2022-08-03

Data Assimilation and Parameter Recovery for Rayleigh-Bénard Convection

Jacob William Murri
Brigham Young University

Data Assimilation and Parameter Recovery for Rayleigh-Bénard Convection

Jacob William Murri

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Jared P Whitehead, Chair
Lennard Bakker
Blake Barker

Department of Mathematics

Brigham Young University

ABSTRACT

Data Assimilation and Parameter Recovery for Rayleigh-Bénard Convection

Jacob William Murri
Department of Mathematics, BYU
Master of Science

Many problems in applied mathematics involve simulating the evolution of a system using differential equations with known initial conditions. But what if one records observations and seeks to determine the causal factors which produced them? This is known as an inverse problem. Some prominent inverse problems include data assimilation and parameter recovery, which use partial observations of a system of evolutionary, dissipative partial differential equations to estimate the state of the system and relevant physical parameters (respectively). Recently a set of procedures called nudging algorithms have shown promise in performing simultaneous data assimilation and parameter recovery for the Lorentz equations and the Kuramoto-Sivashinsky equation. This work applies these algorithms and extensions of them to the case of Rayleigh-Bénard convection, one of the most ubiquitous and commonly-studied examples of turbulent flow. The performance of various parameter update formulas is analyzed through direct numerical simulation. Under appropriate conditions and given the correct parameter update formulas, convergence is also established, and in one case, an analytical proof is obtained.

The Python source code for methods are contained in an open-source GitHub repository at https://github.com/jwp37/RB_parameters.

Keywords: data assimilation, parameter recovery, partial differential equations, nudging, Rayleigh-Bénard convection

Acknowledgements

First, I would like to thank my advisor, Jared Whitehead, for proposing an interesting and difficult problem to work on and providing helpful guidance along the way. I would also like to thank Shane McQuarrie, whose code implementing data assimilation methods for Rayleigh-Bénard convection is very well-documented and served as a foundation for this project's codebase. Further thanks is owed to Vincent Martinez for working through the analysis and helping to clarify ideas, and to Benjamin Pachev for discussions which helped resolve the problem of basis selection for the PWM algorithm.

Second, I would like to thank all the BYU faculty, especially those in the ACME program, and those who served as research mentors to me as an undergraduate (Lennard Bakker, Tyler Jarvis, and Benjamin Webb) The skills and tools I learned from my research and undergraduate math classes at BYU, especially in Python programming, scientific computing, partial differential equations, and mathematical analysis have been invaluable.

I would also like to thank the Brigham Young University College of Physical and Mathematical Sciences, whose generous funding in the form of a Dean's Fellowship made this work (in addition to the coursework required to obtain the masters degree in mathematics) possible to complete in one year.

Lastly, I want to thank my wonderful wife, Lillian, for her love and support throughout my masters degree. She has helped me gain motivation during hard times and learn how to manage my time more effectively in the face of large projects without many deadlines.

Contents

List of Tables

List of Figures

Chapter 1. Introduction

Because so many of the important physical systems in the universe relate to liquids and gases, understanding and predicting fluid dynamics is a crucial scientific endeavor. Indeed, the Earth's oceans, atmosphere, and mantle all represent fluids governed by the laws of physics at varying scales. Modeling fluid motion is also essential in many areas of engineering, including hydraulics, aerospace, industrial, thermal, and chemical. One of the greatest contributions of mathematics to the physical sciences is the ability to model fluids using partial differential equations (PDEs). Applying well-known conservation laws (like conservation of mass, energy, and momentum) to various mathematical objects representing different classes of fluids has led to many different PDE models for fluid dynamics (including the wave equation, Euler equations for an inviscid fluid, the Korteweg—De Vries equation, and the Navier-Stokes equations for compressible and incompressible fluids, among many others). Such PDEs can provide insight into the dynamical evolution of the fluid for many systems when subjected to the tools of mathematical analysis and numerical simulation. In the practical sense, analysis, simulation, and experimental study of fluid dynamics have led to dramatically more efficient design of airplane and jet wings, and improved design of heating and cooling systems in urban interiors such as office buildings.

This work will focus on a particular type of fluid system that may be called "convection due to heating from below." The term "convection" refers to fluid motion caused by temperature differences [1]. For many fluids, this motion is caused by the expansion of the fluid as it is heated (or rather, the relative difference in density between hot and cold fluid). Convection is a well-known physical phenomenon that has been studied for centuries due to its ubiquity. It has been argued, "convection due to nonuniform heating is, without overstatement, the most widespread type of fluid motion in the Universe" [2]. Convection produces chaotic dynamics which elude most classical analysis techniques. Therefore convection is also studied as a prototypical example of a chaotic system. Perhaps the simplest form of

convection, and the one focused on herein, is convection due to heating from below (hereafter referred to as Rayleigh-Bénard convection). Readers likely have experience with this type of convection, as it occurs in commonplace situations like boiling water on a stove (in addition to less commonly-experienced but more fundamental systems like the interior of the earth). As a mathematical representation of this system, consider an infinite horizontal fluid layer bounded by infinite planes above and below, where the temperature is kept at a constant hot temperature on the bottom and a constant cold temperature on the top. Section 1.1 introduces the relevant governing equations and mathematical formalism.

One might expect that if the physical first principles of the mathematical model are correct, the model will produce good predictions which match what occurs in the real world. However this is not always the case. A model may be only approximately correct (or very far from correct), and errors can be introduced in a multitude of ways. Some of these include

- Measurement error. Many PDE models start from an initial state and evolve in time. However, if the initial state is measured incorrectly, if noise is present in the measurements, or if measurement of the full state of the system is impossible (for example, measuring the full state of every particle in the atmosphere), then the predictions the model makes may not match reality. The field of data assimilation seeks to overcome this type of error /y optimally combining noisy and uncertain measurements with a mathematical model derived from first principles.

- Model error. The derivation of many PDEs which model physical systems include many simplifying assumptions which may not always reflect reality exactly. Furthermore, many models contain parameters which many be unknown or uncertain. If these parameters are incorrect, the model may lead to incorrect predictions no matter how detailed or accurate the measurements are. Determining the correct parameters may be characterized as an "inverse problem" because it involves estimating the discrepancy between the model's predictions and reality, and propagating that uncertainty backward rather than forward to determine the relevant differences in the model pa-

rameters.

- Numerical error. Simulating the evolution of a PDE on a computer necessarily involves discretizing both the domain of the system (using some kind of finite grid, or a projection onto a finite dimensional subspace) and the variables of a system (using floating-point arithmetic), and thus introduces some error.

This work seeks to demonstrate methods for overcoming certain types of measurement error and model error when modeling Rayleigh-Bénard convection. To overcome measurement error due to incomplete, inaccurate, or unavailable initial data, a data assimilation technique called nudging, which has been applied to this and several other systems successfully, is employed [3, 4, 5, 6]. Using the nudging approach along with a parameter estimation algorithm, it is possible to overcome model error due to errors in the control parameters (here, the Rayleigh and Prandtl numbers) This could be called parameter estimation, parameter recovery, or parameter calibration. Furthermore, it is possible to do both of these simultaneously (i.e. recover the parameters and state of the system when neither is known exactly).

1.1 Boussinesq Equations

To construct a mathematical model for Rayleigh-Bénard convection, suppose that the fluid in question may be modeled as a continuum, with the fluid velocity at a point (x, z) (this work considers convection in only two spatial dimensions) at time t given by a velocity field $\mathbf{u} : [0, L] \times [0, h] \times \mathbb{R} \to \mathbb{R}^2$ defined on a the two-dimensional box. The fluid must satisfy boundary conditions ((1.8), (1.5)) at the vertical boundaries of the box, and is periodic in the horizontal direction (see (1.6)). For the sake of simplicity, further assume that the fluid is incompressible, which implies that its density is nearly constant. Then mass and momentum

conservation dictate that the fluid obeys the Navier-Stokes equations

$$\begin{gathered}\rho_0 \frac{\partial \mathbf{u}}{\partial t} + \rho_0(\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \eta \Delta \mathbf{u} + \mathbf{F} \\ \nabla \cdot \mathbf{u} = 0,\end{gathered} \tag{1.1}$$

where $\mathbf{F}$ is the external body force per unit volume, ρ_0 is the (nearly constant) density of the fluid, p is the pressure, and η is the dynamic viscosity. In Rayleigh-Bénard convection, the forcing on the fluid is due to buoyancy from temperature differences (which cause small variations in the density of the fluid). Let $\theta : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$ be the scalar temperature field, and assume that the fluid is expansive (meaning it becomes less dense at higher temperatures) with expansion coefficient α. This means that if the mean density of the fluid is ρ_0, then the change in density due to change in temperature is given by

$$\Delta \rho = -\alpha \rho_0 \Delta \theta. \tag{1.2}$$

Letting $\mathbf{z}$ be the unit vector which points in the upward direction, the buoyancy force density is

$$\mathbf{F} = -g \Delta \rho (-\hat{\mathbf{z}}) = g \alpha \rho_0 (\theta - \theta_0) \hat{\mathbf{z}} \tag{1.3}$$

Setting the temperature at the top ($z = h$) of the box to θ_0, the next step is to substitute (1.3) into (1.1). First, it should be noted that (1.1) and (1.3) employ slightly different assumptions. (1.1) assumes that the density of the fluid is essentially constant, while (1.3) assumes that the density can vary when the temperature of the fluid changes. The *Boussinesq approximation* assumes that differences in density only affect the gravitational force, so it is reasonable to include the effect of density change in (1.3) but not in (1.1). Defining $\nu = \eta/\rho_0$, the kinematic viscosity, and making the substitution, one obtains the dimensional Boussinesq

equations

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} &= -\frac{1}{\rho_0}\nabla p + \nu\Delta\mathbf{u} + g\alpha\theta\hat{\mathbf{z}}, \\ \nabla\cdot\mathbf{u} &= 0, \\ \frac{\partial\theta}{\partial t} + \mathbf{u}\cdot\nabla\theta &= \kappa\Delta\theta, \end{aligned} \tag{1.4}$$

where the third equation represents the advection and diffusion of temperature, with the thermal diffusion constant κ. As mentioned earlier, the box is heated from below and cooled on top, so the temperature boundary conditions

$$\theta|_{z=0} = \theta_0 + \delta\theta, \qquad \theta|_{z=h} = \theta_0, \tag{1.5}$$

apply to this problem. This work considers Rayleigh-Bénard convection along with the periodic boundary conditions

$$\theta|_{x=0} = \theta|_{x=L}, \qquad \theta_x|_{x=0} = \theta_x|_{x=L}, \qquad \theta_z|_{x=0} = \theta_z|_{x=L} \tag{1.6}$$

$$\mathbf{u}|_{x=0} = \mathbf{u}|_{x=L}, \qquad \mathbf{u}_x|_{x=0} = \mathbf{u}_x|_{x=L}, \qquad \mathbf{u}_z|_{x=0} = \mathbf{u}_z|_{x=L}, \tag{1.7}$$

and the *no-slip* boundary conditions for the velocity field at the top and bottom plates

$$\mathbf{u}|_{z=0} = \mathbf{u}|_{z=h} = 0. \tag{1.8}$$

1.2 Nondimensionalization

Nondimensionalization involves picking characteristic scales for the relevant physical quantities in the problem (1.4) so that can redefine all of the variables as dimensionless versions of themselves, and identify the truly important dimensionless parameters in the system. All of the variables in the system (1.4) are constructed out of units of length, time, temperature, and mass, so constructing a characteristic scale for each (call them $[L]$, $[T]$, $[\Theta]$, and $[M]$,

respectively) using the six independent physical constants $\rho_0, h, \kappa, \delta\theta, \nu$, and αg which are contained in the system will be sufficient to produce a nondimensionalized set of equations (note that α and g are kept together because they never appear separately in the system equations).

A characteristic mass scale is relevant to only one of the physical constants in the problem, ρ_0. Therefore the mass scale must be chosen in terms of ρ_0. Since the units of ρ_0 are $[M][L]^{-3}$, a good choice would be $[M] = \rho_0[L]^3$.

It remains to choose the scales $[L], [T]$, and $[\Theta]$. The most common nondimensionalization chooses to set the coefficient on the temperature diffusivity term to unity. This may be thought of as selecting the thermal diffusive time scale as the relevant time scale for the problem. It requires choosing the characteristic scales

$$[L] = h, \qquad [T] = h^2/\kappa, \qquad [\Theta] = \delta\theta, \qquad [M] = \rho_0 h^3$$

which results in a pressure scale $\rho_0\kappa^2/h^2$ and velocity scale κ/h. Rescaling all variables appropriately gives

$$\begin{aligned}
\frac{\kappa^2}{h^3}\frac{\partial \mathbf{u}}{\partial t} + \frac{\kappa^2}{h^3}(\mathbf{u}\cdot\nabla)\mathbf{u} &= -\frac{\kappa^2}{h^3}\nabla p + \frac{\kappa\nu}{h^3}\Delta\mathbf{u} + (g\alpha\delta\theta)\,\theta\hat{\mathbf{z}}, \\
\frac{\kappa}{h^2}\nabla\cdot\mathbf{u} &= 0, \\
\frac{\delta\theta\kappa}{h^2}\frac{\partial\theta}{\partial t} + \frac{\delta\theta\kappa}{h^2}\mathbf{u}\cdot\nabla\theta &= \frac{\delta\theta\kappa}{h^2}\Delta\theta,
\end{aligned}$$

where the quantities $\mathbf{u}, \theta, t, p$ are now dimensionless. Dividing out factors and making the definitions

$$\mathrm{Pr} = \frac{\nu}{\kappa}, \qquad \mathrm{Ra} = \frac{g\alpha\delta\theta h^3}{\nu\kappa},$$

the equations become

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p &= \Pr \Delta \mathbf{u} + \Pr \mathrm{Ra}\, \theta \hat{\mathbf{z}}, \\ \nabla \cdot \mathbf{u} &= 0, \\ \frac{\partial \theta}{\partial t} + \mathbf{u}\cdot\nabla\theta &= \Delta\theta. \end{aligned} \tag{1.9}$$

The dimensionless quantities Pr and Ra are known as the Prandtl and Rayleigh numbers respectively. Pr expresses the ratio of momentum to thermal diffusivity, and Ra represents the strength of the buoyancy forcing. The Rayleigh number Ra also designates whether the flow is in a turbulent or laminar regime (the larger the Rayleigh number, the more turbulent the flow due to an increased effective thermal forcing).

Another important nondimensionalization is formulated so that the parameters Pr and Ra are in separate equations. To obtain it, multiply the temperature scale by Pr while dividing the time scale by Pr:

$$[L] = h, \qquad [T] = h^2/\nu, \qquad [\Theta] = \Pr\, \delta\theta, \qquad [M] = \rho_0 h^3$$

which results in a pressure scale $\rho_0\nu^2/h^2$ and velocity scale ν/h (this is equivalent to using a viscous time-scale as the dominant time-scale). In this case, rescaling all variables gives

$$\begin{aligned} \frac{\nu^2}{h^3}\frac{\partial \mathbf{u}}{\partial t} + \frac{\nu^2}{h^3}(\mathbf{u}\cdot\nabla)\mathbf{u} &= -\frac{\nu^2}{h^3}\nabla p + \frac{\nu^2}{h^3}\Delta\mathbf{u} + (g\alpha \Pr\, \delta\theta)\,\theta\hat{\mathbf{z}}, \\ \frac{\nu}{h^2}\nabla\cdot\mathbf{u} &= 0, \\ \Pr\frac{\delta\theta\nu}{h^2}\frac{\partial\theta}{\partial t} + \Pr\frac{\delta\theta\nu}{h^2}\mathbf{u}\cdot\nabla\theta &= \Pr\frac{\delta\theta\kappa}{h^2}\Delta\theta \end{aligned}$$

which simplifies to

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p &= \Delta \mathbf{u} + \mathrm{Ra}\,\theta\hat{\mathbf{z}}, \\ \nabla\cdot\mathbf{u} &= 0, \\ \frac{\partial\theta}{\partial t} + \mathbf{u}\cdot\nabla\theta &= \frac{1}{\mathrm{Pr}}\Delta\theta. \end{aligned} \tag{1.10}$$

1.2.1 Streamfunction-Vorticity Form. Let $\mathbf{u} = (v, w)$. The equation $\nabla\cdot\mathbf{u} = 0$ is equivalent to the statement $v_x + w_z = 0$. Define a scalar *streamfunction* ψ so that $\mathbf{u} = (-\psi_z, \psi_x)$. Writing $\mathbf{u}$ in this way eliminates the need for the equation $\nabla\cdot\mathbf{u} = 0$ because $\psi_{xz} = \psi_{zx}$. Then, define the scalar *vorticity*

$$\zeta := \Delta\psi = \psi_{xx} + \psi_{zz} = w_x - v_z,$$

which is the curl of $\mathbf{u}$ (it is a scalar because $\mathbf{u}$ is two-dimensional). To write an evolution equation for ζ, first explicitly write down the equations for v and w,

$$\begin{aligned} v_t + vv_x + wv_z + p_x &= \mathrm{Pr}\,(v_{xx} + v_{zz}) \\ w_t + vw_x + ww_z + p_z &= \mathrm{Pr}\,(w_{xx} + w_{zz}) + \mathrm{Pr}\,\mathrm{Ra}\,\theta, \end{aligned}$$

then differentiate the first with respect to z and the second with respect to x to find evolution equations for v_z and w_x:

$$\begin{aligned} v_{tz} + v_z v_x + vv_{xz} + w_z v_z + wv_{zz} + p_{xz} &= \mathrm{Pr}\,(v_{xxz} + v_{zzz}) \\ w_{tx} + v_x w_x + vw_{xx} + w_x w_z + ww_{zx} + p_{zx} &= \mathrm{Pr}\,(w_{xxx} + w_{zzx}) + \mathrm{Pr}\,\mathrm{Ra}\,\theta_x \end{aligned}$$

Then subtracting the first from the second (assuming appropriate smoothness) yields

$$\zeta_t + v_x\zeta + v\zeta_x + w_z\zeta + w\zeta_z = \mathrm{Pr}\,\Delta\zeta + PrRa\theta_x.$$

Using $v_x + w_z = 0$, it follows that

$$\frac{\partial \zeta}{\partial t} + \mathbf{u} \cdot \nabla \zeta = \mathrm{Pr}\, \Delta \zeta + \mathrm{Pr}\, \mathrm{Ra}\, \theta_x.$$

1.3 Nudging Framework

The nudging data assimilation algorithm employed here was first introduced for the 2D Navier-Stokes equations in [7]. In the context of Rayleigh-Bénard convection, the *true system*

$$\begin{aligned} \frac{\partial \zeta}{\partial t} + \mathbf{u} \cdot \nabla \zeta &= \mathrm{Pr}\, \Delta \zeta + \mathrm{Pr}\, \mathrm{Ra}\, \theta_x, \\ \frac{\partial \theta}{\partial t} + \mathbf{u} \cdot \nabla \theta &= \Delta \theta, \end{aligned} \tag{1.11}$$

is coupled with the *nudged system* or *assimilating system*

$$\begin{aligned} \frac{\partial \tilde{\zeta}}{\partial t} + \tilde{\mathbf{u}} \cdot \nabla \tilde{\zeta} &= \widetilde{\mathrm{Pr}} \Delta \tilde{\zeta} + \widetilde{\mathrm{Pr}}\widetilde{\mathrm{Ra}} \tilde{\theta}_x + \mu P_N(\zeta - \tilde{\zeta}), \\ \frac{\partial \tilde{\theta}}{\partial t} + \tilde{\mathbf{u}} \cdot \nabla \tilde{\theta} &= \Delta \tilde{\theta}, \end{aligned} \tag{1.12}$$

where P_N is an observation operator which represents observations available to the observer (i.e. the observer can only have partial information about the state). The rest of this work will assume that P_N is a linear projection operator to simplify the calculations. It should be emphasized that this is likely not necessary, but does significantly simplify the underlying analysis and calculations. As mentioned previously, this and similar nudging algorithms have been studied extensively for the convection problem where it has been shown rigorously and computationally that the full state can be adequately recovered (meaning that $\tilde{\zeta} \to \zeta$ and $\tilde{\theta} \to \theta$ as $t \to \infty$) for various types of observations under a variety of different assumptions [3, 4, 5, 6].

Chapter 2. Parameter Estimation Algorithms

The goal of this chapter is to introduce and derive several algorithms which utilize partial observations of the state of a dynamical system over time to concurrently estimate the true state and true values of some of the system's unknown parameters. The algorithms introduced here are first introduced in a general case, then applied to Rayleigh-Bénard convection specifically. In all cases, the assumption is that observations are accurate but incomplete (there is no observational error). This fact is represented mathematically by saying that there is some linear projection P_N which maps from true states to observed states, discarding some information along the way. In applications, this incompleteness often arises from being unable to observe continuously in space; there must necessarily be a discrete grid of sensors measuring the desired quantity which may have large gaps. For example, when measuring temperature or pressure in the atmosphere, the stations or balloons or sensors which make measurements may be spaced hundreds of miles apart. The mathematical analysis that follows usually considers orthogonal projections onto Fourier modes, because if one is able to observe data from N data points, one is able to calculate the first N Fourier modes of the data. One type of error that the models in this work do not consider is observational error or error due to noise. Specifically, it is assumed that the observations, while incomplete, are exact and accurate, and that they are essentially continuous in time.

