specifying the overall temperature or pressure of a gas, the positions and velocities of each individual particle need not be taken into account, greatly simplifying the problem. The micro-scale parameters that comprise the elements of $\mathbf{W}_{out}$ can be trained at minimal cost using linear regression.

The reservoir dynamics themselves can either be posed in mapping or differential form. The differential form is

$$\dot{\mathbf{r}}(t) = f_r^d(\mathbf{r}(t), \mathbf{u}(t)) = \gamma\Big[-\mathbf{r}(t) + \tanh(\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{u}(t) + \sigma_b\mathbb{1})\Big] \tag{3.5}$$

with γ the inverse time constant, σ_b the input bias, $\mathbb{1}$ a vector of 1's, and $t \in \mathbb{R}$.

The mapping form is

$$\mathbf{r}(t+1) = F_r^d(\mathbf{r}(t), \mathbf{u}(t)) = \alpha\tanh(\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{u}(t) + \sigma_b\mathbb{1}) + (1-\alpha)\mathbf{r}(t) \tag{3.6}$$

with $t \in \mathbb{Z}$. The notation is as follows; f describes the differential form, while F describes the mapping form. F_r^d denotes the 'driven' RC, as opposed to 'autonomous' (F_r^a), $\mathbf{r}$ is the reservoir state vector, and $\mathbf{u}$ is the state produced by the true dynamics, which is provided as input data. Eqs.(3.5, 3.6) are denoted the "driven RC" because $\mathbf{u}(t)$ enters directly into the reservoir equation of motion.

The mapping form is equivalent to the differential form when the latter is integrated using the Euler method. This can be shown by setting $\alpha = \gamma\Delta t$ and discretizing Eq.(3.5) with the Euler discretization scheme. While both the mapping and differential forms can be used interchangeably, this detail must be acknowledged as the integration method may affect the RC accuracy. Alternative forms of the RC have been suggested—see [119] for a

review—but so far we have not found any advantages in using those forms.

We determine the micro-scale parameters in $\mathbf{W}_{\text{out}}$ by optimizing the loss

$$\text{minimize} \quad \mathscr{L}_{\text{micro}}(\mathbf{r}, \mathbf{u}) = ||\mathbf{W}_{\text{out}}\mathbf{r} - \mathbf{u}||^2 + \beta||\mathbf{W}_{\text{out}}||^2, \tag{3.7}$$

where $\mathbf{r}$ is a matrix containing $\mathbf{r}(t)$ and $\mathbf{u}$ is a matrix containing $\mathbf{u}(t)$ for all t in the training dataset, and β is a Tikhonov-Miller regularization hyperparameter [162]. This is also known as ridge regression. The solution is

$$\mathbf{W}_{\text{out}} = \mathbf{u}\mathbf{r}^T(\mathbf{r}\mathbf{r}^T + \beta\mathbb{I})^{-1}, \tag{3.8}$$

where $\mathbf{I}$ is the $N \times N$ identity matrix. There is an option to add nonlinearity to the readout operator, which is addressed in section 4.4.5.

Forecasting

When making a forecast, the external forcing $\mathbf{u}(t)$ is replaced by a feedback loop $\mathbf{W}_{\text{out}}\mathbf{r}(t)$. Thus the equation for the "autonomous RC" is

$$\mathbf{r}(t+1) = F_r^a(\mathbf{r}(t)) = \alpha \tanh[\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{W}_{\text{out}}\mathbf{r}(t) + \sigma_b\mathbb{1}] + (1-\alpha)\mathbf{r}(t). \tag{3.9}$$

One can see that $\mathbf{u}(t)$ is replaced by $\mathbf{W}_{\text{out}}\mathbf{r}(t)$ and so in prediction we have $F_r^a(\mathbf{r}(t))$ being an autonomous dynamical system, allowing multi-timestep forecasts via recursion.

3.3.1 Synchronization and Training

Generalized Synchronization refers to the synchronization of two **nonidentical** dynamical systems which, by definition, cannot exhibit identical oscillations [135–137, 163]. For an input system $\mathbf{u}(t)$ and response system $\mathbf{r}(t)$, if there is GS, then there is a function ψ such that $\mathbf{r} = \psi(\mathbf{u})$; note that this relation is only true asymptotically—*i.e.*,

Table 3.1. Global scalar parameters of the RC. These parameters may either be treated as 'macro-scale' model parameters to be optimized during training, or may be used as hyperparameters and tuned manually.

Parameter	Description
ρ_{SR}	Spectral radius (max eigenvalue) of the adjacency matrix $\mathbf{A}$
ρ_A	Density of the adjacency matrix $\mathbf{A}$
N	Degrees-of-Freedom of the reservoir
α	Leak rate (time constant) of the reservoir computer
σ	Strength of input signal
σ_b	Strength of input bias
β	Tikhonov regularization parameter

after a certain finite amount of time has passed for transients to die out. The dynamics of the response system are therefore entirely predictable from the history of the input; in RC, for a contracting system, this property is related to the "echo state property" [104] and "fading memory" [164]. One rarely has an explicit form for ψ, but if there is GS, then that suggests it exists [34, 114]. To our knowledge, the details of the mathematical properties of $\psi(\mathbf{r})$ are not known. Important information about ψ is found in [165–167].

When $\mathbf{u}(t)$ and $\mathbf{r}(t)$ are synchronized, the combined system in $\mathbb{R}^{N+D}$ will lie on an invariant *synchronization manifold* $\mathcal{M}$ [168]. $\mathcal{M}$ must be locally attracting, that is the Lyapunov exponents transverse to the manifold, called conditional Lyapunov exponents, are negative. The stability of the motion on such a manifold has been the subject of numerous inquiries [168, 169] but can overall be summarized by the statement that $\mathcal{M}$ must be normally hyperbolic—*i.e.*, the contraction normal to $\mathcal{M}$ is larger than the contraction tangential to $\mathcal{M}$ [170].

While GS is defined to be $\mathbf{r} = \psi(\mathbf{u})$, we have defined PGS to be $\mathbf{u} = \varphi(\mathbf{r})$ without assuming $\varphi = \psi^{-1}$ in a global sense. PGS assumes that the system is contained in a local region where ψ is smooth, invertible and differentiable—see [142] for a discussion of conditions where this might hold. If the basins of attraction to the synchronization manifold are fractal then different initial conditions of the reservoir may end up on different

attractors.

PGS gives us some advantage in the analysis of RC networks; it assures us that the dynamical properties that the dynamical properties of the stimulus $\mathbf{u}(t)$ and the reservoir $\mathbf{r}(t)$ are now essentially the same. They share global Lyapunov exponents [29], attractor dimensions, and other quantities classifying nonlinear systems [135]. The principal power of PGS in RC is that we may replace the initial non-autonomous reservoir dynamical system with an autonomous system operating on the synchronization manifold.

The function $\varphi(\mathbf{r})$ is approximated in some manner, through training, Eq. (3.2), and then this is substituted for $\mathbf{u}$ in the reservoir dynamics. In previous work on this [114, 115] the authors approximated $\varphi(\mathbf{r})$ as a polynomial expansion in the components $\mathbf{r}_\alpha$ and used a Tikhonov-Miller [162, 171, 172] regularization method, also known as ridge regression [173], to find the coefficients of the powers of $\mathbf{r}_\alpha$. In this paper we follow their example. However, we note that there are a large number of approximation methods for representing functions of many variables [155–161, 174]. Some may provide a useful general representations of $\varphi(\mathbf{r})$ whose value could exceed that of a Taylor series expansion.

The reservoir dynamics acts at the nodes of the *non-autonomous* network equations for $\mathbf{r}(t)$ given by

$$\frac{d\mathbf{r}(t)}{dt} = \mathbf{F}_r[\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta}], \tag{3.10}$$

or the equivalent statement of the dynamics in discrete time.

If PGS takes place, we may replace Eq. (3.10) by the *autonomous* dynamical system

$$\frac{d\mathbf{r}(t)}{dt} = \mathbf{F}_r[\mathbf{r}(t), \varphi(\mathbf{r}(t)), \boldsymbol{\theta}]. \tag{3.11}$$

We use this for forecasting. What information we have about the manner in which driving forces influence the forecasting capability of the RC is now encoded in $\varphi(\mathbf{r})$ through the PGS relation $\mathbf{u} = \varphi(\mathbf{r})$.

3.3.2 The Auxiliary Method for PGS

There are a variety of approaches for determining whether one's selection of $\mathbf{r}(t)$ and $\mathbf{u}(t)$ exhibit PGS. Perhaps the most direct approach is to work with **two identical** reservoirs [34, 136] driven by the same $\mathbf{u}(t)$,

$$\frac{d\mathbf{r}_A(t)}{dt} = \mathbf{F}_r(\mathbf{r}_A(t), \mathbf{u}(t), \boldsymbol{\theta}) \text{ and } \frac{d\mathbf{r}_B(t)}{dt} = \mathbf{F}_r(\mathbf{r}_B(t), \mathbf{u}(t), \boldsymbol{\theta}). \tag{3.12}$$

Solve these two equations, keeping $\mathbf{u}(t)$ fixed and with $\mathbf{r}_A(0) \neq \mathbf{r}_B(0)$; then examine the distance between $\mathbf{r}_A(t)$ and $\mathbf{r}_B(t)$ as t becomes large. If $\mathbf{r}_A(t) \to \mathbf{r}_B(t)$ we have PGS. This 'auxiliary method' was first used in an experimental verification of GS by [140].

The auxiliary method is simply a restatement of the fact that GS is contingent upon the conditonal Lyapunov exponents (CLE) being negative. The CLEs give the rate of error growth of the response system dynamics on $\mathcal{M}$. Given a small perturbation to the RC trajectories—$\hat{\mathbf{r}}(t) = \mathbf{r}(t) + \delta\mathbf{r}(t)$—the error growth rate after linearization is

$$\frac{\mathrm{d}}{\mathrm{d}t}\delta\mathbf{r}(t) = \mathbf{F}_r(\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta}) - \mathbf{F}_r(\hat{\mathbf{r}}(t), \mathbf{u}(t), \boldsymbol{\theta}) \Rightarrow \frac{\mathrm{d}}{\mathrm{d}t}\delta\mathbf{r}(t) = D_r F(\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta}) \cdot \delta\mathbf{r}(t), \tag{3.13}$$

with $D_r F$ the jacobian with respect to $\mathbf{r}$. The CLEs can be calculated by solving this variational equation; for a reservoir dimension of several thousand this calculation can be computationally intensive.

The advantage of the auxiliary method test is that it fast and efficient, unlike the challenges associated with evaluating the CLE directly [27, 29, 34]. While the results of the two methods may be essentially the same, the required computations might be quite unequal. We show the outcomes of both approaches in Fig.(3.4). The CLEs [139] of the reservoir systems being negative mean that the two reservoir states should converge exponentially towards each other.

The auxiliary method is not a direct test for PGS. Rather it is a test that the

GS function ψ exists, is smooth and continuous [136]. While we do not claim that these properties always guarantee that the auxiliary method implies PGS, it is highly suggestive that it does. In addition, there are some limitations in general on the accuracy of the auxiliary method in certain circumstances. The method is only applicable when the driven system does not trend to a fixed point or a limit cycle for arbitrary u, but is instead "driven" by the input. Moreover, one may construct systems for which a function φ exists but for which the auxiliary method will fail. Therefore this test will not always guarantee PGS.

To restate the claim: if the auxiliary method passes then GS exists and the synchronization manifold in all likelihood is locally smooth and differentiable implying the existence of PGS. If the auxiliary test fails then in general GS is not occurring except in certain circumstances such as the presence of multiple basins of attraction. Therefore PGS is almost certainly not occurring. In factorable systems (such as an RC with block diagonal entries in the adjacency matrix) all independent components would have to be tested separately. In our experience, however, the auxiliary method is a useful practical tool for standard formulations of RC.

3.3.3 Testing for PGS

PGS provides us with a test of whether a particular reservoir, with a specific choice of architecture, nodal dynamics and hyperparameters, has the capability to learn the dynamics presented by the data. As described in the previous section, **without training** one can simply evolve $\dot{\mathbf{r}}(t) = \mathbf{F}_r(\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta})$ with the input $\mathbf{u}(t)$ present for two different initial conditions, and then test to see if PGS occurs. If PGS **does not occur** between the reservoir and the data, then our forecasting results will be unwelcome, and a more suitable choice of hyperparameters should be sought.

In Fig.(3.3) we display the predictive consequences of a set of $\boldsymbol{\theta}$ that admits PGS, **TopPanel**, and after that the predictive consequences of a set of $\boldsymbol{\theta}$ that does not admit

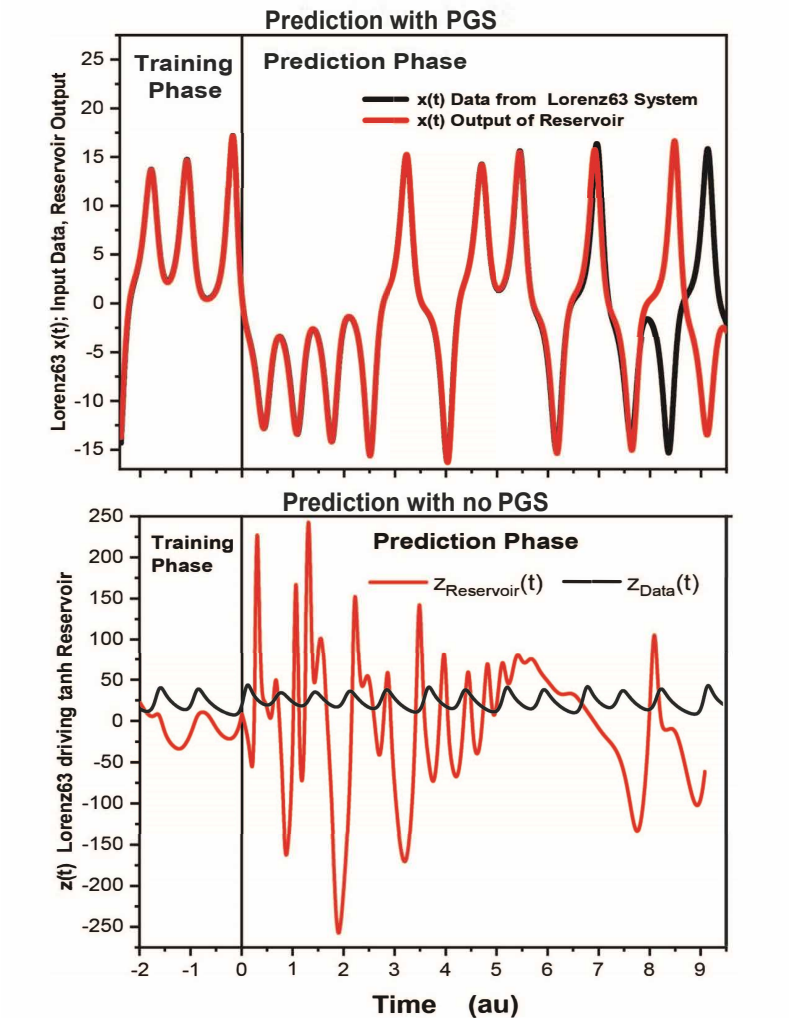

Figure 3.3. Top Prediction by an $N = 2000$ tanh reservoir output (red) receiving input from Lorenz63 dynamics (black) [145]. In $A_{\alpha\,\beta}$: $SR = 0.9$ and $\rho_A = 0.02$. This is in the PGS region. The black vertical line is the end of the "training period". **Bottom** When one selects the hyperparameters **outside** the region of PGS, for example using $N = 2000$, $SR = 1.6$ and $\rho_A = 0.02$ for the tanh reservoir, the function $\varphi(\mathbf{r})$ does not exist. We may expect the reservoir to operate poorly in producing a replica of the input $\mathbf{u}(t)$, as we can see it does.

PGS, **BottomPanel**. This Figure comes from a tanh reservoir driven by data coming from solving the Lorenz63 [145] equations.

Establishing regions of $\boldsymbol{\theta}$ for PGS can greatly reduce the number of RCs with different hyperparameters that must be trained and tested in a grid search over $\boldsymbol{\theta}$ or other identification technique.

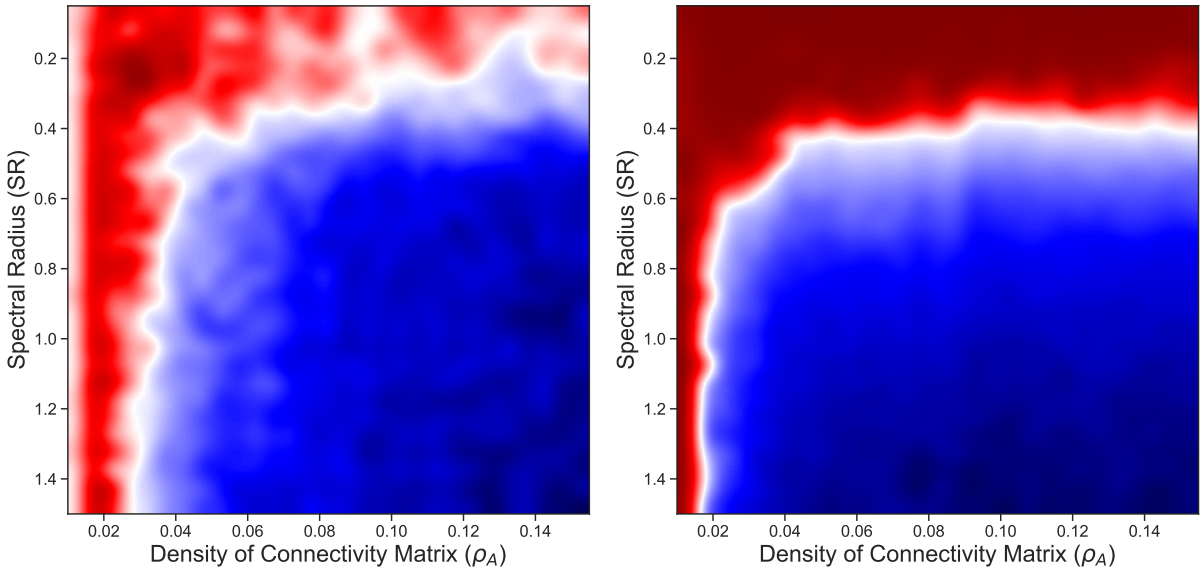

Figure 3.4. We display two ways of computing regions of PGS for a reservoir with Fitzhugh-Nagumo neurons [175, 176] at the nodes (N = 500). Both methods give approximately the same result.
Left Panel The largest conditional Lyapunov exponent (CLE) calculated for Lorenz63 input and a Fitzhugh-Nagumo RC as we vary the hyperparameters SR and ρ_A. **Blue** shows regions with *positive* CLEs, so no PGS. **Red** shows regions of *negative* CLE meaning PGS exists in this region.
Right Panel The error between the response system and the *auxiliary* response system as $t \to \infty$: $\|\mathbf{r}_A(t) - \mathbf{r}_B(t)\|$. If this remains large, there is no PGS. If it goes to zero, there is PGS. Choices for hyperparameters in the **Blue** regions indicate the absence of PGS, while choices in the **Red** regions show PGS.
There is a slight discrepancy between the two calculations. Part of this is certainly caused by numerical errors in estimating the CLEs of a high dimensional dynamical system using a finite trajectory. It is also possible that the auxiliary method initialized at different points could be falling into different basins of attraction resulting in a failed test even though the CLEs may be negative.

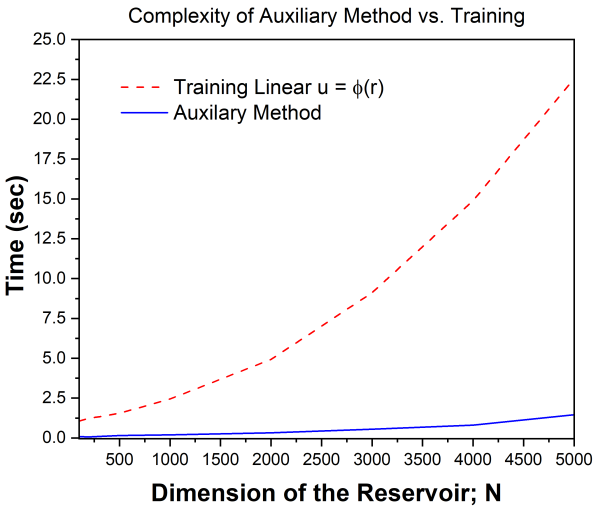

Figure 3.5. Time for testing a single set of hyperparameters for a tanh RC as a function of reservoir dimension using either the auxiliary method or directly through training the linear regression output. The test was executed on a personal desktop with an intel i9-9900K cpu. We recommend using the auxiliary test to find the bounds of the feasible space of hyperparameters for the RC. In addition, if using a cost function based optimization, then one could use the auxiliary method as a prescreen for a proposed set of parameters.

3.3.4 Advantages

The advantage of searching first for PGS comes from the fact that the auxiliary method test is fast and efficient. In practice this property often means that one can look at a much smaller segment of time in the data series than is required for accurate results for training. In addition, the linear regression step does not need to be completed, so searching for PGS is computationally much more efficient than training a reservoir and then evaluating it by predicting at multiple points.

As an example of the computational speedup afforded by the auxiliary method test, we evaluated the timing of the test vs the timing of training an RC as a function of reservoir dimension—Fig.(3.5). The benefit in using the auxiliary test comes for large N RCs.

Even greater benefit is accrued when running on a big system using a parallel reservoir scheme on localized patches such as in section 3.2.3. Generally PGS on one of

the parallel RCs implies PGS on all so we can decrease the computation substantially. The test for the SWE in 3.2.3 took 16 seconds per point searched, while training required a several hundred node CPU cluster with multiple reservoirs running in parallel and each point in the search required 20 minutes. In a Bayesian optimization setting such as in [125] this test can be used as a binary (PGS or no PGS) prescreening function to the cost function evaluation.

As may be seen in Fig.(3.4), the boundary between PGS and nonPGS regions is, in practice, quite sharp. This also seen in Fig.(3.8). So the search for PGS and nonPGS regions in $\boldsymbol{\theta}$ space may be coarse grained to begin and subsequently refined if desired [88]. Additionally, it has been suggested that RC works best at the "edge of stability" [177]. The auxiliary method allows the user to find this "edge" and concentrate the search for the hyperparameters in the border region where it has empirically been shown that the RC works best.

Testing for PGS only tells us that the function $\varphi(\mathbf{r})$ exists, not whether our approximation to it is sufficient for prediction. One would expect a linear approximation in $\mathbf{r}$ to $\varphi(\mathbf{r})$ would predict well only for a small subset of the parameters for which PGS is shown to occur; indeed this is exactly what we find empirically. We suggest that richer approximations for $\varphi(\mathbf{r})$ might expand this subset of good predictions to include much more of the region indicated by the PGS test.

3.4 Using PGS in Forecasting Examples

3.4.1 Two RC Networks

For the purposes of this paper, we have used two quite distinct dynamical systems at the reservoir nodes. One has tanh nonlinearities at the reservoir nodes, and the second has nonlinear FHN [175, 176] oscillators at its nodes. Our methods are the same, giving qualitatively similar results, for the identification of PGS and noPGS regions for these

and many other instantiations of input signals and reservoir activation functions. We have used simple Hodgkin-Huxley neuron models [151, 152, 178] in an RC architecture, but do not report the results in this paper.

These simple and familiar choices emphasize our earlier comment that any smooth dynamics may be used as activation functions in an RC.

The descriptions of these two choices now follows:

The Hyperbolic Tangent RC

We use the adjacency matrix $\mathbf{A}_{\alpha\,\beta}$ and nonlinear activation function tanh() in this RC model. γ adjusts the time scale of the reservoir dynamics, and σ weights the strength of the input signal $\mathbf{u}(t)$

$$\frac{d\mathbf{r}_\alpha(t)}{dt} = \gamma\Big[-r_\alpha(t) + \tanh(A_{\alpha\beta}r_\beta(t) + \sigma W_{\alpha a}u_a(t))\Big]. \tag{3.14}$$

Repeated indices are summed over. The $N \times D$ matrix $W_{\alpha a}$ determines where the input signals are introduced into the reservoir.

In Table (3.1) we display the meaning of the parameters in the tanh RC. For other selections of active units and RC architectures different tables may be useful.

The FHN RC

The equations for the Fitzhugh-Nagumo Model (FHN) [175, 176] operating at reservoir sites $\gamma = 1, 2, .., N$ are

$$\frac{dV_\gamma(t)}{dt} = \frac{1}{\tau}[V_\gamma(t) - \frac{1}{3}V_\gamma(t)^3 - w_\gamma(t)] + \left\{\sum_{\beta=1}^{N} A_{\gamma\beta}V_\beta(t)I0(\gamma,\beta) + \sum_{b=1}^{D} W_{\gamma b}u_b(t)\right\} \tag{3.15}$$

$$\frac{dw_\gamma(t)}{dt} = V_\gamma(t) - \eta w_\gamma(t) + \xi$$

The constants are $\xi = 0.7$, $\eta = 0.8$, $\tau = 0.08$ ms. $I0(\gamma, \beta) = \frac{I_0}{2}[1 + \tanh(K(V_\beta(t) - V_p))]$ is a synaptic current from a presynaptic neuron labeled by β to a postsynaptic neuron labeled by γ; $K = 3/2$, $V_p = 1$, $I_0 = 1.0$.

3.4.2 Data Sources

Lorenz63 Model

The Lorenz-63 [145] equations form a deterministic nonlinear dynamical system that exhibits chaos for broad ranges of parameters. It was originally presented as a three dimensional, **very** reduced, approximation to the partial differential equations for the heating of the lower atmosphere of the earth by insolation. The dynamical equations of motion are

$$\begin{aligned} \frac{dx(t)}{dt} &= \sigma[y(t) - x(t)] \\ \frac{dy(t)}{dt} &= x(t)[\rho - z(t)] - y(t) \\ \frac{dz(t)}{dt} &= x(t)y(t) - \beta z(t) \end{aligned} \tag{3.16}$$

$$\tag{3.17}$$

with time independent parameters often chosen to be $\sigma = 10, \rho = 28, \beta = 8/3$.

The global Lyapunov exponents here are $\{\lambda_1, \lambda_2, \lambda_3\} = [0.9, 0, -14.7]$ calculated using the QR decomposition algorithm given by Eckmann and Ruelle [27].

