COMMUNICATIONS OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 2, Pages 283–344 (October 14, 2022)
https://doi.org/10.1090/cams/10

A FRAMEWORK FOR MACHINE LEARNING OF MODEL ERROR IN DYNAMICAL SYSTEMS

MATTHEW E. LEVINE AND ANDREW M. STUART

ABSTRACT. The development of data-informed predictive models for dynamical systems is of widespread interest in many disciplines. We present a unifying framework for blending mechanistic and machine-learning approaches to identify dynamical systems from noisily and partially observed data. We compare pure data-driven learning with hybrid models which incorporate imperfect domain knowledge, referring to the discrepancy between an assumed truth model and the imperfect mechanistic model as *model error*. Our formulation is agnostic to the chosen machine learning model, is presented in both continuous- and discrete-time settings, and is compatible both with model errors that exhibit substantial memory and errors that are memoryless.

First, we study memoryless linear (w.r.t. parametric-dependence) model error from a learning theory perspective, defining excess risk and generalization error. For ergodic continuous-time systems, we prove that both excess risk and generalization error are bounded above by terms that diminish with the square-root of T, the time-interval over which training data is specified.

Secondly, we study scenarios that benefit from modeling with memory, proving universal approximation theorems for two classes of continuous-time recurrent neural networks (RNNs): both can learn memory-dependent model error, assuming that it is governed by a finite-dimensional hidden variable and that, together, the observed and hidden variables form a continuous-time Markovian system. In addition, we connect one class of RNNs to reservoir computing, thereby relating learning of memory-dependent error to recent work on supervised learning between Banach spaces using random features.

Numerical results are presented (Lorenz '63, Lorenz '96 Multiscale systems) to compare purely data-driven and hybrid approaches, finding hybrid methods less data-hungry and more parametrically efficient. We also find that, while a continuous-time framing allows for robustness to irregular sampling and desirable domain-interpretability, a discrete-time framing can provide similar or better predictive performance, especially when data are undersampled and the vector field defining the true dynamics cannot be identified. Finally, we demonstrate numerically how data assimilation can be leveraged to learn hidden dynamics from noisy, partially-observed data, and illustrate challenges in representing memory by this approach, and in the training of such models.

Received by the editors July 13, 2021, and, in revised form, May 11, 2022, August 17, 2022, and August 21, 2022.

2020 *Mathematics Subject Classification.* Primary 68T30, 37A30, 37M10; Secondary 37M25, 41A30.

Key words and phrases. Dynamical systems, model error, statistical learning, random features, recurrent neural networks, reservoir computing.

The work of the first and second authors was supported by NIH RO1 LM012734 "Mechanistic Machine Learning". The first author was also supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1745301. The second author was also supported by NSF (award AGS-1835860), NSF (award DMS-1818977), the Office of Naval Research (award N00014-17-1-2079), and the AFOSR under MURI award number FA9550-20-1-0358 (Machine Learning and Physics-Based Modeling and Simulation). The computations presented here were conducted in the Resnick High Performance Center, a facility supported by Resnick Sustainability Institute at the California Institute of Technology.

1. Introduction

1.1. **Background and literature review.** The modeling and prediction of dynamical systems and time-series is an important goal in numerous domains, including biomedicine, climatology, robotics, and the social sciences. Traditional approaches to modeling these systems appeal to careful study of their mechanisms, and the design of targeted equations to represent them. These carefully built *mechanistic models* have impacted humankind in numerous arenas, including our ability to land spacecraft on celestial bodies, provide high-fidelity numerical weather prediction, and artificially regulate physiologic processes, through the use of pacemakers and artificial pancreases, for example. This paper focuses on the learning of *model error*: we assume that an imperfect mechanistic model is known, and that data are used to improve it. We introduce a framework for this problem, focusing on distinctions between Markovian and non-Markovian model error, providing a unifying review of relevant literature, developing some underpinning theory related to both the Markovian and non-Markovian settings, and presenting numerical experiments which illustrate our key findings.

To set our work in context, we first review the use of data-driven methods for time-dependent problems, organizing the literature review around four themes comprising Sections 1.1.1 to 1.1.3 and 1.1.5; these are devoted, respectively, to pure data-driven methods, hybrid methods that build on mechanistic models, non-Markovian models that describe memory, and applications of the various approaches. Having set the work in context, in Section 1.2 we detail the contributions we make, and describe the organization of the paper.

1.1.1. *Data-driven modeling of dynamical systems.* A recent wave of machine learning successes in data-driven modeling, especially in imaging sciences, has shown that we can demand even more from existing models, or that we can design models of more complex phenomena than heretofore. Traditional models built from, for example, low order polynomials and/or linearized model reductions, may appear limited when compared to the flexible function approximation frameworks provided by neural networks and kernel methods. Neural networks, for example, have a long history of success in modeling dynamical systems [50, 83, 86, 112, 139–141] and recent developments in deep learning for operators continue to propel this trend [12, 92, 93, 102].

The success of neural networks arguably relies on balanced expressivity and generalizability, but other methods also excel in learning parsimonious and generalizable representations of dynamics. A particularly popular methodology is to perform sparse regression over a dictionary of vector fields, including the use of thresholding approaches (SINDy) [19] and L_1-regularized polynomial regression [145–147, 159]. Nonparametric methods, like Gaussian process models [136], have also been used widely for modeling nonlinear dynamics [27, 46, 81, 165]. While a good choice of kernel is often essential for the success of these methods, recent progress has been made towards automatic hyperparameter tuning via parametric kernel flows [61]. Successes with Gaussian process models were also extended to high dimensional problems by using random feature map approximations [131] within the context of data-driven learning of parametric partial differential equations (PDEs) and solution operators [113]. Advancements to data-driven methods based on Koopman operator theory and dynamic

mode decomposition also offer exciting new possibilities for predicting nonlinear dynamics from data [1, 82, 160].

It is important to consider whether to model in discrete- or continuous-time, as both have potential advantages. The primary positive for continuous-time modeling lies in its flexibility and interpretability. In particular, continuous-time approaches are more readily and naturally applied to irregularly sampled time-series data, e.g. electronic health record data [142], than discrete-time methods. Furthermore, this flexibility with respect to timestep enables simple transferability of a model learnt from discrete-time data at one timestep, to new settings with a different timestep and indeed to variable timestep settings; the learned right-hand-side can be used to generate numerical solutions at any timestep. On the other hand, applying a discrete-time model to a new timestep either requires exact alignment of subsampled data or some post-processing interpolation step. Continuous-time models may also provide greater interpretability than discrete-time methods when the right-hand-side of the ordinary differential equation (ODE) is a more physically interpretable object than the Δt-solution operator (e.g. for equation discovery [72]).

Traditional implementations of continuous-time learning require accurate estimation of time-derivatives of the state, but this may be circumvented using approaches that leverage autodifferentiation software [73, 118, 142] or methods which learn from statistics derived from time-series, such as moments or correlation functions [150]. Keller and Du [77], Du et al. [39] provide rigorous analysis demonstrating how inference of a continuous-time model from discrete-time data must be conducted with great care; they prove how stable and consistent linear multistep methods for continuous-time integration may not possess the same guarantees when used for the inverse problem, i.e. discovery of dynamics. Queiruga et al. [128] provide pathological illustrations of this phenomenon in the context of Runge-Kutta methods.

Discrete-time approaches, on the other hand, are easily deployed when train and test data sample rates are the same. For applications in which data collection is easily configured (e.g. simulated settings, available automatic sensors, etc.), discrete-time methods are typically much easier to implement and test than continuous-time methods. Moreover, they allow for "nonintrusive" model correction, as additions are applied outside of the numerical integrator; this may be relevant for practical integration with complex simulation software. In addition, discrete-time approaches can be preferable when there is unavoidably large error in continuous-time inference Chorin and Lu [30], Lu et al. [100].

Both nonparametric and parametric model classes are used in the learning of dynamical systems, with the latter connecting to the former via the representer theorem, when Gaussian process regression [136] is used [20, 49, 62].

1.1.2. *Hybrid mechanistic and data-driven modeling.* Attempts to transform domains that have relied on traditional mechanistic models, by using purely data-driven (i.e. *de novo* or "learn from scratch") approaches, often fall short. Now, there is a growing recognition by machine learning researchers that these mechanistic models are very valuable [110], as they represent the distillation of centuries of data collected in countless studies and interpreted by domain experts. Recent studies have consistently found advantages of hybrid methods that blend mechanistic knowledge and data-driven techniques; Willard et al. [169] provide a thorough review of this shift amongst scientists

and engineers. Not only do these hybrid methods improve predictive performance [121], but they also reduce data demands [129] and improve interpretability and trustworthiness, which is essential for many applications. This is exemplified by work in autonomous drone landing [153] and helicopter flying [130], as well as predictions for COVID-19 mortality risk [155] and COVID-19 treatment response [127].

The question of how best to use the power of data and machine learning to leverage and build upon our existing mechanistic knowledge is thus of widespread current interest. This question and research direction has been anticipated over the last thirty years of research activity at the interface of dynamical systems with machine learning [98, 141, 170], and now a critical mass of effort is developing. A variety of studies have been tackling these questions in weather and climate modeling [43, 76]; even in the imaging sciences, where pure machine learning has been spectacularly successful, emerging work shows that incorporating knowledge of underlying physical mechanisms improves performance in a variety of image recognition tasks [5].

As noted and studied by Ba et al. [5], Freno and Carlberg [45] and others, there are a few common high-level approaches for blending machine learning with mechanistic knowledge: (1) use machine learning to learn additive residual corrections for the mechanistic model [43, 62, 72, 99, 101, 144, 153, 174]; (2) use the mechanistic model as an input or feature for a machine learning model [14, 89, 121]; (3) use mechanistic knowledge in the form of a differential equation as a final layer in a neural network representation of the solution, or equivalently define the loss function to include approximate satisfaction of the differential equation [25, 134, 135, 154]; and (4) use mechanistic intuition to constrain or inform the machine learning architecture [59, 106]. Many other successful studies have developed specific designs that further hybridize these and other perspectives [45, 60, 70, 173]. In addition, parameter estimation for mechanistic models is a well-studied topic in data assimilation, inverse problems, and other mechanistic modeling communities, but recent approaches that leverage machine learning for this task may create new opportunities for accounting for temporal parameter variations [109] and unknown observation functions [95].

An important distinction should be made between physics-informed *surrogate* modeling and what we refer to as *hybrid* modeling. Surrogate modeling primarily focuses on replacing high-cost, high-fidelity mechanistic model simulations with similarly accurate models that are cheap to evaluate. These efforts have shown great promise by training machine learning models on expensive high-fidelity simulation data, and have been especially successful when the underlying physical (or other domain-specific) mechanistic knowledge and equations are incorporated into the model training [134] and architecture [106]. We use the term hybrid modeling, on the other hand, to indicate when the final learned system involves interaction (and possibly feedback) between mechanism-based and data-driven models [121].

In this work, we focus primarily on hybrid methods that learn residuals to an imperfect mechanistic model. We closely follow the discrete-time hybrid modeling framework developed by [62], while providing new insights from the continuous-time modeling perspective. The benefits of this form of hybrid modeling, which we and many others have observed, are not yet fully understood in a theoretical sense. Intuitively, nominal mechanistic models are most useful when they encode key nonlinearities that

are not readily inferred using general model classes and modest amounts of data. Indeed, classical approximation theorems for fitting polynomials, Fourier modes, and other common function bases directly reflect this relationship by bounding the error with respect to a measure of complexity of the target function (e.g. Lipschitz constants, moduli of continuity, Sobolev norms, etc.) [36, Chapter 7]. Recent work by E et al. [41] provides a priori error bounds for two-layer neural networks and kernel-based regressions, with constants that depend explicitly on the norm of the target function in the model-hypothesis space (a Barron space and a reproducing kernel Hilbert space, resp.). At the same time, problems for which mechanistic models only capture low-complexity trends (e.g. linear) may still be good candidates for hybrid learning (over purely data-driven), as an accurate linear model reduces the parametric burden for the machine-learning task; this effect is likely accentuated in data-starved regimes. Furthermore, even in cases where data-driven models perform satisfactorily, a hybrid approach may improve interpretability, trustworthiness, and controllability without sacrificing performance.

Hybrid models are often cast in Markovian, memory-free settings where the learned dynamical system (or its learned residuals) is solely dependent on the observed states. This approach can be highly effective when measurements of all relevant states are available or when the influence of the unobserved states is adequately described by a function of the observables. This is the perspective employed by Shi et al. [153], where they learn corrections to physical equations of motion for an autonomous vehicle in regions of state space where the physics perform poorly— these residual errors are *driven* by un-modeled turbulence during landing, but can be *predicted* using the observable states of the vehicle (i.e. position, velocity, and acceleration). This is also the perspective taken in applications of high-dimensional multiscale dynamical systems, wherein sub-grid closure models parameterize the effects of expensive fine-scale interactions (e.g. cloud simulations) as functions of the coarse variables [11,18,55,79,117,137,150,158]. The result is a hybrid dynamical system with a physics-based equation defined on the coarse variables with a Markovian correction term that accounts for the effects of the expensive fine scale dynamics.

1.1.3. *Non-Markovian data-driven modeling.* Unobserved and unmodeled processes are often responsible for model errors that cannot be represented in a Markovian fashion within the observed variables alone. This need has driven substantial advances in memory-based modeling. One approach to this is the use of delay embeddings [157]. These methods are inherently tied to discrete time representations of the data and, although very successful in many applied contexts, are of less value when the goal of data-driven learning is to fit continuous-time models; this is a desirable modeling goal in many settings.