The simplest paradigm for a parameter estimation algorithm is the "discrete" or "point-in-time update", which has been given the more memorable name "relax, then punch." At the beginning, the correct values of some system parameters are unknown, and the observer has only partial information about the system state. Nevertheless, given an initial guess for the values of the unknown system parameters, the algorithm may integrate forward a "nudged system" which forces relaxation of the assimilating state variables towards the observable

portion of the true state variables. After integrating the system forward over some interval of time, it is assumed that the error between the true system and the assimilating system is proportional to the model error which comes from the difference between the true parameter values and the estimated (or guessed) values. When this time is reached, the algorithm selects new values for the parameters using a procedure or formula (a few examples of which are outlined in this section) and updates them accordingly. This is the aspect of the algorithm referred to as the "punch". Then the procedure repeats itself; at each iteration the system is integrated forward farther in time and an instantaneous parameter update applied. The central idea of this algorithm is alternating between estimating the true state and estimating the true parameters. In the "relax" phase of the algorithm, integrating the nudged system forward in time essentially produces the best possible estimate of the full state of the system given the model error. Then the "punch" uses the better estimate of the state to produce a better estimate of the parameters, which is then integrated forward to obtain a better estimate of the state, and so on. As long as the error in the state can be bounded by some constant times the model error, certain non-degeneracy conditions are satisfied, and each successive state estimate and parameter estimate is better than the last, this algorithm will eventually estimate the true values of the parameters and the full state of the system, up to a small error.

2.1 Derivation of Algorithms

2.1.1 CHL Algorithm for Estimating a Single Parameter. The so-called "CHL" algorithm originates from a paper of Carlson, Hudson and Larios [8], hence its name. What follows is a general derivation of this algorithm, whereas the original presentation in [8] was specifically developed to estimate the viscosity in 2D Navier-Stokes. Consider the system

$$\dot{x} = \lambda L x + F(x) \tag{2.1}$$

on the domain $\Omega \subset \mathbb{R}^m$, with $x : \Omega \times [0, \infty) \to \mathbb{R}^n$, $\dot{x}$ representing its time derivative, L being a linear differential operator, $\lambda \in \mathbb{R}$, and F being a smooth nonlinear operator. Assume that x has either periodic or homogeneous Dirichlet boundary conditions on $\partial\Omega$, and that 2.1 is well-posed in the sense that $x(t) \in L^2(\Omega)$ for all times t and is sufficiently smooth. To derive the CHL algorithm for estimating the parameter λ it is also assumed that (2.1)) is coupled with the nudged system

$$\dot{\tilde{x}} = \tilde{\lambda} L\tilde{x} + F(\tilde{x}) - \mu P_N(\tilde{x} - x), \tag{2.2}$$

where $\mu > 0$ is a nudging parameter and P_N is a linear projection such that $P_N(x)$ represents the partial information from x available to the observer. It is assumed that P_N is idempotent, and commutes with L. Additional properties of P_N may be needed to ensure that the necessary theory outlined below can be rigorously established, but those assumptions are determined below as needed (see [7] for a further discussion on the required type of interpolant). An example of a projection that satisfies these properties is a projection onto N Fourier modes. It is further assumed that (2.1) is a dissipative system such that if $\tilde{\lambda} = \lambda$ in (2.2), then $\tilde{x} \to x$, i.e. the standard nudging algorithm originally introduced in [7] will work asymptotically in time. This framework is sufficiently broad that most cases of single-parameter estimation can be fit into this framework. To proceed with the derivation, let $u := \tilde{x} - x$ and write down the time-evolution of u:

$$\dot{u} = \dot{\tilde{x}} - \dot{x} = \tilde{\lambda} L\tilde{x} - \lambda Lx + F(\tilde{x}) - F(x) - \mu P_N(\tilde{x} - x).$$

Simplifying and using the simple identity

$$\tilde{a}\tilde{b} - ab = \tilde{a}(\tilde{b} - b) + (\tilde{a} - a)b,$$

we have that

$$\dot{u} = \tilde{\lambda} Lu + (\tilde{\lambda} - \lambda)Lx + F(\tilde{x}) - F(x) - \mu P_N u.$$

Then apply the projection P_N throughout and take an $L^2(\Omega)$-inner product with $P_N u$ to obtain

$$\langle P_N u, P_N \dot{u} \rangle = \tilde{\lambda} \langle P_N u, L P_N u \rangle + (\tilde{\lambda} - \lambda) \langle P_N u, L P_N x \rangle + \langle P_N u, P_N (F(\tilde{x}) - F(x)) \rangle - \mu \langle P_N u, P_N u \rangle.$$

The key identity here is that

$$\langle P_N u, P_N \dot{u} \rangle = \int_\Omega (P_N u) \frac{\partial}{\partial t} (P_N u) = \frac{1}{2} \frac{d}{dt} \int_\Omega |P_N u|^2 = \frac{1}{2} \frac{d}{dt} \|P_N u\|^2,$$

where $\| \cdot \|$ represents the norm on $L^2(\Omega)$. Hence it follows that

$$\frac{1}{2} \frac{d}{dt} \|P_N u\|^2 = \tilde{\lambda} \langle P_N u, L P_N u \rangle + (\tilde{\lambda} - \lambda) \langle P_N u, L P_N x \rangle + \langle P_N u, P_N (F(\tilde{x}) - F(x)) \rangle - \mu \|P_N u\|^2. \tag{2.3}$$

The central idea of the CHL algorithm is start out at time t_0 with a guess $\tilde{\lambda} = \lambda_0$. Then the system (2.1)-(2.2) is integrated forward. If the true value of λ were known and one set $\tilde{\lambda} = \lambda$, it is expected that nudging would eventually force $\tilde{x} \to x$, or in other words, $u \to 0$. In the case that $\tilde{\lambda} \neq \lambda$ (where the discrepancy between the two is referred to as model error), $\|u\|$ can be expected to decrease but not converge all of the way to zero. The key assumption is that there will be a time t_1 when the model error is the dominant factor preventing $\|u\|$ from decreasing further; and therefore at the time t_1,

$$\frac{d}{dt} \left[\|P_N u\|^2 \right] \Big|_{t=t_1} \approx 0. \tag{2.4}$$

Given that (2.4) holds, it is possible for the observer to solve (2.3) for the true value of λ, as long as quantities which the observer does not have access to are appropriately dealt with. One way of dealing with these terms is to eliminate them, which amounts to assuming that they are small. Another approach is to approximate the terms with data that the observer does have access to. Both of these approaches are outlined below.

Eliminate the terms. To justify eliminating terms from (2.3), note that $\|u\|$ is expected to be relatively small at $t = t_1$, so that terms which are quadratic in the error are negligibly small unless they are multiplied by μ (which may be very large). This assumption means that the terms

$$\langle P_N u, LP_N u\rangle, \text{ and } \langle P_N u, P_N(F(\tilde{x}) - F(x))\rangle,$$

should be neglected, leading to

$$0 \approx (\tilde{\lambda} - \lambda)\langle P_N u, LP_N x\rangle|_{t=t_1} - \mu\|P_N u\|^2|_{t=t_1}. \tag{2.5}$$

As long as $\langle P_N u, LP_N x\rangle|_{t=t_1} \neq 0$, solving for λ yields the relation

$$\lambda = \tilde{\lambda} - \mu\frac{\|P_N u\|^2}{\langle P_N u, LP_N x\rangle}\Big|_{t=t_1}$$

Up until t_1, the parameter estimate $\tilde{\lambda}$ has been set to the value λ_0. Then at time t_1, the CHL algorithm updates the parameter $\tilde{\lambda}$ to the new best guess as to the true value of the parameter λ:

$$\tilde{\lambda} = \lambda_1 := \lambda_0 - \mu\frac{\|P_N u\|^2}{\langle P_N u, LP_N x\rangle}\Big|_{t=t_1}.$$

This process is then repeated by integrating forward again until $\|u\|$ stops decreasing again, choosing a new λ_2, etc. In general, the update formula is given by

$$\lambda_n := \lambda_{n-1} - \mu\frac{\|P_N u\|^2}{\langle P_N u, LP_N x\rangle}\Big|_{t=t_n}.$$

Estimate the terms. Another approach is to estimate the nonlinear term using the best information about x available to the observer, namely $P_N x$. This means making the substitution

$$\langle P_N u, P_N(F(\tilde{x}) - F(x))\rangle \approx \langle P_N u, P_N(F(P_N\tilde{x}) - F(P_N x))\rangle,$$

so that at time $t = t_1$

$$0 \approx \tilde{\lambda}\langle P_N u, LP_N u\rangle|_{t=t_1} + (\tilde{\lambda}-\lambda)\langle P_N u, LP_N x\rangle|_{t=t_1} + \langle P_N u, P_N(F(P_N\tilde{x}) - F(P_N x))\rangle|_{t=t_1} - \mu\|P_N u\|^2|_{t=t_1},$$

so if $\langle P_N u, LP_N x\rangle|_{t=t_1} \neq 0$, then

$$\lambda = \frac{\tilde{\lambda}\langle P_N u, LP_N\tilde{x}\rangle + \langle P_N u, P_N(F(P_N\tilde{x}) - F(P_N x))\rangle - \mu\|P_N u\|^2}{\langle P_N u, LP_N x\rangle}\Big|_{t=t_1}.$$

As before, this leads to the parameter update formula

$$\lambda_n = \frac{\lambda_{n-1}\langle P_N u, LP_N\tilde{x}\rangle + \langle P_N u, P_N(F(P_N\tilde{x}) - F(P_N x))\rangle - \mu\|P_N u\|^2}{\langle P_N u, LP_N x\rangle}\Big|_{t=t_n}.$$

Nondegeneracy Condition. In each case above it has been necessary to assume that $\langle P_N u, LP_N x\rangle \neq 0$ at the parameter update time. This requirement, commonly referred to as a *nondegeneracy condition*, is worth elaborating on. If at a given time $t = t_1$ this condition does not hold, i.e. $\langle P_N u, LP_N x\rangle = 0$, then it is impossible to solve (2.3) for the true value of λ because that term vanishes. This makes sense because under this condition, all values of λ would satisfy (2.3); hence the equation is degenerate. Looking back at the system evolution equations (2.1)-(2.2) it is clear that if $Lx = 0$ at a particular time (which would force $\langle P_N u, LP_N x\rangle = 0$), then solving for λ is impossible; any value of λ would lead to the same dynamics if $Lx = 0$ held over an interval of time.

However, this requirement is not particularly onerous to satisfy. If the requirement is not satisfied at a particular time t, then the algorithm can usually wait longer until it is satisfied. If this is not the case, then the CHL algorithm will break, i.e. CHL can *not* recover the true parameter when this non-degeneracy condition is *not* satisfied.

Estimating Ra **with Known** Pr**.** Now that the CHL algorithm has been derived in general, the discussion that follows will begin to apply it to the problem of estimating parameters in the vorticity-temperature formulation of Rayleigh-Bénard convection. For

now, suppose that the Prandtl number Pr is known, but the Rayleigh number Ra is unknown, and that information about the full state $(\mathbf{u}, \theta)$ is unavailable; the observer only has access to $(P_N(\mathbf{u}), P_N(\theta))$; where P_N is a projection onto a finite number N of Fourier modes. This system, in vorticity-temperature formulation, is

$$\begin{aligned}
&\frac{\partial \zeta}{\partial t} + \mathbf{u} \cdot \nabla \zeta = \Pr \Delta \zeta + \Pr \operatorname{Ra} \theta_x, \\
&\frac{\partial \theta}{\partial t} + \mathbf{u} \cdot \nabla \theta = \Delta \theta, \\
&\frac{\partial \tilde{\zeta}}{\partial t} + \tilde{\mathbf{u}} \cdot \nabla \tilde{\zeta} = \Pr \Delta \tilde{\zeta} + \Pr \widetilde{\operatorname{Ra}} \, \tilde{\theta}_x - \mu P_N \left(\tilde{\zeta} - \zeta \right), \\
&\frac{\partial \tilde{\theta}}{\partial t} + \tilde{\mathbf{u}} \cdot \nabla \tilde{\theta} = \Delta \tilde{\theta}.
\end{aligned} \tag{2.6}$$

where $\mu > 0$ is a nudging parameter. Now let $w := \tilde{\zeta} - \zeta$ be the error in the vorticity. Then the time evolution of w can be written as

$$\frac{\partial w}{\partial t} = \frac{\partial \tilde{\zeta}}{\partial t} - \frac{\partial \zeta}{\partial t} = - \left(\tilde{\mathbf{u}} \cdot \nabla \tilde{\zeta} - \mathbf{u} \cdot \nabla \zeta \right) + \Pr \Delta \left(\tilde{\zeta} - \zeta \right) + \Pr \left(\widetilde{\operatorname{Ra}} \, \tilde{\theta}_x - \operatorname{Ra} \theta_x \right) - \mu P_N \left(\tilde{\zeta} - \zeta \right).$$

Simplifying and applying the projection operator P_N throughout (which as noted, commutes with derivatives, is linear, and is idempotent), it follows that

$$P_N(w_t) = -P_N \left(\tilde{\mathbf{u}} \cdot \nabla \tilde{\zeta} - \mathbf{u} \cdot \nabla \zeta \right) + \Pr P_N(\Delta w) + \Pr \left(\widetilde{\operatorname{Ra}} \, P_N(\tilde{\theta}_x) - \operatorname{Ra} P_N(\theta_x) \right) - \mu P_N w.$$

Letting $\mathbf{v} := \tilde{\mathbf{u}} - \mathbf{u}$, and $\eta := \tilde{\theta} - \theta$, it follows that

$$\begin{aligned}
\tilde{\mathbf{u}} \cdot \nabla \tilde{\zeta} - \mathbf{u} \cdot \nabla \zeta &= \tilde{\mathbf{u}} \cdot \nabla w + \mathbf{v} \cdot \nabla \zeta, \\
\widetilde{\operatorname{Ra}} \, \tilde{\theta}_x - \operatorname{Ra} \theta_x &= \widetilde{\operatorname{Ra}} \, \eta_x + \left(\widetilde{\operatorname{Ra}} - \operatorname{Ra} \right) \theta_x,
\end{aligned}$$

so

$$P_N(w_t) = -P_N(\tilde{\mathbf{u}} \cdot \nabla w) - P_N(\mathbf{v} \cdot \nabla \zeta) + \Pr P_N(\Delta w) + \Pr \widetilde{\operatorname{Ra}} P_N(\eta_x) + \Pr \left(\widetilde{\operatorname{Ra}} - \operatorname{Ra} \right) P_N(\theta_x) - \mu P_N w.$$

At this point, take the L^2-inner product of both sides with $P_N w$. Then

$$\langle P_N w, P_N(w_t)\rangle = \int_\Omega P_N w P_N w_t = \int_\Omega \frac{\partial}{\partial t}\left[\frac{1}{2}P_N w^2\right] = \frac{1}{2}\frac{d}{dt}\int_\Omega P_N w^2 = \frac{1}{2}\frac{d}{dt}\left[\|P_N w\|^2\right]$$

and

$$\langle P_N w, P_N(\Delta w)\rangle = \int_\Omega P_N w \Delta P_N w = -\int_\Omega |\nabla P_N w|^2 = -\|\nabla P_N w\|^2$$

so that the equation becomes

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}\left[\|P_N w\|^2\right] = &-\langle P_N w, P_N(\tilde{\mathbf{u}}\cdot\nabla w)\rangle - \langle P_N w, P_N(\mathbf{v}\cdot\nabla\zeta)\rangle - \mathrm{Pr}\,\|\nabla P_N w\|^2 \\ &+ \mathrm{Pr}\,\widetilde{\mathrm{Ra}}\langle P_N w, P_N(\eta_x)\rangle + \mathrm{Pr}\left(\widetilde{\mathrm{Ra}} - \mathrm{Ra}\right)\langle P_N w, P_N(\theta_x)\rangle - \mu\|P_N w\|^2.\end{aligned} \tag{2.7}$$

Following the derivation above, integrate the system forward until time $t = t_1$ at which $\|w\|$ stops decreasing, i.e.

$$\frac{d}{dt}\left[\|P_N w\|^2\right]\Big|_{t=t_1} \approx 0. \tag{2.8}$$

Then one can (approximately) solve (2.3) for the true value of Ra. If one eliminates terms quadratic in the error (except those coupled with μ) and assume that the nondegeneracy condition for this algorithm holds, i.e.

$$\langle P_N w, P_N(\theta_x)\rangle|_{t=t_1} \neq 0,$$

then one can solve for the (approximate) true value of Ra and assign it to be the next estimate $\widetilde{\mathrm{Ra}}$:

$$\widetilde{\mathrm{Ra}}|_{t>t_1} = \widetilde{\mathrm{Ra}}|_{t\in[t_0,t_1)} - \frac{\mu}{\mathrm{Pr}}\frac{\|P_N w\|^2}{\langle P_N w, P_N(\theta_x)\rangle}\Big|_{t=t_1}.$$

If, instead, one tries to estimate the terms rather than eliminate them, then the update becomes

$$\widetilde{\mathrm{Ra}}|_{t>t_1} = \frac{\widetilde{\mathrm{Ra}}|_{t\in[t_0,t_1)}\langle P_N w, P_N(\tilde{\theta}_x)\rangle - \frac{1}{\mathrm{Pr}}\langle P_N w, P_N(g)\rangle - \|\nabla P_N w\|^2 - \frac{\mu}{\mathrm{Pr}}\|P_N w\|^2}{\langle P_N w, P_N(\theta_x)\rangle}\Big|_{t=t_1}.$$

where

$$g := P_N(\tilde{\mathbf{u}}) \cdot P_N(\nabla w) + P_N(\mathbf{v}) \cdot P_N(\nabla \zeta). \tag{2.9}$$

While this update formula is much more complicated than the first one, it does not drastically increase computational expense (the major expense in the algorithm is due to the integration of the system forward in time). Furthermore, this "complex" update formula often ensures convergence of the parameters and state, while the first "simple" one frequently fails to force convergence (see Sections 4.2.1 and 4.2.2).

Estimating Pr **with Known** Ra**.** The derivation for the parameter updates in this case follows the same steps as before. Consider the system

$$\begin{aligned}
&\frac{\partial \zeta}{\partial t} + \mathbf{u} \cdot \nabla \zeta = \mathrm{Pr}\, \Delta \zeta + \mathrm{Pr}\, \mathrm{Ra}\, \theta_x, \\
&\frac{\partial \theta}{\partial t} + \mathbf{u} \cdot \nabla \theta = \Delta \theta, \\
&\frac{\partial \tilde{\zeta}}{\partial t} + \tilde{\mathbf{u}} \cdot \nabla \tilde{\zeta} = \widetilde{\mathrm{Pr}} \left(\Delta \tilde{\zeta} + \mathrm{Ra}\ \tilde{\theta}_x \right) - \mu P_N \left(\tilde{\zeta} - \zeta \right), \\
&\frac{\partial \tilde{\theta}}{\partial t} + \tilde{\mathbf{u}} \cdot \nabla \tilde{\theta} = \Delta \tilde{\theta}.
\end{aligned} \tag{2.10}$$

Following the derivation above shows that once the error $\|w\|$ stops decreasing at time $t = t_1$ it follows that

$$\begin{aligned}
0 \approx &-\langle P_N w, P_N(\tilde{\mathbf{u}} \cdot \nabla w) \rangle - \langle P_N w, P_N(\mathbf{v} \cdot \nabla \zeta) \rangle + \widetilde{\mathrm{Pr}} \langle P_N w, P_N \left(\Delta w + \mathrm{Ra}\, \eta_x \right) \rangle \\
&+ \left(\widetilde{\mathrm{Pr}} - \mathrm{Pr} \right) \langle P_N w, P_N \left(\Delta \zeta + \mathrm{Ra}\, \theta_x \right) \rangle - \mu \| P_N w \|^2.
\end{aligned} \tag{2.11}$$

If one eliminates terms quadratic in the error except those coupled with μ and assume the nondegeneracy condition

$$\langle P_N w, P_N \left(\Delta \zeta + \mathrm{Ra}\, \theta_x \right) \rangle |_{t=t_1} \neq 0$$

holds, then the update formula is given by

$$\widetilde{\mathrm{Pr}}|_{t>t_n} = \widetilde{\mathrm{Pr}}|_{t \in [t_{n-1}, t_n)} - \mu \frac{\| P_N w \|^2}{\langle P_N w, P_N \left(\Delta \zeta + \mathrm{Ra}\, \theta_x \right) \rangle}.$$

If one tries to estimate terms rather than eliminating them, the (possibly more accurate) update formula is given by

$$\widetilde{\mathrm{Pr}}|_{t>t_n} = \frac{-\langle P_N w, P_N(g)\rangle + \widetilde{\mathrm{Pr}}|_{t\in[t_{n-1},t_n)}\langle P_N w, P_N\left(\Delta\tilde{\zeta} + \mathrm{Ra}\,\tilde{\theta}_x\right)\rangle - \mu\|P_N w\|^2}{\langle P_N w, P_N\left(\Delta\zeta + \mathrm{Ra}\,\theta_x\right)\rangle}$$

where g is defined as in (2.9).