We chose this model to provide three dimensional input $\mathbf{u}(t) = \{x(t), y(t), z(t)\}$ from the Lorenz63 system, as it is a test bed for many new ideas involving chaotic time series.

Lorenz-96 Model

This model of Lorenz [146, 147] is widely used in geophysics to examine new methods of data assimilation [179]. It has the structure of 'stations" at $x_a(t)$ on a ring

forced by a constant f. For $f \approx 8.0$ the $\mathbf{x}(t)$ are chaotic when $D \geq 3$.

The dynamical equations introduced by Lorenz [146, 147]:

$$\frac{dx_a(t)}{dt} = x_{a-1}(t)(x_{a+1}(t) - x_{a-2}(t)) - x_a(t) + f \tag{3.18}$$

and $a = 1, 2, ..., D$; $x_{-1}(t) = x_{D-1}(t)$; $x_0(t) = x_D(t)$; $x_{D+1}(t) = x_1(t)$. f is a fixed parameter which we take to be in the range 8.0 to 8.2 where the solutions to these dynamical equations are chaotic [179, 180]. The equations for the states $x_a(t)$; $a = 1, 2, ..., D$ are meant to describe 'stations' on a periodic spatial lattice. We use $D = 5$.

The Lyapunov exponents are $\{\lambda_1, \ldots, \lambda_5\} = [0.6, 0.0, -0.4, -1.4, -3.8]$ and were evaluated via the QR decomposition algorithm given by Eckmann and Ruelle [27].

Data from this model are used in Section (3.5).

Shallow Water Equations

These are equations for three fields the East-West and North-South velocities, and the fluid height $[u(x, y, t), v(x, y, t), h(x.y.t)] = [\mathbf{V}(\mathrm{x,y,t}), h(x, y, t)]$ and describe the flow of a thin ($\approx$ 10km) layer of fluid on a two dimensional mid-latitude plane tangent to the earth ($\approx$ 6400km in radius) [148–150]. They form a core contribution to much more complex models of the atmosphere/ocean system in weather and climate prediction models.

$$\frac{\partial \mathbf{V}(x, y, t)}{\partial t} + \eta(\hat{z} \times (h(x, y, t)\mathbf{V}(x, y, t))) + \nabla[gh(x, y, t) + \frac{\mathbf{V}(x, y, t)^2}{2}] = 0$$

and

$$\frac{\partial h(x, y, t)}{\partial t} + \nabla \cdot (h(x, y, t)\mathbf{V}(x, y, t)) = 0. \tag{3.19}$$

$\hat{z}$ is a unit vector normal to the (x,y) plane of the fluid flow. g is the strength of gravity and

$$\eta = \frac{\partial_x v(x,y,t) - \partial_y u(x,y,t)}{h(x,y,t)}, \tag{3.20}$$

is the potential vorticity.

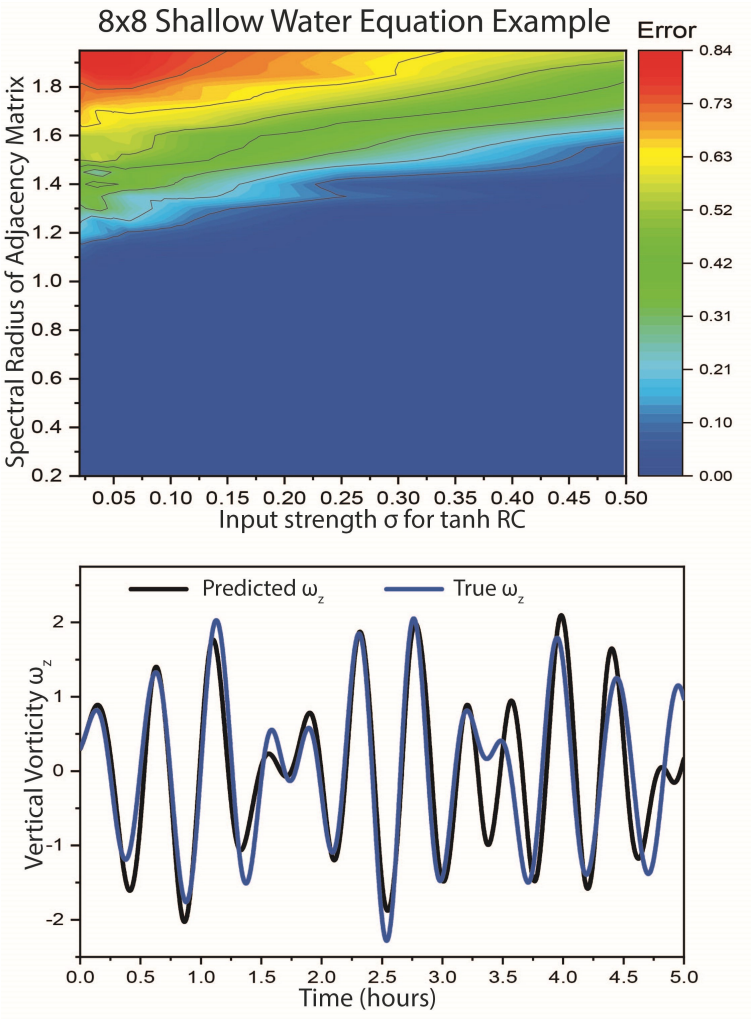

Figure 3.6. Top Plot Contour plot of **PGS** and **no PGS** regions. Blue/Purple indicates a region of parameters in a localized tanh reservoir model (N = 5000) which shows PGS or noPGS with a driving signal from the 8×8 Shallow Water Equations (SWE) [148, 149] as $\mathbf{u}(t) \in \mathbb{R}^{192}$. The red region shows no PGS.
Bottom Plot Forecast for the normalized vertical vorticity at a particular point on the 192 dimensional $8 \times 8 \times 3$ grid. The forecast starts here at time 0 after a spinup period which is not shown. The localized reservoir algorithm and details of the SWE are found in section 3.2.3.

Shallow Water Equation Results

Accurate numerical solutions to the SWEs on a grid have been investigated in detail by Sadourny [148] who concluded that a potential-enstrophy conserving scheme is effective. The details of this scheme can be found in Section 2 of [148]. We use a form of the SWEs with three dynamical variables: surface height $\mathbf{h}(x, y, t)$, and the $\mathbf{u}(x, y, t)$ and $\mathbf{v}(x, y, t)$ components of velocity. We solve the SWEs numerically on a discretized grid of size $N_\Delta = 8$ in two horizontal directions, resulting in an 8×8 grid. Including the three dynamical variables, this yields a D = 192-dimensional dynamical system.

Following the scheme used in [110] on a 1 dimensional grid, we use this discretized numerical integration of the SWEs to drive a set of localized reservoirs arranged in 16 overlapping local "patches" on a 2 dimensional grid. Each patch receives input from a subset of 48 local variables of the total 192-dimensional input vector. The 48 variables input to each local reservoir consist of 16 $u(t)$, 16 $v(t)$ and 16 $h(t)$ that are located at the 16 points on a local patch of the grid. Each local reservoir is used to predict 12 (4 $u(t)$, 4 $v(t)$, 4 $h(t)$) of these after training, thus creating the overlapping scheme.

From the dynamical variables $\{u(x, y, t), v(x, y, t), h(x, y, t)\}$ we compare the reservoir output for normalized height and for the normalized vertical vorticity $\omega_z(x, y, t)$:

$$\omega_z(x, y, t) = \frac{\partial v(x, y, t)}{\partial x} - \frac{\partial u(x, y, t)}{\partial y}. \tag{3.21}$$

with their counterparts in the data.

Even in this complicated set of overlapping localized RCs, it is straightforward and computationally efficient to apply our PGS test to the data. Applying the auxiliary test we see—Fig.(3.6)—that there is a broad region where our 16 reservoir scheme synchronizes with the data. This test is much more computationally efficient than evaluating the reservoir by training, thus giving us guidance as to where to focus our search. Then, after a traditional search over this smaller grid of hyperparameters, a set of hyperparameters

were found that produce reasonable and robust predictions over a short time scale. The test enables us to significantly reduce the number of hyperparameters searched.

We have also tested our procedures on other autonomous dynamical system drivers of an RC: (1) the Colpitts oscillator [181] and (2) the Double Scroll system [182]. The success of these investigations produces no additional guidance on using PGS, so we do not report on them any further in this paper.

Driven Dynamical Systems; A Biophysical Example

The examples we have discussed until now have addressed data streams $\mathbf{u}(t)$ from autonomous dynamical systems as represented in Eq. (3.3). Here we extend the discussion to a driven dynamical system.

A scientific area where RC may well find broad application is biophysics, in particular neurobiology. Neurons individually or in a functional biophysical circuit exhibit only a fixed point behavior ("resting potential") when they do not receive external stimuli $\mathbf{I}(t)$. Clearly one does not need RC to predict the future of such a response.

When driven by electrical forces, usually a stimulating current $\mathbf{I}(t)$, the dynamics for the same neuron model [151, 152] is conditional on the forcing. The equations for the data stream change from Eq. (3.3) to the non-autonomous data source system:

$$\frac{d\mathbf{u}(t)}{dt} = \mathbf{F}_u(\mathbf{u}(t), \mathbf{I}(t))), \tag{3.22}$$

and, as $\mathbf{u}(t)$ depends on the selected forces in $\mathbf{I}(t)$, information about this must be conveyed to the RC. This physical setting for data sources is important well beyond biophysics.

As an example to show how RC works in this situation we have chosen to examine a data stream arising from the forcing of the simplest neuron models: that of Hodgkin-Huxley (HH). We use essentially the one they established 70 years ago [151, 152, 178]. The HH model is perhaps the earliest biophysical model of neuron oscillations. This neuron, a

nonlinear electrical oscillator, has ion channels in its cell membrane through which Na^+ ions and K^+ ions flow, as well as a 'leak' channel which represents the fact that the cell membrane leaks charge.

We selected this model as it is the basis for the characterization of much more complex neuron structures.

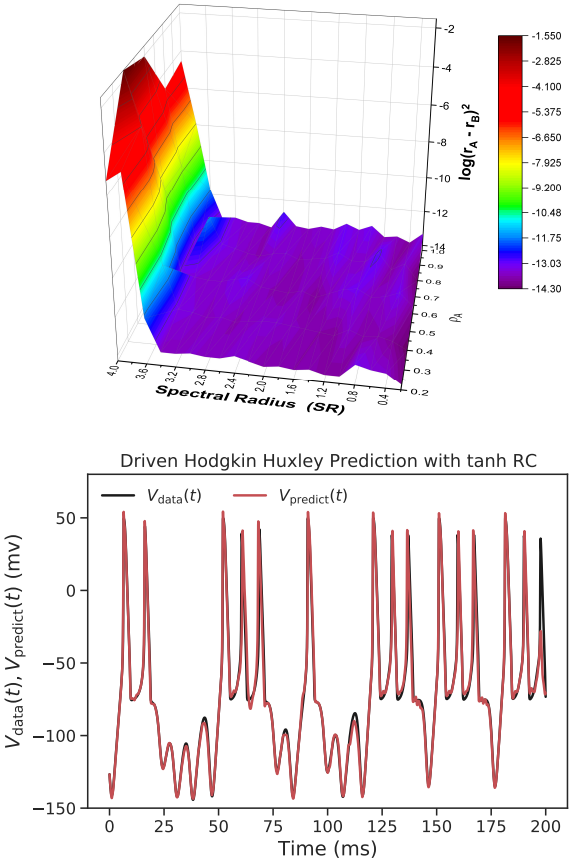

Figure 3.7. Top Panel Three dimensional display of the PGS and nonPGS regions in the case of an NaKL Hodgkin-Huxley neuron [151, 152, 178], driven by an external current with a waveform derived from the components associated with the Lorenz63 [145] model, and presented to a tanh RC. These regions were selected using the auxiliary method, and we show the logarithm of $\|r_A(t) - r_B(t)\|$ for large times as a function of SR and ρ_A. The regions colored blue/purple are where PGS is found. The nonuniform surface reflects the residual roundoff error using single precision arithmetic in the calculations. **Bottom Panel** The cross membrane voltage showing both the known data and the prediction of the trained reservoir in the case of an NaKL Hodgkin-Huxley neuron [151, 152, 178], driven by an external current with a waveform derived from the components associated with the Lorenz63 [145] model, and presented to a tanh RC. The parameters SR = 3.65 and $\rho_A = 0.02$ were selected to be in the PGS region of the top panel.

The Hodgkin-Huxley model is governed by the following four first-order differential equations

$$C\frac{dV(t)}{dt} = I_{inj}(t) + g_{Na}m(t)^3h(t)(E_{Na} - V(t))$$
$$+g_K n(t)^4(E_K - V(t)) + g_L(E_L - V(t)).$$

$$\frac{dG(t)}{dt} = \frac{G_0(V(t)) - G(t)}{\tau_G(V(t))} \qquad G(t) = \{m(t), h(t), n(t)\}$$
$$G_0(V) = \frac{1}{2}[1.0 + \tanh\left(\frac{V - V_G}{\Delta V_G}\right)]$$
$$\tau_G(V) = \tau_{G0} + \tau_{G1}\left(1 - \tanh^2\left(\frac{V - V_G}{\Delta V_G}\right)\right) \tag{3.23}$$

In these equations the $\{g_{Na}, g_K\}$ are maximum conductances for the Na and K ion channels, the $\{E_{Na}, E_K\}$ are reversal potentials for those ion channels, and $I_{inj}(t)$ is the external stimulating current injected into the neuron. The overall strength of an ion channel is set by the maximal conductances.

The gating variables $G(t) = \{m(t), h(t), n(t)\}$ are taken to satisfy first order kinetic equations and each lies between zero and unity, as they are effectively the probability that the ion channel is open.

The quantities $G_0(V)$ and $\tau_G(V)$ are the voltage dependent activation function and the voltage dependent time constant of the gating variable $G(t)$. The forcing to the cell $I_{inj}(t)$, here a scalar, is known to us. The 19 parameters entering Eq. (3.23) are selected and further discussed in [52] as well as in many other places [151, 152, 178, 179].

We, for this example, have chosen the forcing current to be a function $I_{inj}(t)$ taken to be proportional to the x(t) component of the Lorenz63 model. The Physics reasons for that are addressed in [52].

We have found that, if we convey the data stream for the four Hodgkin-Huxley state variables, along with **only** the x(t) component from the Lorenz63 model, then following our

guiding path to working with an RC, we arrive at quite good predictions of the HH data stream (not shown here). However, if we add the information about the other Lorenz63 state variables, $\{y(t), z(t)\}$, the forecasting capability of the RC (here a tanh reservoir) is very much enhanced.

The procedure is to perform training and predicting with RC on the HH System (4 Dimensions) + L63 Dimensional System (3 Dimensions). The key difference between the driven and autonomous cases is that throughout the prediction process, the 3 dimensions of L63 are driven by their true values. This means that the 4 dimensions of the NaKL system are linked together to the 3 dimensions of the L63 system through the mixing process in the reservoir. The statistical stationarity of the L63 model allows predictions forward in time.

In Figure (3.7) we show, in the **top panel** the regions of PGS and nonPGS when a Hodgkin-Huxley data stream is presented to an N = 1000 dimensional tanh reservoir. The parameters for the prediction in the **bottom panel** are chosen at the edge of stability (SR = 3.65) showing the benefit of first looking at the PGS regions and then searching for hyperparameters.

3.5 Assessing Successful Reservoir Properties

After running the test for PGS and performing a hyperparameter search, the question arises of how to guarantee stable forecasting. One often encounters the two situations in RC:

- The forecast starts out close to the data but then quickly diverges and becomes non-physical
- The forecast is "good" for certain initial starting conditions but not for others.

The problem of finding a set of hyperparameters that will give robust predictions is one of the main challenges in reservoir computing and RNNs in general. The definition of

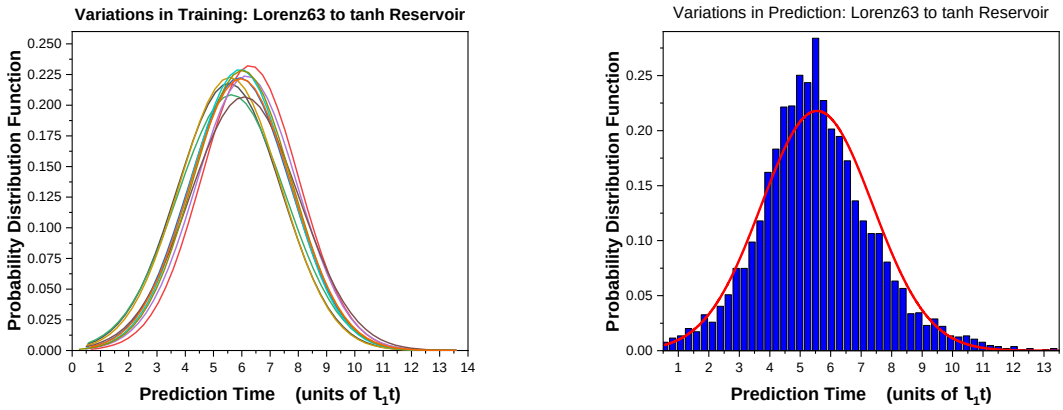

Figure 3.8. Left Panel Gaussian fit to the prediction time for 10 N=2000 tanh RCs trained on Lorenz63 data with the same hyperparameters but different random seeds and training data. The prediction time is the time for which the prediction stays close to the true values; the calculation is detailed in the appendix. Each reservoir predicts 4000 randomly selected training points. These points are different for each reservoir. The 10 RC's prediction times overlap closely; the (mean(10 reservoirs)) = 5.92 and the (RMS deviation(10 reservoirs)) = 0.24. This shows the robustness of this set of hyperparameters to training data and randomization of the reservoir layers. **Right Panel** A histogram and the Gaussian fit to it from the **Left Panel** better displays the variation shown in one hyperparameter setting of the reservoir computer.

a robust set of hyperparameters is one in which neither the randomness of the adjacency matrix $A_{\alpha\,\beta}$ or the training data set causes the reservoir to fail in prediction. An example of robustness is shown in Fig.(3.8) for a Lorenz63 input system. Many sets of parameters work well for a particular example or point on the attractor. It is more challenging to find a set of parameters that predict well for many initial conditions and for different instantiations of the RC.

The typical approach for evaluating ML predictions with the mean squared error over a test set does not capture a key feature of RC; a well trained RC should be able to give good short term predictions for **all** initial starting points **and** be stable in the medium to long term. This feature is called attractor reconstruction [114]. Instead of a test set, we propose an additional criterion for RC evaluation; **a well trained RC reproduces the spectrum of Lyapunov exponents of the input system** F_u; Fig.(3.9) for an example.

Lyapunov exponents (LEs) characterize the average global error growth rate of a dynamical system [25] along directions in phase space. One can calculate the N LEs of the forecast reservoir $\dot{\mathbf{r}}(t) = \mathbf{F}_r(\mathbf{r}(t), \varphi(\mathbf{r}(t), \boldsymbol{\theta})$ and compare the largest of them to the D exponents of the input system. If the D largest LEs match and the smaller $N - D$ exponents of the RC are negative, then the two systems will have the same global behavior, increasing the likelihood of robust, stable predictions. It is important to note that the exponents of the input system can be calculated directly from experimental data [34].

We show this calculation for the Lorenz96 system [146] in Fig.(3.9). Our results show that when more of the spectrum of LE's are matched by the RC, the better the average predictions. In situations where it is difficult to exhaustively test the RC, perhaps because the model is expensive to run or there is limited data, evaluating the Lyapunov exponents of the forecasting reservoir will guarantee that the global error growth of the RC is the same as the data.

The LE criterion should be used as a check on the robustness of the system; this is *in lieu* of checking robustness by training and predicting over thousands of initial

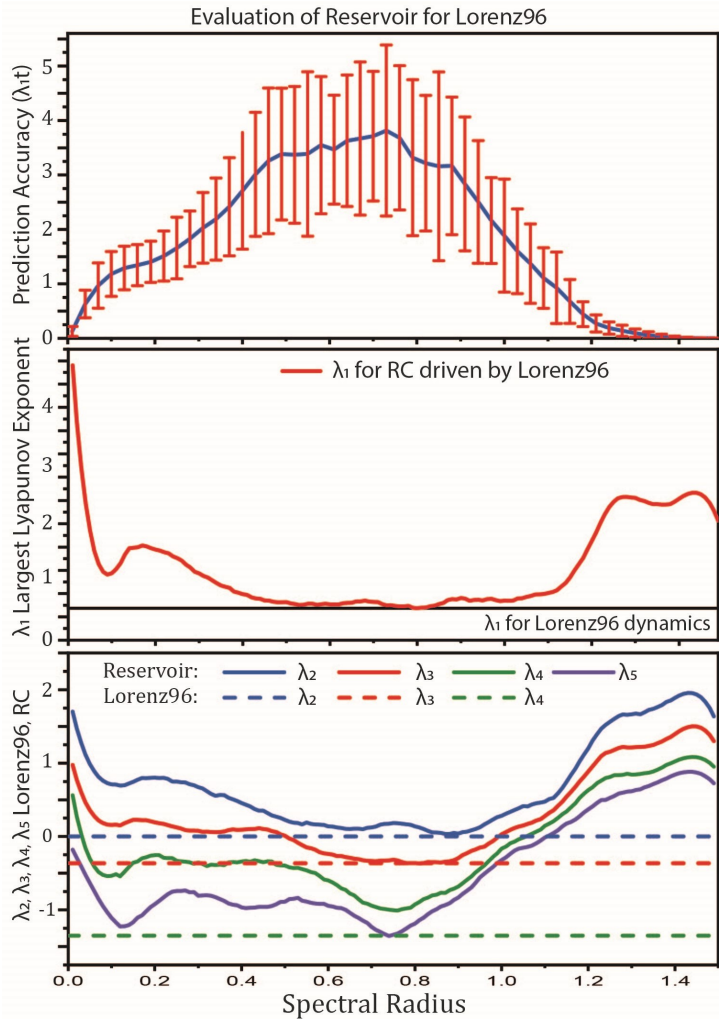

Figure 3.9. Top Average prediction time of a N=2000 tanh reservoir as a function of SR when driven by a $D = 5$ Lorenz96 $\mathbf{u}(t)$ [146]. The time is in units of $\lambda_1 t$. The error bars indicate variation in prediction depending on the stability of the input stimulus. **Middle** The largest Lyapunov exponent, λ_1 of the forecast reservoir and λ_1 of the input system (black line) as a function of SR. **Bottom** $\lambda_2, ..., \lambda_5$. The next four Lyapunov exponents of the forecast reservoir . The method for computing the Lyapunov Exponents of an RC is discussed in [27, 34, 102]. Theoretically one of the LEs should always be 0, the slight discrepancy between 0 and the LEs is caused by numerical errors in estimating the LEs of a high dimensional dynamical system from a finite time sequence. Of course, we have displayed only a small subset of the LE's of the trained reservour computer in this Figure.

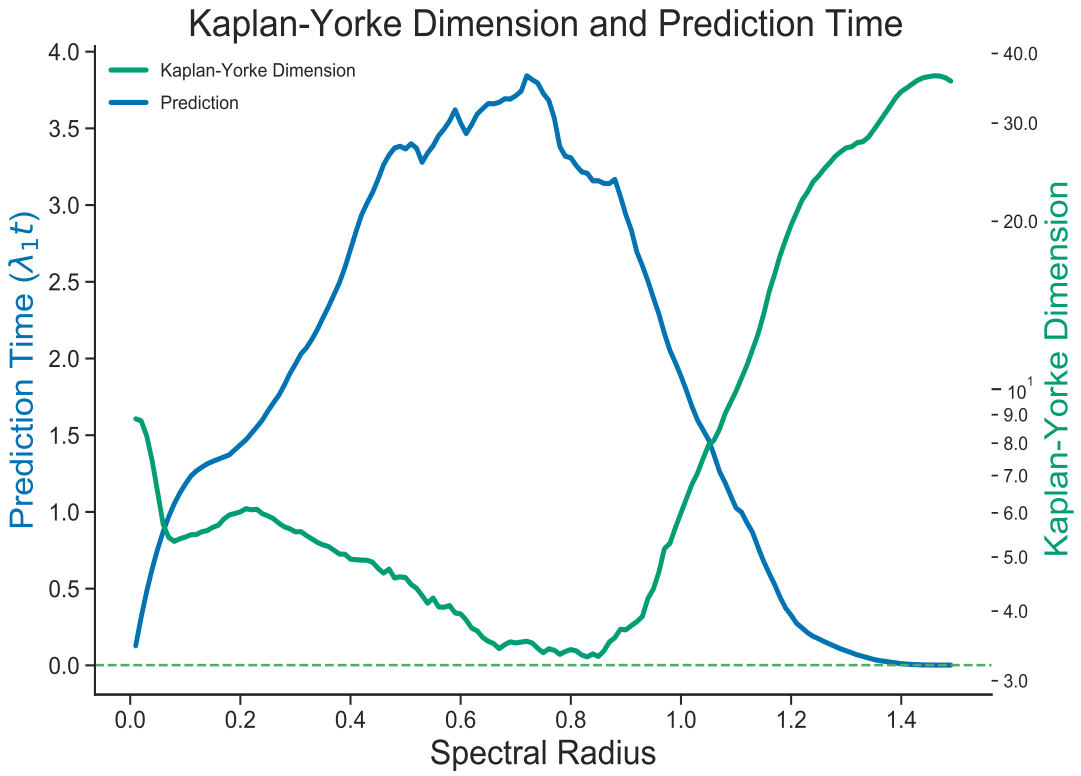

Figure 3.10. The Kaplan-Yorke fractal dimension [32, 33] (Green) of the tanh forecasting reservoir (N=2000) trained by a Lorenz96 D = 5 system as a function of SR plotted along with the Prediction Time in units of $\lambda_1 t$ (Blue); this is the same system and RC as for the data shown in Fig.(3.9). The predicting reservoir KY dimension is an estimate of the dimensionality of the synchronization manifold where the RC resides. The KY dimension of the 5D L96 system is shown as the dashed line. As SR crosses ≈ 1.1, corresponding to the largest Conditional Lyapunov Exponent of the reservoir crossing 0, the KY dimension of the reservoir increases rapidly. This corresponds to the reservoir moving off the low dimensional PGS manifold. One expects that as this occurs, the forecasting capability of the RC will vanish quickly.

conditions Fig.(3.8). These data may not be available for experimental systems or simply computationally infeasible.