An alternative to understanding memory is via the Mori-Zwanzig formalism, which is a fundamental building block in the presentation of memory and hidden variables and may be employed for both discrete-time and continuous-time models. Although initially developed primarily in the context of statistical mechanics, it provides the basis for understanding hidden variables in dynamical systems, and thus underpins many generic computational tools applied in this setting [31, 54, 178]. It has been successfully applied to problems in fluid turbulence [40, 119] and molecular dynamics [66, 90]. Lin and Lu [94] demonstrate connections between Mori-Zwanzig and delay

embedding theory in the context of nonlinear autoregressive models using Koopman operator theory. Indeed, Gilani et al. [49] shows a correspondence between the Mori-Zwanzig representation of the Koopman operator and Taken's delay-embedding flow map. Studies by Ma et al. [105], Wang et al. [166] demonstrate how the Mori-Zwanzig formalism motivates the use of recurrent neural networks (RNNs) [51, 143] as a deep learning approach to non-Markovian closure modeling. Harlim et al. [62] also use the Mori-Zwanzig formalism to deduce a non-Markovian closure model, and evaluate RNN-based approximations of the closure dynamics. Closure modeling using RNNs has recently emerged as a new way to learn memory-based closures [24, 62, 75].

Although the original formulation of Mori-Zwanzig as a general purpose approach to modeling partially observed systems was in continuous-time [31], many practical implementations adopt a discrete-time picture [30, 35, 94]. This causes the learned memory terms to depend on sampling rates, which, in turn, can inhibit flexibility and interpretability.

Recent advances in continuous-time memory-based modeling, however, may be applicable to these non-Markovian hybrid model settings. The theory of continuous-time RNNs (i.e. formulated as differential equations, rather than a recurrence relation) was studied in the 1990s [8, 47], albeit for equations with a specific additive structure. This structure was exploited in a continuous-time reservoir computing (RC) approach by Lu et al. [103] for reconstructing chaotic attractors from data. Comparisons between RNNs and RC (a subclass of RNNs with random parameters fixed in the recurrent state) in discrete-time have yielded mixed conclusions in terms of their relative efficiencies and ability to retain memory [23, 48, 126, 164]. Recent formulations of continuous-time RNNs have departed slightly from the additive structure, and have focused on constraints and architectures that ensure stability and accuracy of the resulting dynamical system [22, 42, 115, 118, 142, 152]. In addition, significant theoretical work has been performed for linear RNNs in continuous-time [91]. Nevertheless, these various methods have not yet been formulated within a hybrid modeling framework, nor has their approximation power been carefully evaluated in that context. A recent step in this direction, however, is the work by Gupta and Lermusiaux [58], which tackles non-Markovian hybrid modeling in continuous-time with neural network-based delay differential equations (DDEs).

1.1.4. *Noisy observations and data assimilation.* For this work we consider settings in which the observations may be both noisy and partial; the observations may be partial either because the system is undersampled in time or because certain variables are not observed at all. We emphasize that ideas from statistics can be used to smooth and/or interpolate data to remove noise and deal with undersampling [33] and to deal with missing data [108]; and ideas from data assimilation [3, 88, 138] can be used to remove noise and to learn about unobserved variables [26, 52, 53]. In some of our experiments we will use noise-free data in continuous-time, to clearly expose issues separate from noise/interpolation; but in other experiments we will use methodologies from data assimilation to enhance our learning [26].

1.1.5. *Applications of data-driven modeling.* In order to deploy hybrid methods in real-world scenarios, we must also be able to cope with noisy, partial observations. Accommodating the learning of model error in this setting, as well as state estimation, is an

active area of research in the data assimilation (DA) community [13, 43, 125]. Learning dynamics from noisy data is generally nontrivial for nonlinear systems—there is a chicken-and-egg problem in which accurate state estimation typically relies on the availability of correct models, and correct models are most readily identified using accurate state estimates. Recent studies have addressed this challenge by attempting to jointly learn the noise and the dynamics. Gottwald and Reich [53] approach this problem from a data assimilation perspective, and employ an Ensemble Kalman Filter (EnKF) to iteratively update the parameters for their dynamics model, then filter the current state using the updated dynamics. A recent follow-up to this work applies the DA-approach to partially-observed systems, and learns a model on a space of discrete-time delay-embeddings [52]. Similar studies were performed by Brajard et al. [16], and applied specifically in model error scenarios [15, 43, 168]. Ayed et al. [4] focus on learning a continuous-time neural network representation of an ODE from partial observations, and learning a separate encoder neural network to map a historical warmup sequence to likely initial conditions in the un-observed space. Kaheman et al. [72] approach this problem from a variational perspective, performing a single optimization over all noise sequences and dynamical parameterizations. Nguyen et al. [114] use an Expectation-Maximization (EM) perspective to compare these variational and ensemble-based approaches, and further study is needed to understand the trade-offs between these styles of optimization. Chen et al. [26] study an EnKF-based optimization scheme that performs joint, rather than EM-based learning, by running gradient descent on an architecture that backpropagates through the data assimilator.

We note that data assimilators are themselves dynamical systems, which can be tuned (using optimization and machine learning) to provide more accurate state updates and more efficient state identification. However, while learning improved DA schemes is sometimes viewed as a strategy for coping with model error [177], we see the optimization of DA and the correction of model errors as two separate problems that should be addressed individually.

When connecting models of dynamical systems to real-world data, it is also essential to recognize that available observables may live in a very different space than the underlying dynamics. Recent studies have shown ways to navigate this using autoencoders and dynamical system models to jointly learn a latent embedding and dynamics in that latent space [21]. Proof of concepts for similar approaches primarily focus on image-based inputs, but have potential for applications in medicine [95] and reduction of nonlinear PDEs [106].

1.2. **Our contributions.** Despite this large and recent body of work in data-driven learning methods and hybrid modeling strategies, many challenges remain for understanding how to best combine mechanistic and machine-learned models; indeed, the answer is highly dependent on the application. Here, we construct a mathematical framework that unifies many of the common approaches for blending mechanistic and machine learning models; having done so we provide strong evidence for the value of hybrid approaches. Our contributions are listed as follows:

(1) We provide an overarching framework for learning model error from (possibly noisy) data in dynamical systems settings, studying both discrete- and continuous-time models, together with both memoryless (Markovian) and memory-dependent representations of the model error. This formulation is

agnostic to choice of mechanistic model and class of machine learning functions.
(2) We study the Markovian learning problem in the context of ergodic continuous-time dynamics, proving bounds on excess risk and generalization error.
(3) We present a simple approximation theoretic approach to learning memory-dependent (non-Markovian) model error in continuous-time, proving a form of universal approximation for two families of memory-dependent model error defined using recurrent neural networks.
(4) We describe numerical experiments which: (a) demonstrate the utility of learning model error in comparison both with pure data-driven learning and with pure (but slightly imperfect) mechanistic modeling; (b) compare the benefits of learning discrete- versus continuous-time models; (c) demonstrate the utility of autodifferentiable data assimilation to learn dynamics from partially observed, noisy data; (d) explain issues arising in memory-dependent model error learning in the (typical) situation where the dimension of the memory variable is unknown.

In Section 2, we address contribution (1) by defining the general settings of interest for dynamical systems in both continuous- and discrete-time. We then link these underlying systems to a machine learning framework in Sections 3 and 4; in the former we formulate the problem in the setting of statistical learning, and in the latter we define concrete optimization problems found from finite parameterizations of the hypothesis class in which the model error is sought. Section 5 is focused on specific choices of architectures, and underpinning theory for machine learning methods with these choices: we analyze linear methods from the perspective of learning theory in the context of ergodic dynamical systems (contribution (2)); and we describe an approximation theorem for continuous-time hybrid recurrent neural networks (contribution (3)). Finally, Section 6 presents our detailed numerical experiments; we apply the methods in Section 5 to exemplar dynamical systems of the forms outlined in Section 2, and highlight our findings (contribution (4)).

2. Dynamical systems setting

In the following, we use the phrase *Markovian model error* to describe model error expressible entirely in terms of the observed variable at the current time, the memoryless situation; *non-Markovian model error* refers to the need to express the model error in terms of the past history of the observed variable.

We present a general framework for modeling a dynamical system with Markovian model error, first in continuous-time (Section 2.1) and then in discrete-time (Section 2.2). We then extend the framework to the setting of non-Markovian model error (Section 2.3), including a parameter ε which enables us to smoothly transition from scale-separated problems (where Markovian closure is likely to be accurate) to problems where the unobserved variables are not scale-separated from those observed (where Markovian closure is likely to fail and memory needs to be accounted for).

It is important to note that the continuous-time formulation necessarily assumes an underlying data-generating process that is continuous in nature. The discrete-time formulation can be viewed as a discretization of an underlying continuous system, but can also represent systems that are truly discrete.

The settings that we present are all intended to represent and classify common situations that arise in modeling and predicting dynamical systems. In particular, we stress two key features. First, we point out that mechanistic models (later referred to as a vector field f_0 or flow map Ψ_0) are often available and may provide predictions with reasonable fidelity. However, these models are often simplifications of the true system, and thus can be improved with data-driven approaches. Nevertheless, they provide a useful starting point that can reduce the complexity and data-hunger of the learning problems. In this context, we study trade-offs between discrete- and continuous-time framings. While we begin with fully-observed contexts in which the dynamics are Markovian with respect to the observed state x, we later note that we may only have access to partial observations x of a larger system (x, y). By restricting our interest to prediction of these observables, we show how a latent dynamical process (e.g. a RNN) has the power to reconstruct the correct dynamics for our observables.

2.1. **Continuous-time.** Consider the following dynamical system

$$\dot{x} = f^\dagger(x), \quad x(0) = x_0, \tag{2.1}$$

and define $\mathsf{X}_s := C([0, s]; \mathbb{R}^{d_x})$. If $f^\dagger \in C^1(\mathbb{R}^{d_x}; \mathbb{R}^{d_x})$ then (2.1) has solution $x(\cdot) \in \mathsf{X}_T$ for any $T < T_{\max} = T_{\max}(x_0) \in \mathbb{R}^+$, the maximal interval of existence.

The primary model error scenario we envisage in this section is one in which the vector field $f^\dagger$ can only be partially known or accessed: we assume that

$$f^\dagger = f_0 + m^\dagger,$$

where f_0 is known to us and $m^\dagger$ is not known. For any $f_0 \in C^1(\mathbb{R}^{d_x}; \mathbb{R}^{d_x})$ (regardless of its fidelity), there exists a function $m^\dagger(x) \in C^1(\mathbb{R}^{d_x}; \mathbb{R}^{d_x})$ such that (2.1) can be rewritten as

$$\dot{x} = f_0(x) + m^\dagger(x). \tag{2.2}$$

However, for this paper, it is useful to think of $m^\dagger$ as being small relative to f_0; the function $m^\dagger$ accounts for *model error.* While the approach in (2.2) is targeted at learning residuals of f_0, $f^\dagger$ can alternatively be reconstructed from f_0 through a different function $m^\dagger(x) \in C^1(\mathbb{R}^{2d_x}; \mathbb{R}^{d_x})$ using the form

$$\dot{x} = m^\dagger(x, f_0(x)). \tag{2.3}$$

Both approaches are defined on spaces that allow perfect reconstruction of $f^\dagger$. However, the first formulation hypothesizes that the missing information is additive; the second formulation provides no such indication. Because the first approach ensures substantial usage of f_0, it has advantages in settings where f_0 is trusted by practitioners and model explainability is important. The second approach will likely see advantages in settings where there is a simple nonadditive form of model error, including coordinate transformations and other (possibly state-dependent) nonlinear warping functions of the nominal physics f_0. Note that the use of f_0 in representing the model error in the augmented-input setting of (2.3) includes the case of not leveraging f_0 at all. It is, hence, potentially more useful than simply adopting an $x-$dependent model error; but it requires learning a more complex function.

The augmented-input method also has connections to model stacking [172] or bagging [17]; this perspective can be useful when there are N model hypotheses:

$$\dot{x} = m^\dagger\big(x, f_0^{(1)}(x), \ \dots \ f_0^{(N)}(x); \theta\big).$$

The residual-based design in (2.2) relates more to model boosting [149].

Our goal is to use machine learning to approximate these corrector functions $m^\dagger$ using our nominal knowledge f_0 and observations of a trajectory $\{x(t)\}_{t=0}^T \in \mathsf{X}_T$, for some $T < T_{\max}(x_0)$, from the true system (2.1). In this work, we consider only the case of learning $m^\dagger(x)$ in equation (2.2). For now the reader may consider $\{x(t)\}_{t=0}^T$ given without noise so that, in principle, $\{\dot{x}(t)\}_{t=0}^T$ is known and may be leveraged. In practice this will not be the case, for example if the data are high-frequency but discrete in time; we address this issue in what follows.

2.2. **Discrete-time.** Consider the following dynamical system

$$x_{k+1} = \Psi^\dagger(x_k) \tag{2.4}$$

and define $\mathsf{X}_K := \ell^\infty\big(\{0, \dots, K\}; \mathbb{R}^{d_x}\big)$. If $\Psi^\dagger \in C(\mathbb{R}^{d_x}; \mathbb{R}^{d_x})$, the map yields solution $\{x_k\}_{k\in\mathbb{Z}^+} \in \mathsf{X}_\infty := \ell^\infty\big(\mathbb{Z}^+; \mathbb{R}^{d_x}\big)$.[1] As in the continuous-time setting, we assume we only have access to an approximate mechanistic model $\Psi_0 \in C(\mathbb{R}^{d_x}; \mathbb{R}^{d_x})$, which can be corrected using an additive residual term $m^\dagger \in C(\mathbb{R}^{d_x}; \mathbb{R}^{d_x})$:

$$x_{k+1} = \Psi_0(x_k) + m^\dagger(x_k), \tag{2.5}$$

or by feeding Ψ_0 as an input to a corrective warping function $m^\dagger \in C(\mathbb{R}^{2d_x}; \mathbb{R}^{d_x})$

$$x_{k+1} = m^\dagger(x_k, \Psi_0(x_k));$$

we focus our experiments on the additive residual framing in (2.5).