Estimating Pr **with Known** Pr Ra**.** If, rather than assuming that Ra is known, Pr Ra is known, then the system can be written as

$$\begin{aligned}
&\frac{\partial\zeta}{\partial t} + \mathbf{u}\cdot\nabla\zeta = \mathrm{Pr}\,\Delta\zeta + \mathrm{Pr}\,\mathrm{Ra}\,\theta_x,\\
&\frac{\partial\theta}{\partial t} + \mathbf{u}\cdot\nabla\theta = \Delta\theta,\\
&\frac{\partial\tilde{\zeta}}{\partial t} + \tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta} = \widetilde{\mathrm{Pr}}\Delta\tilde{\zeta} + \mathrm{Pr}\,\mathrm{Ra}\;\tilde{\theta}_x - \mu P_N\left(\tilde{\zeta}-\zeta\right),\\
&\frac{\partial\tilde{\theta}}{\partial t} + \tilde{\mathbf{u}}\cdot\nabla\tilde{\theta} = \Delta\tilde{\theta}.
\end{aligned} \tag{2.12}$$

Following the derivations above shows that once the error $\|w\|$ stops decreasing at time $t = t_1$, it follows that

$$\begin{aligned}
0 \approx &-\langle P_N w, P_N(\tilde{\mathbf{u}}\cdot\nabla w)\rangle - \langle P_N w, P_N(\mathbf{v}\cdot\nabla\zeta)\rangle + \widetilde{\mathrm{Pr}}\langle P_N w, P_N\Delta w\rangle\\
&+\left(\widetilde{\mathrm{Pr}} - \mathrm{Pr}\right)\langle P_N w, P_N\Delta\zeta)\rangle + \mathrm{Pr}\,\mathrm{Ra}\langle P_N w, P_N\eta_x\rangle - \mu\|P_N w\|^2.
\end{aligned} \tag{2.13}$$

If one eliminates terms quadratic in the error except those coupled with μ and assume the nondegeneracy condition

$$\langle P_N w, P_N\Delta\zeta)\rangle|_{t=t_1} \neq 0$$

holds, then the update formula is given by

$$\widetilde{\mathrm{Pr}}|_{t>t_n} = \widetilde{\mathrm{Pr}}|_{t\in[t_{n-1},t_n)} - \mu\frac{\|P_N w\|^2}{\langle P_N w, P_N\Delta\zeta\rangle}.$$

If one tries to estimate terms rather than eliminating them, a (possibly more accurate) update formula is given by

$$\widetilde{\mathrm{Pr}}|_{t>t_n} = \frac{\widetilde{\mathrm{Pr}}|_{t\in[t_{n-1},t_n)}\langle P_N w, P_N \Delta\tilde{\zeta}\rangle - \langle P_N w, P_N(g)\rangle + \mathrm{Pr}\,\mathrm{Ra}\langle P_N w, P_N \eta_x\rangle - \mu\|P_N w\|^2}{\langle P_N w, P_N\left(\Delta\zeta + \mathrm{Ra}\,\theta_x\right)\rangle}$$

where g is defined as in (2.9).

2.1.2 CHL Algorithm for Estimating Multiple Parameters. The CHL algorithm can be used for estimating multiple parameters, as long as the system contains more than one equation, and each parameter of interest is found in exactly one equation. The derivation proceeds in a similar manner; each equation is used to derive a separate update. Note that in this case it is necessary to use a different nondimensionalization of the Rayleigh-Bénard equations (see (1.10)) which allows the Pr and Ra to be located in separate equations. For added flexibility, a nudging term is placed on the temperature equation as well. The system is as follows:

$$\begin{aligned}
&\frac{\partial\zeta}{\partial t} + \mathbf{u}\cdot\nabla\zeta = \Delta\zeta + \mathrm{Ra}\,\theta_x,\\
&\frac{\partial\theta}{\partial t} + \mathbf{u}\cdot\nabla\theta = \mathrm{Pr}^{-1}\Delta\theta,\\
&\frac{\partial\tilde{\zeta}}{\partial t} + \tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta} = \Delta\tilde{\zeta} + \widetilde{\mathrm{Ra}}\,\tilde{\theta}_x - \mu_1 P_N\left(\tilde{\zeta}-\zeta\right),\\
&\frac{\partial\tilde{\theta}}{\partial t} + \tilde{\mathbf{u}}\cdot\nabla\tilde{\theta} = \widetilde{\mathrm{Pr}}^{-1}\Delta\tilde{\theta} - \mu_2 P_N\left(\tilde{\theta}-\theta\right)
\end{aligned} \tag{2.14}$$

Following the same steps in the derivations above, once the errors $\|w\|$ and $\|\eta\|$ stop decreasing at time $t = t_1$, it follows that

$$0 \approx -\langle P_N w, P_N g\rangle + \langle P_N w, P_N \Delta w\rangle + \widetilde{\mathrm{Ra}}\langle P_N\tilde{\eta}_x, P_N w\rangle - (\widetilde{\mathrm{Ra}} - \mathrm{Ra})\langle P_N\theta_x, P_N w\rangle - \mu_1\|P_N w\|^2,$$

and

$$0 \approx -\langle P_N\eta, P_N h\rangle + \widetilde{\mathrm{Pr}}^{-1}\langle P_N\eta, P_N\Delta\eta\rangle + \left(\widetilde{\mathrm{Pr}}^{-1} - \mathrm{Pr}^{-1}\right)\langle P_N\eta, P_N\Delta\theta\rangle - \mu_2\|P_N\eta\|^2,$$

where

$$h := P_N(\tilde{\mathbf{u}}) \cdot P_N(\nabla \eta) + P_N(\mathbf{v}) \cdot P_N(\nabla \theta),$$

and g is defined as in (2.9). If one eliminates terms quadratic in the error except those coupled with μ and assume the nondegeneracy conditions

$$\langle P_N \theta_x, P_N w \rangle \neq 0, \qquad \langle P_N \eta, P_N \Delta\theta \rangle \neq 0$$

hold, then the update formulae are given by

$$\widetilde{\mathrm{Ra}}|_{t>t_n} = \widetilde{\mathrm{Ra}}|_{t\in[t_{n-1},t_n)} - \mu_1 \frac{\|P_n w\|^2}{\langle P_N \theta_x, P_N w \rangle}\bigg|_{t=t_1}$$

and

$$\widetilde{\mathrm{Pr}}^{-1}|_{t>t_n} = \widetilde{\mathrm{Pr}}^{-1}|_{t\in[t_{n-1},t_n)} - \mu_2 \frac{\|P_n \eta\|^2}{\langle P_N \eta, P_N \Delta\theta \rangle}.$$

Similar updates are available as done above if instead of eliminating the nonlinear terms, they are approximated with projections of the observed true state.

2.1.3 PWM Algorithm. This algorithm originates from a paper of Pachev, Whitehead, and McQuarrie [9]. In that work, the authors propose a formula for updating a parameter "continuously" and concurrently with the integration of the nudged system by performing a parameter update at each time step. This work considers the update derivation in [9], within the context of the "relax and punch" algorithm.

Consider a partial differential equation given by

$$\dot{x} = \sum_{k=1}^{p} \lambda_k L_k x + F(x),$$

where $\dot{x}$ is the time-derivative of x, $\{L_k\}_{k=1}^p$ are spatial linear differential operators, $\{\lambda_k\}_{k=1}^p$ are parameters, and F is a smooth (potentially nonlinear) differential operator (with a sufficient amount of dissipation to give the system a finite dimensional attractor [10]. Given

an orthogonal projection P_N, this system is coupled with the nudged system

$$\dot{\tilde{x}} = \sum_{k=1}^{p} \tilde{\lambda}_k L_k \tilde{x} + F(\tilde{x}) - \mu P_N(\tilde{x} - x). \tag{2.15}$$

Then the error $u := \tilde{x} - x$ evolves according to

$$\dot{u} = \sum_{k=1}^{p} \tilde{\lambda}_k L_k \tilde{x} + F(\tilde{x}) - \dot{x} - \mu P_N u.$$

Apply the projection P_N throughout and take an inner product with $P_N u$; it follows that

$$\frac{1}{2}\frac{d}{dt}\left[\|P_N u\|^2\right] = \left[\left\langle \sum_{k=1}^{p} \tilde{\lambda}_k L_k P_N \tilde{x} + P_N F(\tilde{x}) - P_N \dot{x}, P_N u \right\rangle\right] - \mu \|P_N u\|^2.$$

achev, Whitehead, and McQuarrie's [9] original observation was that if one can select the parameter estimates $\{\tilde{\lambda}_k(t)\}_{k=1}^{p}$ in order to enforce

$$\left\langle \sum_{k=1}^{p} \tilde{\lambda}_k L_k P_N \tilde{x} + P_N F(\tilde{x}) - P_N \dot{x}, P_N u \right\rangle = 0, \tag{2.16}$$

over some interval $[t_0, t_1]$, then $\|P_N u\|$ will satisfy

$$\|P_N u(t)\| = \|P_N u(t_0)\| e^{-\mu(t-t_0)}$$

over that interval [9]. The key idea was that because the error in the state along the projected modes will decrease exponentially, the parameter estimates will eventually have to converge to the true parameters (otherwise, the error in state would not be able to become small). However, it turns out that selecting the parameters to enforce (2.16) is not actually necessary, and can be counterproductive. Indeed, computational evidence shows that choosing to satisfy (2.16) rather than following the principles in Section 2.1.3 often causes the PWM algorithm to fail to converge (see Section 4.3.1 for computational evidence and further detail). Instead,

looking at relation (2.15), the goal is to enforce $\{\tilde{\lambda}\}_{k=1}^{p}$ so that

$$\sum_{k=1}^{p} \tilde{\lambda}_k L_k \tilde{x} + F(\tilde{x}) - \dot{x} = 0. \tag{2.17}$$

However, several immediate concerns arise. First of all, $\dot{x}$ is not fully available to the observer; only $P_N \dot{x}$ (assuming that observations are sufficiently dense in time). Secondly, the p variables $\{\tilde{\lambda}_k\}_{k=1}^{p}$ cannot be chosen to enforce what is effectively an infinite number of linear conditions. Even if the projection P_N of both sides is taken, as long as $N > p$ the system is still overdetermined. To make the system solvable, p linearly independent vectors $\{b_j\}_{j=1}^{p}$ must be chosen, then the free variables $\{\tilde{\lambda}_k\}_{k=1}^{p}$ can be chosen to solve the system along these directions; yielding the p equations

$$\left\langle \sum_{k=1}^{p} \tilde{\lambda}_k L_k \tilde{x} + F(\tilde{x}) - \dot{x}, b_j \right\rangle = 0, \qquad j = 1, \ldots, p.$$

Defining the $p \times p$ matrix A by $A_{jk} = \langle L_k \tilde{x}, b_j \rangle$ and the p-vector $\mathbf{g}$ by $g_j = \langle \dot{x} - F(\tilde{x}, b_j \rangle$, the system becomes $A\tilde{\boldsymbol{\lambda}} = \mathbf{g}$, where $\tilde{\boldsymbol{\lambda}} = (\tilde{\lambda}_1, \ldots, \tilde{\lambda}_p)$. As long as the vectors $b_1, \ldots b_p$ are in the image of the orthogonal projection P_N (i.e. they are observable modes), then $\langle b_j, v \rangle = \langle b_j, P_N v \rangle$, so the observer can calculate all of the entries of A and $\mathbf{b}$ and solve the linear system for $\tilde{\boldsymbol{\lambda}}$. Here, as in the CHL algorithm, there is also a non-degeneracy condition, namely that the matrix A is nonsingular, or that $\det A \neq 0$. If this condition holds, then $\tilde{\boldsymbol{\lambda}} = A\mathbf{g}$ is a formula for a parameter update. Note also that this parameter update does not explicitly depend on the previous parameter estimates, as it does in the CHL algorithm.

Choosing a Basis. One might wonder how to select the vectors $\{b_j\}_{j=1}^{p}$. While at the level of the analysis it does not seem to matter what the $\{b_j\}_{j=1}^{p}$ are as long as they lie within the image of the projection P_N (see Sections 3.1 and 3.2), in practice the choice of the basis is critical to ensuring convergence of the state and parameters (see Section 4.3.1).

By defining a $p \times N$ matrix B where B_{ji} is ith projected mode (under P_N) of b_j, and an $N \times p$ matrix L by setting L_{ik} to be the ith projected mode of $L_k \tilde{x}$, then it is clear that

$A = BL$. Similarly, let $\mathbf{f}$ be the vector whose ith component is the ith projected mode of $\dot{x} - F(\tilde{x})$; then $\mathbf{g} = B\mathbf{f}$. Choosing the vectors $\{b_j\}_{j=1}^{p}$ amounts to selecting the entries of the matrix B, and the system $A\tilde{\boldsymbol{\lambda}} = \mathbf{g}$ can be rewritten as $BL\tilde{\boldsymbol{\lambda}} = B\mathbf{f}$. Recall that the original goal was to choose $\tilde{\boldsymbol{\lambda}}$ to satisfy (2.17), or at least to satisfy it on the N projected modes which are observable. In the new notation, the system (2.17) can be written $L\tilde{\boldsymbol{\lambda}} = \mathbf{f}$, which is indeed an overdetermined system of N equations in p variables. Making the choice $B = L^T$ the system becomes $L^T L\tilde{\boldsymbol{\lambda}} = L^T\mathbf{f}$, which has the solution

$$\tilde{\boldsymbol{\lambda}} = (L^T L)^{-1} L^T \mathbf{f},$$

which is the ordinary least squares solution to $L\tilde{\boldsymbol{\lambda}} = \mathbf{f}$. Hence this choice of B leads to a choice of parameter estimates $\tilde{\boldsymbol{\lambda}}$ which minimizes $\|L\tilde{\boldsymbol{\lambda}} - \mathbf{f}\|^2$. This is equivalent to minimizing the quantity

$$\left\| P_N \left(\sum_{k=1}^{p} \tilde{\lambda}_k L_k \tilde{x} + F(\tilde{x}) - \dot{x} \right) \right\|.$$

This makes choosing $B = L^T$, or in other words, choosing $b_j = P_N L_k \tilde{x}$, as the optimal choice in the sense outlined above.

Estimating Ra **with known** Pr**.** Consider the system (2.6). The discussion that follows will apply the derivation in the previous section to this sytem. Letting $w = \tilde{\zeta} - \zeta$ be the error in vorticity, it follows that

$$\frac{\partial w}{\partial t} = \frac{\partial \tilde{\zeta}}{\partial t} - \frac{\partial \zeta}{\partial t} = -\tilde{\mathbf{u}} \cdot \nabla \tilde{\zeta} + \Pr \Delta \tilde{\zeta} + \Pr \widetilde{\mathrm{Ra}} \tilde{\theta}_x - \mu P_N \left(\tilde{\zeta} - \zeta \right) - \zeta_t,$$

where $\zeta_t := \partial \zeta / \partial t$. Simplifying and applying the projection operator P_N throughout (which as noted, commutes with derivatives, is linear, and is idempotent), it follows that

$$P_N w_t = \Pr P_N(\Delta \tilde{\zeta}) + \Pr \widetilde{\mathrm{Ra}} P_N(\tilde{\theta}_x) - \left[P_N(\tilde{\mathbf{u}} \cdot \nabla \tilde{\zeta}) + P_N \zeta_t \right] - \mu P_N w$$

At this point, the goal is to choose $\widetilde{\mathrm{Ra}}$ to enforce

$$\mathrm{Pr}\, P_N(\Delta\tilde{\zeta}) + \mathrm{Pr}\,\widetilde{\mathrm{Ra}} P_N(\tilde{\theta}_x) - \left[P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t\right] = 0 \tag{2.18}$$

to the extent possible. Section 2.1.3 dictates choosing $b_1 = P_N(\tilde{\theta}_x)$ and seeking to satisfy (2.18) along this direction. Taking the inner product with b_1 yields

$$\mathrm{Pr}\,\widetilde{\mathrm{Ra}}\|P_N\tilde{\theta}_x\|^2 = \langle P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t + \mathrm{Pr}\, P_N(\Delta\tilde{\zeta}), P_N\tilde{\theta}_x\rangle$$

which yields the update equation

$$\widetilde{\mathrm{Ra}}|_{t>t_n} = \left.\frac{\left\langle \frac{1}{\mathrm{Pr}}\left[P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t\right] + P_N(\Delta\tilde{\zeta}), P_N\tilde{\theta}_x\right\rangle}{\|P_N\tilde{\theta}_x\|^2}\right|_{t=t_n}$$

as long as $\|P_N\tilde{\theta}_x\| \neq 0$, which is a weaker nondegeneracy condition than in the CHL case.

Estimating Pr **with known** Ra**.** To derive the PWM update formula for (2.10), let $w = \tilde{\zeta} - \zeta$ be the error in vorticity. It follows that

$$\frac{\partial w}{\partial t} = \frac{\partial\tilde{\zeta}}{\partial t} - \frac{\partial\zeta}{\partial t} = -\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta} + \widetilde{\mathrm{Pr}}\Delta\tilde{\zeta} + \widetilde{\mathrm{Pr}}\,\mathrm{Ra}\,\tilde{\theta}_x - \mu P_N\left(\tilde{\zeta} - \zeta\right) - \zeta_t,$$

where $\zeta_t := \partial\zeta/\partial t$. Simplifying and applying the projection operator P_N throughout (which as noted, commutes with derivatives, is linear, and is idempotent), it follows that

$$P_N w_t = \widetilde{\mathrm{Pr}}\left[P_N(\Delta\tilde{\zeta}) + \mathrm{Ra}\, P_N(\tilde{\theta}_x)\right] - \left[P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t\right] - \mu P_N w$$

At this point, the goal is to choose $\widetilde{\mathrm{Pr}}$ to enforce

$$\widetilde{\mathrm{Pr}}\left[P_N(\Delta\tilde{\zeta}) + \mathrm{Ra}\, P_N(\tilde{\theta}_x)\right] - \left[P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t\right] = 0 \tag{2.19}$$

to the extent possible. Section 2.1.3 dictates choosing the basis vector $b_1 = P_N(\Delta\tilde{\zeta} + \mathrm{Ra}\,\tilde{\theta}_x)$ and seeking to satisfy (2.19) along this direction. Taking the inner product with b_1 yields

$$\mathrm{Pr}\,\|P_N(\Delta\tilde{\zeta} + \mathrm{Ra}\,\tilde{\theta}_x)\|^2 = \langle P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t, P_N(\Delta\tilde{\zeta} + \mathrm{Ra}\,\tilde{\theta}_x)\rangle$$

which yields the update equation

$$\widetilde{\mathrm{Pr}}|_{t>t_n} = \left.\frac{\langle P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t, P_N(\Delta\tilde{\zeta} + \mathrm{Ra}\,\tilde{\theta}_x)\rangle}{\|P_N(\Delta\tilde{\zeta} + \mathrm{Ra}\,\tilde{\theta}_x)\|^2}\right|_{t=t_n}$$

as long as $\|P_N(\Delta\tilde{\zeta} + \mathrm{Ra}\,\tilde{\theta}_x)\| \neq 0$, which is a weaker nondegeneracy condition than in the CHL case.

Estimating Ra **and** Pr**.** Unlike for CHL, the PWM algorithm is capable of estimating Pr and Ra simultaneously while nudging the vorticity alone. In other words, one need only consider (2.12) in order to recover both of the relevant non-dimensional parameters. Starting with this system, let $w = \tilde{\zeta} - \zeta$ be the error in vorticity, it follows that

$$\frac{\partial w}{\partial t} = \frac{\partial\tilde{\zeta}}{\partial t} - \frac{\partial\zeta}{\partial t} = -\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta} + \widetilde{\mathrm{Pr}}\Delta\tilde{\zeta} + \widetilde{\mathrm{Pr}}\widetilde{\mathrm{Ra}}\tilde{\theta}_x - \mu P_N\left(\tilde{\zeta} - \zeta\right) - \zeta_t,$$

where $\zeta_t := \partial\zeta/\partial t$. Simplifying and applying the projection operator P_N throughout (which as noted, commutes with derivatives, is linear, and is idempotent), it follows that

$$P_N w_t = \widetilde{\mathrm{Pr}}P_N(\Delta\tilde{\zeta}) + \widetilde{\mathrm{Pr}}\widetilde{\mathrm{Ra}}P_N(\tilde{\theta}_x) - \left[P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t\right] - \mu P_N w$$

At this point, the goal is to choose $\widetilde{\mathrm{Pr}}$ and $\widetilde{\mathrm{Ra}}$ to enforce

$$\widetilde{\mathrm{Pr}}P_N(\Delta\tilde{\zeta}) + \widetilde{\mathrm{Pr}}\widetilde{\mathrm{Ra}}P_N(\tilde{\theta}_x) - \left[P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t\right] = 0 \tag{2.20}$$

to the extent possible. Section 2.1.3 dictates choosing $b_1 = P_N(\Delta\tilde{\zeta})$ and $b_2 = P_N(\tilde{\theta}_x)$ and seeking to satisfy (2.20) along these directions. In practice, the Gram-Schmidt procedure

would be employed to orthonormalize this basis and make it more numerically stable, but that level of detail is unnecessary for the present analysis. The conditions $\widetilde{\mathrm{Pr}}$ and $\widetilde{\mathrm{Ra}}$ must be chosen to satisfy are

$$\widetilde{\mathrm{Pr}}\|P_N(\Delta\tilde{\zeta})\|^2 + \widetilde{\mathrm{Pr}}\widetilde{\mathrm{Ra}}\langle P_N(\tilde{\theta}_x), P_N(\Delta\tilde{\zeta})\rangle = \langle P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t, P_N(\Delta\tilde{\zeta})\rangle$$
$$\widetilde{\mathrm{Pr}}\langle P_N(\Delta\tilde{\zeta}), P_N(\tilde{\theta}_x)\rangle + \widetilde{\mathrm{Pr}}\widetilde{\mathrm{Ra}}\|P_N(\tilde{\theta}_x)\|^2 = \langle P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t, P_N(\tilde{\theta}_x)\rangle$$

which can be written in matrix form:

$$\begin{pmatrix} \|P_N(\Delta\tilde{\zeta})\|^2 & \langle P_N(\tilde{\theta}_x), P_N(\Delta\tilde{\zeta})\rangle \\ \langle P_N(\Delta\tilde{\zeta}), P_N(\tilde{\theta}_x)\rangle & \|P_N(\tilde{\theta}_x)\|^2 \end{pmatrix} \begin{pmatrix} \widetilde{\mathrm{Pr}} \\ \widetilde{\mathrm{Pr}}\widetilde{\mathrm{Ra}} \end{pmatrix} = \begin{pmatrix} \langle P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t, P_N(\Delta\tilde{\zeta})\rangle \\ \langle P_N(\tilde{\mathbf{u}}\cdot\nabla\tilde{\zeta}) + P_N\zeta_t, P_N(\tilde{\theta}_x)\rangle \end{pmatrix}. \tag{2.21}$$

Letting A be the matrix on the left-hand side of (2.21) and b be the vector on the right-hand side of (2.21), the parameter update takes the form

$$\widetilde{\mathrm{Pr}}|_{t>t_n} = (A^{-1}b)_1|_{t=t_n}, \qquad \widetilde{\mathrm{Ra}}|_{t>t_n} = \frac{(A^{-1}b)_2}{(A^{-1}b)_1}\Big|_{t=t_n}$$

as long as the matrix A is nonsingular ($\det A \neq 0$ is the nondegeneracy condition in this case), and as long as $(A^{-1}b)_1 \neq 0$. This second condition is new, but makes sense because the Rayleigh-Bénard system becomes ill-posed if the Prandtl number is close to zero.