One could also use the matching of LEs as a cost function over which to evaluate the best hyperparameters. This calculation is, however, quite expensive on a cpu but may work more efficiently given a GPU implementation. There are also reasonably efficient approximation techniques for estimating the largest exponent [183].

One can gain additional insight by plotting the forecast against the Kaplan-Yorke fractal dimension of the synchronization manifold Fig.(3.10). The KY dimension is calculated using the LEs of the autonomous forecast reservoir when it has moved onto the synchronization manifold. The lower the dimension of the synchronization manifold, the better the predictions will be. If there are too many degrees of freedom there is movement in directions that do not correspond to physically meaningful signals; thus the effect of the extra dimensions is to add the equivalent of noise into the reservoir. Another view is that a lower dimensional synchronization manifold corresponds to a smooth and continuous φ which is quantified by the KY dimension.

We speculate that the lowest dimension of the synchronization manifold could be related to the minimum embedding dimension of the system [34]—not necessarily the fractal dimension of those systems. However, when calculated for the Lorenz 63 [145], Lorenz 96 [146] and Colpitt's Oscillator we could find no direct correspondence between the dimension of the synchronization manifold and the embedding dimension. Our failure in this regard does not mean that the correspondence does not exist.

The results presented in Fig.(3.9) match the suggestion that the reservoir operates best at "the edge of chaos" [102, 184, 185], that is, the maximal prediction time of the reservoir corresponds to a SR just less than 1. [177] makes the point that the "edge of chaos" is not necessarily always the best point for predictions of the reservoir.

3.6 Conclusion and Discussion

In this paper we have addressed the following topics:

- Generalized Synchronization for a dynamical system $\mathbf{r}(t)$ driven by a signal $\mathbf{u}(t)$

$$\frac{d\mathbf{r}(t)}{dt} = \mathbf{F}_r(\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta}), \tag{3.24}$$

  in which category Reservoir Computing (RC) belongs, has long been seen as the condition that there is some function $\psi(\mathbf{u}) = \mathbf{r}$ connecting the network $\mathbf{F}_r(\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta})$ coordinates $\mathbf{r}$ to the unidirectional drive signal $\mathbf{u}$. After a discussion of the possibility that this may translate to a similar connection $\mathbf{u} = \varphi(\mathbf{r})$, which we called Predictive Generalized Synchronization (PGS) we concluded, based on our present knowledge, that this connection need not be global inverses of each other. We suggested, for example, in the case of multistability [143, 186, 187] of the dynamics of $\mathbf{r}$ and/or $\mathbf{u}$, such a global relation might be difficult to establish.

  To bypass this subtle mathematical issue, we used PGS, $\mathbf{u} = \varphi(\mathbf{r})$ in the discussion of RC, as this is all that is required to investigate how we might focus on the manner in which RC forecasts so well.

- We introduced a computationally efficient numerical test, based on PGS, and using the 'auxiliary method', to guide hyperparameter selections in RCs resulting in very good forecasting.

- We delineated the ideas for the use of PGS with a few simple low illustrative models [145–147] presented as $\mathbf{u}(t)$ to an RC for forecasting, then we turned to an important geophysical model [148–150], and finished with a discussion of a biophysical model of neuron dynamics [151, 152]. The last item produces data from a driven dynamical system (the neuron), and the data depend on an injection of current to

stimulate the neuron into interesting oscillations. The RC must obtain information about the driving force as it is trained.

- We explored a metric for a "well trained" RC network using the reproduction of the input system's Lyapunov exponent spectrum.

3.6.1 Issues to Address

The discussion in this paper directs attention to at least these issues:

- It appears interesting to complete the investigation of driven systems such as the neuron model we considered in Section (3.4.2). What information is required,in detail, about the forces stimulating the driven system of interest. This is especially of interest when the dynamics, if any, of those forces may not be known.

- With reference to the previous item, one should investigate how we can use RC to forecast in a non-stationary environment [188, 189], [138] Chapter 13, and references therein, see also [190]. This may be an important question for analysis of observed data.

- We used a polynomial representation for $\varphi(\mathbf{r})$, Section (3.7.1), following a path drawn in the literature [114]. However, we have also successfully used radial basis functions [155–161, 174] in this regard. This material is not presented in this paper.

- Once parameters for PGS regions for a given $\mathbf{u}(t)$ and a selected $\mathbf{F}_r(\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta})$ are found, one may still wish to seek choices in those regions yielding the 'best' forecasting, Section (3.5). In this regard, not presented here, we have found the algorithm, Differential Evolution [88], to be quite helpful.

Finally, we address the question of why all this could be very interesting for the use of ML in forecasting Physical, Geophysical, Biophysical, and other observed data. It might seem that requiring the use of a high dimensional 'reservoir' $\mathbf{F}_r(\mathbf{r}(t), \mathbf{u}(t), \boldsymbol{\theta})$ to forecast

low dimensional observed dynamics is not promising. However, from the work of [120] on implementing reservoirs in hardware, it appears that one can build special purpose computers to forecast the future of observed data and provide a fast, rather easily realized forecast machine. The simplicity of many choices of activation functions at the nodes of the RC lends itself to realizing this in practice.

3.7 Appendix

3.7.1 Polynomial Expansion

GS assures us that the dynamical properties of the stimulus $\mathbf{u}(t)$ and the reservoir $\mathbf{r}(t)$ are now essentially the same. They share global Lyapunov exponents [29], attractor dimensions, and other classifying nonlinear system quantities [135].

The principal power of PGS in RC is that we may replace the initial non-autonomous reservoir dynamical system

$$\frac{d\mathbf{r}_\alpha(t)}{dt} = F_\alpha[\mathbf{r}(t), \mathbf{u}(t)], \tag{3.25}$$

with an autonomous system operating on the synchronization manifold [168]

$$\frac{d\mathbf{r}_\alpha(t)}{dt} = F_\alpha[\mathbf{r}(t), \varphi(\mathbf{r}(t))]. \tag{3.26}$$

In practice, the function $\mathbf{u} = \varphi(\mathbf{r})$ is approximated in some manner, through training, and then this is substituted for $\mathbf{u}$ in the reservoir dynamics. In previous work on this [114, 115] the authors approximated $\varphi(\mathbf{r})$ via a polynomial expansion in the components $\mathbf{r}_\alpha;\ \alpha = 1, 2, ..., N$, and used a regression method to find the coefficients of the powers of $\mathbf{r}_\alpha$.

This means we write $u_a(t) = \varphi_a(\mathbf{r}(t)) = \sum_{\alpha,\beta=1}^{N} J_{a\alpha}\ r_\alpha(t) + Z_{a\alpha\beta}\ r_\alpha(t) r_\beta(t) + \ldots$, and we evaluate the coefficients $\{\mathbf{J}, \mathbf{Z}, \ldots\}$ by minimizing with respect to the constant matrices $J_{a\alpha}$ and $Z_{a\alpha\beta}$

$$\sum_t \left[u_a(t) - \left\{ \sum_{\alpha,\beta=1}^{N} J_{a\alpha}\, r_\alpha(t) + Z_{a\alpha\beta}\, r_\alpha(t) r_\beta(t) + \ldots \right\} \right]^2 + \text{regularization term.} \quad (3.27)$$

See the discussions in [162, 171, 172, 191, 192].

The dimension of $J_{a\,\alpha}$ is D by N. The dimension of $Z_{a\,\alpha\,\beta}$ is D by $\frac{N(N+1)}{2}$ as it is symmetric in $\{\alpha, \beta\}$. If one simplifies to keeping only 'diagonal' terms in $\{\alpha\,\beta\}$, then the second term in Eq. (3.27) is $\mathbf{Z}_{a\,\alpha}[r_\alpha]^2$ and this has dimension D by N.

We use this polynomial representation for $\varphi(\mathbf{r})$, noting there are many ways of approximating multivariate functions of $\mathbf{r}$ [155–159, 161].

3.7.2 What if only one component of the data is known ?

If we only know one component of the time series, say $s(t_n) = s(n)$, which is a scalar, we can define an M-dimensional proxy space of vectors $\mathbf{S}(n) \in \mathbb{R}^M$ via time delays [34, 138, 193] as

$$\begin{aligned} \mathbf{S}(n) &= [s(n), s(n-\tau), ..., s(n-(M-1)\tau)] \\ &= [S_1(n), S_2(n), ..., S_M(n)] \end{aligned} \quad (3.28)$$

From the definition of the components of $\mathbf{S}(n) = \{S_k(n) = s(n-(k-1)\tau)\}$ for $\mathbf{S} \in \mathbb{R}^M$ we have in time steps of τ,

$$\mathbf{S}(n+1) = \mathbf{H}(\mathbf{S}(n)), \quad (3.29)$$

We know something about $\mathbf{H}(\mathbf{S})$: noting that $S_k(n+1) = S_{k-1}(n)$ for k = 2,...,M. For $k = 1$, $S_1(n+1) = s(n+1) = H_1(\mathbf{S}(n))$. $H_1(\mathbf{S})$ is a scalar function of the M-dimensional variables $\mathbf{S}$.

We have, for $H_1(\mathbf{S})$, the same problem we addressed in representing a function of many variables $\varphi(\mathbf{r})$. This is an easier issue than representing $\varphi(\mathbf{r})$, as it is a vector in $\mathbb{R}^D$, while $H_1(\mathbf{S})$ is a scalar.

To 'train' $H_1(\mathbf{S})$ we note that $S_1(n+1) = s(n+1)$, so $s(n+1) = H_1(\mathbf{S}(n))$. We know the values of s(n), so we can 'train' $H_1(\mathbf{S}(n))$ by expanding it in a Taylor series in $\mathbf{S}(n)$, for example, and then determine the expansion coefficients in that series expansion as we did in Eq. (3.27) for $\varphi(\mathbf{r})$.

It is important to keep nonlinear terms in $\mathbf{S}(n)$ in the representation of $H_1(\mathbf{S}(n))$. If we were to take $H_1(\mathbf{S})$ to be linear, we would miss the nonlinear terms in the dynamical equations producing $s(t_n)$.

3.7.3 Prediction Quality

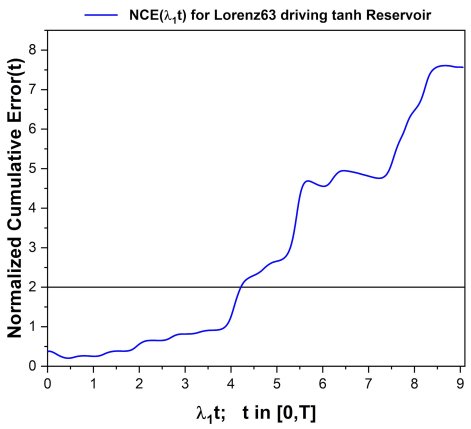

Figure 3.11. NCE($\lambda_1 t$), Eq.(3.30), Lorenz63 input to a tanh RC; $N = 2000$. NCE remains quite small for a long time. As the two time series separate NCE(t) rises. The dashed line suggests when the quality of the prediction has "ended."

To compare prediction times, a metric is required to estimate when the RC's capability to accurately predict $\mathbf{u}(t)$ ends. It needs to allow that the RC could be close to but not precisely the input for a number of time steps before diverging. A good metric for

this **Normalized Cumulative Error** (NCE):

$$\mathbf{NCE}(\mathbf{t}) = \sum_{t=1}^{T} \frac{1}{t}[\mathbf{u}(t) - \varphi(\mathbf{r}(t))]^2. \tag{3.30}$$

T is the length of the prediction phase. NCE(λ_1t) stays small and flat for a long time before rising rapidly. One can see this in Fig.(3.11).

Chapter 3, in full, is a reprint of the material as it appears in Jason A. Platt, Adrian S. Wong, Randall Clark, Stephen G. Penny, and H. D. I. Abarbanel. "Robust forecasting using predictive generalized synchronization in reservoir computing". In: Chaos *31 (2021), p. 123118.* URL: *https://doi.org/10.1063/5.0066013. The dissertation author was the primary investigator and author of this paper.*

Chapter 4

Practical Reservoir Computing

Parts of this chapter are adapted from [3].

4.1 Introduction

The previous chapter was an examination of the dynamics of an RC and how GS could be used to understand the function of the RNN. In this chapter, through the examination of architectural and procedural choices for the RC design, we present "best in class" results for a wide range of simple dynamical systems. We then explore the method of localization for scaling the RC to larger spatiotemporally chaotic systems. We note that because the structure of the RC is nearly identical to the canonical RNN [194], many of the findings will be relevant for general RNNs as well.

Although published guides to the general use of RC exist [195] and there have been multiple studies on specific applications of RC with dynamical systems [119], there has been little guidance on best practices for implementing RC for forecasting chaotic dynamical systems. A practitioner must piece together design decisions by drawing on potentially conflicting information from multiple sources. We intend here to point the researcher or practitioner to strategies that have proven more successful, and to avoid those that we show to be 'traps' (*i.e.*, commonly published design decisions that have little or even negative impact on overall performance). Additionally, using ML to forecast chaotic

dynamical systems presents a number of its own unique challenges when compared to standard practices in ML. There is a high potential for misunderstanding by practitioners and developers.

This chapter aims to lay out in clear terms the problem being solved by RC for the specific task of forecasting chaotic dynamical systems, guidance on how to train and test the RC models, as well as the lessons we have learned in applying these methods over the last several years on the specifics of implementations for particular problems.

4.2 RC Theory

4.2.1 Stability

Recalling the earliest works on RCs [104, 121], setting the spectral radius $\rho_{SR} < 1$ is given as a guarantee of the "echo state property" *i.e.*, generalized synchronization and fading memory—see section 4.2.2. Over the years the caveats in the original derivation, for example that the condition holds only when $\mathbf{u}(t) = 0$ is an input, have generally been lost; the claim is therefore that the ρ_{SR} should always be set to a value less than 1. Despite this statement being proved empirically incorrect [185, 196], it is usually one of the first statements one comes across when studying the RC literature. In the following we give an alternative analysis based on dynamical stability theory.

Intuitively, local stability around a fixed point requires that small perturbations do not result in large movements to the state of the system [197]. If the RC were unstable, a small variation in training data would result in vastly different reservoir states, making finding a readout $\mathbf{W}_{\text{out}}$ practically if not theoretically impossible [198]. Therefore finding conditions for local instability can in practice give us conditions on the trainability of the RC.

The fixed points of a nonlinear map $\mathbf{r}(n+1) = F_r(\mathbf{r}(n))$ are solutions of the equation

$\mathbf{r}_\star = F_r(\mathbf{r}_\star)$. Therefore, for our reservoir Eq.(3.6)

$$\mathbf{r}(t+1) = F_r^d(\mathbf{r}(t), \mathbf{u}(t)) = \alpha \tanh(\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{u}(t) + \sigma_b \mathbb{1}) + (1-\alpha)\mathbf{r}(t)$$

we find can find the equilibrium

$$\mathbf{r}_\star = \tanh(\mathbf{A}\mathbf{r}_\star + \mathbf{W}_{\text{in}}\mathbf{u} + \sigma_b \mathbb{1}), \tag{4.1}$$

which has solution $\mathbf{r}_\star = \mathbf{0}$ when $\mathbf{u} = \mathbf{0}$, and $\sigma_b = 0$.

The stability of the system is governed by the eigenvalues of the Jacobian of the map evaluated at $\mathbf{r}_\star$. If the magnitude of the largest eigenvalue is < 1 then the system is stable around the fixed point [199]; one caveat is that local stability does not imply global stability where a perturbed orbit will stay in the neighborhood of the unperturbed orbit. The Jacobian of a nonlinear map is $\mathbf{DF}_r = \frac{\partial F_r}{\partial r}\Big|_{r_\star}$. For the RC, taking the derivative of Eq.(3.6) with respect to $\mathbf{r}$

$$\mathbf{DF}_r^d = \frac{\partial F_r}{\partial r}\Big|_{r_\star} = \alpha \cdot \text{diag}(\mathbb{1} - \tanh(\mathbf{A}\mathbf{r}_\star + \mathbf{W}_{\text{in}}\mathbf{u} + \sigma_{\text{b}}\mathbb{1})^2)\mathbf{A} + (1-\alpha)\mathbb{I}$$

and plugging in $\mathbf{r}_\star = \tanh(\mathbf{A}\mathbf{r}_\star + \mathbf{W}_{\text{in}}\mathbf{u} + \sigma_b\mathbf{I})$ from Eq.(4.1) to simplify, we find

$$\mathbf{DF}_r^d = \frac{\partial F_r}{\partial r}\Big|_{r_\star} = \alpha \cdot \text{diag}(\mathbb{1} - \mathbf{r}_\star^2)\mathbf{A} + (1-\alpha)\mathbb{I}, \tag{4.2}$$

where diag$(\cdot)$ denotes the formation of a diagonal matrix. We can use this equation to visualize the areas of stability of the RC. If we assume $\mathbf{r}^\star \sim 0$ for $\tilde{\mathbf{u}} = \mathbf{W}_{\text{in}}\mathbf{u}$ small then $\tanh(A\mathbf{r}^\star + \tilde{\mathbf{u}}) \sim \tanh\tilde{\mathbf{u}} + \mathbf{r}^\star\text{diag}(\text{sech}^2(\tilde{\mathbf{u}}))\text{A}$ and we solve for

$$\mathbf{r}^\star = \tanh(\tilde{\mathbf{u}})(\mathbb{I} + \text{diag}(\text{sech}^2(\tilde{\mathbf{u}}))\text{A})^{-1}. \tag{4.3}$$

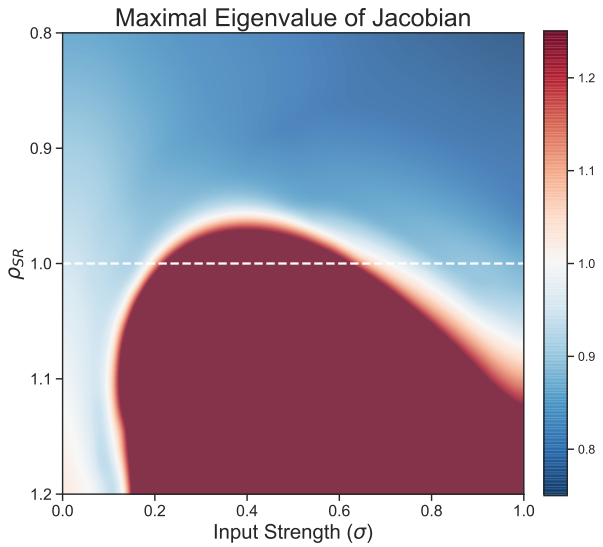

Figure 4.1. The magnitude of the maximal eigenvalue of the Jacobian matrix Eq.(4.2)—$N = 200, \alpha = 0.5, \rho_A = 0.9$. When the eigenvalue of the Jacobian is < 1 then the fixed point equilibrium of the RC is stable; this is a general property of discrete dynamical systems [199]. If the fixed point is stable then orbits of the RC as it is driven around the fixed point will tend to stay in the neighborhood of that point. This would suggest that the RC is trainable (*i.e.*, a "good" $\mathbf{W}_{\text{out}}$ can be found)—see section 4.2.2 for a more general criterion for trainability. An eigenvalue above 1 indicates an unstable fixed point, making the RC sensitive to perturbations (*e.g.*, noise) in the driving signal. The white line indicates $\rho_{SR} = 1$, the generally accepted limit for the spectral radius. This example indicates that one should be careful when assuming a requirement that $\rho_{SR} < 1$.

With the fixed points given we can plot the maximal eigenvalue of $\mathbf{DF}_r^d$. An example is given in Fig.(4.1). In this example, the RC is stable for values of ρ_{SR} that are much greater than 1. This also gives a clear example that for nonzero input, there can be regions where ρ_{SR} is less than one but the RC is unstable.

Having a globally stable RC is most likely necessary, but certainly not a sufficient condition for obtaining a $\mathbf{W}_{\text{out}}$ that gives good predictions. For instance, it has been noted many times that the RC operates best at the "edge of stability"[177, 184, 198] and in general the RC is sensitive to parameter variation. An unstable RC is, however, rather unlikely to have a linear map between $\mathbf{r}$ and $\mathbf{u}$.

LEs in Reservoir Computing

When the LEs of the **autonomous** Eq.(3.9) RC match the LEs of the input data generated from the dynamical system f_u then the RC is said to have "reconstructed" the attractor [124]. A reconstructed attractor is strongly correlated with properties such as the physicality and robustness of forecasts [2]. The LEs of f_u can be estimated from the data with no knowledge of the dynamics [26].

To calculate the LEs of the autonomous reservoir we note that the Jacobian/linear propagator of Eq.(3.9) is

$$\mathbf{DF}_r^a(n) = \alpha \cdot \text{diag}(\mathbb{1} - \tanh[\mathbf{W}\mathbf{r}(\text{n}) + \sigma_{\text{b}}\mathbb{1}]^2)\mathbf{W} + (1-\alpha)\mathbb{I} \tag{4.4}$$

with diag denoting a diagonal matrix and the constant matrix $\mathbf{W} = \mathbf{A} + \mathbf{W}_{\text{in}}\mathbf{W}_{\text{out}}$. This propagator may be used in the algorithms given in Geist et al., [200] using the procedure given in [116, 124] to calculate the LEs.

4.2.2 Generalized Synchronization

To reiterate the main point of chapter 3 and as described in detail by [2, 124], the RC works by making use of the concept of generalized synchronization (GS) [135].

The definition of GS is: for a drive system $\mathbf{u} \in \mathbb{R}^D$ and response system $\mathbf{r} \in \mathbb{R}^N$, they are synchronized if there exists a function ψ such that $\mathbf{r} = \psi(\mathbf{u})$ [135]. Intuitively this implies that the dynamics of the response system are entirely predictable from the history of the input; in RC, for a contracting system, this property is related to the "echo state property" [104] and "fading memory" [164]. The requirement to find a matrix $\mathbf{W}_{\text{out}}$ such that $\mathbf{W}_{\text{out}}\mathbf{r}(t) = \mathbf{u}(t)$ is tantamount to finding a linear approximation to the local inverse of the generalized synchronization function—we call this statement predictive generalized synchronization [2]. We would like to note that the statement of the existence of the inverse of ψ is in general not a global, but a local property due to the necessity of ψ being differentiable [142] for ψ^{-1} to exist. Grigoryeva et. al., [201, 202] details general conditions for differential GS to exist in an RC. When ψ does not meet these criteria then the RC is untrainable; this statement is connected to the stability criterion explored in section 4.2.1.

The concept of GS is useful for visualizing the operation of the RC. When $\mathbf{u}(t)$ and $\mathbf{r}(t)$ are synchronized, the combined state $\mathbb{R}^{N+D}$ will lie on an invariant *synchronization manifold* $\mathbf{M}$ [168]. This manifold is generally low dimensional [2] and so the dynamics of the RC are taking place on this low dimension structure in phase space. In addition, the concept of synchronization shows that there is a certain amount of "spinup" time needed for the transient initial conditions to contract to the synchronization manifold. The amount of time needed for spinup is directly related to the strength of the conditional Lyapunov exponents (CLEs) [203] of the synchronization manifold. The statement of stability of the motion on the manifold [168, 169] is that the contraction normal to the manifold must be larger than the contraction tangential to it [170], thus giving a statement of the stability of the RC.

The CLEs and the constraint that they be negative is in fact closely related to the stability conditions we discuss in section 4.2.1. If the RC is unstable everywhere, $\mathbf{DF}_r^d > 1$, then the CLEs will always be positive and the RC will not synchronize with the input data. As a last note, it has been observed that RCs work best when at the edge of stability

[177, 184, 198]; this edge corresponds to the largest CLE being ~ 0 and $\mathbf{DF}_r^d \sim 1$ [2].

4.3 Data Generation for Numerical Experiments

We begin with a discussion of the training and testing data used in the following numerical experiments. The use of input data is different for dynamical systems forecasting than for the typical ML use case. Typically in ML there are standard training and testing datasets *e.g.*, MNIST [204], that one can acquire in order to directly test newly proposed models against published results from previous models [205]. For dynamical systems forecasting we instead have certain standard models from which one can generate data. Therefore it is important to make sure one is observing best practices when constructing ones own training, validation and testing sets.

We would like to construct independent training, validation and testing data for training and testing our ML models. In order to be confident in our results, the validation/testing data must be independent from both one another and the training data. Furthermore, we would like the testing data to sample as much of the phase space of f_u as possible, so that we can be confident that the RC generalizes to parts of the input with different properties than which it has been trained on. All data have been generated with these principles in mind Fig.(4.2).

Additionally, we report forecast times in terms of the Lyapunov timescale of the input dynamical system. This timescale $\tau_\lambda = 1/\lambda_1$, gives the natural timescale of error growth in the system and is therefore a useful measure of how "good" the forecast is. Not even a perfect model can predict a chaotic dynamical system forever—numerical errors will eventually cause divergence—and therefore τ_λ gives us a metric of a reasonable timescale for prediction.

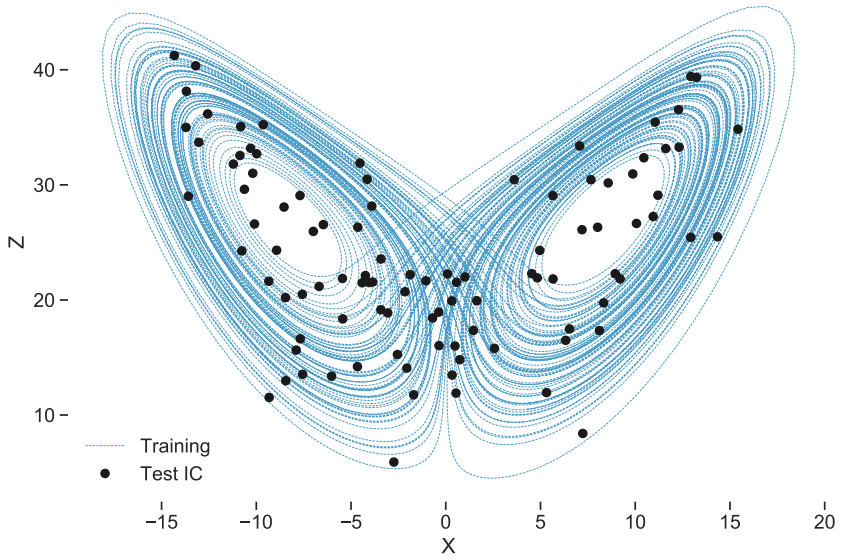

Figure 4.2. Training data (blue) and test forecast initial conditions (black) for the L63 system. For the training data it is important that the data cover most parts of the input attractor. This ensures that the RC does not overfit a local section of the dynamics and that it can generalize. The difference in dynamics over the attractor can be formalized by the finite time Lyapunov exponents (FTLE) [26, 30], which denote the localized error growth rates. These can be positive or negative for the L63 system depending on the initial condition. Likewise for the testing data, the initial conditions for the test forecasts should widely sample the attractor. It is important that the testing data be long term forecasts and not one-step forecasts in order to capture the broadband frequencies of the chaotic dynamics.