Note that the discrete-time formulation can be made compatible with continuous-time data sampled uniformly at rate Δt (i.e. $x(k\Delta t) = x_k$ for $k \in \mathbb{N}$). To see this, let $\Phi^\dagger(x_0, t) := x(t)$ be the solution operator for (2.1) (and Φ_0 defined analogously for f_0). We then have

$$\Psi^\dagger(v) := \Phi^\dagger(v, \Delta t), \tag{2.6a}$$

$$\Psi_0(v) := \Phi_0(v, \Delta t), \tag{2.6b}$$

which can be obtained via numerical integration of $f^\dagger$, f_0, respectively.

2.3. **Partially observed systems (continuous-time).** The framework in Sections 2.1 and 2.2 assumes that the system dynamics are Markovian with respect to observable x. Most of our experiments are performed in the fully-observed Markovian case. However, this assumption rarely holds in real-world systems. Consider a block-on-a-spring experiment conducted in an introductory physics laboratory. In principle, the system is strictly governed by the position and momentum of the block (i.e. f_0), along with a few scalar parameters. However (as most students' error analysis reports will note), the dynamics are also driven by a variety of external factors, like a wobbly table or a poorly greased track. The magnitude, timescale, and structure of the influence of these different factors are rarely known; and yet, they are somehow encoded in the discrepancy between the nominal equations of motion and the (noisy) observations of this multiscale system.

[1] Here we define $\mathbb{Z}^+ = \{0, \dots, \}$, the non-negative integers, including zero.

Thus we also consider the setting in which the dynamics of x is not Markovian. If we consider x to be the observable states of a Markovian system in dimension higher than d_x, then we can write the full system as

$$\dot{x} = f^\dagger(x, y), \quad x(0) = x_0, \tag{2.7a}$$

$$\dot{y} = \frac{1}{\varepsilon} g^\dagger(x, y), \quad y(0) = y_0. \tag{2.7b}$$

Here $f^\dagger \in C^1(\mathbb{R}^{d_x} \times \mathbb{R}^{d_y}; \mathbb{R}^{d_x})$, $g^\dagger \in C^1(\mathbb{R}^{d_x} \times \mathbb{R}^{d_y}; \mathbb{R}^{d_y})$, and $\varepsilon > 0$ is a constant measuring the degree of scale-separation (which is large when ε is small). The system yields solution[2] $x(\cdot) \in \mathsf{X}_T, y(\cdot) \in \mathsf{Y}_T$ for any $T < T_{\max}(x(0), y(0)) \in \mathbb{R}^+$, the maximal interval of existence. We view y as the complicated, unresolved, or unobserved aspects of the true underlying system.

For any $f_0 \in C^1(\mathbb{R}^{d_x}; \mathbb{R}^{d_x})$ (regardless of its fidelity), there exists a function $m^\dagger(x, y) \in C^1(\mathbb{R}^{d_x} \times \mathbb{R}^{d_y}; \mathbb{R}^{d_x})$ such that (2.7) can be rewritten as

$$\dot{x} = f_0(x) + m^\dagger(x, y), \tag{2.8a}$$

$$\dot{y} = \frac{1}{\varepsilon} g^\dagger(x, y). \tag{2.8b}$$

Now observe that, by considering the solution of equation (2.8b) as a function of the history of x, the influence of $y(\cdot) \in \mathsf{Y}_t$ on the solution $x(\cdot) \in \mathsf{X}_t$ can be captured by a parameterized (w.r.t. t) family of operators $m_t^\dagger : \mathsf{X}_t \times \mathbb{R}^{d_y} \times \mathbb{R}^+ \mapsto \mathbb{R}^{d_x}$ on the historical trajectory $\{x(s)\}_{s=0}^t$, unobserved initial condition $y(0)$, and scale-separation parameter ε such that

$$\dot{x}(t) = f_0\big(x(t)\big) + m_t^\dagger\big(\{x(s)\}_{s=0}^t;\ y(0),\ \varepsilon\big). \tag{2.9}$$

Our goal is to use machine learning to find a Markovian model, in which x is part of the state variable, using our nominal knowledge f_0 and observations of a trajectory $\{x(t)\}_{t=0}^T \in \mathsf{X}_T$, for some $T < T_{\max}(x_0, y_0)$, from the true system (2.7); note that $y(\cdot)$ is not observed and nothing is assumed known about the vector field $g^\dagger$ or the parameter ε.

Note that equations (2.7), (2.8) and (2.9) are all equivalent formulations of the same problem and have identical solutions. The third formulation points towards two intrinsic difficulties: the unknown "function" to be learned is in fact defined by a family of operators $m_t^\dagger$ mapping the Banach space of path history into $\mathbb{R}^{d_x}$; secondly the operator is parameterized by $y(0)$ which is unobserved. We will address the first issue by showing that the operators $m_t^\dagger$ can be arbitrarily well-approximated from within a family of differential equations in dimension $\mathbb{R}^{2d_x+d_y}$; the second issue may be addressed by techniques from data assimilation [3,88,138] once this approximating family is learned. We emphasize, however, that we do not investigate the practicality of this approach to learning non-Markovian systems and much remains to be done in this area.

It is also important to note that these non-Markovian operators $m_t^\dagger$ can sometimes be adequately approximated by invoking a Markovian model for x and simply learning function $m^\dagger(\cdot)$ as in Section 2.1. For example, when $\varepsilon \to 0$ and the y dynamics, with

[2] With Y_T defined analogously to X_T.

x fixed, are sufficiently mixing, the averaging principle [9, 122, 161] may be invoked to deduce that

$$\lim_{\varepsilon\to 0} m_t^\dagger\big(\{x(s)\}_{s=0}^t;\ y(0),\varepsilon\big) = m^\dagger(x(t))$$

for some $m^\dagger$ as in Section 2.1. This fact is used in section 3 of [71] to study the learning of closure models for linear Gaussian stochastic differential equations (SDEs).

It is highly advantageous to identify settings where Markovian modeling is sufficient, as it is a simpler learning problem. We find that learning $m_t^\dagger$ is necessary when there is significant memory required to explain the dynamics of x; learning $m^\dagger$ is sufficient when memory effects are minimal. In Section 6, we show that Markovian closures can perform well for certain tasks even when the scale-separation factor ε is not small. In Section 3 we demonstrate how the family of operators $m_t^\dagger$ may be represented through ODEs, appealing to ideas which blend continuous-time RNNs with an assumed known vector field f_0.

2.4. **Partially observed systems (discrete-time).** The discrete-time analog of the previous setting considers a mapping

$$x_{k+1} = \Psi_1^\dagger(x_k, y_k), \tag{2.10a}$$

$$y_{k+1} = \Psi_2^\dagger(x_k, y_k) \tag{2.10b}$$

with $\Psi_1^\dagger \in C(\mathbb{R}^{d_x}\times\mathbb{R}^{d_y};\mathbb{R}^{d_x})$, $\Psi_2^\dagger \in C(\mathbb{R}^{d_x}\times\mathbb{R}^{d_y};\mathbb{R}^{d_y})$, yielding solutions $\{x_k\}_{k\in\mathbb{Z}^+} \in \mathsf{X}_\infty$ and $\{y_k\}_{k\in\mathbb{Z}^+} \in \mathsf{Y}_\infty$. We assume unknown $\Psi_1^\dagger, \Psi_2^\dagger$, but known approximate model Ψ_0 to rewrite (2.10) as

$$x_{k+1} = \Psi_0(x_k) + m^\dagger(x_k, y_k), \tag{2.11a}$$

$$y_{k+1} = \Psi_2^\dagger(x_k, y_k). \tag{2.11b}$$

We can, analogously to (2.9), write a solution in the space of observables as

$$x_{k+1} = \Psi_0\big(x_k\big) + m_k^\dagger\big(\{x_s\}_{s=0}^k,\ y_0\big) \tag{2.12}$$

with $m_k^\dagger : \mathsf{X}_k \times \mathbb{R}^{d_y} \to \mathbb{R}^{d_x}$, a function of the historical trajectory $\{x_s\}_{s=0}^k$ and the unobserved initial condition y_0. If this discrete-time system is computed from the time Δt map for (2.1) then, for $\varepsilon \ll 1$ and when averaging scenarios apply as discussed in Section 2.3, the memoryless model in (2.5) may be used.

3. Statistical learning for ergodic dynamical systems

Here, we present a learning theory framework within which to consider methods for discovering model error from data. We outline the learning theory in a continuous-time Markovian setting (using possibly discretely sampled data), and point to its analogs in discrete-time and non-Markovian settings.

In the discrete-time settings, we assume access to discretely sampled training data $\{x_k = x(k\Delta t)\}_{k=0}^K$, where Δt is a uniform sampling rate and we assume that $K\Delta t = T$. In the continuous-time settings, we assume access to continuous-time training data $\{\dot{x}(t), x(t)\}_{t=0}^T$; Section 6.2.1 discusses the important practical question of estimating $\dot{x}(t), x(t)$ from discrete (but high frequency) data. In either case, consider the problem

of identifying $m \in \mathcal{M}$ (where $\mathcal{M}$ represents the model, or hypothesis, class) that minimizes a loss function quantifying closeness of m to $m^\dagger$. In the Markovian setting we choose a measure μ on $\mathbb{R}^{d_x}$ and define the loss

$$\mathcal{L}_\mu(m, m^\dagger) := \int_{\mathbb{R}^{d_x}} \|m(x) - m^\dagger(x)\|_2^2 d\mu(x).$$

If we assume that, at the true $m^\dagger$, $x(\cdot)$ is ergodic with invariant density μ, then we can exchange time and space averages to see, for infinitely long trajectory $\{x(t)\}_{t\geq 0}$,

$$\begin{aligned}\mathcal{I}_\infty(m) &:= \lim_{T\to\infty} \frac{1}{T}\int_0^T \|m(x(t)) - m^\dagger(x(t))\|_2^2 dt \\ &= \int_{\mathbb{R}^{d_x}} \|m(x) - m^\dagger(x)\|_2^2 d\mu(x) \\ &= \mathcal{L}_\mu(m, m^\dagger).\end{aligned}$$

Since we may only have access to a trajectory dataset of finite length T, it is natural to define

$$\mathcal{I}_T(m) := \frac{1}{T}\int_0^T \|m(x(t)) - m^\dagger(x(t))\|_2^2 dt$$

and note that, by ergodicity,

$$\lim_{T\to\infty} \mathcal{I}_T(m) = \mathcal{L}_\mu(m, m^\dagger).$$

Finally, we can use (2.2) to get

$$\mathcal{I}_T(m) = \frac{1}{T}\int_0^T \|\dot{x}(t) - f_0(x(t)) - m(x(t))\|_2^2 dt. \tag{3.1}$$

This, possibly regularized, is a natural loss function to employ when continuous-time data is available, and should be viewed as approximating $\mathcal{L}_\mu(m, m^\dagger)$. We can use these definitions to frame the problem of learning model error in the language of statistical learning [162].

If we let $\mathcal{M}$ denote the hypothesis class over which we seek to minimize $\mathcal{I}_T(m)$ then we may define

$$m_\infty^* = \operatorname*{argmin}_{m\in\mathcal{M}} \mathcal{L}_\mu(m, m^\dagger) = \operatorname*{argmin}_{m\in\mathcal{M}} \mathcal{I}_\infty(m), \quad m_T^* = \operatorname*{argmin}_{m\in\mathcal{M}} \mathcal{I}_T(m).$$

The *risk* associated with seeking to approximate $m^\dagger$ from the class $\mathcal{M}$ is defined by $\mathcal{L}_\mu(m_\infty^*, m^\dagger)$, noting that this is 0 if $m^\dagger \in \mathcal{M}$. The risk measures the intrinsic error incurred by seeking to learn $m^\dagger ger$ from the restricted class $\mathcal{M}$, which typically does not include $m^\dagger ger$; it is an approximation theoretic concept which encodes the richness of the hypothesis class $\mathcal{M}$. The risk may be decreased by increasing the expressiveness of $\mathcal{M}$. Thus risk is independent of the data employed. *Empirical risk minimization* refers to minimizing $\mathcal{I}_T$ (or a regularized version) rather than $\mathcal{I}_\infty$, and this involves the specific instance of data that is available. To quantify the effect of data volume on learning $m^\dagger$ through empirical risk minimization, it is helpful to introduce the following two concepts. The *excess risk* is defined by

$$R_T := \mathcal{I}_\infty(m_T^*) - \mathcal{I}_\infty(m_\infty^*) \tag{3.2}$$

and represents the additional approximation error incurred by using data defined over a finite time horizon T in the estimate of $m^\dagger$. The *generalization error* is

$$G_T := \mathcal{I}_T(m_T^*) - \mathcal{I}_\infty(m_T^*) \tag{3.3}$$

and represents the discrepancy between training error, which is defined using a finite trajectory, and idealized test error, which is defined using an infinite length trajectory (or, equivalently, the invariant measure μ), when evaluated at the estimate of the function $m^\dagger$ obtained from finite data. We return to study excess risk and generalization error in the context of linear (in terms of parametric-dependence) models for $m^\dagger$, and under ergodicity assumptions on the data generating process, in Section 5.2.

We have introduced a machine learning framework in the continuous-time Markovian setting, but it may be adopted in discrete-time and in non-Markovian settings. In Section 4, we define appropriate objective functions for each of these cases.

Remark 3.1. The developments we describe here for learning in ODEs can be extended to the case of learning SDEs; see [10, 85]. In that setting, consistency in the large T limit is well-understood. It would be interesting to build on the learning theory perspective described here to study statistical consistency for ODEs; the approaches developed in the work by McGoff et al. [107], Su and Mukherjee [156] are potentially useful in this regard.

4. Parameterization of the loss function

In this section, we define explicit optimizations for learning (approximate) model error functions $m^\dagger$ for the Markovian settings, and model error operators $m_t^\dagger$ for the non-Markovian settings; both continuous- and discrete-time formulations are given. We defer discussion of specific approximation architectures to the next section. Here we make a notational transition from optimization over (possibly nonparametric) functions $m \in \mathcal{M}$ to functions parameterized by θ that characterize the class $\mathcal{M}$.