2.2 Comparison of Algorithms

Both the CHL and PWM algorithms appear to perform well at recovering unknown parameters for a dissipative dynamical system. It appears that both algorithms rely on the "relax and punch" concept that the system needs to first relax via nudging so that the state converges up to a model error determined by the difference in parameters. After this convergence in the state is satisfied then the system is "punched" via an update in the parameters and the system is run again. As numerically demonstrated below, both approaches ap-

pear to work well for Rayleigh-Bénard convection, recovering both the Rayleigh and Prandtl numbers under certain reasonable conditions.

A distinct advantage that PWM has over CHL is that it is more conducive to estimation of multiple parameters simultaneously even if all of the parameters are contained in a single equation. This observation is due to the choice of the basis vectors b_j as described above for PWM. On the other hand PWM also requires the numerical estimation of the projected time derivative which can lead to significant numerical error (see [9] for a thorough discussion of this issue for the continuous version of the same algorithm). At the same time, incorporating the time derivative into the algorithm update in PWM eliminates the need to show that this term is bounded as is done when rigorously establishing convergence for CHL in [11, 12]. This indicates that while PWM may be prone to numerical error from the temporal spacing of the observations, it is more grounded theoretically, and less prone to errors that may creep in from potentially unbounded (or exponentially large) growth in the projected time derivative.

Chapter 3. Theoretical Analysis of Convergence

3.1 Formalizing the PWM Algorithm

To formalize the PWM algorithm for a given system and put it in a rigorous setting, a few more assumptions are necessary. Firstly, assume that the system is d-dimensional and can be written as

$$\dot{x} = Lx + F(x), \tag{3.1}$$

where $\dot{x} = dx/dt$, $x : [0, \infty) \to \mathbb{R}^d$, and L is a linear combination of linear operators $\{L_j\}_{j=1}^p$ mapping from $\mathbb{R}^d$ to $\mathbb{R}^d$ having the form

$$L = \sum_{j=1}^{p} \alpha_j L_j.$$

It is assumed that $F : \mathbb{R}^d \to \mathbb{R}^d$ is smooth, and that $\{\alpha_j\}_{j=1}^p \subset \mathbb{R}$ are the unknown system parameters. It is also required that the system (3.1) is dissipative, which means that it possesses a finite-dimensional global attractor (if L is autonomous) or a pullback attractor (if L is non-autonomous). Therefore, given an initial point $x_0 \in \mathbb{R}^d$, there must exist a unique solution $x(t; x_0)$ of (3.1) defined for all $t \geq 0$, and such a solution must eventually be bounded. Denote the trajectory of this unique global solution $\{x(t; x_0\}_{t\geq 0}$.

Let P_N be the orthogonal projection onto the span of an orthogonal basis $\{e_i\}_{i=1}^N \subset \mathbb{R}^d$, where $p \leq N \leq d$. It is assumed that the observations are available continuously in time; that is, the observer has access to $P_N x$ and $P_N \dot{x}$ at all times t. To include the possibility of different nudging parameters for different modes of the system, let $M = \text{diag}(\mu_1, \ldots, \mu_d)$, where $\mu_1, \ldots, \mu_d > 0$, be a diagonal matrix, and consider the nudged system

$$\dot{\tilde{x}} = \tilde{L}\tilde{x} + F(\tilde{x}) - MP_N(\tilde{x} - x), \tag{3.2}$$

where

$$\tilde{L} = \sum_{j=1}^{p} \tilde{\alpha}_j L_j.$$

and $\{\tilde{\alpha}_j\}_{j=1}^p$ are the parameter estimates. For convenience, the following discussion will use the vector notation $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_p)$ and $\tilde{\boldsymbol{\alpha}} = (\tilde{\alpha}_1, \ldots, \tilde{\alpha}_p)$ when necessary. Because $\mu_1, \ldots, \mu_d > 0$, the system (3.2) is also dissipative. Let

$$\tilde{x}^0(t) = \tilde{x}(t; \tilde{x}_0, \alpha^0, \{P_N x(t; x_0)\}_{t\geq 0}) = \tilde{x}(t; \tilde{x}_0, \alpha^0)$$

be the solution corresponding to the initial value $\tilde{x}_0$, parameter estimates $\tilde{\alpha} = \alpha^0$, and

observations $\{P_N x(t, x_0)\}_{t\geq 0}$ from the system (3.1). Allow the system to proceed forward in time until a sufficiently large time $t_1 > 0$ when the system has relaxed.

Let $u := \tilde{x} - x$ be the error in state. Then $\dot{u}$ can be written

$$\dot{u} = \tilde{L}\tilde{x} + F(\tilde{x}) - \dot{x} - MP_N u.$$

As before (see Section 2.1.3), the PWM algorithm seeks to choose $\tilde{\boldsymbol{\alpha}}$ in order to enforce

$$\left[\tilde{L}\tilde{x} + F(\tilde{x}) - \dot{x}\right]\Big|_{t=t_1} = 0 \tag{3.3}$$

to the extent that it is possible. Indeed, (3.3) is a system of p variables in d equations. To ensure solvability of the system, vectors $\{b_j^0\}_{j=1}^p \subset P_N(\mathbb{R}^d)$ are chosen (an optimal choice would be $b_j^0 = P_N L_j \tilde{x}|_{t=t_1}$; see Section 2.1.3) to construct a $p \times p$ system

$$\langle \tilde{L}\tilde{x} + F(\tilde{x}) - \dot{x}, b_j\rangle|_{t=t_1} = 0, \qquad j = 1, \dots p.$$

As before, let the $p \times p$ matrix A^0 and the vector $\mathbf{g}^0 \in \mathbb{R}^p$ have entries

$$A^0_{jk} = \langle L_k \tilde{x}, b_j\rangle|_{t=t_1}, \qquad g^0_j = \langle \dot{x} - F(\tilde{x}, b_j\rangle|_{t=t_1}$$

so that the system above can be written $A^0\tilde{\boldsymbol{\alpha}} = \mathbf{g}^0$. Note that each of the entries of A and g can be calculated with information available to the observer. Assuming that A^0 is nonsingular at time t_1, the parameters are updated using the formula

$$\alpha^1 = (A^0)^{-1}\mathbf{g}^0|_{t=t_1},$$

then the new nudged system $\tilde{x}^1(t; \tilde{x}_1, \alpha^1)$ is considered for $t > t_1$.

The first step in proving the convergence of the PWM algorithm is to find an explicit

representation for the parameter error. Using the relations

$$\sum_{j=1}^{p} \alpha_j^1 \langle L_j \tilde{x}, b_i \rangle = \langle \dot{x} - F(\tilde{x}, b_i \rangle, \qquad i = 1, \ldots, p$$

and making the substitution

$$\langle \dot{x}, b_i \rangle = \sum_{j=1}^{p} \alpha_j \langle L_j x, b_i \rangle + \langle F(x), b_i \rangle, \qquad i = 1, \ldots, p$$

(see (3.1)) yields

$$\sum_{j=1}^{p} \alpha_j^1 \langle L_j \tilde{x}, b_i \rangle = \sum_{j=1}^{p} \alpha_j \langle L_j x, b_i \rangle + \langle F(x) - F(\tilde{x}, b_i \rangle, \qquad i = 1, \ldots, p.$$

Now subtract $\sum_{j=1}^{p} \alpha_j \langle L_j \tilde{x}, b_i \rangle$ from both sides to obtain

$$\sum_{j=1}^{p} (\alpha_j^1 - \alpha_j) \langle L_j \tilde{x}, b_i \rangle = -\sum_{j=1}^{p} \alpha_j \langle L_j u, b_i \rangle - \langle F(\tilde{x}) - F(x), b_i \rangle, \qquad i = 1, \ldots, p. \tag{3.4}$$

Define a $p \times p$ matrix U^0 and a vector $\mathbf{f}^0 \in \mathbb{R}^p$ whose entries are

$$U_{ij}^0 = -\langle L_j u, b_i \rangle, \qquad f_i^0 = -\langle F(\tilde{x} - F(x), b_i \rangle$$

Then (3.4) can be written more succinctly as

$$A^0(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\alpha}) = U^0 \boldsymbol{\alpha} + \mathbf{f}$$

Using the assumption that A^0 is invertible at time t_1, it follows that there is explicitly represent the parameter error in terms of the true parameters:

$$\tilde{\boldsymbol{\alpha}} - \boldsymbol{\alpha} = (A^0)^{-1}(U\boldsymbol{\alpha} + \mathbf{f})$$

Then the above process is repeated. At the $(n+1)$th stage, it is assumed that $\alpha^0, \ldots, \alpha^n$ have been constructed under the proper conditions, i.e. that the orthonormal bases B^i have been identified satisfying the condition $\{b_j^i\}_{j=1}^p \subset P_N(\mathbb{R}^d)$ at times $t = t_{i+1}$ and that $\det A^i|_{t=t_{i+1}} \neq 0$ for $i = 0, \ldots, n-1$ as well. Letting $\tilde{x}_n \in \mathbb{R}^d$ be arbitrary, one then considers the solution $\tilde{x}^n(t) = \tilde{x}(t; \tilde{x}_n, \alpha^n)$ corresponding to initial value $\tilde{x}_n$ and parameter values α^{n-1}. Once again, suppose let t_{n+1} be sufficiently large that $\tilde{x}^n(t_{n+1})$ has relaxed. Then, a basis $B^n = (b_1^n, \ldots, b_p^{n-1}) \subset P_N(\mathbb{R}^d)$ is identified, and, assuming that $\det A^n|_{t=t_{n+1}} \neq 0$, the next parameter update is given by

$$\alpha^{n+1} = (A^n)^{-1} g^n|_{t=t_{n+1}}.$$

According to the error representation above, it is clear that

$$\alpha^{n+1} - \alpha = (A^n)^{-1}(U^n \boldsymbol{\alpha} + \mathbf{f}^n)\Big|_{t=t_{n+1}}.$$

Our goal is to establish the following bound for some constants $C, c > 0$:

$$(A^n)^{-1}(U^n \boldsymbol{\alpha} + \mathbf{f}^n)\Big|_{t=t_{n+1}} \leq C \frac{|\alpha^n - \alpha|}{(\det A^n|_{t=t_{n+1}} \mu^c)}.$$

Provided that μ is chosen large enough, this will ensure that there exists some $\beta < 1$ for which

$$|\alpha^{n+1} - \alpha| \leq \beta |\alpha^n - \alpha|,$$

therefore establishing geometric convergence of the parameter estimates to the true values.

3.2 Statement and Proof of Convergence

Before proving the theorem, it is necessary to set forth some new notation and a few more assumptions. Define

$$A_l^k := A^k|_{t=t_{l+1}}, \qquad U_l^k := U^k|_{t=t_{l+1}}, \qquad f_l^k := |_{t=t_{l+1}}.$$

and let

$$u^n = \tilde{x}(t, \tilde{x}_n, \alpha^n) - x(t; x_0), \qquad \Delta\alpha^n = \alpha^n - \alpha.$$

The following additional assumptions will be made:

(i) By time $t = 0$ the system $x(t; x_0)$ has relaxed to be within the absorbing ball of the dynamics, so that there exists $R > 0$ such that $|x(t; x_0)| \leq R$ for all $t > 0$.

(ii) Each parameter update produces parameters for which the nudged system (3.2) is still well-posed and still has global solutions with absorbing ball bounds of the same order.

Theorem 3.1. *Assume that $F : \mathbb{R}^d \to \mathbb{R}^d$ is locally Lipschitz and that the coefficients of each $L_j : \mathbb{R}^d \to \mathbb{R}^d$ are uniformly bounded in time. Suppose that there exist $T > 0$ and constants $q, C_T > 0$ such that the following model error estimate holds for all $t \geq T$:*

$$|\tilde{x}(t; \tilde{x}_0, \tilde{\alpha}, \{P_N x(t; x_0)\}_{t\geq 0}) - x(t; x_0)| \leq \frac{C_T}{\mu^q}|\tilde{\alpha} - \alpha| \tag{3.5}$$

for all $t \geq T$. Then for each $n \geq 1$, there exist constants $r, C_n > 0$ such that

$$|\det A_n^n||\alpha^{n+1} - \alpha| \leq \frac{C_n}{\mu^r}|\alpha^n - \alpha|. \tag{3.6}$$

Proof. Let the initial parameter estimates $\alpha^0 \in \mathbb{R}^p$ be fixed, and let $t_0 = 0$. Let t_1 be large enough that (3.5) holds at time t_1. Then

$$|u^1| \leq \frac{C_T}{\mu^q}|\Delta\alpha^0|$$

holds for some power $q > 0$, where $\mu = \min\{\mu_1, \ldots, \mu_d\}$. Then, using representation of the error above, it follows that

$$A_0^0 \Delta\alpha^1 = U_0^0 \alpha + f_0^0.$$

Since each L_j has bounded coefficients by hypothesis, $C_L := |L| < \infty$. Therefore

$$|U_0^0| \leq C_L |u^1|$$

based on the definition of U, since B^1 is an orthonormal basis. Letting C_F^n denote the local Lipschitz constant of F at time $t = t_n$, it follows that

$$|f_0^0| \leq |F(\tilde{x}_1) - F(x_1)| \leq C_F^1 |u^1|.$$

Utilizing the fact that there exists a constant $c_d > 0$ such that

$$|\det M||M^{-1}| \leq c_d |A|^{d-1}$$

for any that for $d \times d$ matrix M,

$$|\Delta\alpha^1| \leq |(A_0^0)^{-1}| \left(|U_0^0||\alpha| + |f_0^0| \right)$$

$$|\det A_0^0||\Delta\alpha^1| \leq \frac{c_d}{|} A_0^0|^{d-1} (C_L + C_F^1) \frac{C_T}{\mu^q} |\Delta\alpha^0|.$$

Letting

$$C^1 := \frac{c_d}{|A_0^0|^{d-1}} (C_L + C_F^1) C_T,$$

we have that

$$|\det A_0^0||\Delta\alpha^1| \leq \frac{C}{\mu^q} |\Delta\alpha^0|.$$

Suppose that (3.6) holds for all $m = 1, \ldots, n$. Now consider the case $m = n+1$. Suppose $t_{n+1} > 0$ is large enough that (3.5) holds at time t_n. Then

$$|u^n| \leq \frac{C_T}{\mu^q} |\Delta\alpha^0|$$

holds for some power $q > 0$, where $\mu = \min\{\mu_1, \ldots, \mu_d\}$. Then, using the representation of

the error, it follows that

$$A_n^n \Delta\alpha^{n+1} = U_n^n \alpha + f_n^n.$$

We also have $|U_n^n| \leq C_L |u^n|$, and letting C_F^n denote the local Lipschitz constant of F at time $t = t_n$, it follows that

$$|f_n^n| \leq |F(\tilde{x}_n) - F(x_n)| \leq C_F^n |u^n|.$$

Then

$$|\Delta\alpha^{n+1}| \leq |(A_n^n)^{-1}| \left(|U_n^n||\alpha| + |f_n^n|\right)$$

$$|\det A_n^n||\Delta\alpha^{n+1}| \leq \frac{c_d}{|}A_n^n|^{d-1}(C_L + C_F^1)\frac{C_T}{\mu^q}|\Delta\alpha^n|.$$

Letting

$$C^n := \frac{c_n}{|A_n^n|^{d-1}}(C_L + C_F^n)C_T,$$

we have that

$$|\det A_n^n||\Delta\alpha^{n+1}| \leq \frac{C}{\mu^q}|\Delta\alpha^n|$$

as claimed. □

Chapter 4. Computational Analysis of Convergence

4.1 Methodology

Writing code to simulate Rayleigh-Bénard convection along with data assimilation and parameter recovery was a large undertaking. Rather than write the partial differential equation solvers from scratch, I made use of the python package Dedalus, version 2.0. Dedalus is a "flexible, open-source, parallelized computational framework for solving general partial differential equations using spectral methods" [13]. Dedalus features an object-oriented design,

symbolic manipulation through a computer algebra system, and good performance. More information about Dedalus is available at `http:dedalus-project.org`.

Using Dedalus had several advantages particularly appropriate to the needs of this project. Firstly, Dedalus uses spectral methods (specifically a first-order generalized tau formulation which discretizes equations into banded matrices). Variables within equations are represented as field objects, which have the capability to quickly transition between Cartesian grid representation and spectral grid representation. Given the boundary conditions for Rayleigh-Bénard convection, it was appropriate to use a Fourier basis in the horizontal direction and a Chebyshev basis in the vertical direction, and Dedalus made this easy. The spectral numerical methods offered high performance, and the ease of transitioning between spectral and Cartesian grid representations of field objects made implementing projection operators easy. Implementing a projection operator was also simple: one could transform a field by simply transitioning to spectral representation, zeroing out certain modes, and transitioning back to grid representation.

Because Dedalus allows the user to input the governing equations of the system as a parseable string, it was easy to make small modifications to the system as necessary. For example, given an initial guess for parameter estimates for Pr and Ra, it was possible to fix these parameters and treat the terms which they multiply implicitly. Then when updates to the parameter estimates were made, the difference between the new parameter estimates and the original guesses, which is relatively small compared to the parameter estimates themselves, could be treated explicitly. Compared to the simple approach of treating all of the terms explicitly (which may result in large timestep restrictions), this method has the advantage of allowing the largest portion of the terms to be treated implicitly.

Another advantage of Dedalus was its automatic paralellization. While some PDEs are relatively cheap to simulate, Rayleigh-Bénard convection is very computationally expensive. Dedalus's built-in parallelization using MPI enabled computations to be performed in a matter of hours rather than days or weeks.

File input and output was handled automatically by Dedalus and by code written by Shane McQuarrie [14]. Dedalus allows the user to record snapshots of the state of the system as well as custom calculations based on the state of the system at specified time intervals. In addition to enabling seamless recording of data, this allowed some processing and analysis to be performed on the data before it was written to files.

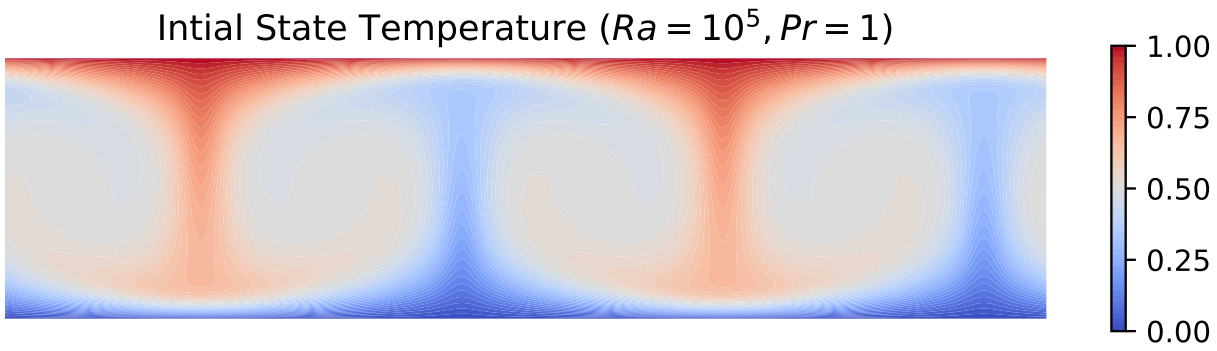
All simulations were performed on a 384×192 grid, with points having equally spaced horizontal coordinates and Chebyshev collocation points as vertical coordinates. The domain was $[0, 4] \times [0, 1]$, meaning that the horizontal dimension was four times as large as the vertical.

All states were initialized using an initial state with $\mathrm{Ra} = 10^5$ and $\mathrm{Pr} = 1$ that had been run out for a long time to ensure that all statistics had constant time averages. The initial conditions used for the true system are shown in Figure 4.1. The assimilating (nudged) system was initialized at a low-mode projection of these states (following [14]), and with the parameter estimates changed from the true values according to whether multiparameter, Ra-only, or Pr-only estimation was being tested.

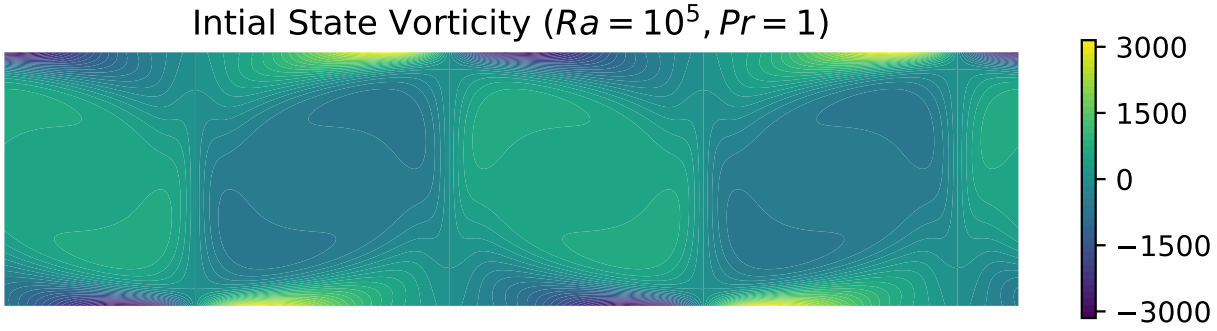
4.2 CHL Algorithm

4.2.1 Estimating a Single Parameter. Figures 4.2 and 4.3 show the performance of the CHL algorithm when estimating either only Ra (with known Pr) or only Pr (with known Ra), and how the performance depends upon which update formula is selected and to what degree temperature nudging is used. In each figure, the algorithms are started from the same initial state. The true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$. Each of the algorithms use $\mu = 8000$, and at time $t = 0.1$ start updating every 0.02 units of simulation time.