4.3.1 Generative Models

The dynamical models used for data generation have been chosen for their applicability to certain use cases as well as their ability to test particular abilities of reservoir computing. The Rossler system and Colpitt's oscillator are both 3 dimensional systems with long time scale dynamics—*i.e.*, τ_λ large, slow error growth rates. The Colpitt's oscillator is particularly interesting because its dynamical equations of motion include an exponential term, making it the most nonlinear of the models studied here. The degree of nonlinearity of the dynamics can create greater difficulty for prediction with a linear readout operator. The Lorenz attractor (L63) is another 3 dimensional system but with shorter time scale dynamics than Rossler or Collpitts. The Lorenz 1996 system can be defined with arbitrary size so we can use it to look at how our RC scales with system dimension. Finally the Climate Lorenz Model (CL63) is composed of three L63 systems coupled together with multi-timescale dynamics. RCs tend to focus on a particular time scale [101] so it is important to study multi-timescale dynamics for certain applications such as in a weather prediction context [15, 206]. See the appendix for equations and parameters.

Table 4.1. Dataset timescales. All dynamical systems we consider are made dimensionless, so the "time" here is number of dimensionless steps into the future.

Dataset	Largest LE λ_1	Time Scale $\tau_\lambda = 1/\lambda_1$
Rossler	0.065	15.4
Colpitts	0.07	14.3
L63	0.9	1.1
L96-5D	0.4	2.5
L96-10D	1.1	0.9
CL63	0.9	1.1

4.3.2 Forecast Testing Metric

A common ML benchmark for time series prediction is the one-step prediction mean square error Eq.(3.7). Indeed this is exactly how we train the micro-scale parameters in $\mathbf{W}_{\text{out}}$ to obtain the linear map from $\mathbf{u}(t) \to \mathbf{r}(t) \to \mathbf{u}(t+1)$. As a method for testing, however, one-step prediction is not an appropriate metric; it overemphasizes the high frequency modes in the data at the expense of the long-term stability of the RC forecast.

As shown by Platt et al.,[2], the metric that best represents the ideal behavior of the RC—good short-term predictions coupled with "physical" (the forecast RC stays on the synchronization manifold $\mathbf{M}$) long-term forecasts—is best represented by the reproduction of the input system's LEs. In practice the LEs can be difficult to calculate from high-dimensional experimental data and computationally expensive for the RC. Therefore, a proxy for the reproduction of the LEs has in practice been long-term forecasts [2].

Our standard metric for forecast time is the valid prediction time (VPT) [116]. The VPT is the time t when the accuracy of the forecast exceeds a given threshold. For example,

$$\mathrm{RMSE(t)} = \sqrt{\frac{1}{\mathrm{D}} \sum_{\mathrm{i}=1}^{\mathrm{D}} \left[\frac{\mathrm{u_i^f(t) - u_i(t)}}{\sigma_{\mathrm{i}}} \right]^2} > \epsilon, \tag{4.5}$$

where D is the system dimension, σ is the long term standard deviation of the time series, ϵ is an arbitrary threshold, and u^f is the RC forecast. For the results shown in subsequent section a total of 200 test initial conditions are used to report a distribution of VPTs with ϵ set arbitrarily to 0.3. [116] use $\epsilon \sim 0.5$ in their reported results.

4.4 RC Training

Training the RC model can be separated into training the macro-scale parameters—*e.g.*, spectral radius, leak rate—and the micro-scale parameters *i.e.*, the elements of $\mathbf{W}_{\text{out}}$. Much of the early guidance on RC training [195] included advice on the intuition behind

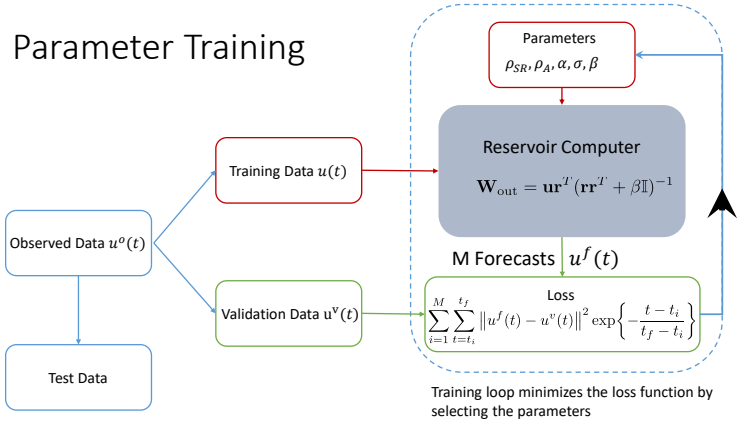

Figure 4.3. Two stage parameter optimization showing the training procedure for the RC. The routine take in as input the data $\mathbf{u}(t)$ which is used for training and comparing the forecast. Parameters for the RC are then chosen and the proposed RC forecasts M times producing $\mathbf{u}^f$. $\mathbf{u}^f$ is then compared directly to the data with the exponential term introduced to countermand the natural error growth due to positive LEs. The cost is then used to inform the next selection of parameters.

hand-tuning the macro parameters for a particular problem as hyperparameters. For instance, a widely cited guideline is that the spectral radius must be below 1—see section 4.2.1—but that is based on many assumptions that do not necessarily hold for a particular RC application. Modern optimization algorithms and increased computational power have decreased the need to hand-tune all of these macro parameters as hyperparameters. Instead, they can be incorporated directly into the objective function for training via optimization. For intuition on the effect of these parameters on the RC we refer the reader to Lukovsevivcius [195].

We employ a two step optimization approach, described by [4, 125] and illustrated in Fig.(4.3) where one first fixes the macro-scale parameters, trains the micro-scale parameters through linear regression Eq.(3.8) and then tests the completed RC on a series of long forecasts. For the macro loss function we have had success using the scaled error

$$\mathscr{L}_{\text{macro}} = \sum_{i=1}^{M}\sum_{t=t_i}^{t_f} \left\|u^f(t) - u(t)\right\|^2 \exp\left\{-\frac{t-t_i}{t_f-t_i}\right\},$$

with M being the number of forecasts used for comparison. We reiterate that the training for the macro-scale parameters does not use the one-step-ahead prediction error because that is not a good proxy for the reproduction of the LE spectrum. The exponential weighting in 4.4 is added to approximately offset the exponentially growing error in the forecast due to positive LEs.

Penny et al., [4] (Fig.12) examined the effect of increasing numbers of M on the cost function landscape and showed that increasing M smooths out the loss function landscape and enables better convergence to the global minimum. However, a compromise is necessary as increasing M will substantially increase computational complexity. Griffith et al., [125] showed that using $M = 1$ causes the RC to be overly sensitive to unstable sets in the dataset. We have empirically found that using 15-20 or so forecasts is generally enough for these simple models, provided that those forecasts are made at statistically independent points. Ideally the data would well sample the input system attractor, but in practice that cannot always be the case.

We set the random seed for the reservoir instantiation during the optimization to reduce the noise in the optimization function. After training, however, we have not found the random seed leading to substantially different prediction times over a large number of test samples—see Fig 7 in [2] for an illustration. This does not mean that the prediction for any individual test will be exactly the same but that on aggregate the randomness does change the mean of the distribution substantially. This result may be different in the "small" reservoir regime of $N <\sim 100$.

4.4.1 Bayesian Optimization

Rather than hand-tune each macro-scale parameter as a hyperparameter, we suggest instead using an optimization procedure such as the two step optimization routine described above. While any global search algorithm can be used *e.g.*, differential evolution, we use a Bayesian optimization technique using surrogate modeling. Bayesian optimization is

a protocol for optimizing expensive nonlinear functions that works by "minimiz[ing] the expected deviation from the extremum" of a target loss function [207]. We have found success with the efficient global optimization algorithm of Jones et al., [208] as implemented by [209]. This algorithm evaluates the loss function over a number of sampled points and then a Gaussian process regression is fit to the surface produced. One can then optimize over this interpolated "surrogate" model to identify promising regions of parameter space and then iterate by reevaluating the loss function in these regions. Given that we need only train a few macro-scale parameters, this surrogate optimization technique renders the optimization problem computationally tractable.

With these algorithms, one can jettison the hand-tuning of hyperparameters and simply optimize all the macro-scale RC parameters for a given problem. This procedure produces good performance for a wide range of simple dynamical models Fig.(4.4) in a reasonable amount of time.

We will now go into greater detail on the choices that must be made outside of the optimization routine (by definition hyperparameters) and how they can affect the solution to the RC problem. The experiments are conducted by fixing/varying a particular hyperparameter or macro-scale parameter from table 1 and then optimizing over the rest of the macro-scale parameters. It will be stated if we are not optimizing over the un-varied parameters.

4.4.2 Reservoir Dimension

The reservoir dimension N of the RC is related to the memory capacity of the network [121, 195]. Larger reservoirs do tend to produce more accurate forecasts with longer VPTs than smaller networks. However, there are diminishing returns where ever larger reservoirs are needed for ever smaller improvements in predictability. For example, consider the L63 system in Fig.(4.5). Using a reservoir size of just 250 nodes, we see a VPT only a couple Lyapunov time scales less than when using a reservoir of size 2000.

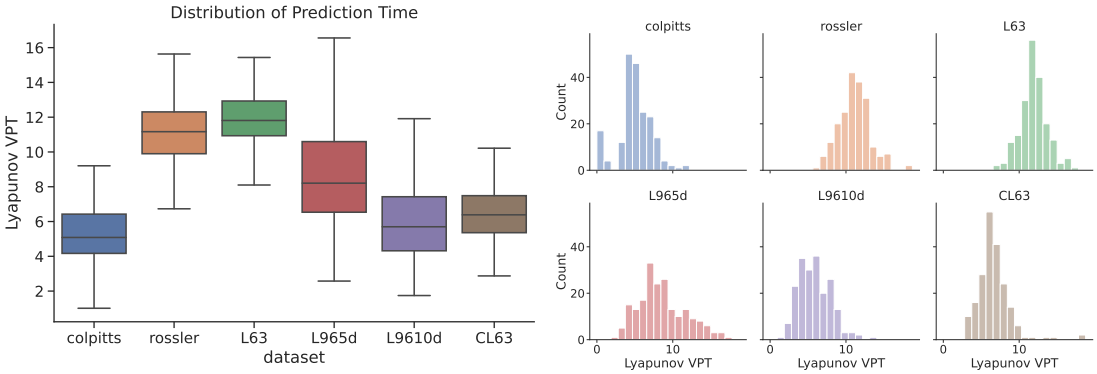

Figure 4.4. **(left)** Valid prediction time for the best fit parameters of the RC over different datasets. Note the distribution of forecast times. The RC predicts well on all the models given to it. Results are shown for $N = 2000$, results for $N = 250$ are shown in Fig.(4.5).
(right) Histogram view of the distribution of VPT for the different models. The distributions are close to Gaussian but with heavier tails. The outliers correlate with the FTLEs of the input system. When the FTLEs are low negative values, meaning the input data is very stable, then prediction times can be extremely long and vice versa. While some of the variability in prediction time is caused by randomness and training in the RC, other variability is intrinsic to the dynamics we are attempting to predict.

Therefore if our application only requires a few Lyapunov time scales of prediction skill, then it is possible to use a much smaller and more computationally efficient RC.

Because optimization routines are far more computationally intensive than forecasting with the RC, it is practical to ask whether one can optimize/train using a smaller reservoir and then scale up to larger reservoir to generate a forecast. The answer is that one can indeed train a small reservoir and then use those same macro-scale parameters with a larger reservoir to see substantial improvement in prediction skill. This strategy was used by [4]. However, an RC optimized at the larger size will tend to produce more accurate forecasts. If computational resources allow, it is appropriate to perform a final optimization using the reservoir size that will be used for the production RC. The left side of Fig.(4.5) shows the effect of increasing the reservoir dimension without re-optimizing parameters. The RC performance tends to saturate as the reservoir dimension increases.

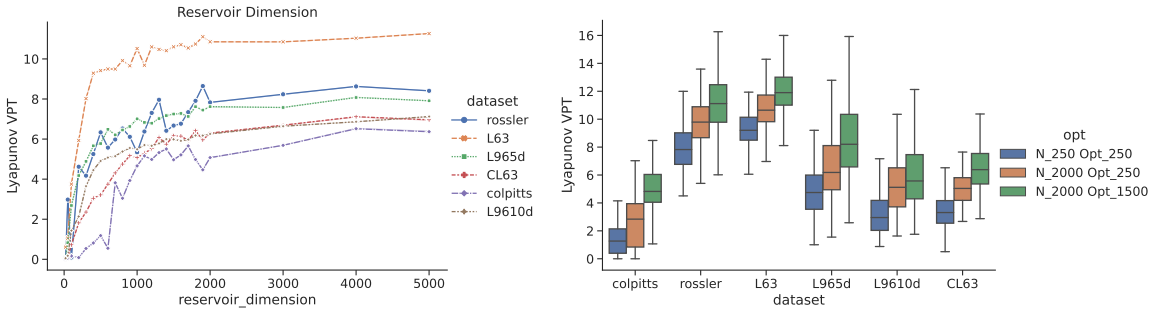

Figure 4.5. (left) The effect of increasing reservoir dimension without reoptimization of the global parameters. RCs optimized for $N = 2000$.
(right) VPT for a 3 different RCs 1) small $N = 250$ RC optimized at $N = 250$ 2) an RC with identical global parameters to 1, but with a dimension scaled up to $N = 2000$ 3) an RC with $N = 2000$ optimized for $N = 1500$.

4.4.3 Spinup Time

The spinup time is the time it takes for a trained RC to converge from its initial condition (usually either set to zero or randomly chosen) onto the synchronization manifold to which it is driven by the input data. The number of steps needed for spinup is related to:

1. How far the initial condition of the RC is from the synchronization manifold.

2. The CLEs of the generalized synchronization manifold; these are related to the "echo state property" and the spectral radius of the adjacency matrix $\mathbf{A}$.

3. The RC approaches the synchronization manifold asymptotically so while the error tolerance is also important, it probably does not need to converge to within 10^{-12}. Convergence during spinup and error growth during prediction are both governed by $\delta x(t) \approx \delta x_0 e^{\lambda t}$ with λ giving the negative CLEs for spinup or the positive LEs for forecasting.

In practice it is easy to tell empirically how much time is needed—see Fig.(4.6)—usually for these small models only $\mathcal{O}(10)$ time steps are necessary. For larger systems the spinup time can be significantly higher.

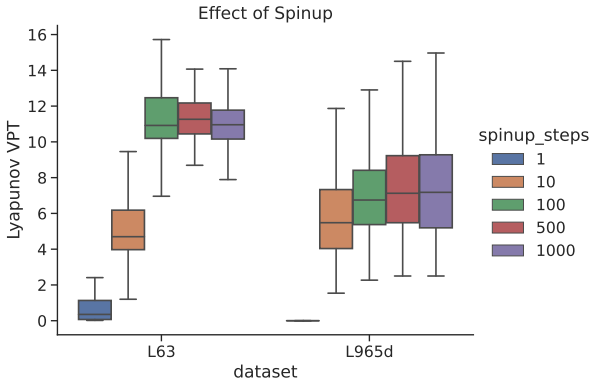

Figure 4.6. Effect of setting the number of spinup steps on the VPT of the RC for L63 and L96-5D. Longer spinup times reduce the initial error of the forecast as the RC and input data synchronize together.

4.4.4 Input Bias

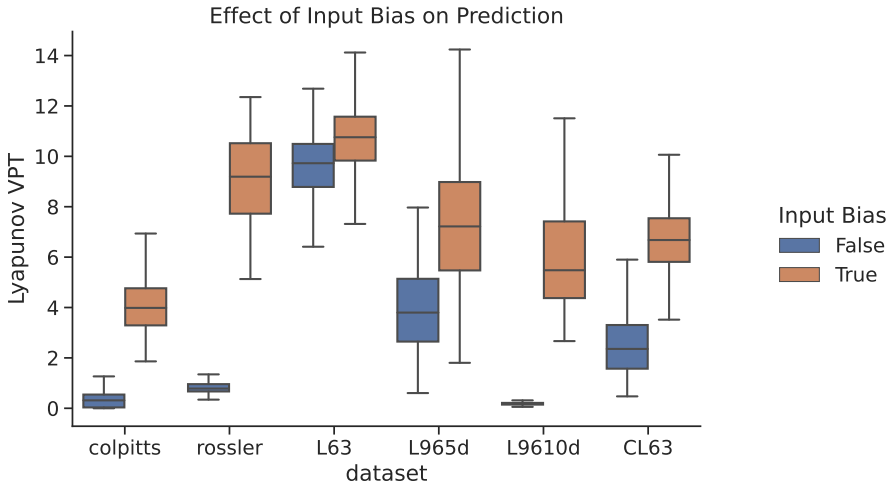

Figure 4.7. There are two sets of parameters: 1) no input bias 2) input bias. Both sets of parameters are optimized separately but with the same reservoir size. The readout is linear for both. The input bias is a constant σ_b, which is optimized Eq.(3.6). σ_b sets the nonlinear regime of the tanh. When $\sigma_b = 0$ the optimization is not able to find any parameters that work for many of the models. However, when σ_b is optimized all models produce accurate and reliable forecasts.

The input bias σ_b can have a significant impact on RC prediction skill, but it has consistently been left out of most published RC studies focused on forecasting chaotic dynamics. We see in Fig.(4.7) that if we do not include this term in the RC when optimizing

macro-scale parameters then for almost all models the forecast skill is negligible. The input bias sets the extent of the 'nonlinearity' in the reservoir Fig.(4.8); $\sigma_b = 0$ corresponds to the reservoir operating mostly in the linear part of the tanh function around 0, while a high bias term corresponds to the RC operation being in a more nonlinear part of the regime.

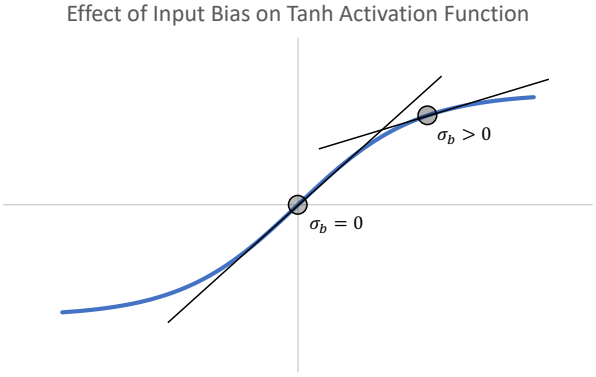

Figure 4.8. Effect of input bias σ_b on tanh activation function. When $\sigma_b \approx 0$, the $\tanh(\cdot)$ activation function operates mostly around 0 in the "linear" regime of the activation function. A nonzero σ_b will push the operating point to the nonlinear regime of the $\tanh(\cdot)$ activation function, possibly giving more expressive results. The bias also affects the sensitivity of the RC to input. Around the origin the RC has maximal gain, while a high bias term will reduce the gain shown in the figure by the slope of the lines.

4.4.5 Readout Functions: Linear, Biased, Quadratic

We generalize the readout $\mathbf{W}_{\text{out}}\mathbf{r}$ to $\mathbf{W}_{\text{out}}Q(\mathbf{r})$ for some function Q. An example in the literature is by Lu et al., [124] who use a quadratic readout function $Q(\mathbf{r}) = \left[\mathbf{r}(t), \mathbf{r}^2(t)\right]$. As another example, [195] recommended a biased output where $Q(\mathbf{r}, \mathbf{u}) = \left[\mathbf{r}(t), \mathbf{u}(t-1)\right]$ so the previous input is fed directly into $\mathbf{W}_{\text{out}}$. We have tested a number of options and found that the readout does not have much impact on the quality of the forecasts made by the RC Fig.(4.9) as long as the parameters are well optimized and the input bias σ_b is not set to 0. While these results seemingly contradict those reported and used in the previously cited studies, we note that when the bias term is not included—see Fig.(4.7)—

then changing the readout layer does indeed improve the predictions. We speculate that the readout in this case is injecting a certain amount of nonlinearity, essential for good predictions, into a system that is operating around the linear regime of the activation function.

In addition to the injection of nonlinearity into the RC, the improved performance of the RC with input bias may be due to the breaking of the symmetry of the reservoir equations [210]. Herteux et al. showed that for the simple Lorenz 63 system, the symmetry of the equations leads to the formation of a "mirror attractor" that causes divergence of the RC off the synchronization manifold and onto an alternate trajectory. Adding either input bias or a quadratic readout resolved the issue. These results were extended by [211] to show that, espeicially for small reservoirs, breaking the symmetries of the input data can be extremely important for prediction. Therefore, while we do not see any change in prediction for the different readout functions once input bias is added, perhaps for small reservoirs ($<\sim 100$) a nonlinear readout could become important.

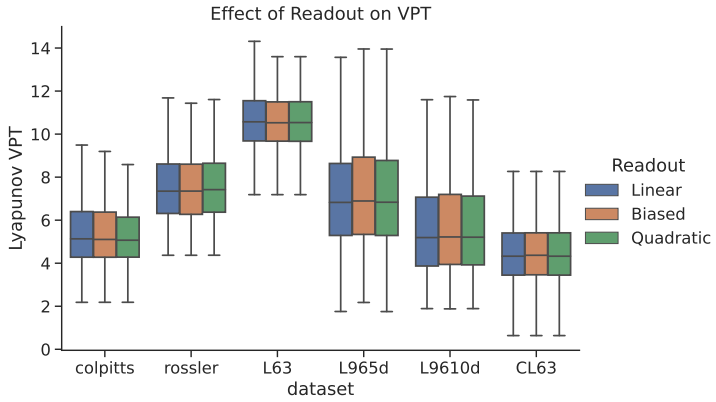

Figure 4.9. Predictions for 3 different kinds of readout methods. Linear: $\hat{\mathbf{u}}(t) = \mathbf{W}_{\text{out}}\mathbf{r}(t)$, Biased: $\hat{\mathbf{u}}(t) = \mathbf{W}_{\text{out}}[\mathbf{r}(t), \mathbf{u}(t-1)]$, Quadratic: $\hat{\mathbf{u}}(t) = \mathbf{W}_{\text{out}}[\mathbf{r}(t), \mathbf{r}^2(t)]$.

4.4.6 Amount of Training Data

The data are used differently in the two steps of the training routine—see Eq.(4.4). The data should be separated into the M long forecasts that are used to compute the loss

$\mathscr{L}_{\text{macro}}$ and the data used to train $\mathbf{W}_{\text{out}}$ in Eq.(3.8). More and longer test forecasts for the global loss will help guarantee 1) the generalizability of the RC to segments of the data the RC might not have seen, 2) stability of the trained RC, and 3) the optimality of the global parameters found during training. The amount of data that is used for the micro-scale loss function governs the accuracy of the linear map $\mathbf{W}_{\text{out}}$. Experiments indicate diminishing returns, where increasing the size of the training data leads to diminishing improvements in forecast skill Fig.(4.10).

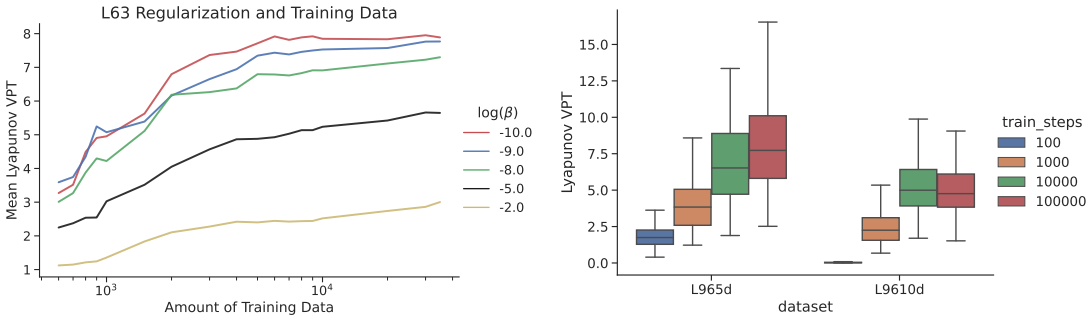

Figure 4.10. (left) L63, RC N=250 for different values of regularization without reoptimizing all the parameters. Diminishing returns with increasing amounts of data. (right) Here we re-optimize the RC for each amount of training data. We still see a clear reduction in the rate of increasing skill as the amount of training data increases.

4.4.7 Normalization

Normalization is a standard practice in ML. Applying normalization is usually among the first suggestions to anyone analyzing a new dataset [97]. We must emphasize, however, that when forecasting dynamical systems the standard methods for normalizing datasets can have deleterious effects. To illustrate, we introduce two normalization schemes,

1. Normalize each variable separately by subtracting the mean and dividing by the standard deviation

$$u_i^{\text{norm}} = \frac{u_i - \text{mean}(\text{u}_\text{i})}{\text{std}(\text{u}_\text{i})};\ i \in \Big[1, 2, \ldots, D\Big].$$

2. Normalize the variables by the joint mean and max/min of the variables

$$u_i^{\text{norm}} = \frac{u_i - \text{mean}(\mathbf{u})}{\max \mathbf{u} - \min \mathbf{u}} \; i \in \Big[1, 2, \ldots, D\Big].$$

We assume that the variables in the dynamical system have been nondimensionalized. The statistical functions mean, max, and min are calculated from all the training data and thus represent "climatological" statistics. The VPTs resulting from both normalization schemes are shown in Fig.(4.11).

As shown in Figure 4.11, normalization scheme 2 vastly outperforms scheme 1. Our hypothesis is that normalizing each variable separately (particularly subtracting the mean) destroys the cross-variable information necessary to reconstruct the relationships in the dynamical system. There is a significant amount of information stored in the relationships between the variables as well as their magnitudes. The second scheme compensates for the normalization via σ and σ_b to preserve all the information in the data while the first scheme destroys that information.