In all the numerical experiments in this paper, we study the use of continuous- and discrete-time approaches to model data generated by a continuous-time process. The setup in this section reflects this setting, in which two key parameters appear: T, the continuous-time horizon for the data; and Δt, the frequency of the data. The latter parameter will always appear in the discrete-time models; but it may also be implicit in continuous-time models which need to infer continuous-time quantities from discretely sampled data. We relate T and Δt by $K\Delta t = T$. We present the general forms of $\mathcal{J}_T(\theta)$ (with optional regularization terms $R(\theta)$). Optimization via derivative-based methodology requires either analytic differentiation of the dynamical system model with respect to parameters or the use of autodifferentiable ODE solvers [142].

4.1. **Continuous-time Markovian learning.**

Here, we approximate the Markovian closure term in (2.2) with a parameterized function $m(x;\, \theta)$. Assuming full knowledge of $\dot{x}(t), x(t)$, we learn the correction term for the flow field by minimizing the following objective function of θ:

$$\mathcal{J}_T(\theta) = \frac{1}{T}\int_0^T \left\| \dot{x}(t) - f_0(x(t)) - m(x(t);\, \theta) \right\|^2 dt + R(\theta). \tag{4.1}$$

Note that $\mathcal{J}_T(\theta) = \mathcal{I}_T\big(m(\,\cdot\,;\,\theta)\big) + R(\theta)$; thus the proposed methodology is a regularization of the empirical risk minimization described in the preceding section.

Notable examples that leverage this framing include: the paper [72], where θ are coefficients for a library of low-order polynomials and $R(\theta)$ is a sparsity-promoting regularization defined by the SINDy framework; the paper [174], where θ are parameters of a deep neural network (DNN) and L_2 regularization is applied to the weights; the paper [153], where θ are DNN parameters and $R(\theta)$ encodes constraints on the Lipschitz constant for m provided by spectral normalization; and the paper [167] which applies this approach to the Lorenz '96 Multiscale system using neural networks with an L_2 regularization on the weights.

4.2. **Discrete-time Markovian learning.** Here, we learn the Markovian correction term in (2.5) by minimizing:

$$\mathcal{J}_T(\theta) = \frac{1}{K}\sum_{k=0}^{K-1} \Big\| x_{k+1} - \Psi_0(x_k) - m(x_k;\,\theta)\Big\|^2 + R(\theta). \tag{4.2}$$

This is the natural discrete-time analog of (4.1) and may be derived analogously, starting from a discrete analog of the loss $\mathcal{L}_\mu(m, m^\dagger)$ where now μ is assumed to be an ergodic measure for (2.4). If a discrete analog of (3.1) is defined, then parameterization of m, and regularization, leads to (4.2). This is the underlying model assumption in the work by Farchi et al. [43].

4.3. **Continuous-time non-Markovian learning.** We can attempt to recreate the dynamics in x for (2.9) by modeling the non-Markovian residual term. A common approach is to augment the dynamics space with a variable $r \in \mathbb{R}^{d_r}$ leading to a model of the form

$$\dot{x} = f_0(x) + f_1(r, x;\,\theta), \tag{4.3a}$$

$$\dot{r} = f_2(r, x;\,\theta). \tag{4.3b}$$

We then seek a d_r large enough, and then parametric models $\{f_j(r, x; \cdot)\}_{j=1}^2$ expressive enough, to ensure that the dynamics in x are reproduced by (4.3). Note that, although the model error in x is non-Markovian, as it depends on the history of x, we are seeking to explain observed x data by an enlarged model, including hidden variables r, in which the dynamics of $[x, r]$ is Markovian.

When learning hidden dynamics from partial observations, we must jointly infer the missing states $r(t)$ and the, typically parameterized, governing dynamics f_1, f_2. Furthermore, when the family of parametric models is not closed with respect to translation of r it will also be desirable to learn r_0; when x is observed noisily, it is similarly important to learn x_0.

To clarify discussions of (4.3) and its training from data, let $u = [x, r]$ and f be the concatenation of the vector fields given by f_0, f_1, f_2 such that

$$\dot{u} = f(u;\,\theta), \tag{4.4}$$

with solution $u(t;\,v, \theta)$ solving (4.4) (and, equivalently, (4.3)) with initial condition v (i.e. $u(0;\,v, \theta) = v$). Consider observation operators H_x, H_r, such that $x = H_x u$, and

$r = H_r u$, and further define noisy observations of x as

$$z(t) = x(t) + \eta(t),$$

where η is i.i.d. observational noise. We now outline three optimization approaches to learning from noisily, partially observed data z.

4.3.1. *Optimization; hard constraint for missing dynamics.* Since (4.3) is deterministic, it may suffice to jointly learn parameters and initial condition $u(0) = u_0$ by minimizing [142]:

$$\mathcal{J}_T(\theta, u_0) = \frac{1}{T}\int_0^T \left\| z(t) - H_x u(t;\, u_0, \theta) \right\|^2 dt + R(\theta). \tag{4.5}$$

A similar approach was applied in [4], but where initial conditions were learnt as outputs of an additional DNN encoder network that maps observation sequences (of fixed length and temporal discretization) to initial conditions.

4.3.2. *Optimization; weak constraint for missing dynamics.* The hard constraint minimization is very sensitive for large T in settings where the dynamics is chaotic. This can be ameliorated, to some extent, by considering the objective function

$$\begin{aligned} \mathcal{J}_T\big(\theta, u(t)\big) = \frac{1}{T}\int_0^T \left\| z(t) - H_x u(t) \right\|^2 dt \\ + \lambda \frac{1}{T}\int_0^T \left\| \dot{u}(t) - f(u(t);\, \theta) \right\|^2 dt. \end{aligned} \tag{4.6}$$

This objective function is employed in [118], where it is motivated by the weak-constraint variational formulation (4DVAR) arising in data assimilation [88].

4.3.3. *Optimization; data assimilation for missing dynamics.* The weak constraint approach may still scale poorly with T large, and still relies on gradient-based optimization to infer hidden states. To avoid these potential issues, we follow the recent work of [26], using filtering-based methods to estimate the hidden state. This implicitly learns initializations and it removes noise from data. It allows computation of gradients of the resulting loss function back through the filtering algorithm to learn model parameters. We define a filtered state

$$\hat{u}_{t,\tau} \coloneqq \hat{u}_t\Big(\tau;\, \hat{v}, \theta_{\mathrm{DYN}}, \theta_{\mathrm{DA}}, \big\{z(t+s)\big\}_{s=0}^{\tau}\Big)$$

as an estimate of $u(t+\tau)|\{z(t+s)\}_{s=0}^{\tau}$ when initialized at $\hat{u}_{t,0} = \hat{v}$.[3] In this formulation, we distinguish θ_{DYN} as parameters for modeling dynamics via (4.3), and θ_{DA} as hyperparameters governing the specifics of a data assimilation scheme. Examples of θ_{DA} are the constant gain matrix K that must be chosen for 3DVAR, or parameters of the inflation and localization methods deployed within Ensemble Kalman Filtering. By parameterizing these choices as θ_{DA}, we can optimize them jointly with model parameters θ_{DYN}. To do this, let $\theta = [\theta_{\mathrm{DYN}}, \theta_{\mathrm{DA}}]$ and minimize

$$\mathcal{J}_T(\theta) = \frac{1}{(T-\tau_1-\tau_2)\tau_2}\int_{t=0}^{T-\tau_1-\tau_2}\int_{s=0}^{\tau_2} \left\| z(t+\tau_1+s) - H_x u(s;\, \hat{u}_{t,\tau_1}, \theta) \right\|^2 ds\, dt. \tag{4.7}$$

[3] In practice we have found that setting $\hat{v} = 0$ works well.

Here, τ_1 denotes the length of assimilation time used to estimate the state which initializes a parameter-fitting over window of duration τ_2; this parameter-fitting leads to the inner-integration over s. This entire picture is then translated through t time units and the objective function is found by integrating over t. Optimizing (4.7) can be understood as a minimization over short-term forecast errors generated from all assimilation windows. The inner integral takes a fixed start time t, applies data assimilation over a window $[t, t+\tau_1]$ to estimate an initial condition $\hat{u}_{t,\tau_1}$, then computes a short-term (τ_2) prediction error resulting from this DA-based initialization. The outer integral sums these errors over all available windows in long trajectory of data of length T.

In our work, we perform filtering using a simple 3DVAR method, whose constant gain can either be chosen as constant or can be learnt from data. When constant, a natural choice is $K \propto H_x^T$, and this approach has a direct equivalence to standard warmup strategies employed in RNN and RC training [121, 164]. The paper [26] suggests minimization of a similar objective, but considers more general observation operators h, restricts the outer integral to non-overlapping windows, and solves the filtering problem with an EnKF with known state-covariance structure.

Remark 4.1. To motivate learning parameters of the data assimilation we make the following observation: for problems in which the model is known (i.e. θ_{DYN} is fixed) we observe successes with the approach of identifying 3DVAR gains that empirically outperform the theoretically derived gains in [87]. Similar is to be expected for parameters defining inflation and localization in the EnKF.

Remark 4.2. Specific functional forms of f_1, f_2 (and their corresponding parameter inference strategies) reduce (4.3) to various approaches. For the continuous-time RNN analysis that we discuss in Section 5 we will start by considering settings in which f_1 and f_2 are approximated from expressive function classes, such as neural networks. We will then specify to models in which f_1 is linear in r and independent of x, while f_2 is a single layer neural network. It is intuitive that the former may be more expressive and allow a smaller d_r than the latter; but the latter connects directly to reservoir computing, a connection we make explicitly in what follows. Our numerical experiments in Section 6 will be performed in both settings: we will train models from the more general setting; and by carefully designed experiments we will shed light on issues arising from over-parameterization, in the sense of choosing to learn a model in dimension higher than that of the true observed-hidden model, working in the setting of linear coupling term f_1, depending only on r.

Remark 4.3. The recent paper [58] proposes an interesting, and more computationally tractable, approach to learning model error in the presence of memory. It proposes to learn a closure operator $m_\tau(\cdot\,;\theta)\colon \mathsf{X}_\tau \to \mathbb{R}^{d_x}$ for a DDE with finite memory τ:

$$\dot{x}(t) = f_0\big(x(t)\big) + m_\tau\big(\{x(t-s)\}_{s=0}^{\tau};\,\theta\big); \tag{4.8}$$

neural networks are used to learn the operator m_τ. Alternatively, Gaussian processes are used to fit a specific class of stochastic delay differential equation (SDDE) (4.8) in [150]. However, although delay-based approaches have seen some practical success, in many applications they present issues for domain interpretability and Markovian ODE or PDE closures are more desirable.

4.4. **Discrete-time non-Markovian learning.** In similar spirit to Section 4.3, we can aim to recreate discrete-time dynamics in x for (2.12) with model

$$x_{k+1} = \Psi_0(x_k) + \Psi_1(r_k, x_k;\, \theta), \tag{4.9a}$$

$$r_{k+1} = \Psi_2(r_k, x_k;\, \theta) \tag{4.9b}$$

and objective function

$$\begin{aligned} \mathcal{J}_T(\theta, r_0) &= \frac{1}{K}\sum_{k=0}^{K-1} \left\| x_{k+1} - \Psi_0(x_k) - \Psi_1(r_k, x_k;\, \theta) \right\|^2 + R(\theta) \\ &\text{s.t.} \quad \{r_k\}_{k=1}^{K-1} \quad \text{solves (4.9b).} \end{aligned} \tag{4.10}$$

Observe that estimation of initial condition r_0 is again crucial, and the data assimilation methods discussed in Section 4.3 can be adapted to this discrete-time setting. The functional forms of Ψ_1, Ψ_2 (and their corresponding parameter inference strategies) reduce (4.9) to various approaches, including recurrent neural networks, latent ODEs, and delay-embedding maps (e.g. to get a delay embedding map, Ψ_2 is a shift operator). Pathak et al. [121] use reservoir computing (a random features analog to RNN, described in the next section) with L_2 regularization to study an approach similar to (4.9), but included $\Psi_0(x_k)$ as a feature in Ψ_1 and Ψ_2 instead of using it as the central model upon which to learn residuals. The data-driven super-parameterization approach in [24] also appears to follow the underlying assumption of (4.9). Harlim et al. [62] evaluate hybrid models of form (4.9) both in settings where delay embedding closures are employed and where RNN-based approximations via LSTMs are employed.

5. Underpinning theory

In this section we identify specific hypothesis classes $\mathcal{M}$. We do this using random feature maps [131] in the Markovian settings (Section 5.1), and using recurrent neural networks in the memory-dependent setting. We then discuss these problems from a theoretical standpoint. In Section 5.2 we study excess risk and generalization error in the context of linear models (a setting which includes the random features model as a special case). And we conclude by discussing the use of RNNs [51, Chapter 10] for the non-Markovian settings (discrete- and continuous-time) in Section 5.3; we present an approximation theorem for continuous-time hybrid RNN models. Throughout this section, the specific use of random feature maps and of recurrent neural networks is for illustration only; other models could, of course, be used.