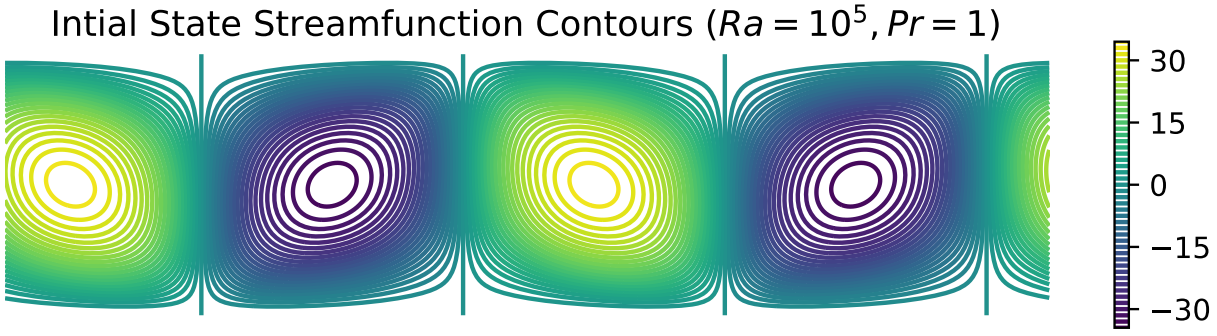
In Figure 4.2, the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Ra}} = 9 \times 10^4$ and $\widetilde{\mathrm{Pr}}$ is held at the true value. The "Simple" update formula which eliminates quadratic terms not coupled with μ fails to force convergence, even when temperature nudging is turned on, and soon blows up (at around $t = 0.35$). Meanwhile, the "Complex" update formula which approximates these terms forces convergence whether or



(a) The temperature satisfies the boundary conditions at the top and bottom plates. It features hot and cold plumes separated by rotational areas.



(b) Vorticity describes the local spinning motion of the fluid; different signs correspond to different directions of rotation.



(c) The streamlines of the fluid, which correspond to the trajectories of particles in a steady flow, are level sets of the streamfunction ψ shown here.

Figure 4.1: The convective state used as initial condition for computational experiments

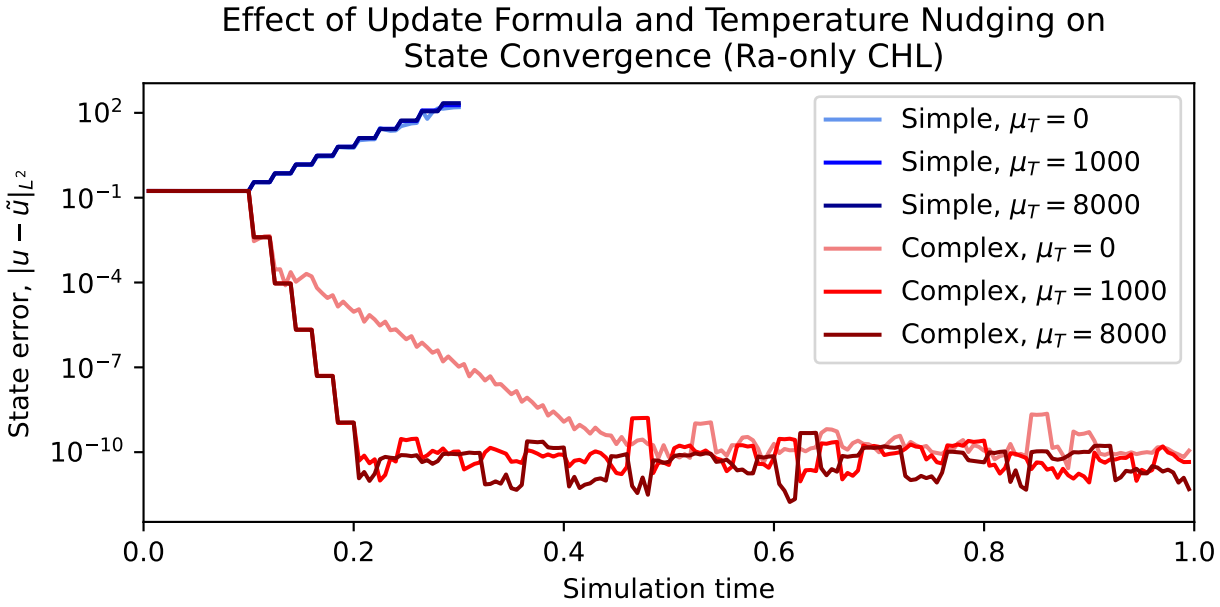
not temperature nudging is used. In each case, the state converges to an error of order 10^{-10}, and $\widetilde{\mathrm{Ra}}$ converges to within a relative error of 10^{-11}. However, the convergence is somewhat faster when the temperature nudging is used (i.e. $\mu_T > 0$).

In Figure 4.3, the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Pr}} = 1.1$ and $\widetilde{\mathrm{Ra}}$ is held at the true value. The "Simple" update formula which eliminates quadratic terms not coupled with μ fails to force convergence, even when temperature nudging is turned on, and soon blows up (at around $t = 0.2$). Meanwhile, the "Complex" update formula which approximates these terms forces convergence whether or not temperature nudging is used. In each case, the state converges to an error of order 10^{-11}, and $\widetilde{\mathrm{Pr}}$ converges to within a relative error of 10^{-12}. However, the convergence is very quick when the temperature nudging is used (i.e. $\mu_T > 0$).

Figures 4.4 and 4.5 show the effect of different relaxation times on the convergence of the single-parameter CHL algorithm. Each of the simulations uses $\mu = 8000$. In each simulation, the initial states for the true systems and assimilating systems are identical, with the assimilating system starting at a low-mode projection of the true state. In each simulation, there is an initial relaxation period of 0.1 units of simulation time. Then a parameter update is made repeatedly after a relaxation interval with a length called the "delay time." The figures show the effect of changing the "delay time" on the convergence. The results seem to show that decreasing the relaxation time increases the rate of convergence until a certain threshold around 0.04 units of time, below which the performance of the algorithm is essentially the same.

In Figure 4.4 the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Ra}} = 9 \times 10^4$ and $\widetilde{\mathrm{Pr}}$ is held at the true value. With each selection of relaxation time, the state and parameters do converge to an error of order 10^{-10}. However, if the relaxation period is above a certain threshold, then the rate of convergence slows slightly.

In Figure 4.5 the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Pr}} = 1.1$ and $\widetilde{\mathrm{Ra}}$ is held at the true value. With each selection of relaxation



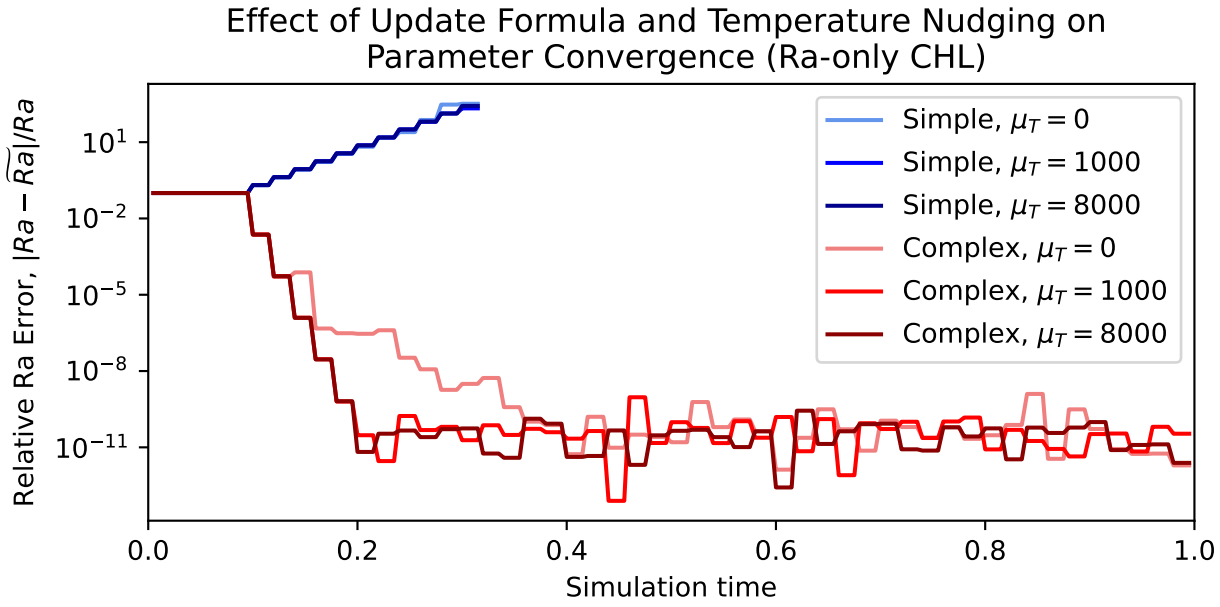
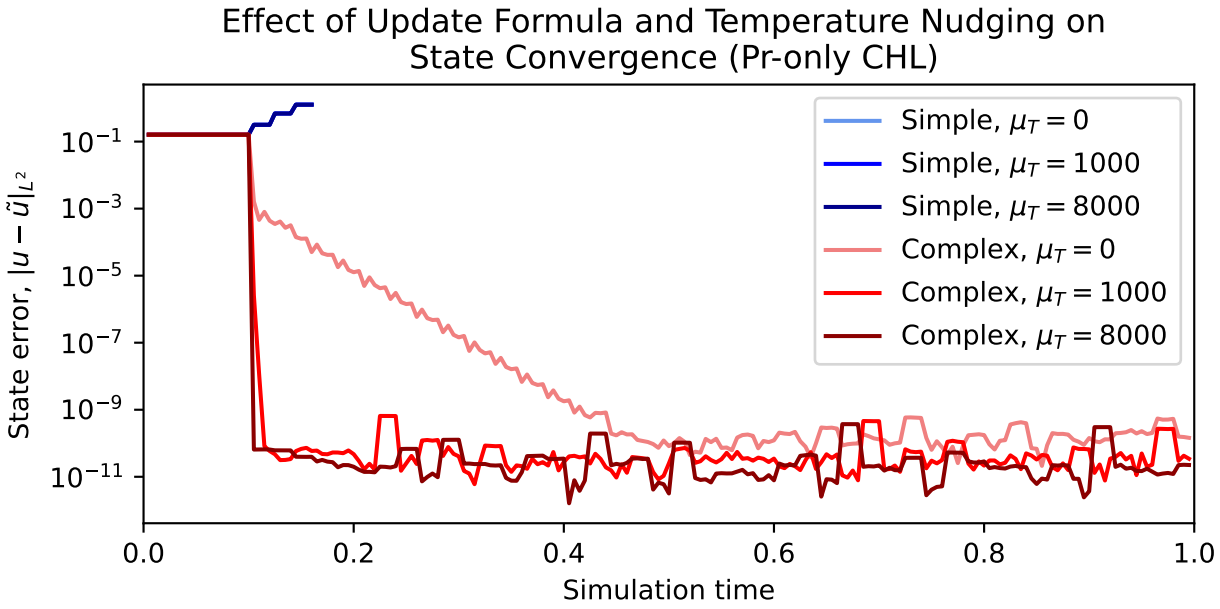


Figure 4.2: Comparing the performance of the Ra-only CHL algorithm when eliminating most terms ("Simple") and estimating them ("Complex")



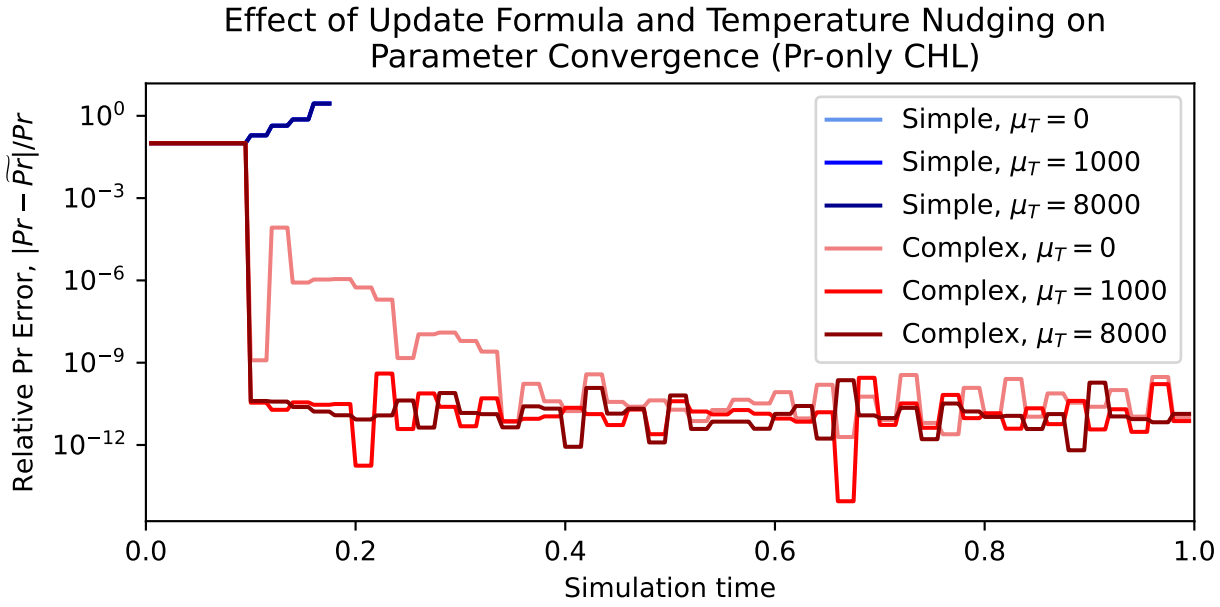


Figure 4.3: Comparing the performance of the Pr-only CHL algorithm when eliminating quadratic terms ("Simple") and approximating them ("Complex")

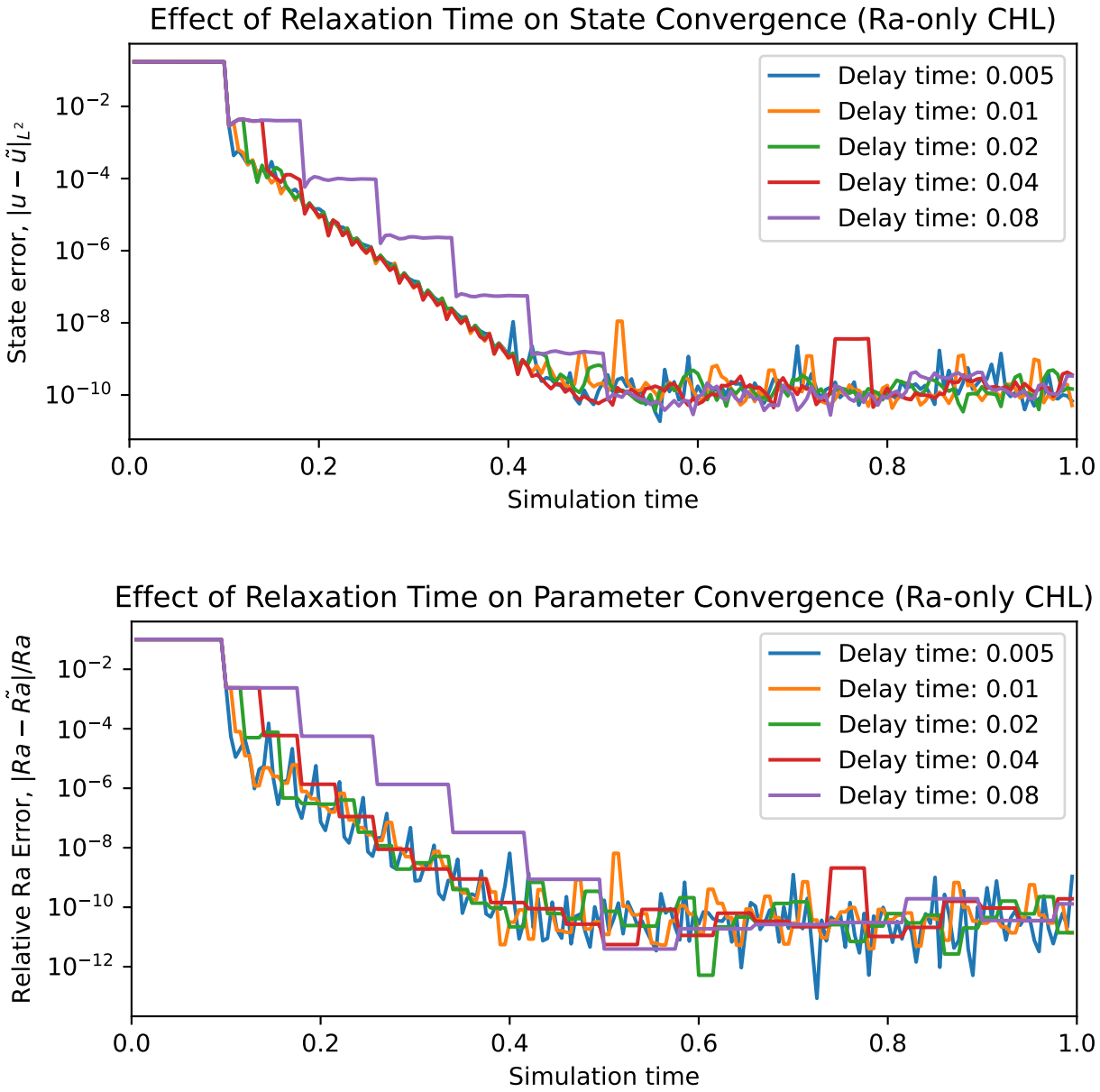


Figure 4.4: Comparing the performance of the Ra-only CHL algorithm when using different relaxation times

time, the state and parameters do converge to an error of order 10^{-10} and 10^{-11}, respectively. Additionally, unlike the Ra-only case above, they appear to converge at approximately the same rate.

4.2.2 Estimating Pr and Ra Simultaneously. In the multiparameter CHL algorithm, it is necessary to use the nondimensionalization (1.10) rather than the nondimensionalization (1.9) used elsewhere in this work. This is because (1.10) places Pr and Ra in separate equations. Due to the fact that the CHL algorithm requires letting the state error reach a steady level, then solving an equation to estimate the true parameter value, it is necessary to solve two equations for two parameters, which was not possible in the usual nondimensionalization. However, in this version as in the single-parameter versions, one can follow two approaches to derive an update formula. The "simple" approach is to eliminate terms which are quadratic in the error but not coupled to μ, while the "complex" approach is to strive to approximate those terms. Figures 4.6-4.8 show the difference in performance between these two approaches. In each figure, the algorithms are started from the same initial state. The true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Pr}} = 1.1$ and $\widetilde{\mathrm{Ra}} = 9 \times 10^4$. Each algorithms use $\mu = 8000$, and at time $t = 0.1$ start updating every 0.02 units of simulation time.

In each case it is clear that the "simple" approach fails to converge, and instead quickly blows up. However, the "complex" approach converges as long as the temperature nudging is turned on. It yields an error of about order 10^{-10} in the state and about 10^{-11} in the parameters. If μ_T, the temperature nudging parameter, is set to 0, then even the "complex" approach fails to converge because it does not update the estimate for Pr correctly.

Figure 4.9 shows the convergence for the "Complex" scheme at each value of μ_T, this time plotted together on the same graph. From these plots it is evident that setting $\mu_T = 0$ causes the Pr estimates to fail to change by much, thus hobbling the convergence of the state and the convergence of the Ra estimates. However, when $\mu_T > 0$ (and especially when $\mu_T = \mu = 8000$) the convergence is relatively quick for both the state and the parameters.