It is possible that we could recover the prediction skill by a skillful selection of $\mathbf{W}_{\text{in}}$. If we optimized the inputs (σ) for each column of $\mathbf{W}_{\text{in}}$ separately as well as σ_b, that would help the RC to compensate for the normalization. However, this introduces additional global parameters that must be optimized.

Generally, we recommend not normalizing the variables when the data is nondimensionalized, as is the case for the dynamical models examined here. We note that [4] used no normalization and still produced successful predictions with the RC models. In a realistic setting, however, when the data is collected from sensors and there are many different unit scales, one has to do some kind of normalization in order for the methods to be numerically stable. We discuss this point further with a simple example in appendix 4.7.1. Because we have shown that normalization can be detrimental when using data to forecast dynamical systems, we emphasize that care must be taken when normalizing any

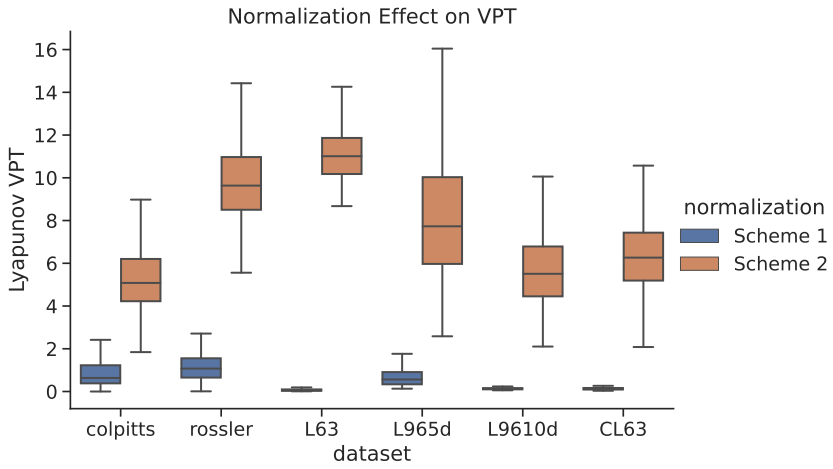

Figure 4.11. Results from normalizing by the two different schemes. The results from scheme 1 can actually be improved slightly by adding a small $\sim 1-2\%$ amount noise to the data—a phenomenon described as well in [116]—but the improvement is marginal. While we are not advocating for a particular normalization scheme, particularly for already nondimensionalized data, it is important to keep in mind that there is essential information in the relationships between the time series that can be destroyed by introducing normalization.

data prior to training.

4.4.8 Effect of Noise

The RC is quite sensitive to additive noise Fig.(4.12). There is a sharp decrease in prediction quality with even a small amount of noise, meaning that any slight perturbation to the training of $\mathbf{W}_{\text{out}}$ causes the errors to multiply rapidly. This is consistent with our observation that even casting double precision floating point numbers in $\mathbf{W}_{\text{out}}$ to single precision can cause similar degradation.

We must address a phenomenon reported by [116] that sometimes a small amount of noise can help predictions. This observation held true when we normalized the data using scheme 1 rather than scheme 2—see the previous section. We note, however, that even with the small observed increase in predictive skill, the VPT was still far below that reported with the scheme 2. While [116] speculated that the noise could regularize the data, we find this is not true in general.

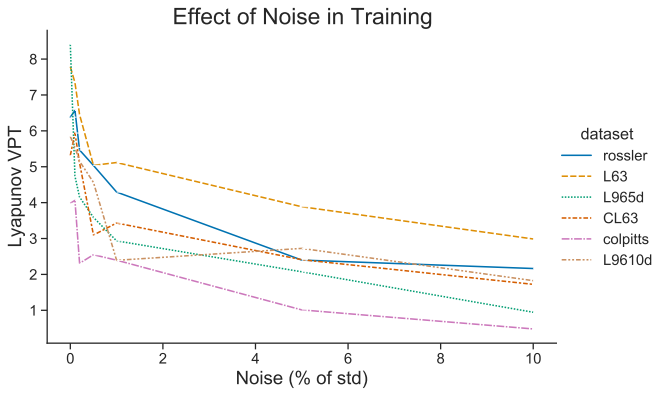

Figure 4.12. Mean VPT as a function of additive Gaussian noise expressed as a percentage of the long term standard deviation of each dynamical variable. The parameters were optimized for each noise level.

Sparse in Time

RC can be quite sensitive to the time step of the input data *i.e.*, how sparse the measurements are in time. In Fig.(4.13) we tested the RC for different time steps Δt, reoptimizing the RC each time. The total time of the training data $T = L\Delta t$ for L number of steps, did not change. We call attention to the result that finer training data does not always increase the prediction time of the RC and that certain models had an optimal time step. In practice one may want to interpolate the input data to try to match the optimal time step.

4.4.9 Sparsity of the Adjacency Matrix A

The sparsity of the adjacency matrix in general makes very little difference to the predictive capability of the RC—this is described by [195] and matches our experience. The benefit of using a very sparse matrix comes from taking advantage of sparse matrix representations in scientific computing software packages to vastly increase computational speed. We have found a set value of 98% or 99% sparsity, $\rho_A = 0.01$ to be sufficient for all applications, provided the resulting matrix is full rank.

Of course there is a limit to this guideline—if the matrix is too sparse (*i.e.*, has no

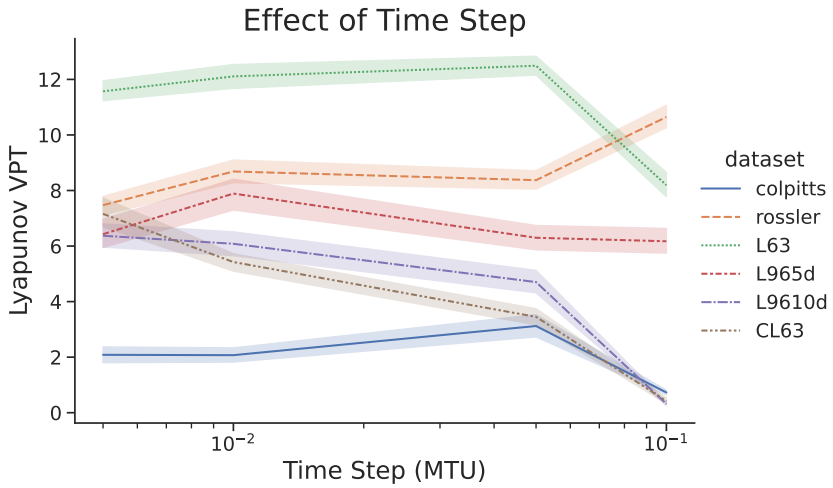

Figure 4.13. VPT as a function of the time step of the RC. The shading denotes the 95% confidence interval for the mean of the distribution. Note the decrease in VPT as the time step increases for the L9610d and CL63 systems and increase for the Rossler system.

nonzero values) then the RC will not work. In addition, when using small reservoirs of say less than N=100 then the randomness of the selection of $\mathbf{A}$ can have a significant impact on the quality of predictions [101]. Slightly larger reservoirs seem to be more robust to random variations in connectivity.

4.5 Scaling to Higher Dimensional Systems

Up to this point we have considered systems with a state dimension $D \leq 10$, and a single RC model has been sufficient to predict their time evolution. However, systems with larger state spaces will require larger reservoirs to make adequate predictions. We highlight this situation in Figure 4.14, which shows the VPT of RC models with increasing reservoir dimension on the Lorenz96 system with 40 nodes. Note that we use a leading Lyapunov exponent of $\lambda_1 \simeq 1.68$ to represent the timescale for this system following [116]. There is no prediction skill until a certain minimum reservoir size is attained, somewhere between 4,800 and 6,000. For systems with even larger state spaces, the reservoir size will have to increase beyond the amount of Random Access Memory (RAM) available and parallelization schemes will have to be considered.

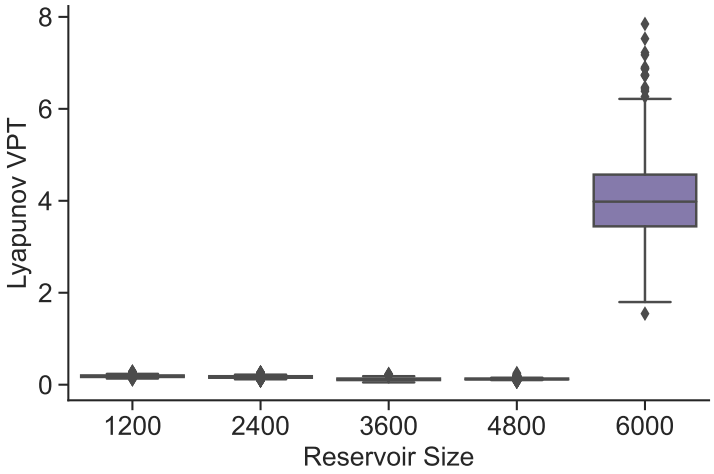

Figure 4.14. VPT from predictions of the 40D Lorenz96 system with a single RC and no localization. Parameters are optimized for each reservoir size. Parameter values are given in [3].

To address these issues, [110] introduced a parallelization strategy which has been used in numerous high dimensional applications, e.g. [4, 116, 126]. In this localization scheme, multiple reservoirs work semi-independently, each making predictions of a subset of the system state. More precisely, the elements of the system state $\mathbf{u} \in \mathbb{R}^D$ are split up into N_g groups, such that each group is assigned an individual RC model that predicts N_{output} of the state vector nodes, with $D = N_g N_{\text{output}}$. At each time step, each group receives N_{halo} of the neighboring state vector values, such that a system with a single spatial dimension has an input dimension for each RC model of $N_{\text{input}} = 2N_{\text{halo}} + N_{\text{output}}$. With this architecture one must decide how to choose N_{output} and N_{halo}. Our goal in this section is to explore these choices within the framework outlined in Sections 4.3 & 4.4.

4.5.1 Varying the halo size and output dimension

Figure 4.15(a) shows the performance of localized RC models with varying output size, N_{output} and halo size, N_{halo}. For output dimensions 2 and 4 (color), we see that the RC model shows its best performance when the halo size is 2. These RC models show no prediction skill with halos smaller than 2, and exhibit diminishing performance as the halo

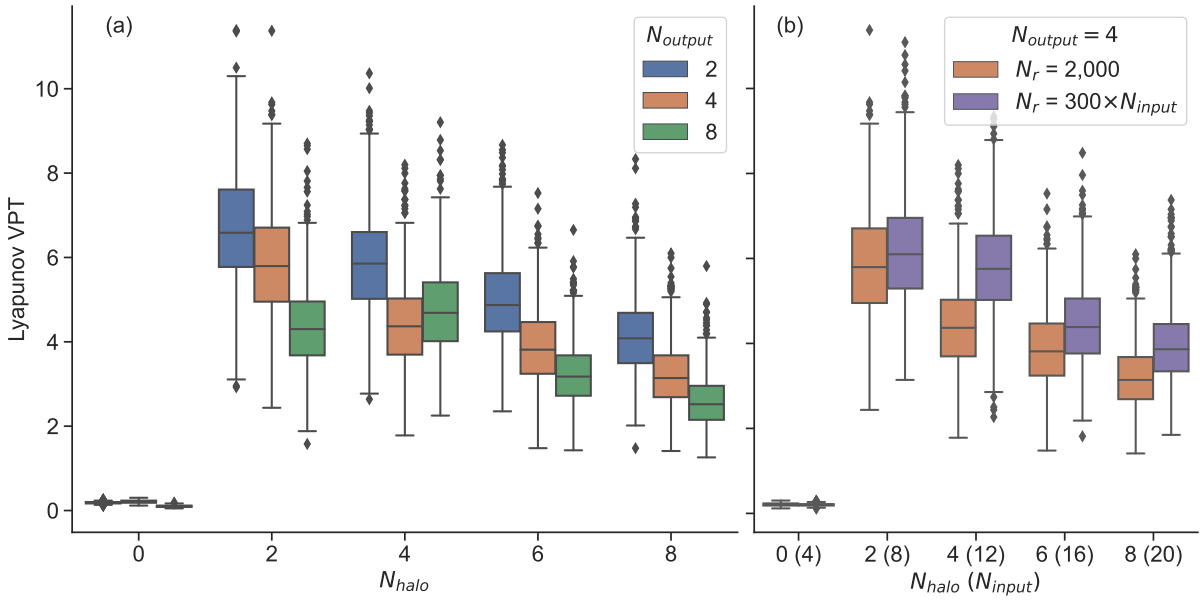

Figure 4.15. VPT of localized RC model predictions for the 40D Lorenz96 system. (a) Performance across various N_{output} (color) and N_{halo} values (x-axis). The reservoir size is fixed at $N_r = 2,000$. (b) Performance for $N_{\text{output}} = 4$, with the reservoir size fixed at 2,000 (orange) and reservoir size increasing with N_{input} (purple). Each histogram is generated from 1,000 samples with parameters optimized for the given reservoir size, output dimension, and halo size. Parameter values are given in [3].

size increases beyond this point. These results suggest that for this particular system, the optimal configuration has a halo size of 2. For the Lorenz96 system, this is unsurprising since we know that the time evolution for each individual node requires information of neighboring states up to 2 nodes away, equation (3.18). For halo sizes smaller than 2, the known interactions between neighboring grid points are not represented. As the halo size increases, the RC model must effectively 'learn' the length scale across which the underlying dynamical interactions occur.

The results suggests that the optimal halo size should be set to a "minimum length scale" relevant to the time evolution of the dynamical system. This value is obviously system dependent, and will require knowledge of how the training data are acquired. If the data arise from a numerical model, the halo size could be related to the numerical stencil used in time integration (e.g. finite difference or finite element scheme). In the case

of observed data, approximating the minimum length scale will require knowledge of the system dynamics.

As the output dimension for each local reservoir increases we see a similar effect as increasing the halo size: the prediction skill decreases. Taken together, prediction skill is optimal when the reservoir "stencil" is as small as possible: each local RC model is trained with exactly the amount of information it needs. This result is in some ways a restatement of [110] (their Figure 5b), who show that prediction performance improves as the number of local reservoirs increases with a fixed system dimension. Using more local reservoirs requires more computational resources, and the benefits must be weighed against the increase in computational cost.

4.5.2 Increasing the reservoir size with input dimension

The previous section showed that prediction skill decreases as the input dimension increases. Here, we test the impact of increasing the reservoir size in order compensate for this reduction in performance. Figure 4.15(b) shows the result of this test for $N_{\text{output}} = 4$, comparing performance with a fixed reservoir size $N_r = 2,000$ (orange) against models where $N_r = 300N_{\text{input}}$. For $N_{\text{halo}} = 4$, the median VPT increases by 1.4 (32%) as a result of increasing the reservoir size from 2,000 to 3,600 (80%). For larger values of N_{input}, the payoff from increasing the reservoir size is even smaller. For example, $N_{\text{halo}} = 6$, the median VPT increases by 0.6 (15%) but requires a reservoir size that is 4,800 (140% larger). The results highlight what is shown in Section 4.4.2 (Figure 4.5). After a certain level of performance is obtained for a given problem size, increasing the reservoir size provides diminishing returns for prediction skill.

4.5.3 Input bias at scale

Finally, we emphasize the importance of the bias term in the RC architecture, which is accentuated in high dimensional systems. Figure 4.16 compares the performance

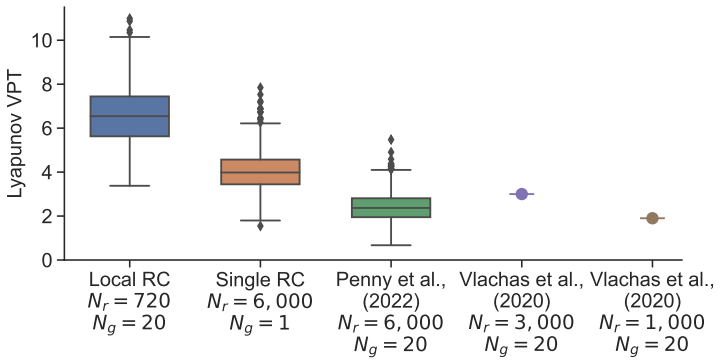

Figure 4.16. VPT of various RC based predictions of the 40D Lorenz96 system. The Local RC (blue) has the configuration $(N_{\text{output}}, N_{\text{halo}}, N_r) = (2, 2, 720)$. The Single RC (orange) uses $N_r = 6,000$ (Figure 4.14). The green box plot shows results from the localized RC Model 3 in [4], which uses $(N_{\text{output}}, N_{\text{halo}}, N_r) = (2, 4, 6000)$. The purple and tan dotted lines indicates the VPT from [116], which use $(N_{\text{output}}, N_{\text{halo}}, N_r) = (2, 4, 3000)$ and $(N_{\text{output}}, N_{\text{halo}}, N_r) = (2, 4, 1000)$, respectively. We note that the VPT computed by [116] used a threshold of 0.5, so we estimate their VPT based on a threshold of 0.3 to match our results. Additionally, [116] compute VPT based on the average NRMSE evolution from 100 sample points, while we compute a histogram of VPT based on each NRMSE from 1,000 sample points. Both the Local RC and Single RC models use an optimized nonzero input bias σ_b, while the other RC designs use $\sigma_b = 0$. Parameter values for the first three models are given in [3]. The amount of training data for the $N_r = 720$ case is $40,000$ with a time step of $dt = 0.01$ which compares to $500,000$ data points in [116].

of two RC models that use an optimized value for σ_b ("Local RC" and "Single RC") against two RC models that use $\sigma_b = 0$ (from [4] and [116]). The RC configuration from [4] uses $N_g = 20$ local reservoirs, each with an output dimension of 2, halo size 4, and reservoir size of 6,000. Even with all other parameters optimized, the VPT is roughly half that of a single RC model with reservoir size 6,000 when an optimized (nonzero) bias term is used. When 20 local reservoirs are used with an optimized bias term and $(N_{\text{output}}, N_{\text{halo}}) = (2, 2)$, the VPT is approximately tripled.

Penny et al. (2021) [4] found it necessary to increase the local reservoir dimension to 6,000 in order to attain reasonable prediction skill. The results in this section show that this was essentially a brute force solution to overcome an inactive bias term. This is an important takeaway for forecasting high dimensional systems, which may have such a large number of localized RC models that increasing the reservoir size to overcome this inactive bias term would become prohibitive.

4.6 Conclusion and Discussion

Reservoir computing is a powerful machine learning method that can be used to successfully predict chaotic time series data. While not as flexible as other RNN methods, the RC has a number of properties that make it a good method of choice for these kinds of tasks. The ability to set the macro-scale properties of the network and the quick training of the micro-scale parameter through linear regression couple together to allow the RC to be trained and deployed quickly and easily. Additionally, the RC has been shown not only to give good predictions but also to react like a physical model to perturbations in the system state (shown as the reproduction of the LEs) [2, 4, 124]. This property is crucial for applications in numerical weather prediction and other fields where it is important to produce both a forecast as well as an uncertainty estimate of that forecast. RCs can also be scaled up to large systems while being accelerated on dedicated hardware

for vast speedups [120], thus making them an exciting candidate for the simulation of high dimensional systems.

We present in this paper an exploration of the options needed for RCs to be trained successfully for chaotic time series forecasting. Almost all the results presented here involve the reoptimization of parameters through the outlined Bayesian optimization procedure for every experiment. The complex effects of the macro-scale parameters on the predictability of the RC and the correlations between these parameters preclude drawing many conclusions from simply varying each one of the parameters individually. When the data, RC training method, or one parameter is changed then a full re-optimization is necessary to compensate. This is one reason why the results presented here differ from those presented in previous literature.

The bias term in the RC equation is found to be criticaly important for all but the L63 system Fig.(4.7), and we emphasize that this is generally neglected in the current literature. Indeed predictions were practically impossible for many of the models without the addition of this term. We additionally show that the form of the readout has little impact on the skillful forecast time, at least for the given dynamical models. While this may be surprising given that the inclusion of an r^2 term in the readout has become a standard practice [110, 116, 124], we note that without the inclusion of bias, the nonlinear readout does have some positive impact (results not shown).

Our further exploration included showing the deleterious effects of a standard ML normalization procedure where each dimension of the input is recentered and rescaled separately [97]. This procedure is important in deep learning where the stability of the optimization procedure using backpropagation can be compromised. Centering all of the data around 0, however, destroys the important relationships between the variables, leading to a very poor predictive capacity for the RC. The RC is not trained through backpropagation, therefore it is advisable to avoid this kind of normalization when attempting to predict chaotic time series.

We also examine the effect of the reservoir dimension N, the amount of training data as well as the time step of the data on the performance of an RC. A general "law of diminishing returns" is exhibited in regards to RC size and the amount of data. This effect can be partially mitigated by re-optimizing the RC using a larger reservoir size instead of scaling up directly. We suggest that more effort should be dedicated to increasing the robustness of the RC to noise, since in real applications this will be a limitation on the usefulness of the method.

Finally, we show how to scale up the method to larger systems through localization, examining the effect of halo size and output dimension. We present results on the 40 dimensional Lorenz system. Comparing to state of the art results for the RC [4, 116], the reported architecture in this paper increases the VPT while decreasing the computational cost. Vlachas et al. (2020) [116] noted "that RC...have slightly lower VPT than GRU and LSTM but require significantly lower training times." Our results now give a clear advantage of RC over the LSTM and GRU in pure predictive capacity as well as training time; additionally the RC has the advantage of reproducing the LEs of the input data, enabling the reproduction of the climatalogical attractor [124]. We hope that this work, by collecting a number of different characteristic models together, gives a clear set of standards to compare as a benchmark when developing new RC techniques.

4.7 Appendix

4.7.1 Malkus Water Wheel

For physical applications, determining the correct normalization scheme can be more difficult when the state variables have different units *e.g.*, temperature, pressure, velocity,.... All of the source models that we used in this study were nondimensionalized. In practice we may instead have data measured in quantities with different units. There are two solutions that we propose:

1. Nondimensionalize the state variables using physical parameters of the system. This does not require knowledge of the full equations of motion, only basic knowledge of the physics involved.

2. Allow the RC to learn the correct scaling laws by setting a separate σ and σ_b for each unit type. For example, one might have a $\sigma_{\text{Temperature}}$ and a σ_{Pressure}. Note that this replaces the scalar σ with a vector σ that has the structure $\sigma = \Big[\sigma_1, \ldots, \sigma_1, \ldots, \sigma_p\Big]$ for p different units for the state variables.

As an illustrative example, we consider the Malkus water wheel [197, 212, 213]. The chaotic water wheel is a physical model of the L63 equations and thus provides a mixed-units example with which to apply the proposed normalization schemes. Following the notation of [212], the three state variables are ω, y and z where ω is the angular velocity of the water wheel $\frac{\mathrm{d}\theta}{\mathrm{d}t}$ for the angle θ in the plane of the wheel, and y/z gives the position of the center of mass (COM) also in the plane of the wheel. The equations of motion are

$$\begin{aligned}
\dot{\omega} &= ay - f\omega \\
\dot{y} &= \omega z - \lambda y \\
\dot{z} &= -\omega y + \lambda(R - z)
\end{aligned}$$

where a is the angular acceleration due to gravity per horizontal displacement of the COM, f^{-1} is the time constant for the axle friction and input water drag, λ is the leakage rate of the water and R is the radius of the wheel. We see that λ^{-1} also gives the relaxation time constant for the COM. The maximal LE when $R = 1\ m$, $a = 1\ (m\ s)^{-1}$, $f = 0.4$, $\lambda = 0.1\ s^{-1}$ is 0.053 giving a time constant of ~ 19 model time units.

To nondimensionalize the system we see that there are two different units for the state variables. ω has units 1/time $\equiv$ 1/T while y and z have units length $\equiv \ell$. Therefore,

if we choose a time scale T and length scale ℓ, we can nondimensionalize by taking $\omega_\star = \omega/T$, $y_\star = \ell y$ and $z_\star = \ell z$. Using knowledge of the physics but without knowing the equations of motion we can write a list of parameters on which the equations may depend—see table 4.2. Again, no a priori knowledge of the equations of motion is needed.

Table 4.2. Deduced parameters of the equations of motion for the Malkus water wheel.

α	axel friction	$kg\ m/s^2$
I	moment of inertia	$kg\ m^2$
λ	leak rate	$1/s$
R	radius	m
$g\sin\phi$	gravity at angle ϕ	m/s^2
M	mass	kg

We can construct T and ℓ using this set of candidate parameters. An obvious choice for ℓ is R, since both have the same units. For T we could use $T = 1/\lambda$ or $T = \sqrt{\frac{RMg\sin\phi}{I}}$. There is no wrong answer as long as all the parameters with the same units are scaled in the same way and one can measure the parameters needed.

As a (somewhat extreme) example, let us say the distance measurements from our sensor are in μm, and ω is measured in rad/s. In this case the RC is not able to predict the system at all Fig.(4.17). This is not really surprising considering the wildly different scales between ω and the x and y variables. We correct this by scaling ω by λ and x and y by R. This leads to reasonably accurate forecasts. Letting the optimization determine the correct scaling between the variables also leads to accurate forecasts.

Chapter 4, in full, has been submitted for publication of the material as it may appear in Jason A. Platt, Stephen G. Penny, Timothy A. Smith, Tse-Chun Chen, and Henry D. I. Abarbanel. "A Systematic Exploration of Reservoir Computing for Forecasting Complex Spatiotemporal Dynamics". In: (Submitted to Neural Networks 2022). arXiv: 2201.08910. The dissertation author was the primary investigator and author of this paper.