5.1. **Markovian modeling with random feature maps.** In principle, any hypothesis class can be used to learn $m^\dagger$. However, we focus on models that are easily trained on large-scale complex systems and yet have proven approximation power for functions between finite-dimensional Euclidean spaces. For the Markovian modeling case, we use random feature maps; like traditional neural networks, they possess arbitrary approximation power [132, 133], but further benefit from a quadratic minimization problem in the training phase, as do kernel or Gaussian process methods. In our case studies, we found random feature models sufficiently expressive, we found optimization easily implementable, and we found the learned models generalized well. Moreover, their linearity with respect to unknown parameters enables a straightforward analysis of excess risk and generalization error in Section 5.2. Details on the derivation and

specific design choices for our random feature modeling approach can be found in Section 8.4, where we explain how we sample D random feature functions $\varphi : \mathbb{R}^{d_x} \to \mathbb{R}$ and stack them to form a vector-valued feature map $\phi : \mathbb{R}^{d_x} \to \mathbb{R}^D$. Given this random function ϕ, we define the hypothesis class

$$\mathcal{M} = \{m : \mathbb{R}^{d_x} \to \mathbb{R}^{d_x} \mid \exists\, C \in \mathbb{R}^{d_x \times D} : m(x) = C\phi(x)\}. \tag{5.1}$$

5.1.1. *Continuous-time.* In the continuous-time framing, our Markovian closure model uses hypothesis class (5.1) and thus takes the form

$$\dot{x} = f_0(x) + C\phi(x(t)).$$

We rewrite (4.1) for this particular case with an L_2 regularization parameter $\lambda \in \mathbb{R}^+$:

$$\mathcal{J}_T(C) = \frac{1}{2T}\int_0^T \left\| \dot{x}(t) - f_0(x(t)) - C\phi\big(x(t)\big) \right\|^2 dt + \frac{\lambda}{2}\|C\|^2. \tag{5.2}$$

We employ the notation $A \otimes B \coloneqq AB^T$ for the outer-product between matrices $A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{l\times n}$, and the following notation for time-average

$$\overline{A}_T \coloneqq \frac{1}{T}\int_0^T A(t)dt$$

of $A \in L^1([0,T], \mathbb{R}^{m\times n})$. The objective function in (5.2) is quadratic and convex in C and thus is globally minimized for the unique C^* which makes the derivative of $\mathcal{J}_T$ zero. Consequently, the minimizer C^* satisfies the following linear equation (derived in Section 8.5):

$$(Z + \lambda I)(C^*)^T = Y. \tag{5.3}$$

Here, $I \in \mathbb{R}^{D\times D}$ is the identity and

$$\begin{aligned} Z &= \overline{[\phi \otimes \phi]}_T \in \mathbb{R}^{D\times D}, \\ Y &= \overline{[\phi \otimes m^\dagger]}_T \in \mathbb{R}^{D\times d_x}. \end{aligned} \tag{5.4}$$

Of course $m^\dagger$ is not known, but $m^\dagger(t) = \dot{x}(t) - f_0(x(t))$ can be computed from data.

To summarize, the algorithm proceeds as follows: (1) create a realization of random feature vector ϕ; (2) compute the integrals in (5.4) to obtain Z, Y; and (3) solve the linear matrix equation (5.3) for C^*. Together this leads to our approximation $m^\dagger(x) \approx m_T^*(x;\, \theta) \coloneqq C^*\phi(x)$.

5.1.2. *Discrete-time.* In discrete-time, our Markovian closure model is

$$x_{k+1} = \Psi_0(x_k) + C\phi(x_k),$$

and is learnt by minimizing

$$\mathcal{J}_T(\theta) = \frac{1}{K}\sum_{k=0}^{K-1} \left\| x_{k+1} - \Psi_0(x_k) - C\phi\big(x(t)\big) \right\|^2 + \frac{\lambda}{2}\|C\|^2. \tag{5.5}$$

The objective function in (5.5) is quadratic in C and thus globally minimized at C^*. As in Section 5.1.1, we can compute Z, Y and solve a linear system for C^* to approximate $m^\dagger(x) \approx m_T^*(x;\, \theta) \coloneqq C^*\phi(x)$. This formulation closely mirrors the fully data-driven linear regression approach in [53].

5.2. **Learning theory for Markovian models with linear hypothesis class.** In this subsection we provide estimates of the excess risk and generalization error in the context of learning $m^\dagger$ in (2.2) from a trajectory over time horizon T. We study ergodic continuous-time models in the setting of Section 4.1. To this end we consider the very general linear hypothesis class given by

$$\mathcal{M} = \{m : \mathbb{R}^{d_x} \to \mathbb{R}^{d_x} \mid \exists\, \theta \in \mathbb{R}^p \,:\, m(x) = \sum_{\ell=1}^{p} \theta_\ell f_\ell(x)\}; \tag{5.6}$$

we note that if the $\{f_\ell\}$ are i.i.d. draws of function ϕ in the case $D = d_x$ then this too reduces to a random features model, but that our analysis in the context of statistical learning does not rely on the random features structure. In fact our analysis can be used to provide learning theory for other linear settings, where $\{f_\ell\}$ represents a dictionary of hypothesized features whose coefficients are to be learnt from data. Nonetheless, universal approximation for random features [131] provides an important example of an approximation class for which the loss function $\mathcal{I}_\infty$ may be made arbitrarily small by choice of p large enough and appropriate choice of parameters, and the reader may find it useful to focus on this case. We also note that the theory we present in this subsection is readily generalized to working with hypothesis class (5.1).

We make the following ergodicity assumption about the data generation process:

Assumption A1. Equation (2.2) possesses a compact attractor $\mathcal{A}$ supporting invariant measure μ. Furthermore the dynamical system on $\mathcal{A}$ is ergodic with respect to μ and satisfies a central limit theorem of the following form: for all Hölder continuous $\varphi : \mathbb{R}^{d_x} \mapsto \mathbb{R}$, there is $\sigma^2 = \sigma^2(\varphi)$ such that

$$\sqrt{T}\left(\frac{1}{T}\int_0^T \varphi\big(x(t)\big)dt - \int_{\mathbb{R}^{d_x}} \varphi\big(x\big)\mu(dx)\right) \Rightarrow N(0,\sigma^2), \tag{5.7}$$

where $\Rightarrow$ denotes convergence in distribution with respect to $x(0) \sim \mu$. Furthermore a law of the iterated logarithm holds: almost surely with respect to $x(0) \sim \mu$,

$$\operatorname{limsup}_{T\to\infty}\Big(\frac{T}{\log\log T}\Big)^{\frac{1}{2}}\left(\frac{1}{T}\int_0^T \varphi\big(x(t)\big)dt - \int_{\mathbb{R}^{d_x}} \varphi\big(x\big)\mu(dx)\right) = \sigma. \tag{5.8}$$

Remark 5.1. Note that in both (5.7) and (5.8) $\varphi(\cdot)$ is only evaluated on (compact) $\mathcal{A}$ obviating the need for any boundedness assumptions on $\varphi(\cdot)$. In the work of Melbourne and co-workers, Assumption A1 is proven to hold for a class of differential equations, including the Lorenz '63 model at, and in a neighbourhood of, the classical parameter values: in [68] the central limit theorem is established; and in [7] the continuity of σ in φ is proven. While it is in general very difficult to prove such results for any given chaotic dynamical system, there is strong empirical evidence for such results in many chaotic dynamical systems that arise in practice. This combination of theory and empirical evidence justifies studying the learning of model error under Assumption A1. Tran and Ward [159] were the first to make use of the theory of Melbourne and coworkers to study learning of chaotic differential equations from time-series.

Given m from hypothesis class $\mathcal{M}$ defined by (5.6) we define

$$\theta^*_\infty = \operatorname{argmin}_{\theta\in\mathbb{R}^p} \mathcal{I}_\infty\big(m(\cdot\,;\theta)\big) = \operatorname{argmin}_{\theta\in\mathbb{R}^p} \mathcal{L}_\mu\big(m(\cdot\,;\theta)\big) \tag{5.9}$$

and

$$\theta_T^* = \operatorname{argmin}_{\theta\in\mathbb{R}^p} \mathcal{I}_T\big(m(\cdot\,;\theta)\big). \tag{5.10}$$

(Regularization is not needed in this setting because the data is plentiful—a continuous-time trajectory—and the number of parameters is finite.) Then $\theta_\infty^*, \theta_T^*$ solve the linear systems

$$A_\infty \theta_\infty^* = b_\infty, \quad A_T \theta_T^* = b_T,$$

where

$$(A_\infty)_{ij} = \int_{\mathbb{R}^{d_x}} \big\langle f_i(x), f_j(x)\big\rangle \mu(dx), \qquad (b_\infty)_j = \int_{\mathbb{R}^{d_x}} \big\langle m^\dagger(x), f_j(x)\big\rangle \mu(dx),$$

$$(A_T)_{ij} = \frac{1}{T}\int_0^T \big\langle f_i\big(x(t)\big), f_j\big(x(t)\big)\big\rangle\, dt, \qquad (b_T)_j = \frac{1}{T}\int_0^T \big\langle m^\dagger\big(x(t)\big), f_j\big(x(t)\big)\big\rangle\, dt.$$

These facts can be derived analogously to the derivation in Section 8.5. Given θ_∞^* and θ_T^* we also define

$$m_\infty^* = m(\cdot\,;\theta_\infty^*), \;\; m_T^* = m(\cdot\,;\theta_T^*).$$

Recall that it is assumed that $f^\dagger, f_0$, and $m^\dagger$ are C^1. We make Assumption A2 regarding the vector fields defining hypothesis class $\mathcal{M}$.

Assumption A2. The functions $\{f_\ell\}_{\ell=0}^p$ appearing in definition (5.6) of the hypothesis class $\mathcal{M}$ are Hölder continuous on $\mathbb{R}^{d_x}$. In addition, the matrix A_∞ is invertible.

Theorem 5.2. *Let Assumptions* A1 *and* A2 *hold. Then the scaled excess risk* $\sqrt{T}R_T$ *in* (3.2) (*resp. scaled generalization error* $\sqrt{T}|G_T|$ *in* (3.3)) *is bounded above by* $\|\mathcal{E}_R\|$ (*resp.* $\|\mathcal{E}_G\|$), *where random variable* $\mathcal{E}_R \in \mathbb{R}^p$ (*resp.* $\mathcal{E}_G \in \mathbb{R}^{p+1}$) *converges in distribution to* $N(0,\Sigma_R)$ (*resp.* $N(0,\Sigma_G)$) *w.r.t.* $x(0)\sim\mu$ *as* $T\to\infty$. *Furthermore, there is constant* $C>0$ *such that, almost surely w.r.t.* $x(0)\sim\mu$,

$$\operatorname{limsup}_{T\to\infty}\Big(\frac{T}{\log\log T}\Big)^{\frac{1}{2}}\big(R_T + |G_T|\big) \le C.$$

The proof is provided in Section 8.1.

Remark 5.3. The convergence in distribution shows that, with high probability with respect to initial data, the excess risk and the generalization error are bounded above by terms of size $1/\sqrt{T}$. This can be improved to give an almost sure result, at the cost of the factor of $\sqrt{\log\log T}$. Theorem 5.2 shows that (ignoring log factors and acknowledging the probabilistic nature of any such statements) trajectories of length $\mathcal{O}(\epsilon^{-2})$ are required to produce bounds on the excess risk and generalization error of size $\mathcal{O}(\epsilon)$.

The bounds on excess risk and generalization error also show that empirical risk minimization (of $\mathcal{I}_T$) approaches the theoretically analyzable concept of risk minimization (of $\mathcal{I}_\infty$) over hypothesis class (5.6). The sum of the excess risk R_T and the generalization error G_T gives

$$E_T := \mathcal{I}_T(m_T^*) - \mathcal{I}_\infty(m_\infty^*).$$

We note that $\mathcal{I}_T(m_T^*)$ is computable, once the approximate solution m_T^* has been identified; thus, when combined with an estimate for E_T, this leads to an estimate for the risk associated with the hypothesis class used.

If the approximating space $\mathcal{M}$ is rich enough, then approximation theory may be combined with Theorem 5.2 to estimate the trajectory error resulting from the learned dynamical system. Such an approach is pursued in Proposition 3 of [175] for SDEs. Furthermore, in that setting, knowledge of rate of mixing/decay of correlations for SDEs may be used to quantify constants appearing in the error bounds. It would be interesting to pursue such an analysis for chaotic ODEs with known mixing rates/decay of correlations. Such results on mixing are less well-developed, however, for chaotic ODEs; see discussion of this point in [68], and the recent work [7].

Work by Zhang et al. [176] demonstrates that error bounds on learned model error terms can be extended to bound error on reproduction of invariant statistics for ergodic SDEs. Moreover, E et al. [41] provide a direction for proving similar bounds on model error learning using nonlinear function classes (e.g. two-layer neural networks).

Finally we remark on the dependence of the risk and generalization error bounds on the size of the model error. It is intuitive that the amount of data required to learn model error should decrease as the size of the model error decreases. This is demonstrated numerically in Section 6.2.3 (cf. Figure 2(a) and 2(b)). Here we comment that Theorem 5.2 also exhibits this feature: examination of the proof in Appendix 8.1 shows that all upper bounds on terms appearing in the excess and generalization error are proportional to $m^\dagger$ itself or to m^*_∞, its approximation given an infinite amount of data; note that $m^*_\infty = m^\dagger$ if the hypothesis class contains the truth.

5.3. **Non-Markovian modeling with recurrent neural networks.** Recurrent Neural Networks (RNNs) are one of the *de facto* tools for modeling systems with memory. Here, we show straightforward residual implementations of RNNs for continuous- and discrete-time, with the goal of modeling non-Markovian model error.

5.3.1. *General case.* Equation (4.3b), and its coupling to (4.3a), constitutes a very general way to account for memory-dependent model error in the dynamics of x. In fact, for f_1, f_2 sufficiently expressive (e.g. random feature functions, neural networks, polynomials), and $d_r \geq d_y$, solutions to (4.3) can approximate solutions to (2.8) arbitrarily well. We make this type of universal approximation theorem concrete in Theorems 5.4 and 5.6. We start by proving Theorem 5.4, which rests on Assumptions A3–A6:

Assumption A3. Functions $f^\dagger, g^\dagger, f_0, f_1, f_2$ are all globally Lipschitz.

Note that this implies that $m^\dagger$ is also globally Lipschitz.

Assumption A4. Fix $T > 0$. There exist $\rho_0 \in \mathbb{R}, \rho_T \in \mathbb{R}$ such that, for equation (2.8), $\big(x(0), y(0)\big) \in B(0, \rho_0)$ implies that $\big(x(t), y(t)\big) \in B(0, \rho_T)\ \forall\ t \in [0, T]$.