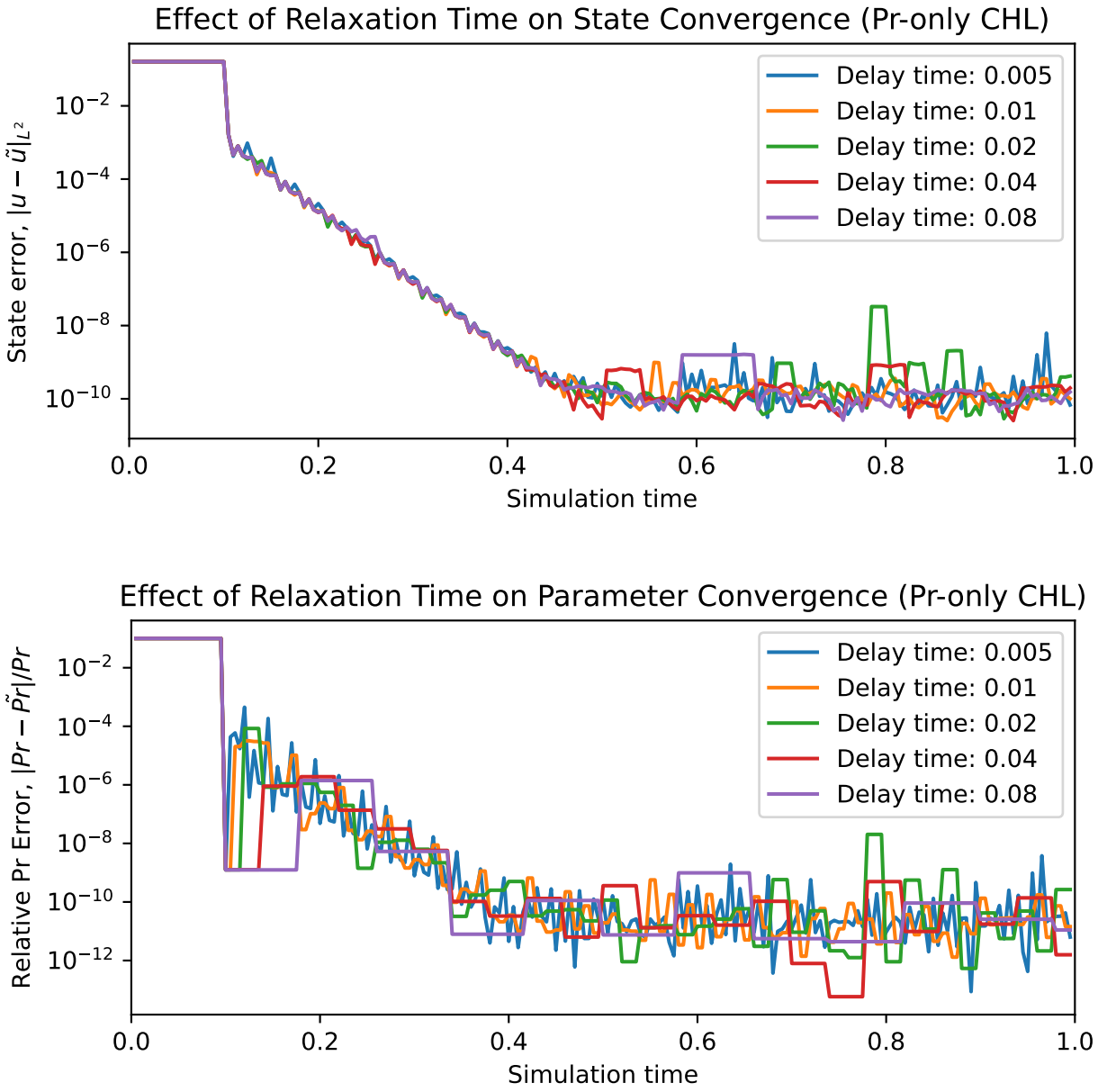


Figure 4.5: Comparing the performance of the Pr-only CHL algorithm when using different relaxation times

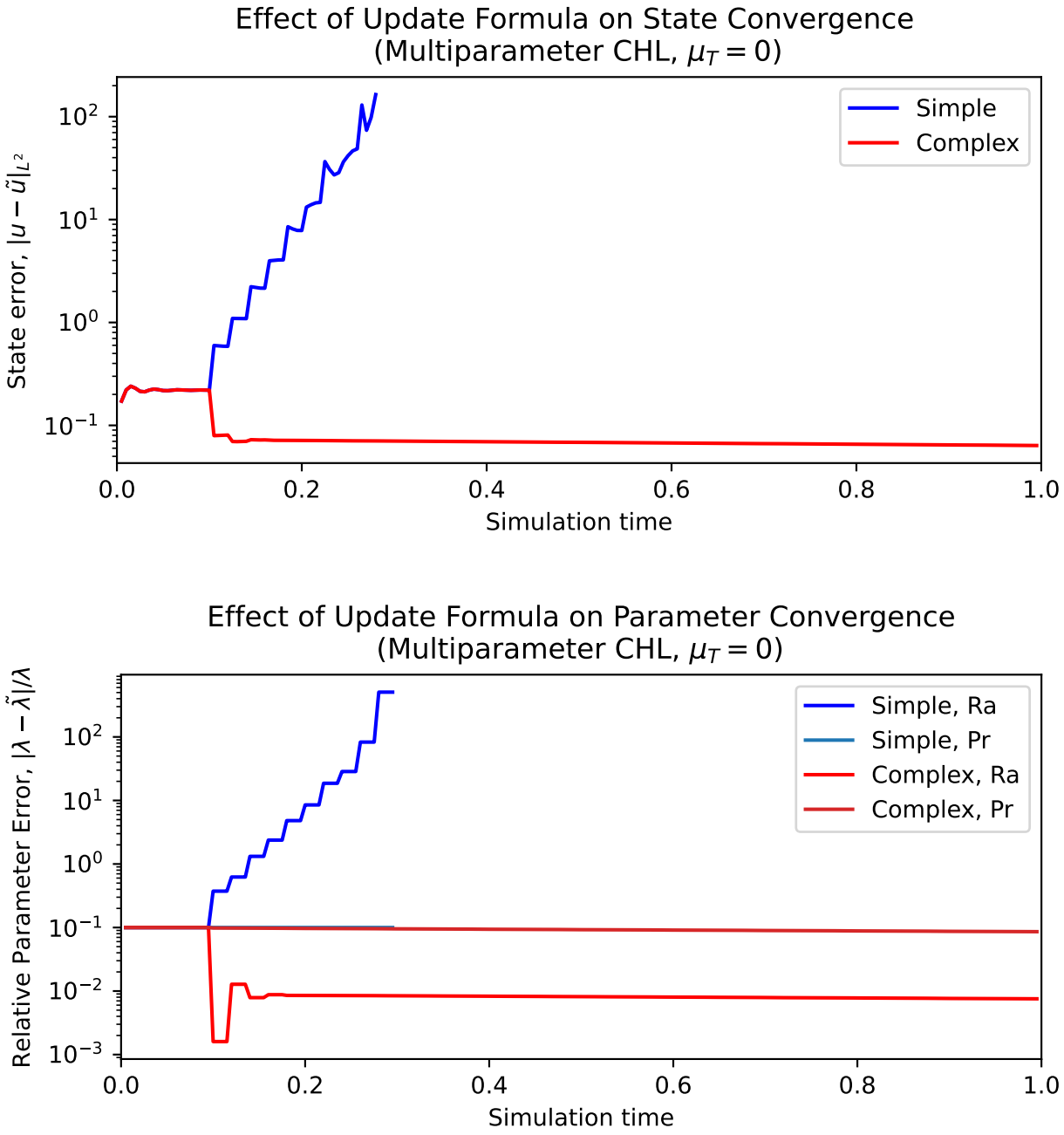


Figure 4.6: Comparing the performance of the multiparameter CHL algorithm when eliminating most terms ("Simple") and estimating them ("Complex"), without temperature nudging

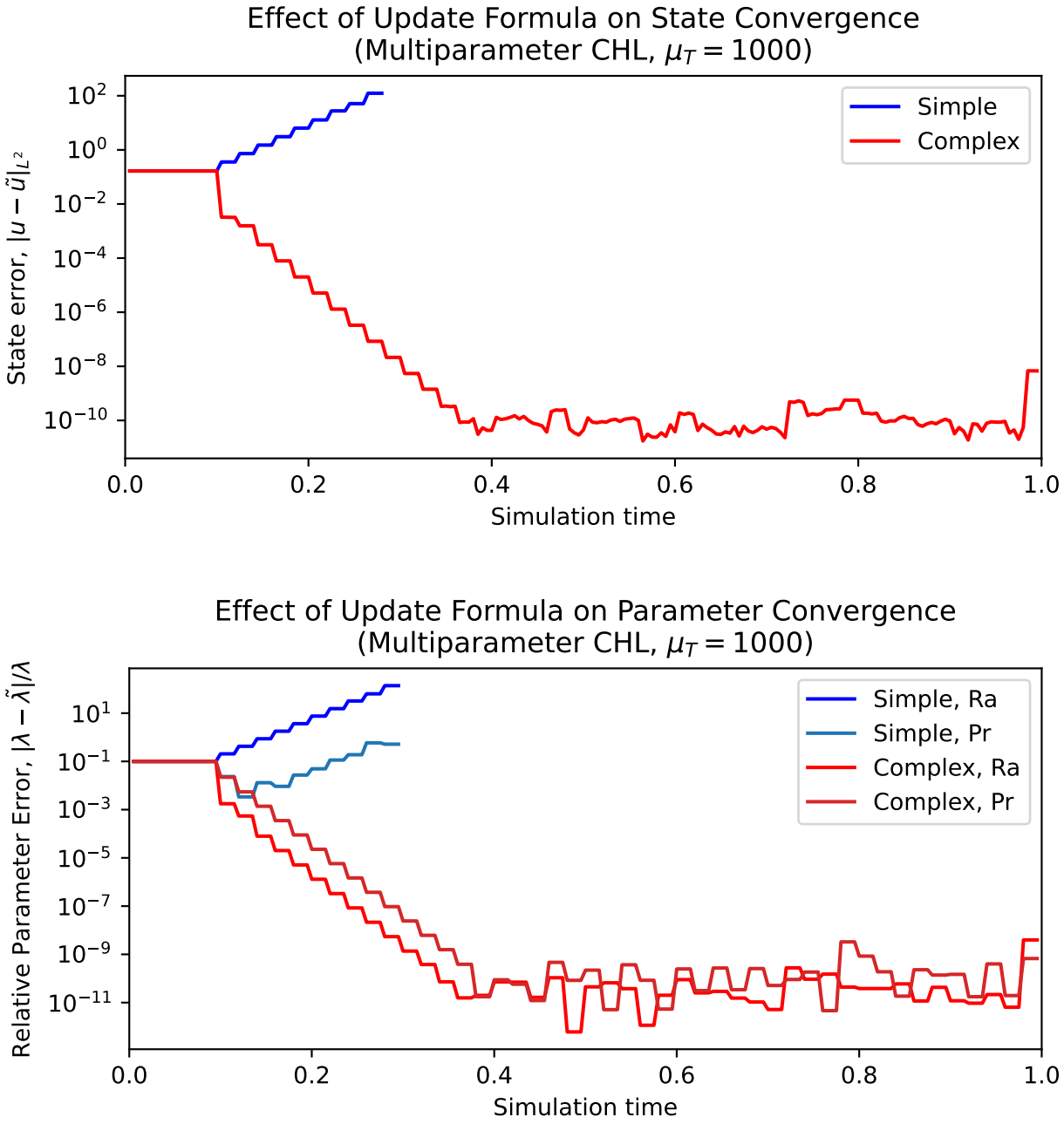


Figure 4.7: Comparing the performance of the multiparameter CHL algorithm when eliminating most terms ("Simple") and estimating them ("Complex"), with some temperature nudging

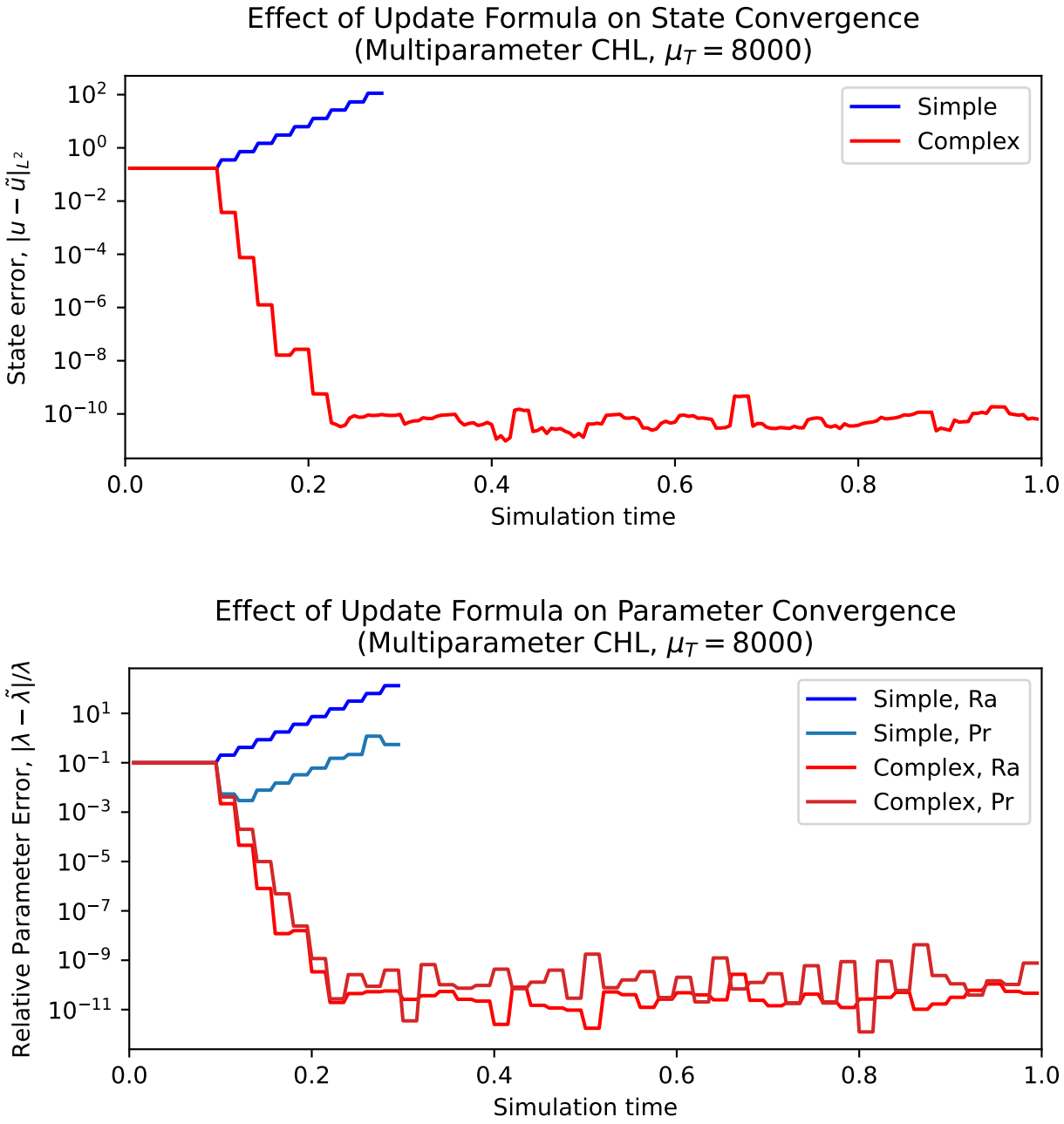
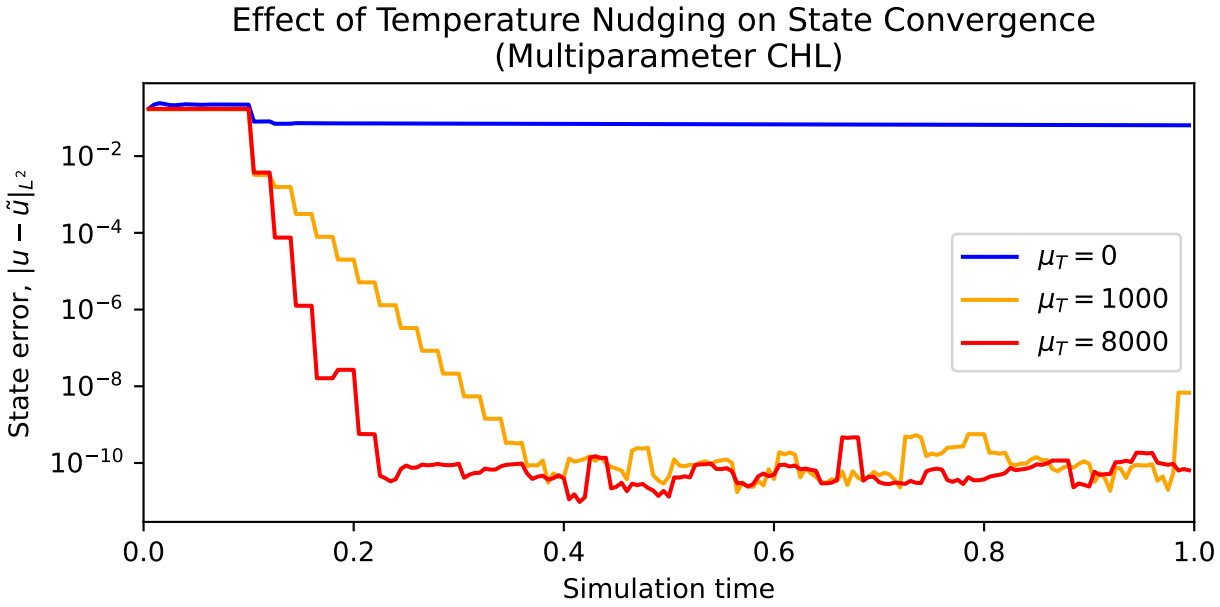


Figure 4.8: Comparing the performance of the multiparameter CHL algorithm when eliminating most terms ("Simple") and estimating them ("Complex"), with full temperature nudging



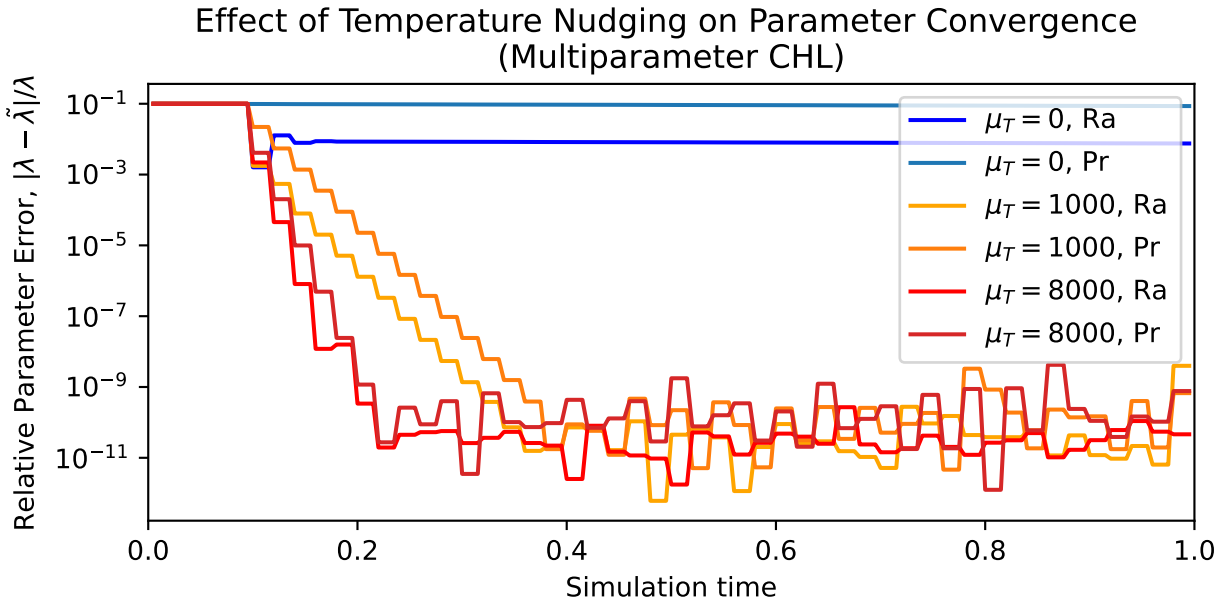


Figure 4.9: Analyzing the effect of temperature nudging on the multiparameter CHL algorithm

4.3 PWM Algorithm

4.3.1 Basis Selection. Figures 4.10-4.12 compare the performance of the PWM algorithm using two different bases and show that the selection of the basis matters in practice. In each figure, the initial states for the true systems and assimilating systems are identical, with the assimilating system starting at a low-mode projection of the true state. "Original," which uses a basis consisting of the projection of the state error and the projection of the x-derivative of the temperature (the linear term coupled with $\mathrm{Pr}\,\mathrm{Ra}$), is compared with "New," which uses the projections of the x-derivative of the temperature and laplacian of the vorticity (the optimal basis according to Section 2.1.3).

In Figure 4.10, the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Pr}} = 1.1$ and $\widetilde{\mathrm{Ra}} = 9 \times 10^4$. Both algorithms used $\mu = 8000$, and at time $t = 0.1$ started updating every 0.02 units of simulation time. The algorithm with the new basis converges quickly and achieves a state error of about 10^{-10} and parameter error of about 10^{-11}, while the algorithm with the original basis makes a bad update which causes the system to blow up at $t \approx 0.12$.

In Figure 4.11, the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Ra}} = 9 \times 10^4$ and $\widetilde{\mathrm{Pr}}$ is held at the true value. Both algorithms used $\mu = 8000$, and at time $t = 0.1$ started updating every 0.02 units of simulation time. The algorithm which uses the new basis converges quickly and achieves a state error of about 10^{-10} and parameter error of about 10^{-11}, while the algorithm with the original basis makes a bad update which causes the system to blow up at $t \approx 0.3$.

In Figure 4.12, the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Pr}} = 1.1$ and $\widetilde{\mathrm{Ra}}$ is held at the true value. Both algorithms used $\mu = 8000$, and at time $t = 0.1$ started updating every 0.02 units of simulation time. The algorithm which uses new basis converges quickly and achieves a state error of about 10^{-10} and parameter error of about 10^{-11}, while the algorithm with the original basis fails to converge and maintains a large error in both the parameter estimates and the system state.