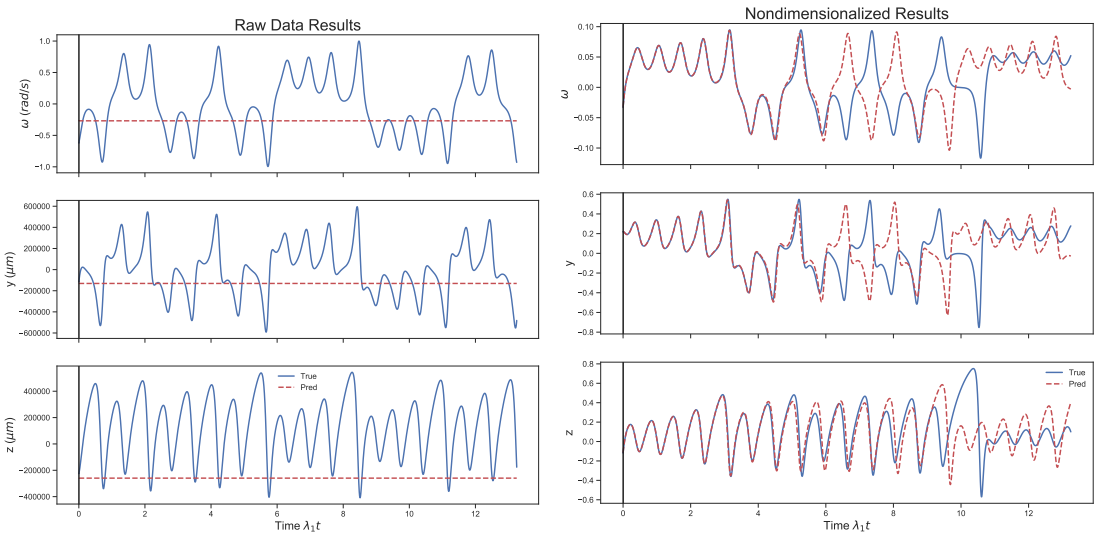

Figure 4.17. **(left)** Our hypothetical sensor measures the COM in μm, leading to an RC that utterly fails. **(right)** By normalizing the measurements correctly we find that the RC can now predict the Malkus water wheel with reasonable accuracy.

Chapter 5

Conclusion

5.1 Application of RC to DA

Parts of this section are taken from [7], previously published by JAMES.

An issue that has not been addressed so far when considering the operation of RC in realistic scenarios is how to incorporate sparsely sampled or partial observations into the forecasting system. Lu et.al. [214] considered the problem of partial observations and showed that it is possible to drive the RC with limited observations in some circumstances by directly inserting the measurements into the RC operation. Penny et.al. [7] makes it clear, however, that this technique breaks down for infrequent observations. The most robust solution to this problem is to integrate RC directly into the DA formalism. By applying DA in the reservoir space of the RC, and then using the composition of an observation operator with the readout in order to compare hidden/reservoir states with observations of the original system, we are able to design a system that is robust to sparse and noisy observations.

With an RNN-DA system there are two perspectives on the value this combination provides. From the perspective of operational forecasting, the RNN provides a simple and low-cost replacement for the production of essential information needed for the online DA cycle. The physical models used in the traditional DA routine may be expensive to integrate, imperfect and may not have a computable Jacobian. Replacing this physical

model with an model derived directly from the data may make the most sense in this situation, at least for some of the necessary quantities such as generating error-covariance matrices. From the machine learning perspective, the DA algorithms allows the RNN hidden/reservoir dynamics to be driven with a noisy and sparsely observed signal. DA methods can produce valid reconstructions of the system state as well as viable initial conditions for short-term forecasts.

DA typically requires a physical model of the process—often assumed to be "almost" perfect—in order to interpolate between the measurements and predictions. Here we substitute this physical model for a surrogate RC data driven model. The advantages of doing so can be computational, given that the RC can often be much faster than integrating the equations [120], or for the reason that the underlying equations are unresolved. In modern DA systems, such as the local ensemble transform Kalman filter (LETKF) [215, 216] the method relies on the correct estimation of the error-covariance matrices that define the level of certainty in the routine. An example estimate of the error-covariance matrix $\mathbf{P}$ for the LETKF is

$$\mathbf{P}_t = \frac{1}{k-1}\mathbf{X}_t^f \left(\mathbf{X}_t^f\right)^T \tag{5.1}$$

where the columns of $\mathbf{X}_t^f$ are perturbations around the forecast ensemble mean at time t. This estimate requires that the perturbations $\mathbf{X}$ of the RC act in the "correct" manner—*i.e.*, that perturbations grow as the LEs, in particular the finite time LEs [30, 34], of the true dynamical system.

While it was shown in chapter 3 that the global LEs of the RC match the LEs of the input system, it is not necessarily the case that the error growth rates over short time scales will as well. In Fig.(5.1) we compare the FTLE and the prediction error as a function of time. The FTLE are not estimated well for short timescales but converge to the global LEs over time. As the LEs converge, however, the prediction error increases so there is a tradeoff between estimation of $\mathbf{P}$ and the accuracy of the RC.

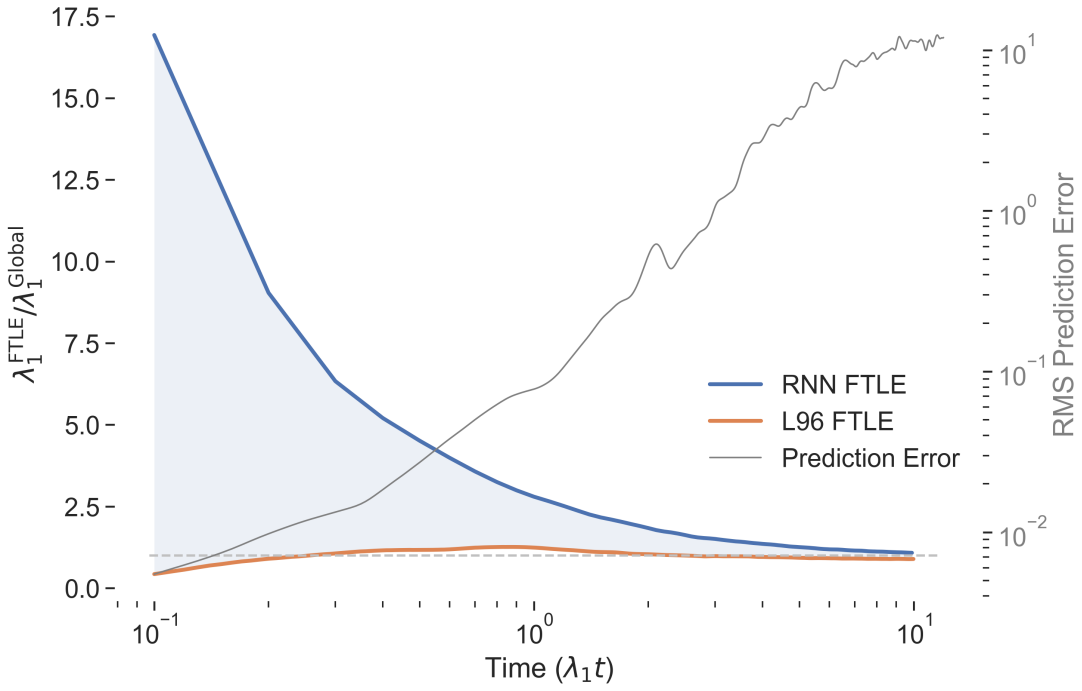

Figure 5.1. Convergence of the leading FTLE (λ_1) for a trained RNN averaged over 100 initial conditions of the L96-6D system. As the RNN is integrated for longer periods of time, the error growth rates generated by the RNN model become more accurate. However, over the same period there is an exponential growth of errors in initial conditions.

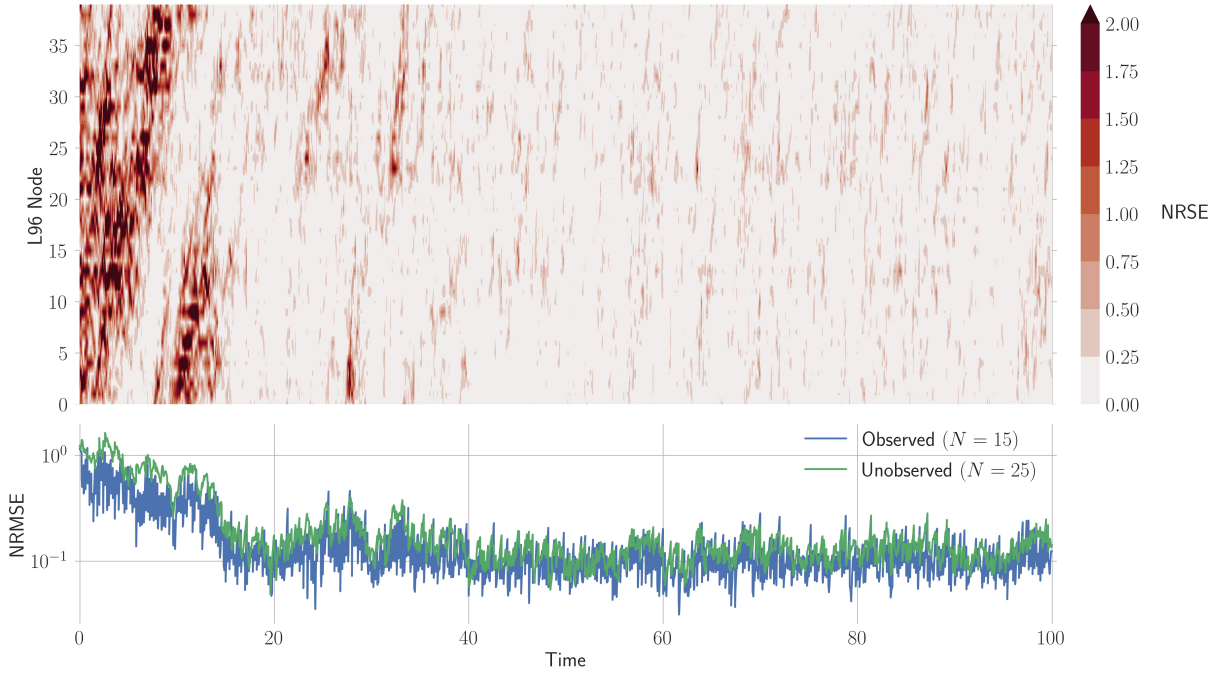

Figure 5.2. Normalized error of the RNN-LETKF based state estimation for the L96-40D system using RNN Model 3. (Top) Normalized Root Square Error (NRSE) shown for each node of the L96 system (y-axis). (Bottom) NRMSE computed separately for the observed and unobserved nodes of the Lorenz system, with 15 nodes of the system observed. Note the y-axis is logarithmic in the lower plot. The error in both plots are normalized by the temporal standard deviation of the true trajectory. The RNN-LETKF uses a 30 member ensemble, with σ_{obs}=σ_{noise}=0.5. The observed nodes of the system are $[0, 3, 5, 8, 10, 14, 16, 19, 20, 25, 27, 30, 34, 36, 39]$. *This figure was generated by Tim Smith of NOAA CIRES [7].*

Using the results from the previous sections to design an RC as well as the LETKF scheme outlined in [7], the RC-LETKF was tested on the L96-40D system Fig.(5.2). After about 20 model time units the DA cycle converges to a low level of error and we are able to predict going forward in time. Thus the RC has successfully been incorporated into a DA routine, using the concepts of GS and the reproduction of the LEs in order to correctly estimate and construct the quantities for use in the Kalman filter.

5.2 Concluding Remarks

Complex systems are phenomena that cannot be reduced to the sum of the individual constituent pieces or derived from the fundamental laws of physics. The great physicist Phil Anderson put it most succinctly in his essay *More is Different* [217] where he states

> At some point we have to stop talking about decreasing symmetry and start calling it increasing complication. Thus, with increasing complication...we expect to encounter fascinating and, I believe, very fundamental questions at each stage in fitting together less complicated pieces into the more complicated system and understanding the basically new types of behavior which can result.

As the number of constituent pieces increases we find emergent behavior which can include everything from superconductivity to memory. The study of these phenomena is as important today as it was 50 years ago when Anderson made his appeal as to the fundamentality of such systems.

In this thesis we began with a discussion of nonlinear dynamics and the application of state and parameter estimation methods to neurobiological systems. Starting from the Bayesian inference formulation of statistical data assimilation, the Euler-Lagrange equations were derived for the equations of motion of a dynamical system forced by measurements and weighted by their uncertainties. From the equations of motion the equivalence to the nudging method of meteorology, based on generalized synchronization, was proposed and applied to the estimation of manufacturing errors in neuromorphic circuits. After demonstrating the ability to estimate parameters in VLSI systems we turned our attention to the Zebra finch song system that underlies major studies of human speech development. Here we see a direct line of inquiry between the dynamics of components and small networks in the brain and questions of learning and psychology that motivate us to study these systems. As shown in chapter 2, direct measurement of individual neurons enable characterization of the ion currents and physical constants that govern their oscillatory behavior. Only in expanding this kind of characterization to many cells,

however, will we be able to continue to answer questions about the collective behavior of such neuronal conurbations. Therefore we ended the chapter with a first look at how to use calcium fluorescence data—collected from many neurons using two photon microscopy—to solve for parameters in the biophysical equations.

From the study of biological networks we turned our attention to recurrent neural networks—specifically reservoir computing—and their application to the forecasting problem. It is impossible to produce long term forecasts of chaotic systems due to the exponential growth of errors quantified by the systems Lyapunov exponent spectrum. Short term forecasts are, however, desirable in particular applications such as weather prediction. RC was shown to be able to successfully produce accurate short term forecasts on simplified weather models. Additionally, the RC model was optimized and scaled up to show state of the art results on higher dimensional systems. That success was explained using generalized synchronization, which enabled the establishment of a computationally efficient test for pretraining the network. Additionally, because GS guarantees that the RC has been driven onto a synchronization manifold which shares dynamical characteristics (*e.g.*, LEs) with the input data, we can test for the reproduction of those characteristics in examining how well we expect the RC to extrapolate to unseen data from the same source.

Finally we showed results from integrating the RC into a state of the art DA optimization routine. This routine enabled us to apply RC to a more realistic scenario where collected data is sparse and noisy. The success of the routine for the 40 dimensional Lorenz systems shows its promise in applying to ever larger systems.

Clearly there is much work to be done in the application of these kinds of data driven methods to real world problems. One area of research that is particularly interesting is the combination of machine learning with dynamical systems knowledge. There has been much fervor in the computer science and physics communities on enforcing conservation laws in neural networks and discovering low dimensional dynamical systems from data. These approaches were explored in the physics inspired neural networks of Raissi et.al.

[218] and the sparse identification of nonlinear dynamical system algorithm of Brunton et.al. [219].

It has been self evident for at least the last 20 years, however, that most real world systems—made of the complex interactions of many constituent components—are not well described by the symmetries of fundamental interactions. Most systems of interest are dissipative and not conservative—*e.g.*, any dynamical system containing friction is dissipative. Dissipative systems do not obey Louiseville's theorem and do not conserve quantities such as energy/momentum. In the vast majority of applications where the phase flow is not incompressible, the attempt to integrate conservation laws and Lagrange symmetries into neural networks may not be applicable.

For the prediction of chaotic systems it may be that although the short term predictions will inevitably diverge, the long term predictions may preserve some invariants. The multiplicative ergodic theorem shows that the LEs and the fractal dimension are both invariant under smooth coordinate transformations and have algorithms that make them feasible to compute from observed date [34]. It should be possible to enforce these conservations laws in ML systems.

The integration of RNNs into forecasting routines and the scientific toolkit is still in its infancy and there is much work to be done. Ideally these tools will become simply another method/routine just like perturbation theory, numerical integration or monte carlo methods are today—something well understood and taken for granted. The questions that need to be answered are how to best train these methods and how to guarantee interpretability and the physicality of predictions so that the correct invariant quantities are conserved. In the future, the techniques will most likely be combined into hybrid physics/machine learning methods to best capture the strengths of both. It is the sincere hope of this author that this thesis has gone some way to explaining the abilities of a few of these techniques and shown useful applications

Chapter 5, in part is adapted from S. G. Penny, T. A. Smith, T.-C. Chen,

J. A. Platt, H.-Y. Lin, M. Goodliff, and H. D. I. Abarbanel. "Integrating Recurrent Neural Networks With Data Assimilation for Scalable Data-Driven State Estimation". In: Journal of Advances in Modeling Earth Systems *14.3 (2022), e2021MS002843.* DOI: *https://doi.org/10.1029/2021MS002843. Stephen Penny was the primary investigator and author of this material while the disertation author was a coauthor.*

Appendix A

A.1 Details of Datasets

A.1.1 Rossler

The Rossler system [220] was proposed as a simplified form of the L63 system with a single second order nonlinearity and only one lobe of the strange attractor. The equations are

$$dx = -(y + z)$$

$$dy = x + 0.2y$$

$$dz = 0.2 + z(x - 5.7).$$

The LEs are $\Big[0.06, 0, -4.9\Big]$.

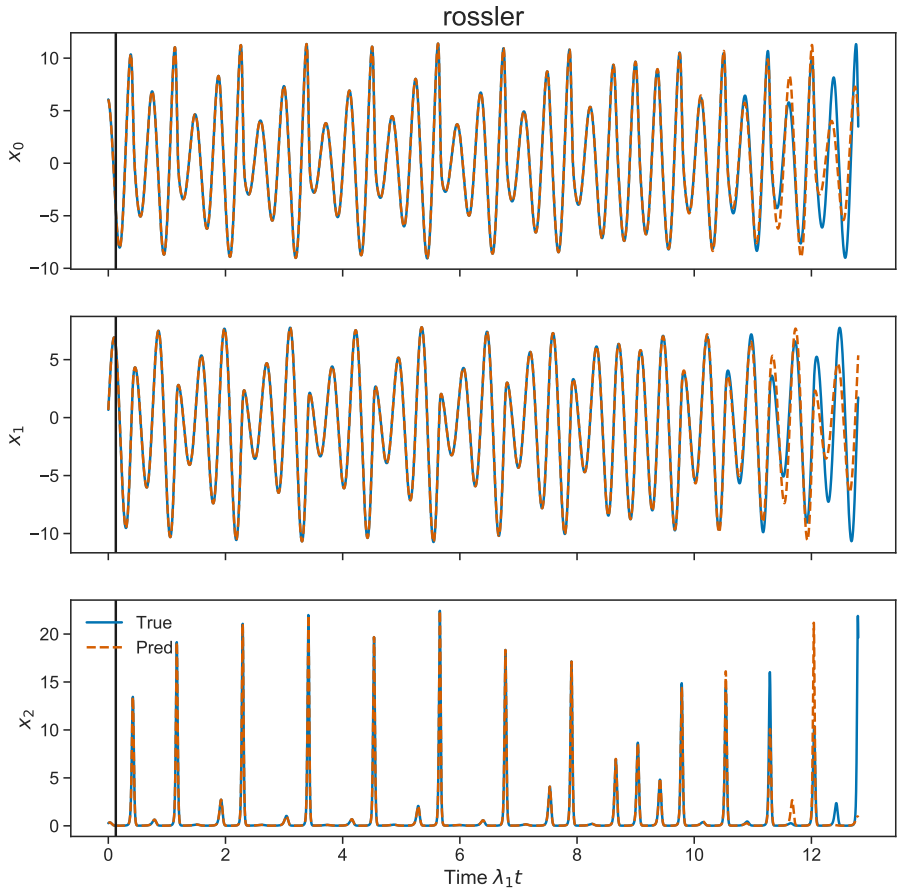

Figure A.1. Rossler Attractor

A.1.2 Colpitts

The Colpitts Oscillator [21] is a three dimensional nonlinear dynamical system describing chaos in a nonlinear circuit. The equations of the system are given

$$\begin{aligned}\frac{dx(t)}{dt} &= \alpha y(t) \\ \frac{dy(t)}{dt} &= -\gamma(x(t)+z(t)) - qy(t) \\ \frac{dz(t)}{dt} &= \eta(y(t)+1-\exp(-x(t)))\end{aligned}$$

with $\alpha = 5$, $\gamma = 0.0797$, $q = 0.6898$ and $\eta = 6.2723$. For $\alpha < 5$ this circuit has limit cycle oscillations.

The Lyapunov exponents are $\{\lambda_1, \lambda_2, \lambda_3\} = \Big[0.09, \quad 0, \quad -0.8\Big]$ calculated via the QR decomposition algorithm given by Eckmann and Ruelle [27].

A.1.3 L63

The Lorenz-63 [23] equations form a deterministic nonlinear dynamical system that exhibits chaos for certain ranges of parameters. It was originally found as a three dimensional, reduced, approximation to the partial differential equations for the heating of the lower atmosphere of the Earth by solar radiation. The dynamical equations of motion are

$$\begin{aligned}\frac{dx(t)}{dt} &= \sigma[y(t)-x(t)] \\ \frac{dy(t)}{dt} &= x(t)[\rho - z(t)] - y(t) \\ \frac{dz(t)}{dt} &= x(t)y(t) - \beta z(t)\end{aligned}$$

with time independent parameters $\sigma = 10, \rho = 28, \beta = 8/3$.

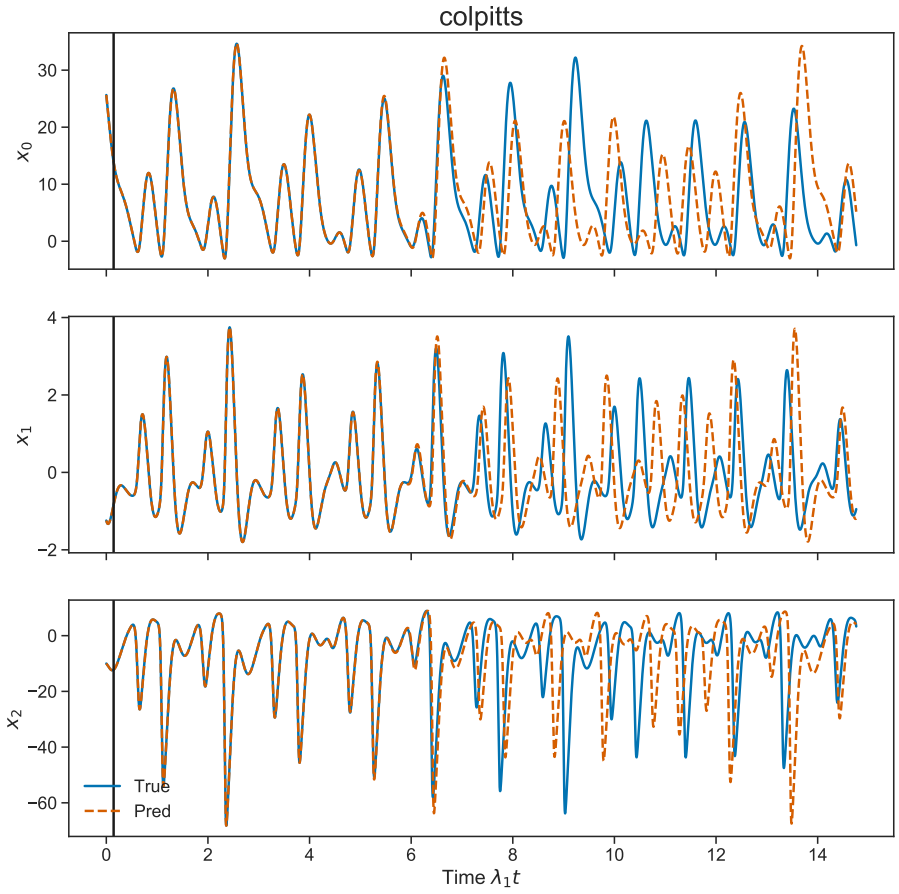

Figure A.2. Colpitt's Oscillator

The Lyapunov exponents are $\{\lambda_1, \lambda_2, \lambda_3\} = \begin{bmatrix} 0.9, & 0, & -14.7 \end{bmatrix}$ calculated using the QR decomposition approach given by Eckmann and Ruelle [27].

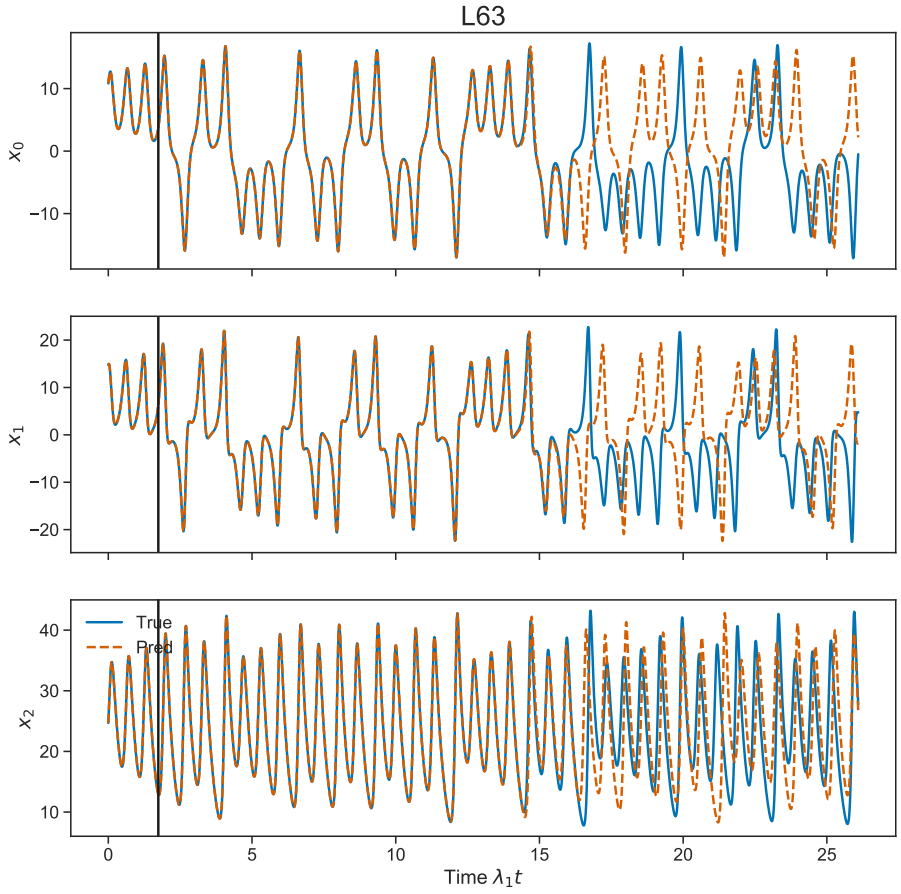

Figure A.3. Lorenz 1963

A.1.4 L96

These dynamical equations were introduced by [221]:

$$\frac{dx_a(t)}{dt} = x_{a-1}(t)(x_{a+1}(t) - x_{a-2}(t)) - x_a(t) + f$$

and $a = 1, 2, ..., D$; $x_{-1}(t) = x_{D-1}(t)$; $x_0(t) = x_D(t)$; $x_{D+1}(t) = x_1(t)$. f is a fixed parameter that we take to be 8.0 where the solutions to these dynamical equations are chaotic [180]. The equations for the states $x_a(t)$; $a = 1, 2, ..., D$ are meant to describe

points on a periodic spatial lattice. We use $D = 5$, $D = 10$ and $D = 40$.