Assumption A5. The hidden state in (4.3), $r \in \mathbb{R}^{d_r}$, has the same dimension as the true hidden state y in (2.8); that is $d_r = d_y$.

Assumption A6. Let functions $f_1(\cdot\,;\,\theta) \in C^1(\mathbb{R}^{d_x} \times \mathbb{R}^{d_y};\, \mathbb{R}^{d_x})$ and $f_2(\cdot\,;\,\theta) \in C^1(\mathbb{R}^{d_x} \times \mathbb{R}^{d_y};\, \mathbb{R}^{d_y})$ be parameterized[4] by $n \in \mathbb{N}$ and $\theta \in \mathbb{R}^n$. Then, for any $\delta > 0$, there exists $n > 0$ and $\theta \in \mathbb{R}^n$ such that

$$\sup_{x,y \in B(0,\rho_T)} \|f^\dagger(x, y) - f_1(x, y;\, \theta)\| \leq \delta$$

[4]Here we define $\mathbb{N} = \{1, 2, \dots, \}$, the strictly positive integers.

and

$$\sup_{x,y\in B(0,\rho_T)} \|g^\dagger(x,y) - f_2(x,y;\,\theta)\| \le \delta.$$

Note that Assumption A6 can be satisfied by any parametric function class possessing a universal approximation property for maps between finite-dimensional Euclidean spaces, such as neural networks, polynomials and random feature methods. Theorem 5.4 transfers this universal approximation property for maps between Euclidean spaces to a universal approximation property for representation of model error with memory; this is a form of infinite dimensional approximation since, via its own dynamics, the memory variable r maps the past history of x into the model error correction term in the dynamics for x.

Theorem 5.4. *Let Assumptions* A3-A6 *hold. Fix any $T > 0$ and $\rho_0 > 0$, let $x(\cdot), y(\cdot)$ denote the solution of* (2.8) *with $\varepsilon = 1$ and let $x_\delta(\cdot), r_\delta(\cdot)$ denote the solution of* (4.3) *with parameters $\theta \in \mathbb{R}^n$. Then, for any $\delta > 0$ and any $T > 0$, there is a parameter dimension $n = n_\delta \in \mathbb{N}$ and parameterization $\theta = \theta_\delta \in \mathbb{R}^{n_\delta}$ with the property that, for any initial condition $\big(x(0), y(0)\big) \in B(0,\rho_0)$ for* (2.8)*, there is initial condition $(x_\delta(0), r_\delta(0)) \in \mathbb{R}^{d_x+d_y}$ for* (4.3)*, such that*

$$\sup_{t\in[0,T]} \|x - x_\delta\| \le \delta.$$

The proof is provided in Section 8.2; it is a direct consequence of the approximation power of f_1, f_2 and the Gronwall Lemma.

Remark 5.5. Note that this existence theorem also holds for $d_r > d_y$ by freezing the dynamics in the excess dimensions and initializing it at, for example, 0. However it is possible for augmentations with $d_r > d_y$ to introduce numerical instability when imperfectly initialized in the excess dimensions, despite their provable correctness when perfectly initialized (see Section 6.4). Nevertheless, we did not encounter such issues when training the general model class on the examples considered in this paper – see Section 6.3).

5.3.2. *Linear coupling.* We now study a particular form RNN in which the coupling term f_1 appearing in (4.3) is linear and depends only on the hidden variable:

$$\dot{x} = f_0(x) + Cr, \tag{5.11a}$$

$$\dot{r} = \sigma(Ar + Bx + c). \tag{5.11b}$$

Here σ is an activation function. The specific linear coupling form is of particular interest because of the connection we make (see Remark 5.9) to reservoir computing. The goal is to choose A, B, C, c so that output $\{x(t)\}_{t\ge0}$ matches output of (2.8), without observation of $\{y(t)\}_{t\ge0}$ or knowledge of $m^\dagger$ and $g^\dagger$. As in the general case from the preceding subsection, inherent in choosing these matrices A, B, C and vector c is a choice of embedding dimension for variable r which will typically be larger than dimension of y itself. The idea is to create a recurrent state r of sufficiently large dimension d_r whose evolution equation takes x as input and, after a final linear transformation, approximates the missing dynamics $m^\dagger(x,y)$.

There is existing approximation theory for discrete-time RNNs [148] showing that a discrete-time analog of our linear coupling setup can be used to approximate discrete-time systems arbitrarily well; see also Theorem 3 of [62]. There is also a general approximation theorem using continuous-time RNNs proved in [47], but it does not apply to the linear-coupling setting. We thus extend the work in these three papers to the context of residual-based learning as in (5.11). We state the theorem after making three assumptions upon which it rests:

Assumption A7. Functions $f^\dagger, g^\dagger, f_0$ are all globally Lipschitz.

Note that this implies that $m^\dagger$ is also globally Lipschitz.

Assumption A8. Let $\sigma_0 \in C^1(\mathbb{R};\mathbb{R})$ be bounded and monotonic, with bounded first derivative. Then $\sigma(u)$ defined by $\sigma(u)_i = \sigma_0(u_i)$ satisfies $\sigma \in C^1(\mathbb{R}^p;\mathbb{R}^p)$.

Assumption A9. Fix $T > 0$. There exist $\rho_0 \in \mathbb{R}, \rho_T \in \mathbb{R}$ such that, for equation (2.8), $\big(x(0), y(0)\big) \in B(0,\rho_0)$ implies that $\big(x(t), y(t)\big) \in B(0,\rho_T)\ \forall\ t \in [0,T]$.

Theorem 5.6. *Let Assumptions* A7-A9 *hold. Fix any $T > 0$ and $\rho_0 > 0$, let $x(\cdot), y(\cdot)$ denote the solution of* (2.8) *with $\varepsilon = 1$ and let $x_\delta(\cdot), r_\delta(\cdot)$ denote the solution of* (5.11) *with parameters $\theta \in \mathbb{R}^n$. Then, for any $\delta > 0$ and any $T > 0$, there is embedding dimension $d_r \in \mathbb{N}$, parameter dimension $n = n_\delta \in \mathbb{N}$ and parameterization $\theta = \theta_\delta = \{A_\delta,\ B_\delta,\ C_\delta, c_\delta\}$ with the property that, for any initial condition $\big(x(0), y(0)\big) \in B(0,\rho_0)$ for* (2.8)*, there is initial condition $(x_\delta(0), r_\delta(0)) \in \mathbb{R}^{d_x+d_r}$ for* (5.11)*, such that*

$$\sup_{t\in[0,T]} \|x - x_\delta\| \le \delta.$$

The complete proof is provided in Section 8.3; here we describe its basic structure. Define $m(t) := m^\dagger\big(x(t), y(t)\big)$ and, with the aim of finding a differential equation for $m(t)$, recall (2.8) with $\varepsilon = 1$ and define the vector field

$$h^\dagger(x,y) := \nabla_x m^\dagger(x,y)[f_0(x) + m^\dagger(x,y)] + \nabla_y m^\dagger(x,y) g^\dagger(x,y). \tag{5.12}$$

Since $\dot m(t)$ is the time derivative of $m^\dagger\big(x(t), y(t)\big)$, when (x,y) solve (2.8) we have

$$\dot m = h^\dagger(x,y).$$

Motivated by these observations, we now introduce a new system of autonomous ODEs for the variables $(x,y,m) \in \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \times \mathbb{R}^{d_x}$:

$$\dot x = f_0(x) + m, \tag{5.13a}$$

$$\dot y = g^\dagger(x,y), \tag{5.13b}$$

$$\dot m = h^\dagger(x,y). \tag{5.13c}$$

To avoid a proliferation of symbols we use the same letters for (x,y) solving equation (5.13) as for (x,y) solving equation (2.8). We now show $m = m^\dagger(x,y)$ is an invariant manifold for (5.13); clearly, on this manifold, the dynamics of (x,y) governed by (5.13) reduces to the dynamics of (x,y) governed by (2.8). Thus $m(t)$ must be initialized at $m^\dagger\big(x(0), y(0)\big)$ to ensure equivalence between the solution of (5.13) and (2.8).

The desired invariance of manifold $m = m^\dagger(x,y)$ under the dynamics (5.13) follows from the identity

$$\frac{d}{dt}\Big(m - m^\dagger(x,y)\Big) = -\nabla_x m^\dagger(x,y)\big(m - m^\dagger(x,y)\big). \tag{5.14}$$

The identity is derived by noting that, recalling (5.12) for the definition of $h^\dagger$, and then using (5.13),

$$\begin{aligned}\frac{d}{dt}m &= h^\dagger(x,y)\\ &= \nabla_x m^\dagger(x,y)[f_0(x)+m^\dagger(x,y)] + \nabla_y m^\dagger(x,y) g^\dagger(x,y)\\ &= \nabla_x m^\dagger(x,y)[f_0(x)+m)] + \nabla_y m^\dagger(x,y) g^\dagger(x,y)\\ &\qquad - \nabla_x m^\dagger(x,y)\big(m - m^\dagger(x,y)\big)\\ &= \frac{d}{dt}m^\dagger(x,y) - \nabla_x m^\dagger(x,y)\big(m - m^\dagger(x,y)\big).\end{aligned}$$

We emphasize this calculation is performed under the dynamics defined by (5.13).

The proof of the RNN approximation property proceeds by approximating vector fields $g^\dagger(x,y), h^\dagger(x,y)$ by neural networks and introducing linear transformations of y and m to rewrite the approximate version of system (5.13) in the form (5.11). The effect of the approximation of the vector fields on the true solution is then propagated through the system and its effect controlled via a straightforward Gronwall argument.

Remark 5.7. The details of training continuous-time RNNs to ensure accuracy and long-time stability are a subject of current research [22, 26, 42, 118] and in this paper we confine the training of RNNs to an example in the general setting, and not the case of linear coupling. Discrete-time RNN training, on the other hand, is much more mature, and has produced satisfactory accuracy and stability for settings with uniform sample rates that are consistent across train and testing scenarios [62]. The form with linear coupling is widely studied in discrete time models. Furthermore, sophisticated variants on RNNs, such as Long-Short Term Memory (LSTM) RNNs [67] and Gated Recurrent Units (GRU) [28], are often more effective, although similar in nature RNNs. However, the potential formulation, implementation and advantages of these variants in the continuous-time setting [115] are not yet understood. We refer readers to [51] for background on discrete RNN implementations and backpropagation through time (BPTT). For implementations of continuous-time RNNs, it is common to leverage the success of the automatic BPTT code written in PyTorch and Tensorflow by discretizing (5.11) with an ODE solver that is compatible with these autodifferentiation tools (e.g. `torchdiffeq` by [142], `NbedDyn` by [118], and `AD-ENKF` by [26]). This compatibility can also be achieved by use of explicit Runge-Kutta schemes [128]. Note that the discretization of (5.11) can (and perhaps should) be much finer than the data sampling rate Δt, but that this requires reliable estimation of $x(t), \dot{x}(t)$ from discrete data.

Remark 5.8. The need for data assimilation [3, 88, 138] to learn the initialization of recurrent neural networks may be understood as follows. Since $m^\dagger$ is not known and y is not observed (and in particular $y(0)$ is not known) the desired initialization for (5.13), and thus also for approximations of this equation in which $g^\dagger$ and $h^\dagger$ are replaced by neural networks, is not known. Hence, if an RNN is trained on a particular trajectory, the initial condition that is required for accurate approximation of (2.8) from an unseen initial condition is not known. Furthermore the invariant manifold $m = m^\dagger(x,y)$ may be unstable under numerical approximation. However if some observations of the trajectory starting at the new initial condition are used, then data assimilation techniques can potentially learn the initialization for the RNN and also stabilize the invariant manifold. Ad hoc initialization methods are common practice [6, 29, 64, 120], and rely on

forcing the learned RNN with a short sequence of observed data to synchronize the hidden state. The success of these approaches likely relies on RNNs' abilities to emulate data assimilators [63]; however, a more careful treatment of the initialization problem may enable substantial advances.

Remark 5.9. Reservoir computing (RC) is a variant on RNNs which has the advantage of leading to a quadratic optimization problem [56, 69, 104]. Within the context of the continuous-time RNN (5.11) they correspond to randomizing (A, B, c) in (5.11b) and then choosing only parameter C to fit the data. To be concrete, this leads to

$$r(t) = \mathcal{G}_t\big(\{x(s)\}_{s=0}^t;\, r(0), A, B, c\big);$$

here $\mathcal{G}_t$ may be viewed as a random function of the path-history of x up to time t and of the initial condition for r. Then C is determined by minimizing the quadratic function

$$\mathcal{J}_T(C) = \frac{1}{2T}\int_0^T \|\dot{x}(t) - f_0(x(t)) - Cr(t)\|^2\, dt + \frac{\lambda}{2}\|C\|^2.$$

This may be viewed as a random feature approach on the Banach space X_T; the use of random features for learning of mappings between Banach spaces is studied by Nelsen and Stuart [113], and connections between random features and reservoir computing were introduced by Dong et al. [37]. In the specific setting described here, care will be needed in choosing probability measure on (A, B, c) to ensure a well-behaved map $\mathcal{G}_t$; furthermore data assimilation ideas [3, 88, 138] will be needed to learn an appropriate $r(0)$ in the prediction phase, as discussed in Remark 5.8 for RNNs.