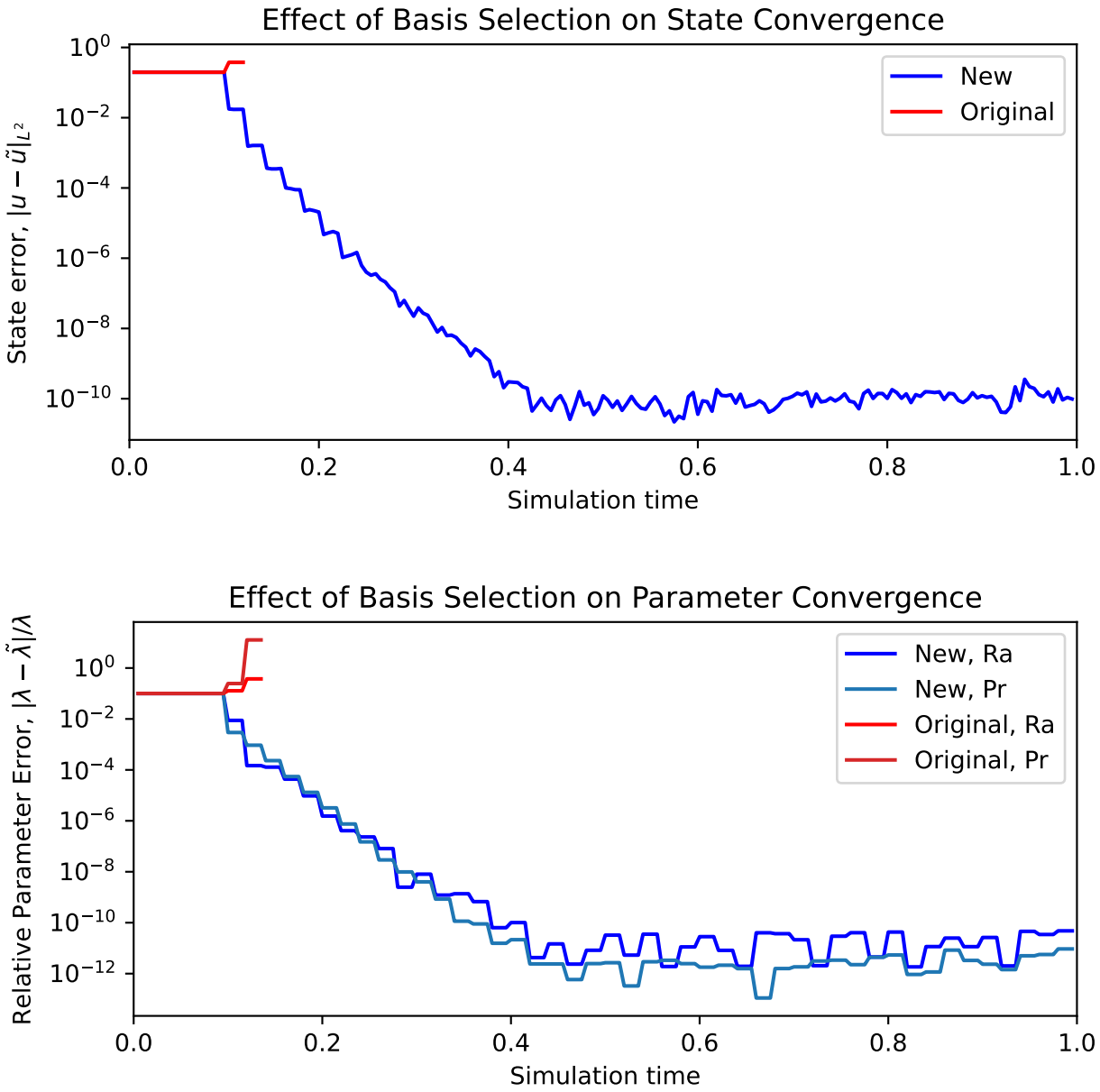


Figure 4.10: Comparing the performance of the multiparameter PWM algorithm using the original basis ("Original") and the new basis ("New")

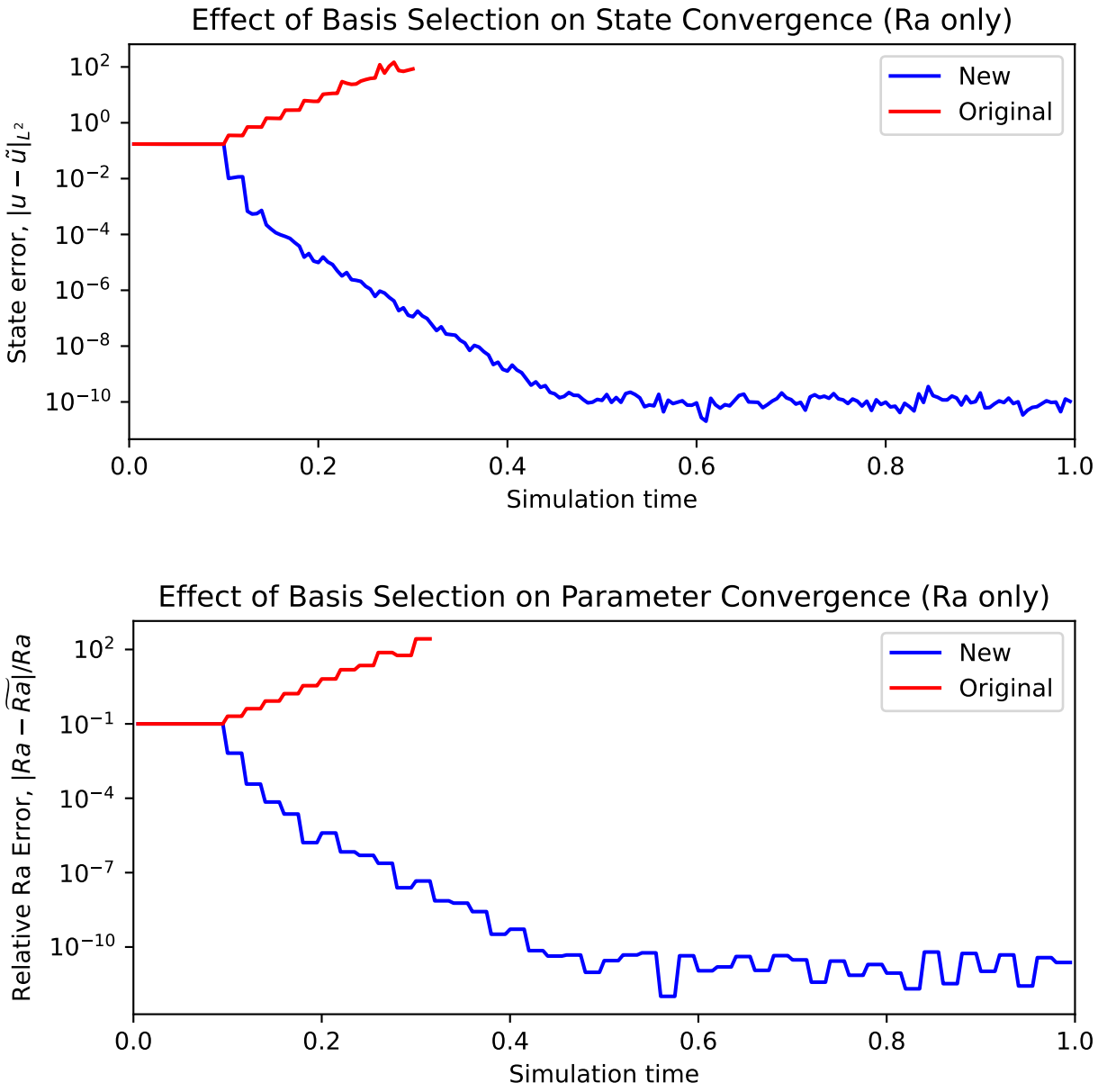


Figure 4.11: Comparing the performance of the Ra-only PWM algorithm using the original basis ("Original") and the new basis ("New")

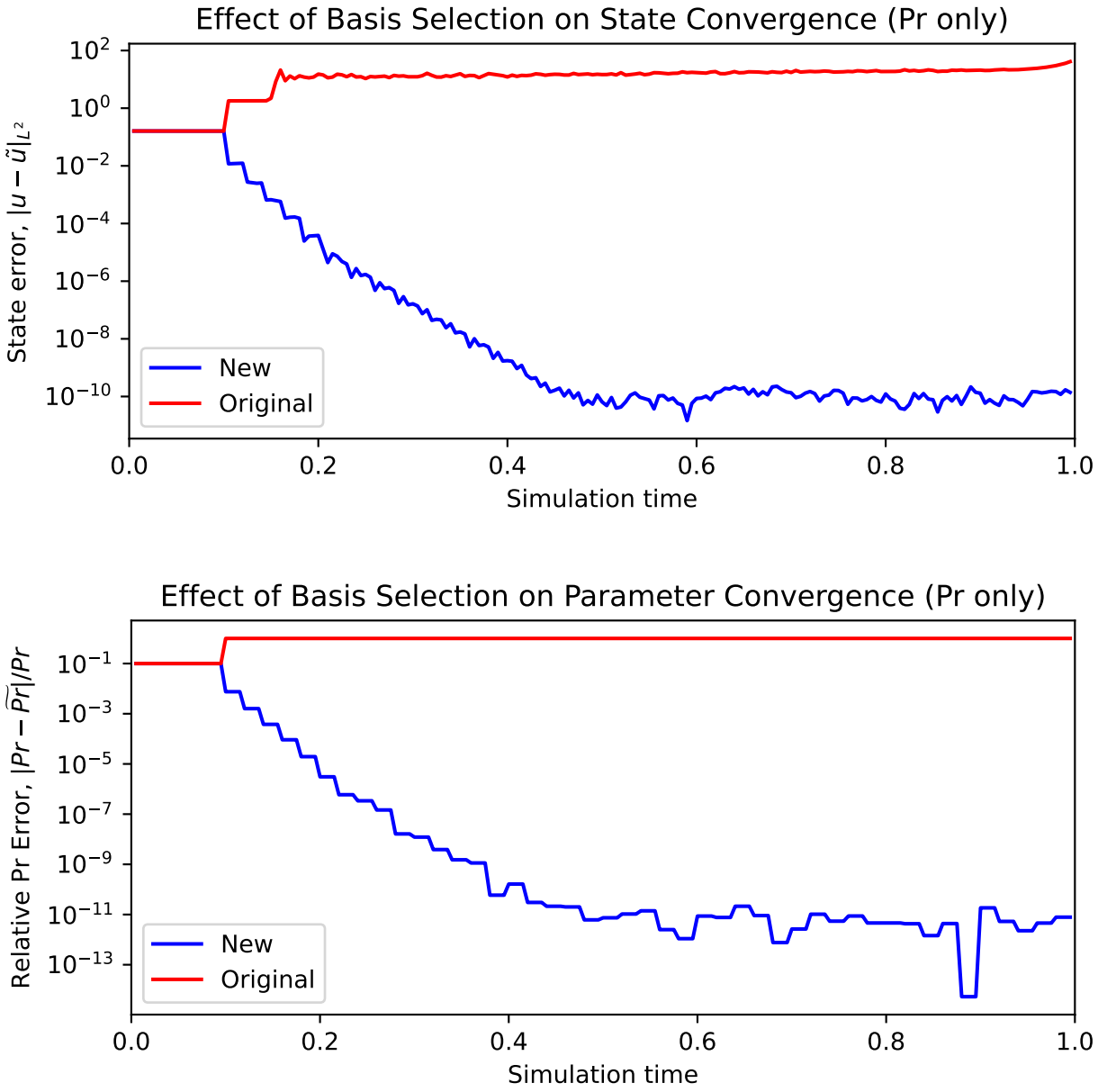


Figure 4.12: Comparing the performance the Pr-only PWM algorithm using the original basis ("Original") and the new basis ("New")

4.3.2 Relaxation Time Selection. Figures 4.13-4.15 show the effect of relaxation time on the convergence of the algorithm. In each figure, the initial states for the true systems and assimilating systems are identical, with the assimilating system starting at a low-mode projection of the true state. In each simulation, there is an initial relaxation period of 0.1 units of simulation time. Then a parameter update is made repeatedly after a relaxation interval with a length called the "delay time." The figures show the effect of changing the "delay time" on the convergence. The results seem to show that decreasing the relaxation time increases the rate of convergence until a certain threshold around 0.02 units of time, below which the performance of the algorithm is essentially the same.

In Figure 4.13, the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Pr}} = 1.1$ and $\widetilde{\mathrm{Ra}} = 9 \times 10^4$. Both algorithms used $\mu = 8000$. The algorithms with longer relaxation times of 0.08 and 0.04 take longer to converge, while all of the other relaxation times appear to converge at about the same rate (for the 0.02 time, the state appears to converge ever so slightly faster).

In Figure 4.14, the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Ra}} = 9 \times 10^4$ and $\widetilde{\mathrm{Pr}}$ is held at the true value. Both algorithms used $\mu = 8000$. The algorithms with the relaxation times of 0.08 take longer to converge, while all of the other relaxation times appear to converge at about the same rate.

In Figure 4.15, the true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Pr}} = 1.1$ and $\widetilde{\mathrm{Ra}}$ is held at the true value. Both algorithms used $\mu = 8000$. The algorithms with longer relaxation times of $(0.04, 0.08)$ take longer to converge, and the algorithm with the relaxation time of 0.02 converges more slowly at first, but soon catches up to the other two.

4.3.3 Temperature Nudging. Figure 4.16 shows the effect of temperature nudging on convergence in the multiparameter PWM algorithm. The true system has $\mathrm{Pr} = 1.0$ and $\mathrm{Ra} = 10^5$, while the assimilating system is initialized with $\widetilde{\mathrm{Pr}} = 1.1$ and $\widetilde{\mathrm{Ra}} = 9 \times 10^4$. Both algorithms used $\mu = 8000$, and at time $t = 0.1$ started updating every 0.02 units of

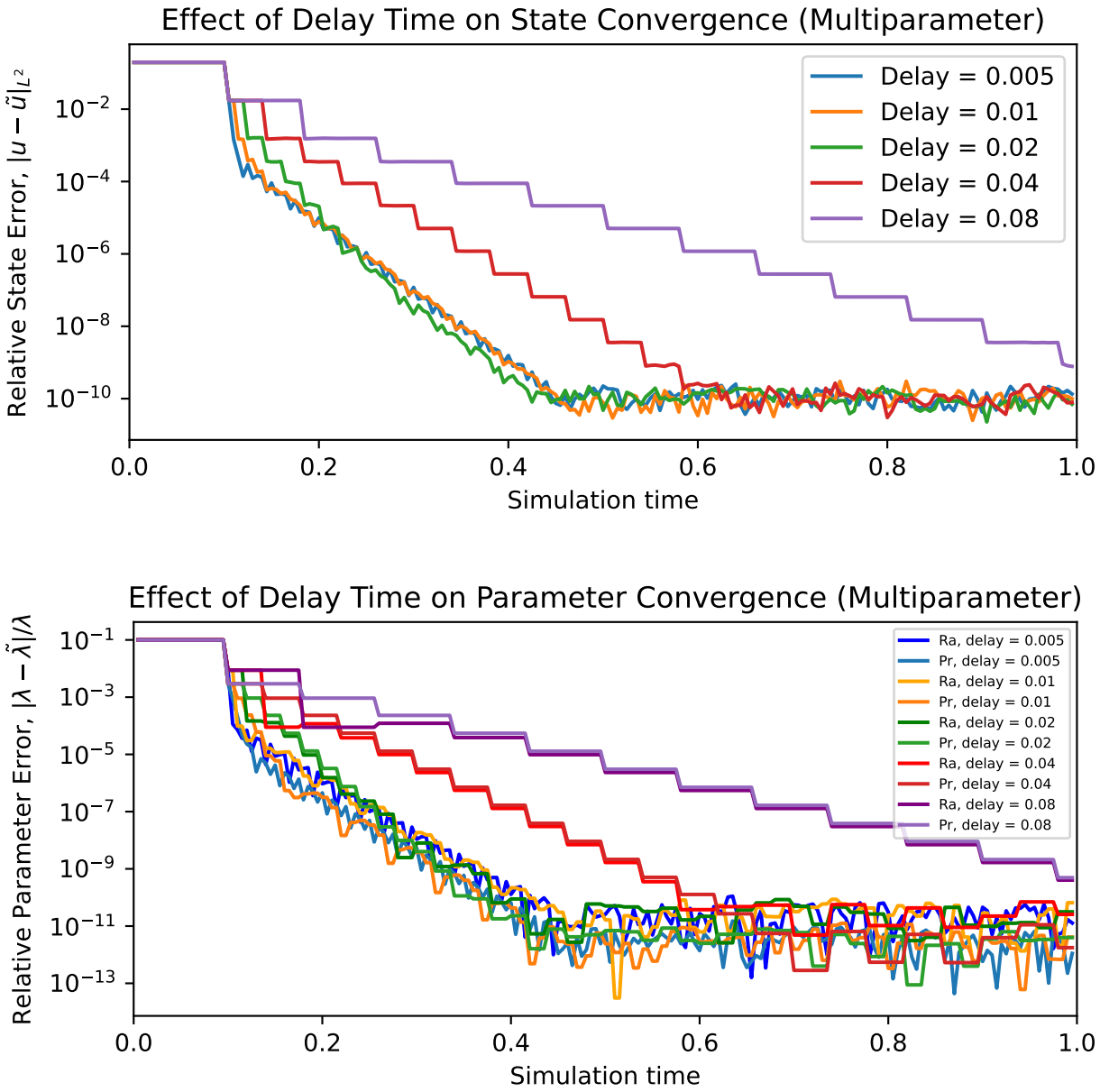


Figure 4.13: Comparing the performance of the multiparameter PWM algorithm at different relaxation times

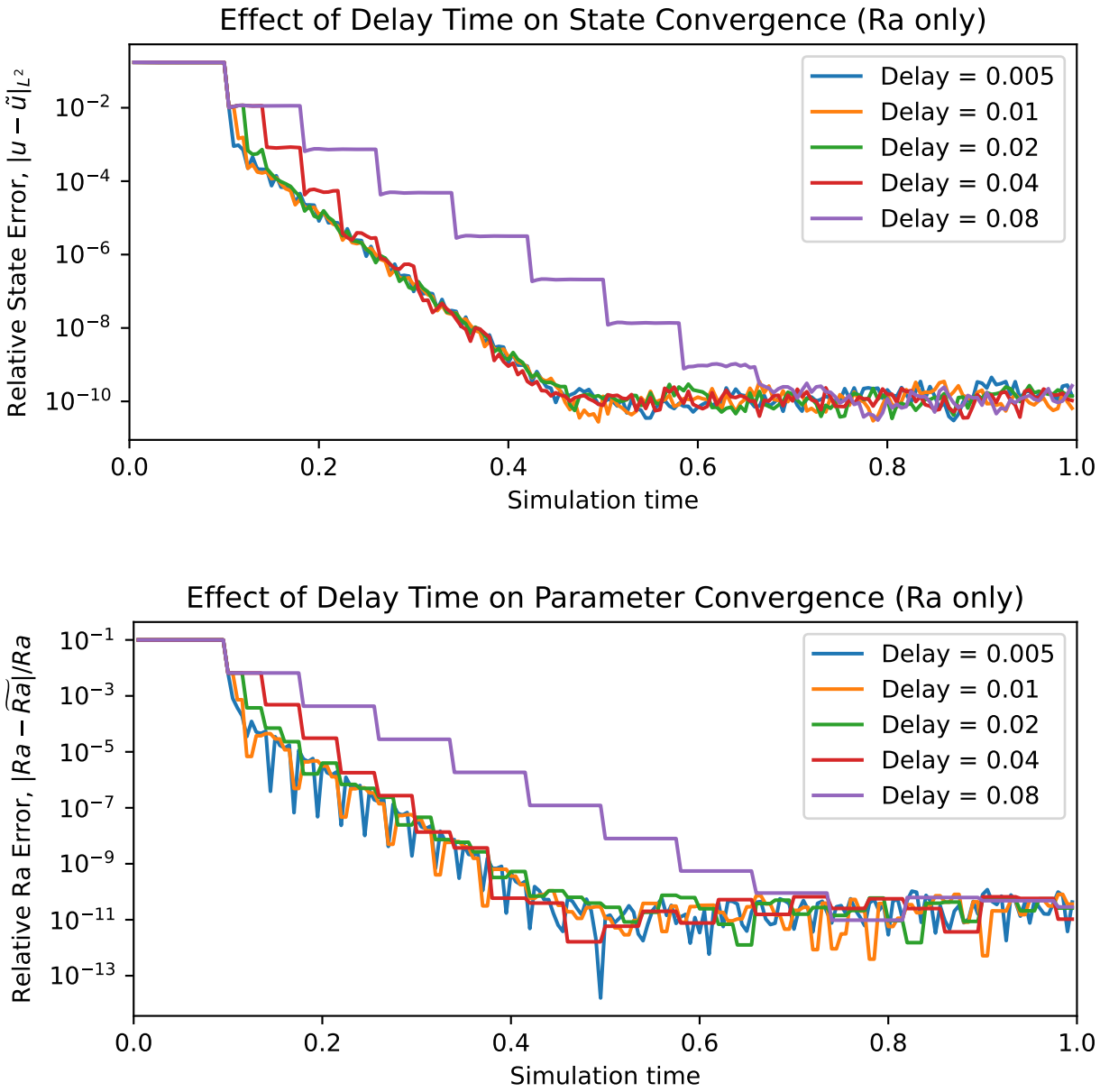


Figure 4.14: Comparing the performance of the Ra only PWM algorithm at different relaxation times