The Lyapunov exponents are $\{\lambda_1, \ldots, \lambda_5\} = \Big[0.4, 0, -0.5, -1.3, -3.5\Big]$ and

$$\{\lambda_1, \ldots, \lambda_{10}\} = \Big[1.1, 0.7, 0.1, 0, -0.4, -0.8, -1.3, -1.9, -2.7, -4.5\Big]$$

calculated using the QR decomposition approach given by Eckmann and Ruelle [27].

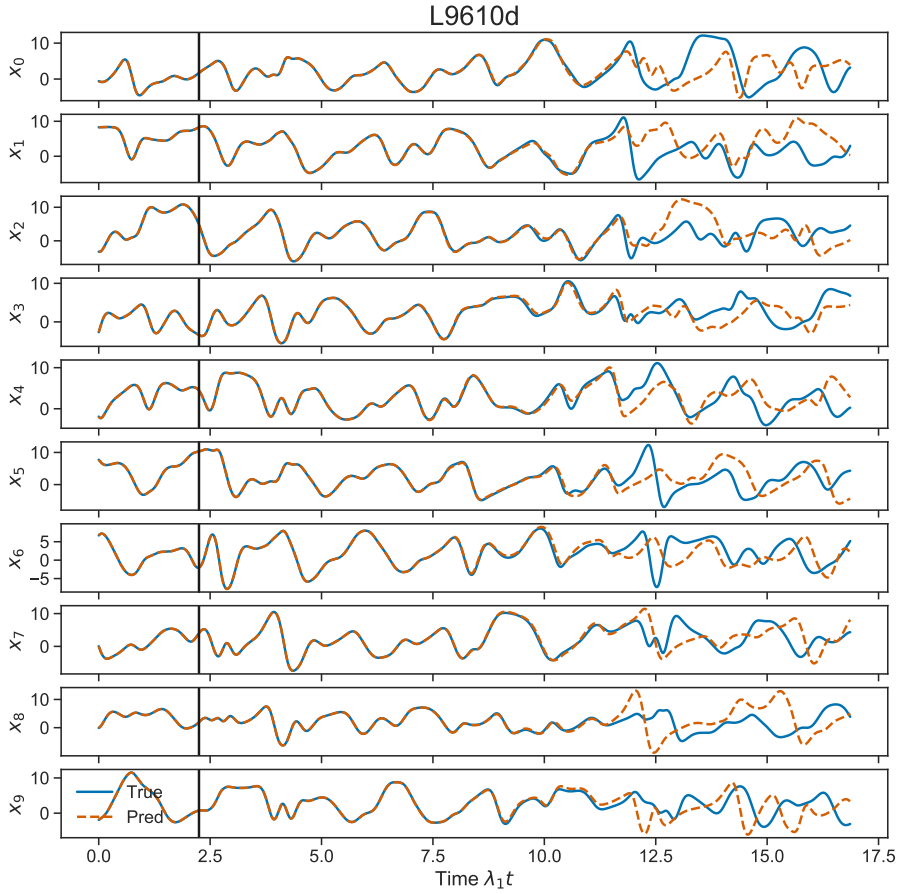

Figure A.4. Lorenz 1996 10 dimensions

A.1.5 CL63

The climate Lorenz 63 system[206] consist of three L63 systems coupled together with multiple time scales. The three layers are an extratropical atmosphere, a tropical atmosphere, and a tropical ocean system. The presence of multiple time scales often

creates a challenge for RC models.

Extratropical atmosphere system

$$\frac{\mathrm{d}x_e}{\mathrm{d}t} = \sigma(y_e - x_e) - \kappa_e(Sx_t + k_1)$$
$$\frac{\mathrm{d}y_e}{\mathrm{d}t} = \rho x_e - y_e - x_e z_e + \kappa_e(Sy_t + k_1)$$
$$\frac{\mathrm{d}z_e}{\mathrm{d}t} = x_e y_e - \beta z_e$$

Tropical atmosphere system

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = \sigma(y_t - x_t) - \kappa(Sx_o + k_2) - \kappa_e(Sx_e + k_1)$$
$$\frac{\mathrm{d}y_t}{\mathrm{d}t} = \rho x_t - y_t - x_t z_t + \kappa(Sy_o + k_2) + \kappa_e(Sy_e + k_1)$$
$$\frac{\mathrm{d}z_t}{\mathrm{d}t} = x_t y_t - \beta z_t + \kappa_z z_o$$

Tropical ocean system

$$\frac{\mathrm{d}x_o}{\mathrm{d}t} = \tau\sigma(y_o - x_o) - \kappa(x_t + k_2)$$
$$\frac{\mathrm{d}y_o}{\mathrm{d}t} = \tau\rho x_o - \tau y_o - \tau S x_o z_o + \kappa(y_t + k_2)$$
$$\frac{\mathrm{d}z_o}{\mathrm{d}t} = \tau S x_o y_o - \tau\beta z_o - \kappa_z z_t$$

$\sigma = 10, \rho = 28, \beta = 8/3., S = 1, k_1 = 10, k_2 = -11, \tau = 0.1, \kappa = 1, \kappa_e = 0.08, \kappa_z = 1$

$LEs = \Big[0.9, 0.4, 0, -0.1, -0.6, -0.8, -1.6, -11.7, -14\Big]$

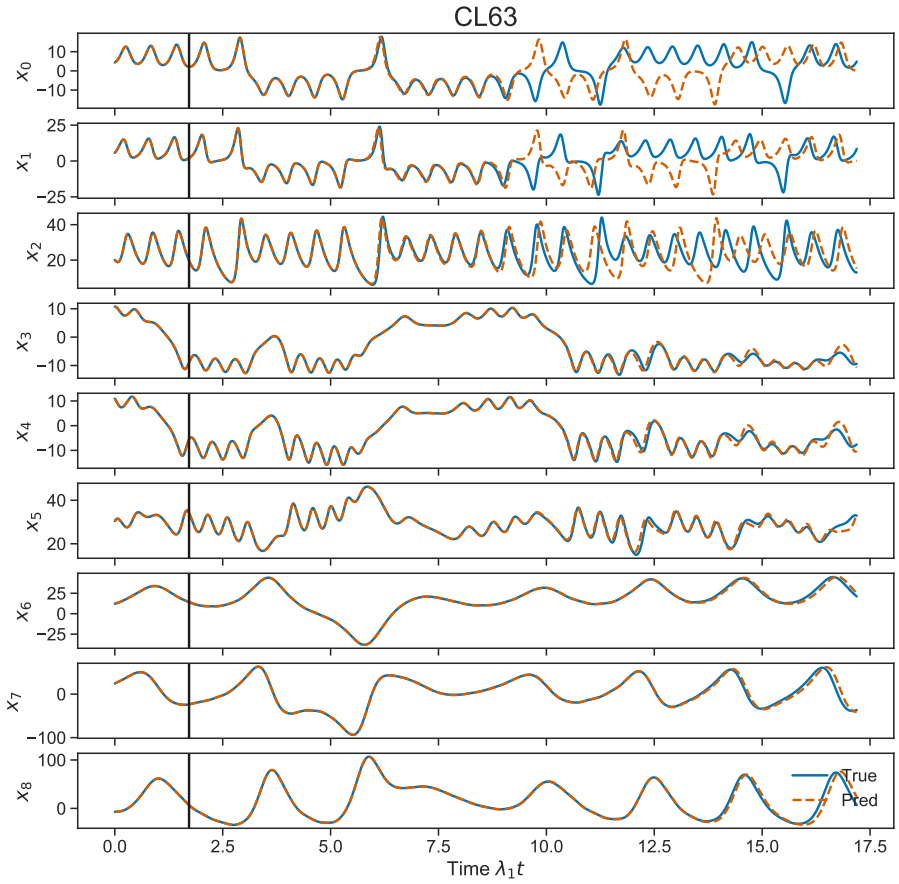

Figure A.5. Climate Lorenz 63 model

Bibliography

[1] Anna Miller et al. "Statistical Data Assimilation: Formulation and Examples From Neurobiology". In: *Frontiers in Applied Mathematics and Statistics* 4 (2018). ISSN: 2297-4687. DOI: 10.3389/fams.2018.00053.

[2] Jason A. Platt et al. "Robust forecasting using predictive generalized synchronization in reservoir computing". In: *Chaos* 31 (2021), p. 123118. URL: https://doi.org/10.1063/5.0066013.

[3] Jason A. Platt et al. "A Systematic Exploration of Reservoir Computing for Forecasting Complex Spatiotemporal Dynamics". In: (Submitted to Neural Networks 2022). arXiv: 2201.08910.

[4] S. G. Penny et al. "Integrating Recurrent Neural Networks With Data Assimilation for Scalable Data-Driven State Estimation". In: *Journal of Advances in Modeling Earth Systems* 14.3 (2022), e2021MS002843. DOI: https://doi.org/10.1029/2021MS002843.

[5] Tse-Chun Chen et al. "Learning dynamical systems and numerical integration methods". In: (Submitted to PNAS 2022). arXiv: 2201.05193.

[6] Randall Clark et al. "Reduced Dimension, Biophysical Neuron Models Constructed From Observed Data". In: *Neural Computation* (2022 in press). DOI: 10.1101/2021.12.03.471194.

[7] Stephen G. Penny et al. "Integrating Recurrent Neural Networks with Data Assimilation for Scalable Data-Driven State Estimation". In: *Journal of Advances in Modelling Earth Systems* (2022). DOI: 10.1029/2021MS002843.

[8] A.D. Lusk, J.P. Platt, and J.A. Platt. "Natural and experimental constraints on a flow law for dislocation-dominated creep in wet quartz". In: *Journal of geophysical research* (2021). DOI: https://doi.org/10.1029/2020JB021302.

[9] Jason Platt et al. "Optimal Operation of a Plug-In Hybrid Vehicle". In: *IEEE Transactions on Vehicular Technology* 67.11 (2018), pp. 10366–10377. DOI: 10.1109/TVT.2018.2866801.

[10] Jason Platt et al. "Equalizer design techniques for dispersive cables with application to the SPS wideband kicker". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 868 (2017), pp. 93–97. ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2017.06.029.

[11] Herodotus. *The Histories*. Ed. by John Marincola and Aubrey De Selincourt. Penguin, 2003.

[12] Murray Gell-Mann. "What is complexity? Remarks on simplicity and complexity by the Nobel Prize-winning author of The Quark and the Jaguar". In: *Complexity* 1.1 (1995), pp. 16–19. DOI: https://doi.org/10.1002/cplx.6130010105. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cplx.6130010105. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cplx.6130010105.

[13] M. E. J. Newman. "Resource Letter CS–1: Complex Systems". In: *American Journal of Physics* 79.8 (2011), pp. 800–810. DOI: 10.1119/1.3590372.

[14] Song-You Hong et al. "An Evaluation of the Software System Dependency of a Global Atmospheric Model". In: *Monthly Weather Review* 141.11 (2013), pp. 4165–4172. DOI: 10.1175/MWR-D-12-00352.1. URL: https://journals.ametsoc.org/view/journals/mwre/141/11/mwr-d-12-00352.1.xml.

[15] Eugenia Kalnay. *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge University Press, 2002. DOI: 10.1017/CBO9780511802270.

[16] J. M. Ottino et al. "Microfluidic systems for chemical kinetics that rely on chaotic mixing in droplets". In: *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 362.1818 (2004), pp. 1087–1104. DOI: 10.1098/rsta.2003.1364.

[17] Roman O. Grigoriev, Michael F. Schatz, and Vivek Sharma. “Chaotic mixing in microdroplets”. In: *Lab Chip* 6 (10 2006), pp. 1369–1372. DOI: 10.1039/B607003E.

[18] F. T. Arecchi et al. “Experimental evidence of chaotic itinerancy and spatiotemporal chaos in optics”. In: *Phys. Rev. Lett.* 65 (20 Nov. 1990), pp. 2531–2534. DOI: 10.1103/PhysRevLett.65.2531. URL: https://link.aps.org/doi/10.1103/PhysRevLett.65.2531.

[19] Xizhe Zang et al. “Applications of Chaotic Dynamics in Robotics”. In: *International Journal of Advanced Robotic Systems* 13.2 (2016), p. 60. DOI: 10.5772/62796.

[20] Daniel Rey et al. “Accurate state and parameter estimation in nonlinear systems with sparse observations”. In: *Physics Letters A* 378.11 (2014), pp. 869–873. ISSN: 0375-9601. DOI: https://doi.org/10.1016/j.physleta.2014.01.027. URL: https://www.sciencedirect.com/science/article/pii/S0375960114000735.

[21] Edwin H. Colpitts. “Oscillation Generation”. US 1624537. Feb. 1918.

[22] M.P. Kennedy. “Chaos in the Colpitts oscillator”. In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 41.11 (1994), pp. 771–774. DOI: 10.1109/81.331536.

[23] Edward N. Lorenz. “Deterministic Nonperiodic Flow”. In: *Journal of the Atmospheric Sciences* 20.2 (Mar. 1963), pp. 130–141. ISSN: 0022-4928. DOI: 10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2.

[24] Robert C. Hilborn et al. “Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers”. In: *Computers in Physics* 8.6 (1994), pp. 689–689. DOI: 10.1063/1.4823351.

[25] A. M. LYAPUNOV. “The general problem of the stability of motion”. In: *International Journal of Control* 55.3 (1992), pp. 531–534. DOI: 10.1080/00207179208934253.

[26] H. D. I. Abarbanel. *The Analysis of Observed Chaotic Data*. Springer-Verlag, New York, 1996.

[27] J. -P. Eckmann and D. Ruelle. “Ergodic theory of chaos and strange attractors”. In: *Rev. Mod. Phys.* 57 (3 July 1985), pp. 617–656. DOI: 10.1103/RevModPhys.57.617.

[28] James D. Bjorken and Sidney David Drell. *Relativistic Quantum Mechanics*. McGraw Hill, 1964.

[29] V. I. Oseledec. "A multiplicative ergodic theorem. Lyapunov characteristic numbers for dynamical systems". In: *Trudy Mosk. Mat. Obsc.* 19 (1968), pp. 197–.

[30] H.D.I. Abarbanel, R. Brown, and M.B. Kennel. "Local Lyapunov exponents computed from observed data". In: *Journal of Nonlinear Science* 2 (1992), pp. 343–365. DOI: https://doi.org/10.1007/BF01208929.

[31] James Theiler. "Estimating fractal dimension". In: *Journal of The Optical Society of America A-optics Image Science and Vision* 7 (1990), pp. 1055–1073.

[32] James L. Kaplan and James A. Yorke. "Chaotic behavior of multidimensional difference equations". In: *Functional Differential Equations and Approximation of Fixed Points*. Ed. by Heinz-Otto Peitgen and Hans-Otto Walther. Berlin, Heidelberg: Springer Berlin Heidelberg, 1979, pp. 204–227. ISBN: 978-3-540-35129-0.

[33] Paul Frederickson et al. "The liapunov dimension of strange attractors". In: *Journal of Differential Equations* 49.2 (1983), pp. 185–207. ISSN: 0022-0396. DOI: https://doi.org/10.1016/0022-0396(83)90011-6.

[34] H. D. I. Abarbanel. *The Analysis of Observed Chaotic Data*. Springer-Verlag, New York, 1996.

[35] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of Basic Engineering* 82.1 (Mar. 1960), pp. 35–45. ISSN: 0021-9223. DOI: 10.1115/1.3662552.

[36] Suzana Herculano-Houzel. "The human brain in numbers: a linearly scaled-up primate brain". In: *Frontiers in Human Neuroscience* 3 (2009). ISSN: 1662-5161. DOI: 10.3389/neuro.09.031.2009. URL: https://www.frontiersin.org/article/10.3389/neuro.09.031.2009.

[37] Matt Carter and Jennifer Shieh. *Guide to Research Techniques in Neuroscience*. Elsevier, 2015. ISBN: 978-0-12-800511-8. DOI: https://doi.org/10.1016/C2013-0-06868-5.

[38] Chih-Ying Su, Karen Menuz, and John. R. Carlson. "Olfactory Perception: Receptors, Cells, and Circuits". In: *Cell* 139 (2009), pp. 45–59.

[39] Gautam Reddy et al. "Antagonism in olfactory receptor neurons and its implications for the perception of odor mixtures". In: *eLife* 7 (Apr. 2018). Ed. by Fred Rieke, e34958. ISSN: 2050-084X. DOI: 10.7554/eLife.34958.

[40] Lu Xu et al. "Widespread receptor-driven modulation in peripheral olfactory coding". In: *Science* 368.6487 (2020). ISSN: 0036-8075. DOI: 10.1126/science.aaz5390.

[41] G. Laurent et al. "Odor Encoding as an Active Dynamical Process: Experiments, Computation, and Theory". In: *Ann. Rev. Neuroscience* 24 (2001), pp. 263–297.

[42] Ramón Huerta. "Learning pattern recognition and decision making in the insect brain". In: *AIP Conference Proceedings*. Vol. 1510. 2013, pp. 101–119. URL: https://doi.org/10.1063/1.4776507.

[43] Ramón Huerta and Thomas Nowotny. "Fast and Robust Learning by Reinforcement Signals: Explorations in the Insect Brain". In: *Neural Computation* 21.8 (2009). PMID: 19538091, pp. 2123–2151. DOI: 10.1162/neco.2009.03-08-733. eprint: https://doi.org/10.1162/neco.2009.03-08-733. URL: https://doi.org/10.1162/neco.2009.03-08-733.

[44] Collins Assisi, Mark Stopfer, and Maxim Bazhenov. "Optimality of sparse olfactory representations is not affected by network plasticity". In: *PLOS Computational Biology* 16.2 (Feb. 2020), pp. 1–20. DOI: 10.1371/journal.pcbi.1007461.

[45] G. Laurent, M. Wehr, and H. Davidowitz. "Temporal representations of odors in an olfactory network". In: *Journal of Neuroscience* (1996).

[46] G. Laurent. "A systems perspective on early olfactory coding". In: *Science* (1999).

[47] M. Wehr and G. Laurent. "Odour encoding by temporal sequences of firing in oscillating neural assemblies". In: *Nature* (1996).

[48] M.I. Rabinovich et al. "Dynamical coding of sensory information with competitive networks". In: *Journal of Physiology* (2000).

[49] M.I. Rabinovich et al. “Dynamical Encoding by Networks of Competing Neuron Groups: Winnerless Competition”. In: *Physical Review Letters* (2001).

[50] Valentin Afraimovich, Mikhael Rabinovich, and Pablo Varona. “Heteroclinic Contours in Neural Ensembles and the Winnerless Competition Principle”. In: *International Journal of Bifurcation and Chaos* (2004).

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

[52] Bryan A Toth et al. “Dynamical estimation of neuron and network properties I: variational methods”. In: *Biological cybernetics* 105.3-4 (2011), pp. 217–237.

[53] Jingxin Ye et al. “Systematic variational method for statistical nonlinear state and parameter estimation”. In: *Physical Review E* 92.5 (2015), p. 052901. DOI: doi.org/10.1103/PhysRevE.92.052901.

[54] Alban Farchi et al. “A comparison of combined data assimilation and machine learning methods for offline and online model error correction”. In: *Journal of Computational Science* 55 (2021), p. 101468. ISSN: 1877-7503. DOI: https://doi.org/10.1016/j.jocs.2021.101468.

[55] Mike Fisher et al. “Weak-constraint and long window 4DVAR”. In: 655 (Nov. 2011), p. 47. DOI: 10.21957/9ii4d4dsq.

[56] Jochen Bröcker. “My view on weak constraint four dimensional variational assimilation”. In: *Summer School/Creative Workshop: Data Assimilation & Inverse Problems* (2013). URL: http://www.inverseproblems.info/reading:summer$%5C_$school$%5C_$2013.

[57] P. S. Laplace. “Memoir on the Probability of Causes of Events”. In: *Mathématique et de Physique,Tome Sixiéme* (1774), pp. 621–656.

[58] N. Kadakia et al. “Symplectic structure of statistical variational data assimilation”. In: *Quarterly Journal of the Royal Meteorological Society* 143.703 (2017), pp. 756–771. ISSN: 1477-870X. DOI: 10.1002/qj.2962. URL: http://dx.doi.org/10.1002/qj.2962.

[59] Zhenming Shun and Peter McCullagh. “Laplace Approximation of High Dimensional Integrals”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 57.4 (1995), pp. 749–760. ISSN: 00359246. URL: http://www.jstor.org/stable/2345941.

[60] Nirag Kadakia. “The Dynamics of Nonlinear Inference”. PhD thesis. UC San Diego, 2017.

[61] R. P. Feynman and A. R. Hibbs. *Quantum Mechanics and Path Integrals*. McGraw-Hill, 1965.

[62] C. W. Gardiner. “Handbook of stochastic methods for physics, chemistry and the natural sciences”. In: Springer Series in Synergetics 13 (2004).

[63] L. D. Landau and E. M. Lifshitz. *Mechanics, Third Edition: Volume 1 (Course of Theoretical Physics)*. 3rd ed. Butterworth-Heinemann, Jan. 1976.

[64] Henry D. I. Abarbanel et al. “A unifying view of synchronization for data assimilation in complex nonlinear networks”. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 27.12 (2017), p. 126802. DOI: 10.1063/1.5001816.

[65] Richard A. Anthes. “Data Assimilation and Initialization of Hurricane Prediction Models”. In: *Journal of Atmospheric Sciences* 31.3 (1974). DOI: 10.1175/1520-0469(1974)031<0702:DAAIOH>2.0.CO;2.

[66] Daniel R. Creveling, Philip E. Gill, and Henry D.I. Abarbanel. “State and parameter estimation in nonlinear systems as an optimal tracking problem”. In: *Physics Letters A* (2008).

[67] Henry Abarbanel. *Predicting the future: completing models of complex systems*. New York: Springer, 2013.

[68] A. Wachter and L. T. Biegler. “On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming”. In: *Mathematical Programming* 106.1 (2006), pp. 25–57.

[69] C Meliza et al. “Estimating parameters and predicting membrane voltages with conductance-based neuron models”. In: *Biological Cybernetics* (2013). DOI: 10.1007/s00422-014-0615-5.

[70] A. Hodgkin and A. Huxley. “A quantitative description of membrane current and its application to conduction and excitation in nerve”. In: *Journal of Physiology* (1952).

[71] R. Jacob Vogelstein et al. “Dynamically Reconfigurable Silicon Array of Spiking Neurons With Conductance-Based Synapses”. In: *IEEE Transactions on Neural Networks* 18.1 (2007), pp. 253–265. DOI: 10.1109/TNN.2006.883007.

[72] Chetan Singh Thakur et al. “Large-Scale Neuromorphic Spiking Array Processors: A Quest to Mimic the Brain”. In: *Frontiers in Neuroscience* 12 (2018). ISSN: 1662-453X. DOI: 10.3389/fnins.2018.00891.

[73] Tim R. Mullen et al. “Real-time neuroimaging and cognitive monitoring using wearable dry EEG”. In: *IEEE Transactions on Biomedical Engineering* 62.11 (2015), pp. 2553–2567. DOI: 10.1109/TBME.2015.2481482.

[74] T. Yu and G. Cauwenberghs. “Analog VLSI biophysical neurons and synapses with programmable membrane channel kinetics”. In: *IEEE Transactions on Biomedical Circuits and Systems* 4 (2010), pp. 139–148.

[75] Jun Wang et al. “Assimilation of Biophysical Neuronal Dynamics in Neuromorphic VLSI”. In: *IEEE Transactions on Biomedical Circuits and Systems* (2017).

[76] Arij Daou et al. “Electrophysiological characterization and computational models of HVC neurons in the zebra finch”. In: *Journal of Neurophysiology* 110 (2013), pp. 1227–1245. URL: http://jn.physiology.org/content/jn/110/5/1227.full.pdf.

[77] Arij Daou and Daniel Margoliash. “Intrinsic neuronal properties represent song and error in zebra finch vocal learning”. In: *Nature Communications* 11.1 (Feb. 2020), p. 952. ISSN: 2041-1723. DOI: 10.1038/s41467-020-14738-7. URL: https://doi.org/10.1038/s41467-020-14738-7.

[78] A C Yu and D Margoliash. “Temporal hierarchical control of singing in birds”. en. In: *Science* 273.5283 (Sept. 1996), pp. 1871–1875.

[79] Michinori Kubota and Ikuo Taniguchi. "Electrophysiological Characteristics of Classes of Neuron in the HVc of the Zebra Finch". In: *Journal of Neurophysiology* 80.2 (1998). PMID: 9705478, pp. 914–923. DOI: 10.1152/jn.1998.80.2.914. URL: https://doi.org/10.1152/jn.1998.80.2.914.

[80] Patrick Dutar, Huan M. Vu, and David J. Perkel. "Multiple Cell Types Distinguished by Physiological, Pharmacological, and Anatomic Properties in Nucleus HVc of the Adult Zebra Finch". In: *Journal of Neurophysiology* 80.4 (1998). PMID: 9772242, pp. 1828–1838. DOI: 10.1152/jn.1998.80.4.1828. URL: https://doi.org/10.1152/jn.1998.80.4.1828.

[81] Richard Mooney. "Different Subthreshold Mechanisms Underlie Song Selectivity in Identified HVc Neurons of the Zebra Finch". In: *Journal of Neuroscience* 20.14 (2000), pp. 5420–5436. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.20-14-05420.2000. eprint: http://www.jneurosci.org/content/20/14/5420.full.pdf. URL: http://www.jneurosci.org/content/20/14/5420.

[82] Winifried Denk, James H. Strickler, and Watt W. Webb. "Two-Photon Laser Scanning Fluorescence Microscopy". In: *Science* 248.4951 (1990), pp. 73–76. DOI: 10.1126/science.2321027.

[83] Thomas Kreuz et al. "Time-resolved and time-scale adaptive measures of spike train synchrony". In: *Journal of Neuroscience Methods* 195.1 (2011), pp. 92–106. ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2010.11.020.

[84] Thomas Kreuz et al. "Monitoring spike train synchrony". In: *Journal of Neurophysiology* 109.5 (2013), pp. 1457–1472. DOI: 10.1152/jn.00873.2012.

[85] T. Kreuz. "SPIKE-distance". In: *Scholarpedia* 7.12 (2012). revision #196763, p. 30652. DOI: 10.4249/scholarpedia.30652.

[86] Alain Nogaret et al. "Automatic Construction of Predictive Neuron Models through Large Scale Assimilation of Electrophysiological Data". In: *Scientific Reports* 6.1 (Sept. 2016), p. 32749. DOI: 10.1038/srep32749.