6. Numerical experiments

In this section, we present numerical experiments intended to test different hypotheses about the utility of hybrid mechanistic and data-driven modeling. We summarize our findings in Section 6.1. We define the overarching experimental setup in Section 6.2.1, then introduce our criteria for evaluating model performance in Section 6.2.2. In the Lorenz '63 (L63) experiments (Section 6.2.3), we investigate how a simple Markovian random features model error term can be recovered using discrete and continuous-time methods, and how those methods scale with the magnitude of error, data sampling rate, availability of training data, and number of learned parameters. In the Lorenz '96 Multiscale (L96MS) experiments (Section 6.2.4), we take this a step further by learning a Markovian random features closure term for a scale-separated system, as well as systems with less scale-separation. As expected, we find that the Markovian closure approach is highly accurate for a scale-separated regime. We also see that the Markovian closure has merit even in cases with reduced scale-separation. However, this situation would clearly benefit from learning a closure term with memory, a topic we turn to in Section 6.3, where we demonstrate that non-Markovian closure models can be learnt from noisy, partially observed data; for low-dimensional cases (e.g. L63), our method of training converges to return a good model with high short-term accuracy and long-term statistical validity. For higher-dimensional cases (e.g. L96MS), we find the method to hold promise, but further research is required in this general area. In Section 6.4, we demonstrate why non-Markovian closures must be carefully initialized and/or controlled (e.g. via data assimilation) in order to ensure their long-term stability and short-term accuracy.

6.1. **Summary of findings from numerical experiments.**

(1) We find that hybrid modeling has better predictive performance than purely data-driven methods in a wide range of settings (see Figure 2(a) and 2(b) of Section 6.2.3): this includes scenarios where f_0 is highly accurate (but imperfect) and scenarios where f_0 is highly inaccurate (but nevertheless faithfully encodes much of the true structure for $f^\dagger$).
(2) We find that hybrid modeling is more data-efficient than purely data-driven approaches (Figure 3 of Section 6.2.3).
(3) We find that hybrid modeling is more parameter-efficient than purely data-driven approaches (Figure 4 of Section 6.2.3).
(4) Purely data-driven discrete-time modeling can suffer from instabilities in the small timestep limit $\Delta t \ll 1$; hybrid discrete-time approaches can alleviate this issue when they are built from an integrator Ψ_0, as this will necessarily encode the correct parametric dependence on $\Delta t \ll 1$ (Figure 5 of Section 6.2.3).
(5) In order to leverage standard supervised regression techniques, continuous-time methods require good estimates of derivatives $\dot{x}(t)$ from the data. Figure 5 of Section 6.2.3 quantifies this estimation as a function of data sample rate.
(6) Non-Markovian model error can be captured by Markovian terms in scale-separated cases. Section 6.2.4 demonstrates this quantitatively in Figure 6, and qualitatively in Figure 7. Beyond the scale-separation limit, Markovian terms will fail for trajectory forecasting. However, Markovian terms may still reproduce invariant statistics in dissipative systems (for example, in cases with short memory-length). Section 6.2.4 demonstrates this quantitatively in Figure 6; Figure 7 offers intuition for these findings.
(7) Non-Markovian description of model error is needed to accurately represent problems where the hidden dynamics is not scale-separated from the observed dynamics. Section 6.3 shows how partial and noisy observations can be exploited by augmented ODE models of form (4.3) when the noise and hidden dynamics are learnt implicitly by autodifferentiable data assimilation. We observe high-quality reconstruction of the L63 system along its first-component when choosing a correct (Figure 8) or overly enlarged (Figure 9) hidden dimension. We also observe promising reconstruction of the L96MS system in its slow components (Figure 10); however, long-time solutions to the learnt model exhibited instabilities inconsistent with the true system.
(8) Non-Markovian models must be carefully initialized, and indeed data assimilation is needed, in order to ensure accuracy (Section 6.4) of invariant statistics (Figure 12), long-term stability (Figure 13), and accurate short-term predictions (Figure 14). We explain observed phenomena in terms of the properties of the desired lower-dimensional invariant manifold which is embedded within the higher dimensional system used as the RNN's basis of approximation.

6.2. **Learning Markovian model errors from noise-free data.**

6.2.1. *Experimental setup.* In the Markovian error modeling experiments described in Sections 6.2.3 and 6.2.4, whether using continuous- or discrete-time models, we train a random features model on noise-free trajectories from the true system (an ODE). The problems we study provably have a compact global attractor and are provably (L63)

or empirically (L96MS) ergodic; the invariant distribution is supported on the global attractor and captures the statistics of long-time trajectories which, by ergodicity, are independent of initial condition. The data trajectories are generated using scipy's implementation of Runge-Kutta 5(4) (via `solve_ivp`) with absolute and relative tolerances both 10^{-9} and maximum step size 10^{-4} [38, 163]. In order to obtain statistical results, we create 5 training trajectories from the true system of interest with initial conditions sampled independently from its attractor. Note that each training trajectory is long enough to explore the attractor, and each is used to train a separate model; the purpose is to observe the variance in learnt models with respect to randomly sampled paths through the attractor. We use the same sampling procedure to generate short independent validation and testing trajectories—we use 7 validation trajectories and 10 testing trajectories (these are short because we only use them to evaluate a model's short term forecast performance; when assessing long-term statistics of a learnt model, we compare to very long simulations from the true system). All plots use error bars to represent empirical estimates of the mean and standard deviation of the presented performance metric, as computed by ensembling the performance of the 5 models (one per training trajectory) over the 10 testing trajectories for a total of 70 random performance evaluations.

Each training procedure also involves an independent draw of the random feature functions as defined in (8.9). A validation step is subsequently performed to optimize the hyperparameters ω, β, as well as the regularization parameter λ. We automate this validation using Bayesian Optimization [111, 116], and find that it typically identifies good hyperparameters within 30 iterations. The entire process of entraining a model to a single, long training trajectory (including hyperparameter validation) typically takes approximately 30 minutes on a single core of a 2.1GHz Skylake CPU with an allocated 1GB RAM. Given a realization of random features and an optimal λ, we obtain the minimizer C^* using the Moore-Penrose Pseudoinverse implemented in scipy (`pinv2`). This learned C^*, paired with its random feature realization, is then used to predict 10 unseen testing trajectories (it is given the true initial condition for each of these testing trajectories).

When implementing in continuous-time, given high frequency but discrete-time data, two computational issues must be addressed: (i) extrapolation of the data to continuous-time; (ii) discretization of the resulting integrals. The approach we adopt avoids "inverse crimes" in which favorable behavior is observed because of agreement between the data generation mechanism (with a specific integrator) and the approximation of the objective functions [32, 74, 171]; see Queiruga et al. [128] for further illustration of this issue and Keller and Du [77], Du et al. [39] for a rigorous analysis of this inversion process in the context of linear multistep integration methods for deep learning. We interpolate the data with a spline, to obtain continuous-time trajectories, and then discretize the integrals using a simple Riemann sum; this strikes a desirable balance between robustness and efficiency and avoids inverse crimes. The discrete-time approaches, however, are able to learn not only model-discrepancy, but also integrator-based discrepancies; hence, the discrete-time methods may artificially appear to outperform continuous-time approaches, when, in fact, their performances might simply be considered to be comparable.

6.2.2. *Evaluation criteria.* Models are evaluated against the test set for their ability to predict individual trajectories, as well as invariant statistics (the invariant measure and the autocorrelation function).

Trajectory Validity Time: Given threshold $\gamma > 0$, we find the first time t_γ at which the norm of discrepancy between true and approximate solutions reaches γ:

$$t_\gamma = \text{argmin}_{t\in[0,T]}\left\{t : \ \|x(t) - x_m(t)\| \geq \gamma\overline{\|x(t)\|}\right\},$$

where $x(t)$ is the true solution to (2.2), $x_m(t)$ is the learned approximation, and the normed time average $\overline{\|x(t)\|}$ is approximated from training data. If the threshold is not violated on $[0, T]$, we define $t_\gamma := T$; this is rare in practice. We take $\gamma = 0.05$ (i.e. 5% relative divergence).

Invariant Distribution: To quantify errors in our reconstruction of the invariant measure, we consider the Kullback-Leibler (KL) divergence [84] between the true invariant measure μ and the invariant measure produced by our learned model μ_m. We approximate the divergence

$$d_{\text{KL}}(\mu, \mu_m) := \int_{\mathbb{R}} \log\left(\frac{d\mu}{d\mu_m}\right) d\mu$$

by integrating kernel density estimates with respect to the Lebesgue measure.

Autocorrelation: We compare the autocorrelation function (ACF) with respect to the invariant distribution of the true and learned models. We approximate the ACF using a fast-Fourier-transform for convolutions Seabold and Perktold [151], and compare them via a normalized L_2 norm of their difference.

6.2.3. *Lorenz '63 (L63).*

Setting. The L63 system [97] is described by the following ODE

$$\begin{aligned} \dot{u}_x &= a(u_y - u_x), \\ \dot{u}_y &= bu_x - u_y - u_x u_z, \\ \dot{u}_z &= -cu_z + u_x u_y \end{aligned} \tag{6.1}$$

whose solutions are known to exhibit chaotic behavior for parameters $a = 10$, $b = 28$, $c = \frac{8}{3}$. We align these equations with our framework, starting from equation (2.1), by letting $x = (u_x, u_y, u_z)^T$ and defining $f^\dagger(x)$ to be the vector field appearing on the right-hand-side in (6.1). We define a discrete solution operator $\Psi^\dagger$ by numerical integration of $f^\dagger$ over a fixed time window Δt corresponding to a uniform data sampling rate, so that the true system is given by (2.1) in continuous-time and (2.6a) in discrete-time.

To simulate scenarios in which our available physics are good, but imperfect, we assume there exists additive unknown model error of form

$$m^\dagger(x) = \epsilon\, m_1(x) \tag{6.2}$$

with function m_1 determining the structure of model error, and scalar coefficient ϵ determining its magnitude. Recall that $f^\dagger = f_0 + m^\dagger$ and we assume f_0 is known to us. Our task is then to learn $f^\dagger$ by learning $m^\dagger$ and adding it to f_0. The discrete solution operator Ψ_0 is obtained as in (2.6b) by numerical integration of f_0 over a fixed time window Δt.

To simplify exposition, we explicitly define $m^\dagger$, then let $f_0 := f^\dagger - m^\dagger$. We first consider the setting where

$$m_1(x) := \begin{bmatrix} 0 \\ bu_x \\ 0 \end{bmatrix} \tag{6.3}$$

(as in [121]) and modulate ϵ in (6.2) to control the magnitude of the error term. In this case, f_0 can be viewed as the L63 equations with perturbed parameter $\tilde{b} = b(1-\epsilon)$, where b is artificially decreased by $100\epsilon\%$.

Then, we consider a more general case of heterogeneous, multidimensional residual error by drawing m_1 from a zero-mean Gaussian Process (GP) with a radial basis kernel (lengthscale 10). We form a map from $\mathbb{R}^3$ into itself by constructing three independent draws from a scalar-valued GP on $\mathbb{R}^3$. The resulting function is visualized in two-dimensional projections in Figure 1.

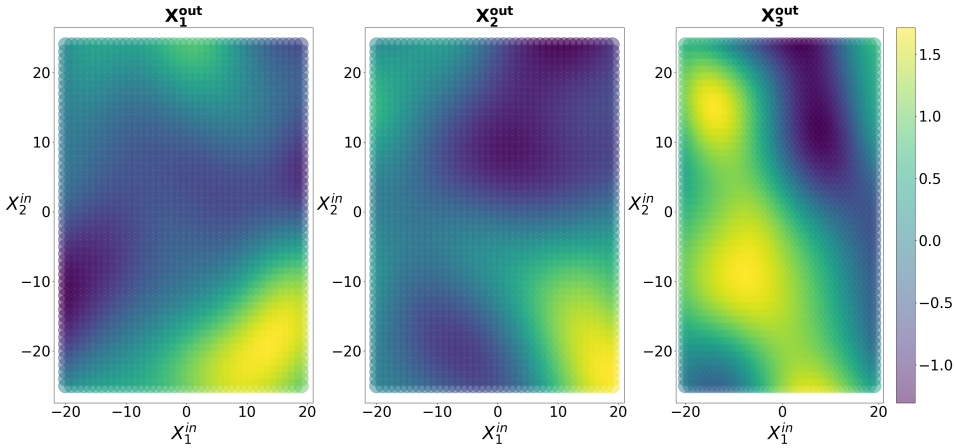
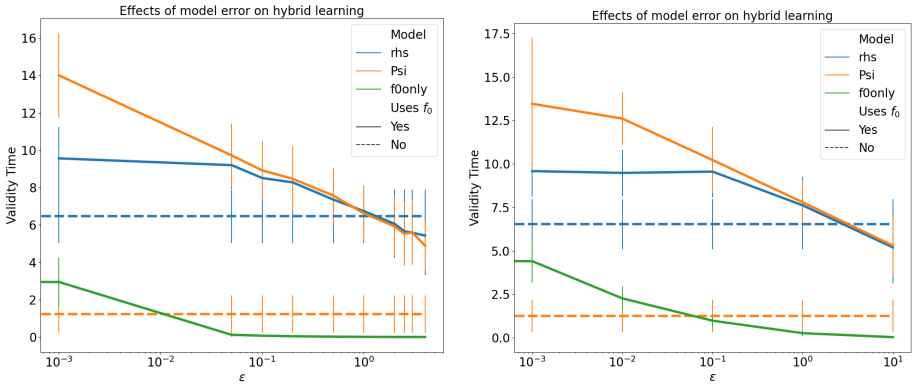


FIGURE 1. Here we visualize an example of the function m_1 in (6.2), which is obtained as a single random draw from a zero-mean Gaussian Process mapping $\mathbb{R}^3 \to \mathbb{R}^3$. We have plotted its output surface as three scalar functions (left to right) of the first two inputs (the plot axes) with the third input component fixed at 0.

Observe that in the continuous-time framing, changes to ϵ do not impact the complexity of the learned error term; however, it does grow the magnitude of the error term. In the discrete-time framing, larger values of ϵ can magnify the complexity of the discrepancy $\Psi_0(x) - \Psi^\dagger(x)$.

Results. We perform a series of experiments with the L63 system in order to illustrate key points about using data to learn model errors in dynamical systems. First, we demonstrate that hybrid modeling tends to outperform data-only and physics-only methods in terms of prediction quality. We first consider model error as in (6.3); see Figure 2(a) in which we study performance (validity time) versus model error amplitude (ϵ), using random feature maps with $D = 200$, and a single trajectory of length $T = 100$ sampled at timestep $\Delta t = 0.001$. Unless otherwise specified, this is also the configuration used in subsequent experiments.