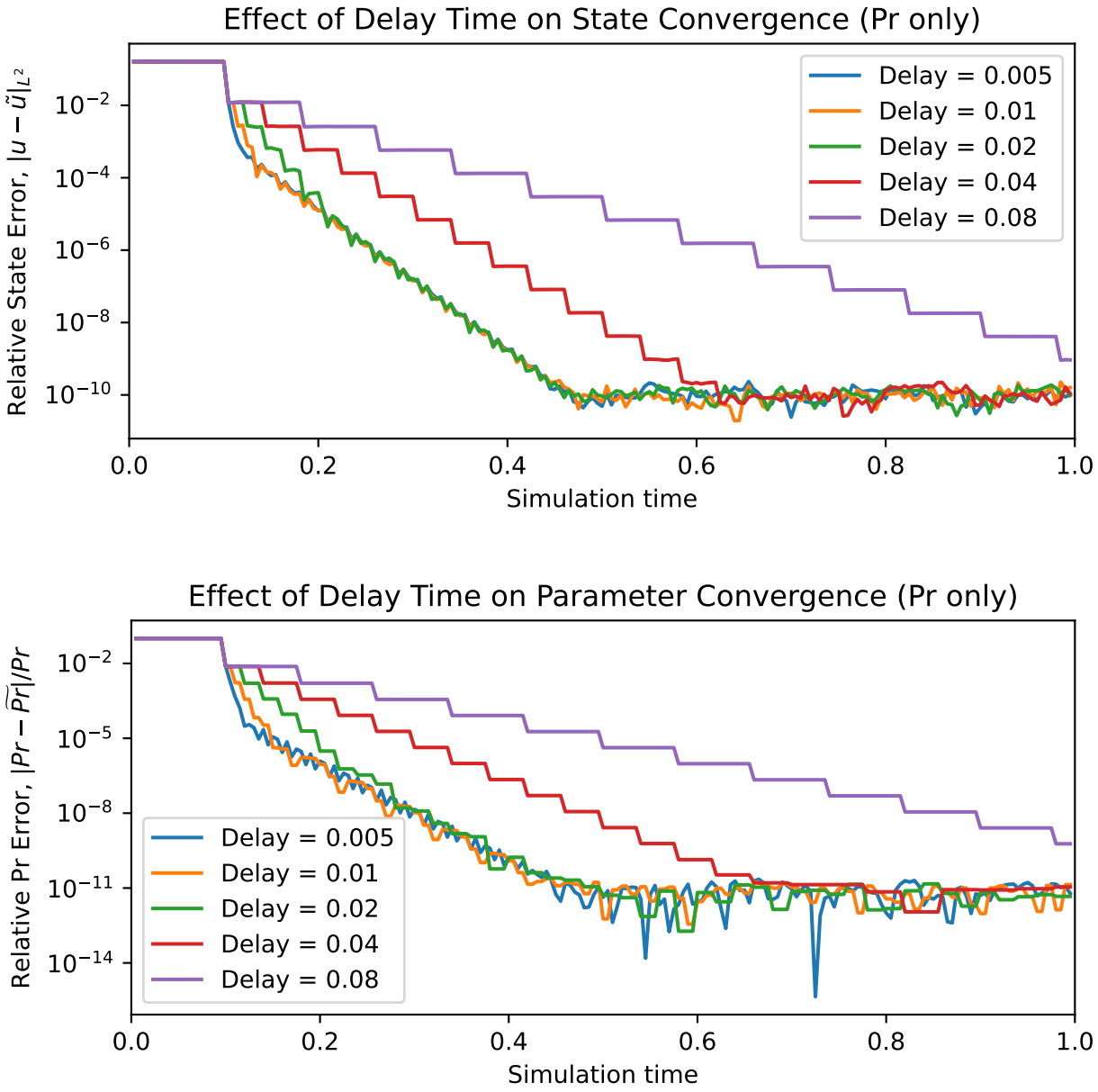


Figure 4.15: Comparing the performance of the Pr only PWM algorithm at different relaxation times

Algorithm	Average State Error	Average Relative Ra Error	Average Relative Pr Error
PWM	2.25×10^{-11}	1.17×10^{-11}	1.50×10^{-12}
CHL	7.43×10^{-11}	4.01×10^{-11}	4.51×10^{-10}

Table 4.1: Averaged errors over the interval $t \in [0.4, 1]$ for PWM and CHL algorithms

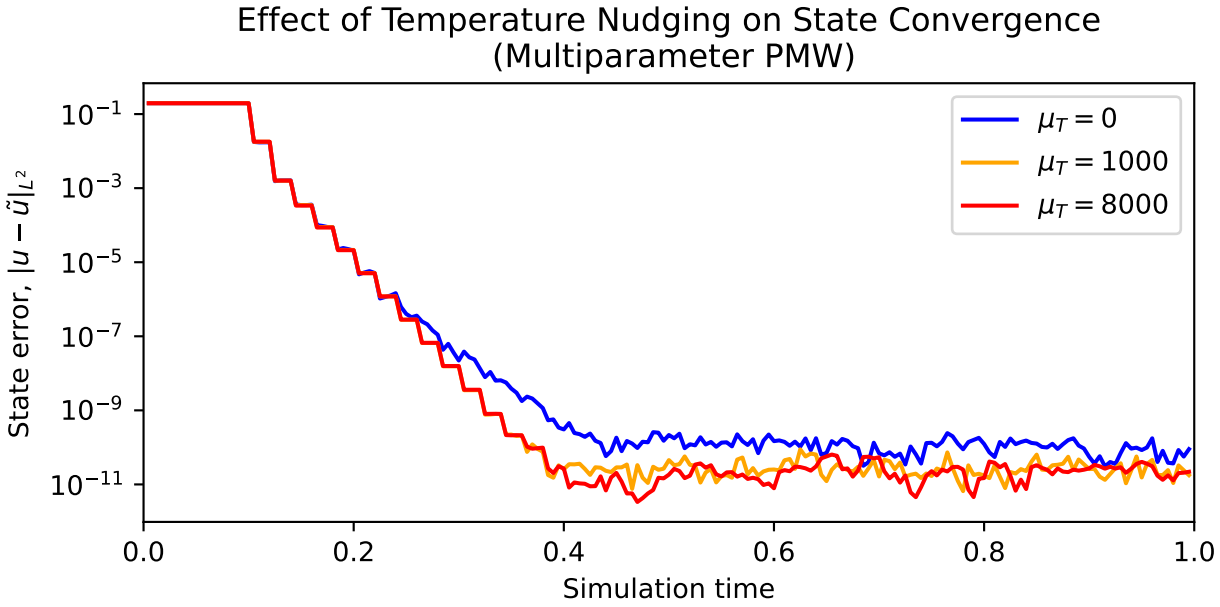
simulation time. The algorithm converges quickly in each case, although with higher values of μ_T it achieves a lower error of about order 10^{-11} in the state and order 10^{-12} in the parameters, compared with the baseline ($\mu_T = 0$) which is roughly an order of magnitude worse in the error.

4.4 Comparing CHL and PWM Algorithms

Figure 4.17 compares the performance of CHL and PWM multiparameter algorithms when utilized on the same initial conditions. Both are started at the same initial state, while their assimilating systems are initialized to a low-mode projection of that state. In each case, the true system has Pr = 1.0 and Ra = 10^5, while the assimilating system is initialized with $\widetilde{\text{Pr}} = 1.1$ and $\widetilde{\text{Ra}} = 9 \times 10^4$. In the simulations shown in the plots below, the algorithms used a relaxation time of 0.02. They used vorticity nudging with $\mu = 8000$ and temperature nudging, also with $\mu_T = 8000$. Note that in Section 4.2.2 (and especially in Figure 4.9), it is documented that without temperature nudging, the CHL multiparameter algorithm fails to converge. Hence, to compare the two algorithms side-by-side it was chosen to include temperature nudging.

In Figure 4.17 it is clear that the CHL algorithm converges more quickly both in state and parameters; the error from the CHL algorithm seems to bottom out at just after time $t = 0.2$. The PWM algorithm converges more slowly (and this is probably not because of the chosen relaxation time; see Section 4.3.2). However, it reaches an error that is smaller than the CHL algorithm. This can be seen in Table 4.1.

Table 4.1 shows that PWM produced a state error and a Ra error which were more than three times smaller than the errors from CHL, while the Pr error was more than 300 times



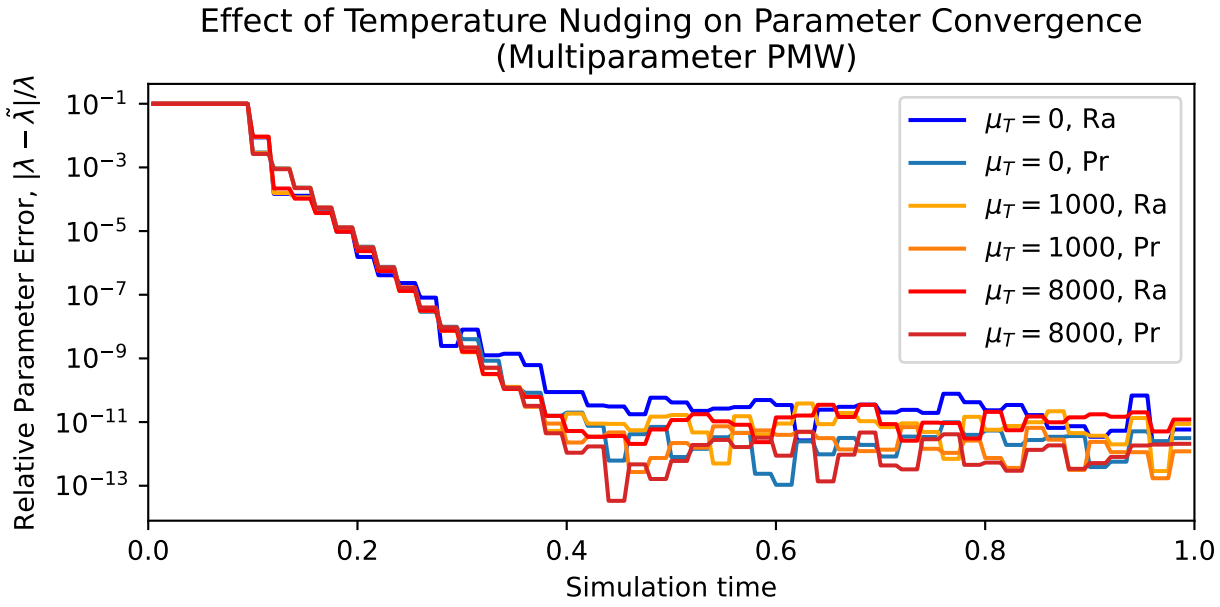


Figure 4.16: Analyzing the effect of temperature nudging on the multiparameter PWM algorithm

smaller. This is an interesting phenomenon which lacks explanation. Indeed, the Pr error was smaller than the Ra error for PWM, while it was larger than the Ra error in CHL.

Chapter 5. Conclusion

This work proposes and derives several algorithms for simultaneous data assimilation and parameter recovery in Rayleigh-Bénard convection. Beyond deriving these algorithms, it is also shown through computational evidence (and mathematical analysis in the case of the PWM algorithm) which of these algorithms succeed in estimating parameters and the state of the convective system correctly.

The mathematical analysis put forward in this work claims that the PWM algorithm will succeed in forcing parameter recovery as long as the chosen nudging parameter μ is chosen large enough. In practice, as long as the basis is chosen well (see Sections 2.1.3 and 4.3.1), the value $\mu = 8000$ is sufficient to force parameter recovery for the situations described here. Choosing a bad basis was enough to prevent the PWM algorithm from converging. Because, from the point of view of the analysis, the choice of basis did not affect convergence, it bears pondering why this might have taken place. The analysis makes several assumptions that may not have been satisfied by this computational setup: it assumes that relaxation times are sufficiently large, that the state of the system lies in an absorbing ball, and that the nudged system with the proposed parameter update is a well-posed system with similar absorbing ball bounds. It must be the case that some or all of these assumptions were not fully satisfied in the case of the PWM simulations which did not converge. Nevertheless, the combination of analytical and computational evidence presented herein clearly demonstrates the potential of the PWM algorithm, which had previously only been tested on the Kuramoto-Sivashinsky equations (see [9]), but is here shown to be effective for Rayleigh-Bénard convection.

Additionally, this work presents interesting new findings related to the CHL algorithm. In particular, the computational evidence presented in Sections 4.2.1 and 4.2.2 shows that the "simple" method of deriving the CHL update formula (wherein terms quadratic in the

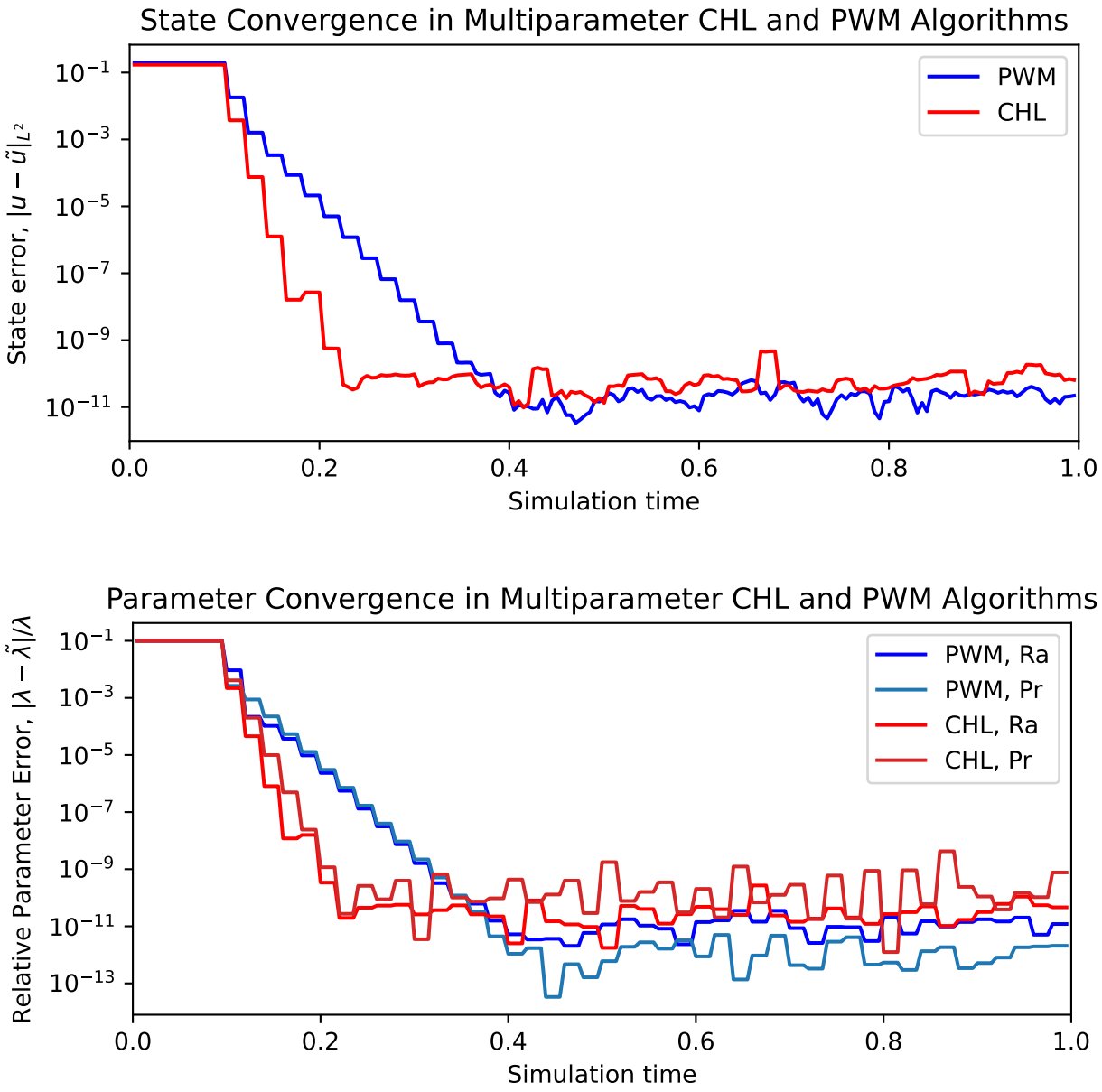


Figure 4.17: Comparing the convergence of CHL and PWM algorithms

error are discarded unless coupled with a nudging parameter) fails to ensure convergence for Rayleigh-Bénard convection in many cases. To force parameter and state convergence, approximating these terms is necessary. This work also derives a multiparameter CHL update formula using a non-standard nondimensionalization (see Section 2.1.2), and demonstrates via numerical simulation that this procedure is successful as long as temperature nudging is used in addition to vorticity nudging (see Sections 4.2.2 and 4.4).

Lastly, the computational framework developed for Rayleigh-Bénard convection using Dedalus demonstrates how that platform can be used for testing parameter recovery algorithms. To the author's knowledge, these types of parameter recovery algorithms have never been demonstrated on a system as computationally expensive as Rayleigh-Bénard convection before.

5.1 Further Work

While this work addresses and answers several important questions about data assimilation and parameter recovery for Rayleigh-Bénard convection, it also opens the door to further research. Because these nudging-based algorithms are so new, there is much more that can be done to increase understanding of how, why, and when these procedures work.

From a mathematical perspective, the derivation of the updates and the proof of convergence for the PWM algorithm in each case rested on a so-called "nondegeneracy condition." Further work is needed to explore and explain these conditions. Relevant questions include, what does the "nondegeneracy condition" tell us about the system? To what extent can the system converge when the nondegeneracy conditions almost fail? Are there algorithmic or mathematical ways to circumvent these conditions?

This work presents a proof of convergence for the PWM algorithm. However, the proof rests on some shaky assumptions, and more work is needed to make the proof fully rigorous and describe how the assumptions are relevant in the computational and practical application of these algorithms. Furthermore, work is needed to establish a similar proof for the CHL

algorithm in the case of Rayleigh-Bénard convection.

Computationally, further study is needed to replicate these results in different dynamical regimes. This work only considers systems with $\mathrm{Ra} = 10^5, \mathrm{Pr} = 1$, but future work could use higher Ra numbers (which correspond to more turbulent regimes) and different Pr numbers (smaller, larger, or infinite) which correspond to different settings for convection. This would be a fruitful area of study because different regimes have different applications (see [14] for exploration of large-Prandtl systems), and because turbulence may affect the performance of these algorithms. It is possible that greater turbulence would aid in convergence because of the greater amount of dissipation; but it could also sufficiently destabilize the system that the types of updates considered here would no longer perform well. It would also be productive to consider convection with different geometries and boundary conditions than those considered here.

This work essentially assumes that error in observations is due to lack of spatial resolution, rather than measurement error. Another interesting future research direction would be to evaluate these types of parameter recovery procedures in situations where there is random noise, or where orthogonal Fourier projections are no longer a good model to represent spatially sparse observations.

Lastly, the scientific community should attempt to apply the PWM and CHL algorithms to different systems (for example, the Navier-Stokes equations) and try to adapt them to estimate even more variables (for instance, forcing in a forced dynamical system).

Bibliography

[1] E. L. Koschmieder. *Béard cells and Taylor vortices*. Press Syndicate of the University of Cambridge, Cambridge, 1993.

[2] Alexander V. Getling. *Rayleigh-Bénard Convection: Structures and Dynamics*. World Scientific, Singapore, 1998.

[3] Aseel Farhat, Michael S Jolly, and Edriss S Titi. Continuous data assimilation for the 2d bénard convection through velocity measurements alone. *Physica D: Nonlinear Phenomena*, 303:59–66, 2015.

[4] MU Altaf, ES Titi, T Gebrael, OM Knio, L Zhao, MF McCabe, and Ibrahim Hoteit. Downscaling the 2d bénard convection equations using continuous data assimilation. *Computational Geosciences*, 21(3):393–410, 2017.

[5] Aseel Farhat, Hans Johnston, Michael Jolly, and Edriss S Titi. Assimilation of nearly turbulent rayleigh–bénard flow through vorticity or local circulation measurements: a computational study. *Journal of Scientific Computing*, 77(3):1519–1533, 2018.

[6] Aseel Farhat, Nathan E Glatt-Holtz, Vincent R Martinez, Shane A McQuarrie, and Jared P Whitehead. Data assimilation in large prandtl rayleigh–bénard convection from thermal measurements. *SIAM Journal on Applied Dynamical Systems*, 19(1):510–540, 2020.

[7] Abderrahim Azouani, Eric Olson, and Edriss S Titi. Continuous data assimilation using general interpolant observables. *Journal of Nonlinear Science*, 24(2):277–304, 2014.

[8] Elizabeth Carlson, Joshua Hudson, and Adam Larios. Parameter recovery for the 2 dimensional navier–stokes equations via continuous data assimilation. *SIAM Journal on Scientific Computing*, 42(1):A250–A270, 2020.

[9] Benjamin Pachev, Jared P Whitehead, and Shane A McQuarrie. Concurrent multi-parameter learning demonstrated on the kuramoto-sivashinsky equation. *arXiv preprint arXiv:2106.06069*, 2021.

[10] James Cooper Robinson and C Pierre. Infinite-dimensional dynamical systems: An introduction to dissipative parabolic pdes and the theory of global attractors. cambridge texts in applied mathematics. *Appl. Mech. Rev.*, 56(4):B54–B55, 2003.

[11] Elizabeth Carlson, Joshua Hudson, Adam Larios, Vincent R Martinez, Eunice Ng, and Jared P Whitehead. Dynamically learning the parameters of a chaotic system using partial observations. *arXiv preprint arXiv:2108.08354*, 2021.

[12] Vincent R Martinez. Convergence analysis of a viscosity parameter recovery algorithm for the 2d navier–stokes equations. *Nonlinearity*, 35(5):2241, 2022.

[13] Keaton J. Burns, Geoffrey M. Vasil, Jeffrey S. Oishi, Daniel Lecoanet, and Benjamin P. Brown. Dedalus: A flexible framework for numerical simulations with spectral methods. *Phys. Rev. Research*, 2:023068, Apr 2020.

[14] Shane Alexander McQuarrie. *Data Assimilation in the Boussinesq Approximation for Mantle Convection*. Brigham Young University, 2018.