[87] Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. "Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix

Adaptation (CMA-ES)". In: *Evolutionary Computation* 11.1 (2003), pp. 1–18. DOI: 10.1162/106365603321828970.

[88] Ranier Storn and Kenneth Price. "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces". In: *Journal of Global Optimization* 11 (1997), pp. 341–359.

[89] W Denk et al. "Anatomical and functional imaging of neurons using 2-photon laser scanning microscopy". In: *Journal of neuroscience methods* 54 (Nov. 1994), pp. 151–62. DOI: 10.1016/0165-0270(94)90189-9.

[90] Joseph B. Dechery and Jason N. MacLean. "Functional triplet motifs underlie accurate predictions of single-trial responses in populations of tuned and untuned V1 neurons". In: *PLOS Computational Biology* 14 (May 2018), pp. 1–23. DOI: 10.1371/journal.pcbi.1006153.

[91] Jason N. MacLean and Rafael Yuste. "Imaging action potentials with calcium indicators." In: *Cold Spring Harbor protocols* 2009 11 (2009).

[92] D Smetters, A Majewska, and R Yuste. "Detecting action potentials in neuronal populations with calcium imaging". In: (June 1999), pp. 215–221. DOI: doi:10.1006/meth.1999.0774.

[93] Peter So. "Two-photon Fluorescence Light Microscopy". In: May 2001. DOI: 10.1038/npg.els.0002991.

[94] Joshua T. Vogelstein et al. "Spike inference from calcium imaging using sequential Monte Carlo methods." In: *Biophysical journal* 97 2 (2009), pp. 636–55.

[95] J. Ye et al. "Improved variational methods in statistical data assimilation". In: *Nonlinear Processes in Geophysics* 22.2 (2015), pp. 205–213. DOI: 10.5194/npg-22-205-2015. URL: https://www.nonlin-processes-geophys.net/22/205/2015/.

[96] Zachary Chase Lipton. "A Critical Review of Recurrent Neural Networks for Sequence Learning". In: *CoRR* abs/1506.00019 (2015). arXiv: 1506.00019.

[97] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA; London, UK, 2016.

[98] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[99] F. A. Gers, J. Schmidhuber, and F. Cummins. "Learning to Forget: Continual Prediction with LSTM". In: *Neural Computation* 12.10 (2000), pp. 2451–2471.

[100] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179.

[101] M. Lukovsevicius and H. Jaeger. "Reservoir computing approaches to recurrent neural network training". In: *Comput. Sci. Rev.* 3 (2009), pp. 127–149.

[102] David Verstraten. "Reservoir Computing: Computation with Dynamical Systems". PhD thesis. University of Ghent, 2009.

[103] Lukosevicius. "Reservoir Computing and Self-Organized Neural Hierarchies". PhD thesis. Jacobs University Bremen, 2011.

[104] Herbert Jaeger. "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note". In: *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report* 148 (2010), pp. 1–47.

[105] Herbert Jaeger. *Long Short-Term Memory in Echo State Networks: Details of a Simulation Study*. 2012.

[106] Herbert Jaeger and Harald Haas. "Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication". In: *Science* 304 (2004), pp. 78–80. DOI: 10.1126/science.1091277.

[107] Benjamin Schrauwen, David Verstraeten, and Jan Van Campenhout. "An overview of reservoir computing: Theory, applications and implementations". In: *ESANN 2007 proceedings - European Symposium on Artificial Neural Networks Bruges (Belgium), 25-27 April 2007*. 2007, pp. 471–482.

[108] Wolfgang Maass, Thomas Natschläger, and Henry Markram. "Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations". In: *Neural Computation* 14 (2002), pp. 2531–2560.

[109] Grzegorz M. Wojcik and Wieslaw A. Kaminski. "Liquid state machine built of Hodgkin-Huxley neurons and pattern recognition". In: *Neurocomputing* 58-60 (2004). Computational Neuroscience: Trends in Research 2004, pp. 245–251. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2004.01.051.

[110] Jaideep Pathak et al. "Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach". In: *Physical Review Letters* 120 (2018), p. 024102.

[111] S. Hochreiter et al. "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies". In: *A Field Guide to Dynamical Recurrent Neural Networks*. Ed. by S. C. Kremer and J. F. Kolen. IEEE Press, 2001.

[112] Y. Bengio, P. Frasconi, and P. Simard. "The problem of learning long-term dependencies in recurrent networks". In: *IEEE International Conference on Neural Networks*. 1993, 1183–1188 vol.3.

[113] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. "On the difficulty of training recurrent neural networks". In: ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, June 2013, pp. 1310–1318.

[114] Brian Hunt. "Machine Learning for Forecasting and Data Assimilation". In: *Talk at NOAA Center for Satellite Applications and Research (STAR); November 7, 2019*. 2019.

[115] Edward Ott. *Using Machine Learning for Prediction of Large Complex, Spatially Extended Systems*. Workshop on Dynamical Methods in Data-based Exploration

of Complex Systems; Max-Planck Institute for the Physics of Complex Systems; Dresden, Germany. Oct. 2019.

[116] P.R. Vlachas et al. "Backpropagation algorithms and Reservoir Computing in Recurrent Neural Networks for the forecasting of complex spatiotemporal dynamics". In: *Neural Networks* 126 (2020), pp. 191–217. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2020.02.016.

[117] N. Sharan, G. Matheou, and P. E. Dimotakis. "Turbulent shear-layer mixing: initial conditions, and direct-numerical and large-eddy simulations." In: *J. Fluid Mech.* 877 (2019), pp. 35–81.

[118] G. Matheou. "Turbulence structure in a stratocumulus cloud". In: *Atmosphere* 9.10 (2018), p. 392.

[119] Gouhei Tanaka et al. "Recent advances in physical reservoir computing: A review". In: *Neural Networks* 115 (2019), pp. 100–123.

[120] Daniel Canaday, Aaron Griffith, and Daniel J. Gauthier. "Rapid time series prediction with a hardware-based reservoir computer". In: *Chaos* 28 (2018), p. 123119.

[121] Herbert Jaeger. "Short term memory in echo state networks. GMD-Report 152". In: *GMD - German National Research Institute for Computer Science*. 2002.

[122] J.J. Steil. "Backpropagation-decorrelation: online recurrent learning with O(N) complexity". In: *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*. Vol. 2. 2004, 843–848 vol.2. DOI: 10.1109/IJCNN.2004.1380039.

[123] A.F. Atiya and A.G. Parlos. "New results on recurrent network training: unifying the algorithms and accelerating convergence". In: *IEEE Transactions on Neural Networks* 11.3 (2000), pp. 697–709. DOI: 10.1109/72.846741.

[124] Zhixin Lu, Brian R. Hunt, and Edward Ott. "Attractor reconstruction by machine learning". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 28.6 (2018), p. 061104. DOI: 10.1063/1.5039508.

[125] Aaron Griffith, Andrew Pomerance, and Daniel J. Gauthier. "Forecasting chaotic systems with very low connectivity reservoir computers". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 29.12 (2019), p. 123108. DOI: 10.1063/1.5120710.

[126] Troy Arcomano et al. "A Machine Learning-Based Global Atmospheric Forecast Model". In: *Geophysical Research Letters* 47.9 (2020), e2020GL087776. DOI: https://doi.org/10.1029/2020GL087776.

[127] Marc Bocquet et al. "Bayesian inference of chaotic dynamics by merging data assimilation, machine learning and expectation-maximization". In: *Foundations of Data Science* 2.1 (2020), pp. 55–80.

[128] Marc Bocquet, Alban Farchi, and Quentin Malartic. "Online learning of both state and dynamics using ensemble Kalman filters". In: *Foundations of Data Science* 3.3 (2021), pp. 305–330.

[129] Casper Kaae Sønderby et al. *MetNet: A Neural Weather Model for Precipitation Forecasting*. 2020. arXiv: 2003.12140 [cs.LG].

[130] Erik Bollt. "On explaining the surprising success of reservoir computing forecaster of chaos? The universal machine learning dynamical system with contrast to VAR and DMD". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 31.1 (2021), p. 013108. DOI: 10.1063/5.0024890.

[131] Daniel J. Gauthier et al. "Next generation reservoir computing". en. In: *Nature Communications* 12.1 (Sept. 2021). DOI: 10.1038/s41467-021-25801-2. (Visited on 11/22/2021).

[132] Randall E Clark et al. "Reduced Dimension, Biophysical Neuron Models Constructed From Observed Data". In: *bioRxiv* (2021). DOI: 10.1101/2021.12.03.471194.

[133] David Verstraeten et al. "An experimental unification of reservoir computing methods". In: *Neural Networks* 20 (2007), pp. 491–403.

[134] Thomas Lymburn et al. "The reservoir's perspective on generalized synchronization". In: *Chaos* 29 (2019), p. 093133.

[135] M. M. Sushchik et al. “Generalized Synchronization of Chaos in Directionally Coupled Chaotic Systems”. In: *Physical Review E* 51 (1995), pp. 980–994.

[136] H. D. I. Abarbanel, N. F. Rul’kov, and M. M. Sushchik. “The Auxiliary Systems Approach to Generalized Synchronization of Chaos”. In: *Physical Review E* 53 (1996), pp. 4528–4535.

[137] L. Kocarev and U Parlitz. “Generalized Synchronization, Predictability, and Equivalence of Unidirectionally Coupled Dynamical Systems”. In: *Physical Review Letters* 76 (1996), pp. 1816–1819.

[138] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis, 2nd ed.* Cambridge University Press, Cambridge, UK, 2004.

[139] L. M. Pecora and T. L. Carroll. “Synchronization in Chaotic Systems”. In: *Physical Review Letters* 64 (1990), pp. 821–825.

[140] D. Y. Tang et al. “Observation of generalized synchronization of chaos in a driven chaotic system”. In: *Physical Review E* 57 (1998), pp. 5247–5251.

[141] GONZALO ALVAREZ and SHUJUN LI. “Some Basic Cryptographic Requirements for Chaos-based Cryptosystems”. In: *International Journal of Bifurcation and Chaos* 16.08 (2006), pp. 2129–2151. DOI: 10.1142/S0218127406015970.

[142] Brian R. Hunt, Edward Ott, and James A. Yorke. “Differentiable generalized synchronization of chaos”. In: *Phys. Rev. E* 55 (4 Apr. 1997), pp. 4029–4034. DOI: 10.1103/PhysRevE.55.4029. URL: https://link.aps.org/doi/10.1103/PhysRevE.55.4029.

[143] Steven W. McDonald et al. “Fractal Basin Boundaries”. In: *Physica D* 17 (1985), pp. 125–153.

[144] Zhixin Lu and Danielle S. Bassett. “Invertible generalized synchronization: A putative mechanism for implicit learning in neural systems”. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 30.6 (2020), p. 063133. DOI: 10.1063/5.0004344.

[145] Edward N Lorenz. "Deterministic nonperiodic flow". In: *Journal of the atmospheric sciences* 20.2 (1963), pp. 130–141.

[146] Edward N. Lorenz. "Predictability – a problem partly solved". In: *Predictability of weather and climate*. Ed. by Tim Palmer and Renate Hagedorn. Cambridge University Press, 2006. Chap. 3, pp. 40–58. ISBN: 9780511617652. DOI: doi.org/10.1017/CBO9780511617652.

[147] E. N. Lorenz and K. A. Emanuel. *Optimal sites for supplementary weather observations: Simulations with a small model*. 1998.

[148] Robert Sadourny. "The Dynamics of Finite-Difference Models of the Shallow Water Equations". In: *Journal of the Atmospheric Sciences* 32 (1975), pp. 680–689.

[149] Joseph Pedlosky. *Geophysical Fluid Dynamics, 2^{nd} Edition*. ISBN 0-387-96387-1. Springer Verlag, 1986.

[150] Geoffrey K. Vallis. *Atmosphere and Ocean Fluid Dynamics: Fundamentals and Large-Scale Circulation*. doi: 10.101.1017/9781107588417. Cambridge University Press, 2017.

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

[152] D. Sterratt et al. *Principles of Computational Modelling in Neuroscience*. Cambridge University Press, 2011, p. 390.

[153] Jeffrey L. Elman. "Finding structure in time". In: *Cognitive Science* 14.2 (1990), pp. 179–211. ISSN: 0364-0213. DOI: https://doi.org/10.1016/0364-0213(90)90002-E.

[154] Allen Hart, James Hook, and Jonathan Dawes. "Embedding and approximation theorems for echo state networks". In: *Neural Networks* 128 (2020), pp. 234–247. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2020.05.013.

[155] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Monographs on Statistics and Applied Probability No. 26. Chapman and Hall/CRC; London, 1986.

[156] Martin D. Buhmann. *Radial Basis Functions (Theory and Implementations)*. Cambridge University Press, 2009.

[157] Martin Casdagli. "NONLINEAR PREDICTION OF CHAOTIC TIME SERIES". In: *Physica D* 35 (1989), pp. 335–356.

[158] S. A. Billings. *Nonlinear System Identification: NARMAX Methods in the time, frequency, and spatio-temporal Domains*. John Wiley and Sons, Ltd, 2013.

[159] D. S. Broomhead and David Lowe. "Multivariable Functional Interpolation and Adaptive Networks". In: *Complex Systems* 2 (1988), pp. 321–355.

[160] Phillipe Guillaume and Alain Huard. "Generalized Multivariate Pade' Approximants". In: *Journal of Approximation Theory* 95 (1998), pp. 203–214.

[161] David W. Scott and Stephen R. Sain. "Multidimensional Density Estimation". In: *Handbook of Statistics, Vol. 24*. 2005 Elsevier B.V., 2005. Chap. 9, pp. 229–261.

[162] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. Wiley:New York, 1977.

[163] K. Pyragas. "Properties of Generalized Synchronization of Chaos". In: *Nonlinear Analysis: Modelling and Control*. Vol. IMI Number 3. Institute of Mathematics and Informatics of Vilnius University. 1998.

[164] S. Boyd and L.O. Chua. "Fading memory and the problem of approximating nonlinear operators with Volterra series". In: *IEEE Trans. Circuits Syst., CAS-32* 11 (1985), pp. 1150–1161.

[165] Jaroslav Stark. "Invariant graphs for forced systems". In: *Physica D* 109 (1997), pp. 163–179.

[166] Jaroslav Stark. "Regularity of invariant graphs for forced systems". In: *Ergod. Th. and Dynam. Sys* 19 (1999), pp. 155–199.

[167] Kresimir Josic. “Synchronization of chaotic systems and invariant manifolds”. In: *Nonlinearity* 13.4 (June 2000), pp. 1321–1336. DOI: 10.1088/0951-7715/13/4/318.

[168] Louis M. Pecora et al. “Fundamentals of synchronization in chaotic systems, concepts, and applications”. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 7.4 (1997), pp. 520–543. DOI: 10.1063/1.166278.

[169] Lou Pecora et al. “Synchronization Stability in Coupled Oscillator Arrays: Solution for Arbitrary Configurations”. In: *International Journal of Bifurcation and Chaos* 10.02 (2000), pp. 273–290. DOI: 10.1142/S0218127400000189.

[170] Ljupco Kocarev, Ulrich Parlitz, and Reggie Brown. “Robust synchronization of chaotic systems”. In: *Phys. Rev. E* 61 (4 Apr. 2000), pp. 3716–3720. DOI: 10.1103/PhysRevE.61.3716. URL: https://link.aps.org/doi/10.1103/PhysRevE.61.3716.

[171] K. Miller. “Least Squares Methods for Ill-Posed Problems with a Prescribed Bound”. In: *SIAM Journal on Mathaematical Analysis* 1 (1970), pp. 52–74.

[172] William H. Press et al. *Numerical recipes: the art of scientific computing 3rd Edition*. Cambridge University Press, 2007. ISBN: 978-0-521-88068-8.

[173] Ghadban Khalaf and Ghazi Shukur. “Choosing Ridge Parameter for Regression Problems”. In: *Communications in Statistics – Theory and Methods* (2005), pp. 1177–1182.

[174] M. J. D. Powell. “Radial Basis Functions for Multivariable Interpolation: A Review”. In: *Algorithms for Approximation*. Oxford: Clarendon Press, 1987, pp. 143–167.

[175] Richard FitzHugh. “Impulses and physiological states in theoretical models of nerve membrane”. In: *Biophysical Journal* 1.6 (1961), p. 445.

[176] J. Nagumo, S. Arimoto, and S. Yoshizawa. “An Active Pulse Transmission Line Simulating Nerve Axon”. In: *Proceedings of the IRE* 50.10 (Oct. 1962), pp. 2061–2070. ISSN: 0096-8390. DOI: 10.1109/JRPROC.1962.288235.

[177] T. L. Carroll. "Do reservoir computers work best at the edge of chaos?" In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 30.12 (2020), p. 121109. DOI: 10.1063/5.0038163.

[178] Alan L Hodgkin and Andrew F Huxley. "A quantitative description of membrane current and its application to conduction and excitation in nerve". In: *The Journal of physiology* 117.4 (1952), pp. 500–544.

[179] H. D. I. Abarbanel. *The Statistical Physics of Data Assimilation and Machine Learning*. Cambridge University Press, 2022, p. 284.

[180] Mark Kostuk. "Synchronization and Statistical Methods for the Data Assimilation of HVC Neuron Models". PhD thesis. University of California San Diego, 2012. URL: http://escholarship.org/uc/item/2fh4d086.

[181] Daniel Creveling. "Parameter and state estimation in nonlinear dynamical systems". PhD thesis. University of California San Diego, 2008. URL: https://escholarship.org/uc/item/7hj6g324.

[182] Daniel J. Gauthier and J. C. Bienfang. "Intermittent loss of synchronization in coupled chaotic oscillators: Toward a new criterion for high-quality synchronization". In: *Physical Review Letters* 77 (1996), pp. 1751–1754.

[183] Michael T. Rosenstein, James J. Collins, and Carlo J. De Luca. "A practical method for calculating largest Lyapunov exponents from small data sets". In: *Physica D: Nonlinear Phenomena* 65.1 (1993), pp. 117–134. ISSN: 0167-2789. DOI: https://doi.org/10.1016/0167-2789(93)90009-P.

[184] Joschka Boedecker et al. "Information processing in echo state networks at the edge of chaos". In: *Theory in biosciences = Theorie in den Biowissenschaften* 131.3 (Sept. 2012), pp. 205–213. ISSN: 1431-7613. DOI: 10.1007/s12064-011-0146-8.

[185] Junjie Jiang and Ying-Cheng Lai. "Model-free prediction of spatiotemporal dynamical systems with recurrent neural networks: Role of network spectral radius". In: *Phys. Rev. Research* 1 (3 Oct. 2019), p. 033056. DOI: 10.1103/PhysRevResearch.1.033056.

[186] A. Kh. Gelig, G. A. Leonov, and V. A. Yakubovich. *Stability of Nonlinear Systems with Non Unique Equilibrium*. Moscow, CCCP, Izdatel'stvo Nauka, 1978.

[187] Denis Efimov. "Global Lyapunov Analysis of Multistable Nonlinear Systems". In: *SIAM Journal on Control and Optimization* 50 (2012), pp. 3132–3154.

[188] Rainer Hegger et al. "Coping with Nonstationaity by Overembedding". In: *Physical Review Letters* 84 (2000), pp. 4092–4095.

[189] Matthew B. Kennel. "Statistical test for dynamical nonstationarity in observed time-series data". In: *Physical Review E* 56 (1997), pp. 316–321.

[190] Masashi Sugiyama and Motoaki Kawanabe. *Machine Learning in Non-Stationary Environments; Introduction to Covariate Shift Adaptation*. MIT Press, 2012, 280 pp.

[191] Andrey Nikolayevich Tikhonov. "On the stability of inverse problems". In: *Doklady Akademii Nauk SSSR* 39 (1943), pp. 195–198.

[192] D. L. Phillips. "A Technique for the Numerical Solution of Certain Integral Equations of the First Kind". In: *Journal of the ACM.* 9 (1962), pp. 84–97.

[193] F. Takens. "Detecting strange attractors in turbulence". In: *Lecture Notes in Math.* 898 (1981), pp. 366–381.

[194] Jeffrey L. Elman. "Finding Structure in Time". In: *Cognitive Science* 14.2 (1990), pp. 179–211. DOI: https://doi.org/10.1207/s15516709cog1402_1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1207/s15516709cog1402_1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1207/s15516709cog1402_1.

[195] Mantas Luko**v**sevi**v**cius. "A Practical Guide to Applying Echo State Networks". In: (Jan. 2012), pp. 659–686. DOI: 10.1007/978-3-642-35289-8_36.

[196] D. Verstraeten, B. Schrauwen, and D. Stroobandt. "Reservoir-based techniques for speech recognition". In: *The 2006 IEEE International Joint Conference on Neural Network Proceedings*. 2006, pp. 1050–1053. DOI: 10.1109/IJCNN.2006.246804.

[197] Steven H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry and Engineering*. Westview Press, 2000.

[198] David Verstraeten and Benjamin Schrauwen. "On the Quantification of Dynamics in Reservoir Computing". In: *Artificial Neural Networks – ICANN 2009*. Ed. by Cesare Alippi et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 985–994. ISBN: 978-3-642-04274-4.

[199] Guanrong Chen. "Stability of Nonlinear Systems". In: Apr. 2005. DOI: 10.1002/0471654507.eme413.

[200] Karlheinz Geist, Ulrich Parlitz, and Werner Lauterborn. "Comparison of Different Methods for Computing Lyapunov Exponents". In: *Progress of Theoretical Physics* 83.5 (May 1990), pp. 875–893. ISSN: 0033-068X. DOI: 10.1143/PTP.83.875.

[201] Lyudmila Grigoryeva, Allen Hart, and Juan-Pablo Ortega. "Chaos on compact manifolds: Differentiable synchronizations beyond the Takens theorem". In: *Phys. Rev. E* 103 (6 June 2021), p. 062204. DOI: 10.1103/PhysRevE.103.062204.

[202] Lyudmila Grigoryeva, Allen Hart, and Juan-Pablo Ortega. *Learning strange attractors with reservoir systems*. 2021. arXiv: 2108.05024 [math.DS].

[203] Louis M. Pecora and Thomas L. Carroll. "Synchronization in chaotic systems". In: *Phys. Rev. Lett.* 64 (8 Feb. 1990), pp. 821–824. DOI: 10.1103/PhysRevLett.64.821.

[204] Li Deng. "The mnist database of handwritten digit images for machine learning research". In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[205] *List of datasets for machine-learning research*. 2021. URL: https://en.wikipedia.org/wiki/List_of_datasets_for_machine-learning_research.

[206] M. Peña and E. Kalnay. "Separating fast and slow modes in coupled chaotic systems". In: *Nonlinear Processes in Geophysics* 11.3 (2004), pp. 319–327. DOI: 10.5194/npg-11-319-2004. URL: https://npg.copernicus.org/articles/11/319/2004/.

[207] J. Mo**v**ckus. "On bayesian methods for seeking the extremum". In: *Optimization Techniques IFIP Technical Conference Novosibirsk, July 1–7, 1974*. Ed. by G. I.

Marchuk. Berlin, Heidelberg: Springer Berlin Heidelberg, 1975, pp. 400–404. ISBN: 978-3-540-37497-8.

[208] Donald Jones, Matthias Schonlau, and William Welch. "Efficient Global Optimization of Expensive Black-Box Functions". In: *Journal of Global Optimization* 13 (Dec. 1998), pp. 455–492. DOI: 10.1023/A:1008306431147.

[209] Mohamed Amine Bouhlel et al. "A Python surrogate modeling framework with derivatives". In: *Advances in Engineering Software* 135 (2019), p. 102662. ISSN: 0965-9978. DOI: https://doi.org/10.1016/j.advengsoft.2019.03.005.

[210] Joschka Herteux and Christoph Räth. "Breaking symmetries of the reservoir equations in echo state networks". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 30.12 (2020), p. 123142. DOI: 10.1063/5.0028993.

[211] Wendson A. S. Barbosa et al. "Symmetry-aware reservoir computing". In: *Phys. Rev. E* 104 (4 Oct. 2021), p. 045307. DOI: 10.1103/PhysRevE.104.045307.

[212] Leslie E. Matson. "The Malkus–Lorenz water wheel revisited". In: *American Journal of Physics* 75.12 (2007), pp. 1114–1122. DOI: 10.1119/1.2785209. URL: https://doi.org/10.1119/1.2785209.

[213] Lucas Illing et al. "Experiments with a Malkus–Lorenz water wheel: Chaos and Synchronization". In: *American Journal of Physics* 80.3 (2012), pp. 192–202. DOI: 10.1119/1.3680533. URL: https://doi.org/10.1119/1.3680533.

[214] Zhixin Lu et al. "Reservoir observers: Model-free inference of unmeasured variables in chaotic systems". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 27.4 (2017), p. 041102.

[215] Craig H Bishop, Brian J Etherton, and Sharanya J Majumdar. "Adaptive sampling with the ensemble transform Kalman filter. Part I: Theoretical aspects". In: *Monthly weather review* 129.3 (2001), pp. 420–436.

[216] Brian R Hunt, Eric J Kostelich, and Istvan Szunyogh. "Efficient data assimilation for spatiotemporal chaos: A local ensemble transform Kalman filter". In: *Physica D: Nonlinear Phenomena* 230.1-2 (2007), pp. 112–126.

[217] P. W. Anderson. “More Is Different”. In: *Science* 177.4047 (1972), pp. 393–396. ISSN: 00368075, 10959203. URL: http://www.jstor.org/stable/1734697 (visited on 04/04/2022).

[218] M. Raissi, P. Perdikaris, and G.E. Karniadakis. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378 (2019), pp. 686–707. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2018.10.045.

[219] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. “Discovering governing equations from data by sparse identification of nonlinear dynamical systems”. In: *Proceedings of the National Academy of Sciences* 113.15 (2016), pp. 3932–3937. DOI: 10.1073/pnas.1517384113.

[220] O.E. Rössler. “An equation for continuous chaos”. In: *Physics Letters A* 57.5 (1976), pp. 397–398. ISSN: 0375-9601. DOI: https://doi.org/10.1016/0375-9601(76)90101-8.

[221] Edward N Lorenz. “Predictability: A problem partly solved”. In: *Predictability of weather and climate*. Ed. by Tim Palmer and Renate Hagedorn. Cambridge, 2006.