(A) (B)

FIGURE 2. These plots show the temporal length of the forecast validity of our learnt models of L63 (6.1), each as a function of model error, as parameterized by ϵ (6.2) (with $D = 200$, $T = 100$, and $\Delta t = 0.001$). Continuous-time methods are shown in blue, discrete-time approaches in orange. Dotted lines indicate purely data-driven methods to learn the entire vector field defining the dynamics; solid lines indicate methods that learn perturbations to the imperfect mechanistic models f_0 or Ψ_0. Integration using the imperfect mechanistic model, without recourse to data, is shown in green. In Figure 2(a), we employ the linear form of model error m_1 defined in (6.3). In Figure 2(b), we let m_1 be a single draw from a Gaussian Process, whose structure is shown in Figure 1. Here, we plot means, with error bars as 1 standard deviation.

We see identical trends in Figure 2(b) for a more general case with the nonparametric model error term constructed from Gaussian processes. Interestingly, we see that for small and moderate amounts of model error ϵ, the hybrid methods substantially outperform data-only and physics-only methods. Eventually, for large enough model discrepancy, the hybrid-methods and data-only methods have similar performance; indeed the hybrid-method may be outperformed by the data-only method at large discrepancies. For the simple model error this appears to occur when the discrepancy term is larger in magnitude than f_0 (e.g. for $b = 28$ and $\epsilon = 2$, the model error term $\epsilon b u_x$ can take on values larger than $f^\dagger$ itself).

Figure 2(b) also shows that a continuous-time approach is favored over discrete-time when using data-only methods, but suggests the converse in the hybrid modeling context. We suspect this is an artifact of the different integration schemes used in data generation, training, and testing phases; the data are generated with a higher-fidelity integrator than the one available in training and testing. For the continuous-time method, this presents a fundamental limitation to the forecast quality (we chose this to avoid having artificially high forecast validity times). However, the discrete-time method can overcome this by not only learning the mechanistic model discrepancy, but also the discrepancy term associated with a mismatched integrator. This typically happens

when a closure is perfectly learnable and deterministic (i.e. our Lorenz '63 example); in this case, the combination of physics-based and integrator-sourced closures can be learned nearly perfectly. In later experiments with a multiscale system, the closures are considered approximate (they model the mean of a noisy underlying process) and the discrete- and continuous-time methods perform more similarly, because the inevitable imperfections of the learned closure term dominate the error rather than the misspecified integrator. Note that approximate closures driven by scale-separation are much more realistic; thus we should not expect the hybrid discrete-time method to dramatically outperform hybrid continuous-time methods unless other limitations are present (e.g. slow sampling rate).

Importantly, the parameter regime for which hybrid methods sustain advantage over the imperfect physics-only method is substantial; the latter has trajectory predictive performance which drops off rapidly for very small ϵ. This suggests that an apparently uninformative model can be efficiently modified, by machine learning techniques, to obtain a useful model that outperforms a *de novo* learning approach.

Next, we show that hybrid methods simplify the machine learning task in terms of complexity of the learned function and, consequently, the amount of data needed for the learning. Figure 3 examines prediction performance (validity time) as a function of training data quantity using random feature maps with $D = 2000$ and a fixed parametric model error ($\epsilon = 0.2$ in (6.2)) and sampling rate $\Delta t = 0.01$. We see that the hybrid methods substantially outperform the data-only approaches in regimes with limited training data. For the continuous-time example, we see an expected trend, where the data-only methods are able to catch up to the hybrid methods with the acquisition of more data. The discrete-time models do not exhibit this behavior, but we expect the data-only discrete-time model to eventually catch up, albeit with additional training data and number of parameters. Note that greater expressivity is also required from data-only methods—our choice of a large $D = 2000$ aims to give all methods ample expressivity, and thus test convergence with respect to training data quantity alone. These results demonstrate that the advantage of hybrid modeling is magnified when training data are limited and cannot fully inform *de novo* learning. Figure 4 further studies the impact of expressivity by again fixing a parametric model error ($\epsilon = 0.05$ in (6.2)), training length $T = 100$, and sampling rate $\Delta t = 0.001$. We see that all methods improve with a larger number of random features, but that relative superiority of hybrid methods is maintained even for $D = 10000$.

Finally, we study trade-offs between learning in discrete- versus continuous-time for the L63 example (6.1). Figure 5 examines prediction performance (validity time) as a function of data sampling rate Δt using random feature maps with $D = 200$ with a fixed parametric model error ($\epsilon = 0.05$ in (6.2)) and an abundance of training data $T = 1000$. We observe that for fast sampling rates ($\Delta t < 0.01$), the continuous-time and discrete-time hybrid methods have similar performance. For $\Delta t > 0.01$, derivatives become difficult to estimate from the data and the performance of the continuous-time methods rapidly decline. However, the discrete-time methods sustain their predictive performance for slower sampling rates ($\Delta t \in (0.01, 0.1)$). At some point, the discrete-time methods deteriorate as well, as the discrete map becomes complex to learn at longer terms because of the sensitivity to initial conditions that is a hallmark of chaotic systems. Here, the discrete-time methods begin to fail around $\Delta t = 0.2$; note that they

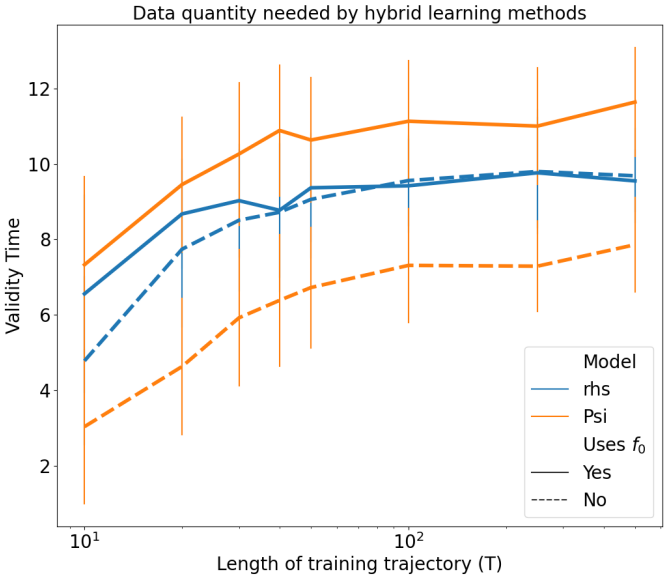


FIGURE 3. Here we examine the performance of the proposed methods as a function of the length of the interval over which the training data is provided, where $\Delta t = 0.01$, ($\epsilon = 0.2$ in (6.2)), and $D = 2000$ are held constant for the L63 example (6.1). See description of Figure 2 for explanation of legend. We observe that all methods improve with increasing training lengths. We see that, in continuous-time, the primary benefit in hybrid modeling is when the training data are limited.

can be extended to longer time intervals by increasing D and amount of training data, but returns diminish quickly.

6.2.4. *Lorenz '96 Multiscale (L96MS) System.*
Setting. Here, we consider the multiscale system [96] of form (2.7), where each variable $X_k \in \mathbb{R}$ is coupled to a subgroup of fast variables $Y_k \in \mathbb{R}^J$. We have $X \in \mathbb{R}^K$ and $Y \in \mathbb{R}^{K\times J}$. For $k = 1 \dots K$ and $j = 1 \dots J$, we write

$$\dot{X}_k = f_k(X) + h_x \overline{Y}_k, \tag{6.4a}$$

$$\dot{Y}_{k,j} = \frac{1}{\epsilon} r_j(X_k, Y_k), \tag{6.4b}$$

$$\overline{Y}_k = \frac{1}{J}\sum_{j=1}^{J} Y_{k,j}, \tag{6.4c}$$

$$f_k(X) = -X_{k-1}(X_{k-2} - X_{k+1}) - X_k + F, \tag{6.4d}$$

$$r_j(X_k, Y_k) = -Y_{k,j+1}(Y_{k,j+2} - Y_{k,j-1}) - Y_{k,j} + h_y X_k, \tag{6.4e}$$

$$X_{k+K} = X_k, \quad Y_{k+K,j} = Y_{k,j}, \quad Y_{k,j+J} = Y_{k+1,j}, \tag{6.4f}$$

where $\epsilon > 0$ is a scale-separation parameter, $h_x, h_y \in \mathbb{R}$ govern the couplings between the fast and slow systems, and $F > 0$ provides a constant forcing. We set $K = 9$, $J =$

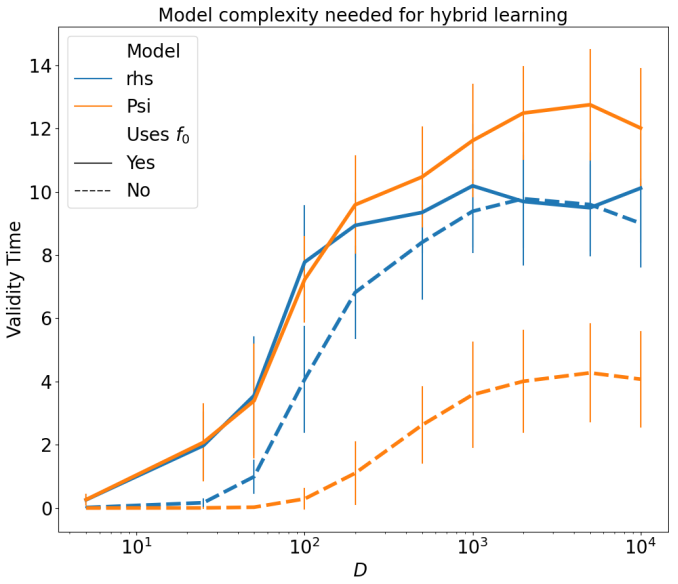


FIGURE 4. Here we examine the performance of the proposed methods as a function of model complexity, where $\Delta t = 0.001$, $\varepsilon = 0.05$, and $T = 100$ are held constant for the L63 example (6.1). See description of Figure 2 for explanation of legend. We observe that all methods improve with increasing number of parameters, and that hybrid methods are especially beneficial when available complexity is limited.

8, $h_x = -0.8$, $h_y = 1$, $F = 10$; this leads to chaotic dynamics for ε small. When studying scale-separation, we consider $\varepsilon \in \{2^{-7}, 2^{-5}, 2^{-3}, 2^{-1}\}$.

We consider the setting in which we learn Markovian random features models in variable X alone, from X data generated by the coupled (X, Y) system. Large scale-separation between the observed (X) and unobserved (Y) spaces can simplify the problem of accounting for the unobserved components; in particular, for sufficient scale-separation, we expect a Markovian term to recover a large majority of the residual errors. In fact, we further simplify this problem by learning a scalar-valued model error M that is applied to each X_k identically in the slow system:

$$\dot{X}_k \approx f_k(X) + M(X_k).$$

This choice stems from observations about statistical interchangeability amongst the slow variables of the system; these properties of the L96MS model in the scale-separated regime are discussed in [44]. We can directly align our reduction of (6.4) with the Markovian hybrid learning framework in (2.2) as follows:

$$\begin{aligned} \dot{X} &\approx f_0(X) + m(X), \\ f_0(X) &\coloneqq [f_1(X), \cdots, f_K(X)]^T, \\ m(X) &\coloneqq [M(X_1), \cdots, M(X_K)]^T. \end{aligned}$$

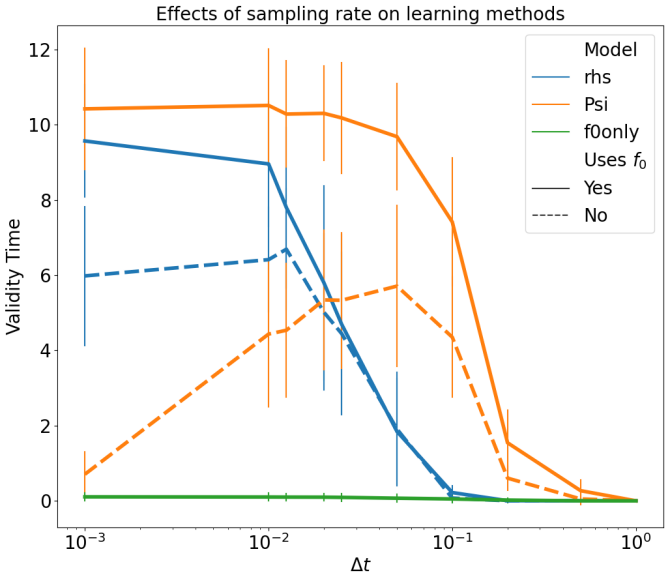


FIGURE 5. This shows temporal forecast validity as a function of the step size of training data for the tested methods in the L63 example (6.1). We hold fixed $D = 200$, $\epsilon = 0.05$, and $T = 1000$. See description of Figure 2 for explanation of legend. We see that while purely data-driven discrete-time methods struggle at short time steps, the hybrid version thrives in this scenario. All approaches, of course, eventually decay as large time steps create more complex forward maps, due to sensitivity to initial conditions. We also see continuous-time methods work well for small time steps, then deteriorate in tandem with quality of estimated derivatives.

Results. We plot the performance gains of our hybrid learning approaches in Figure 6 by considering validity times of trajectory forecasts, estimation of the invariant measure, and ACF estimation. In all three metrics (and for all scale-separations ε), *de novo* learning in discrete ($\Psi^\dagger \approx m$) and continuous-time ($f^\dagger \approx m$) is inferior to using the nominal mechanistic model f_0. We found that the amount of data used in these experiments is insufficient to learn the full system from scratch. On the other hand, hybrid models in discrete ($\Psi^\dagger \approx \Psi_0 + m$) and continuous-time ($f^\dagger \approx f_0 + m$) noticeably outperformed the nominal physics.

Surprisingly, Figure 6 shows that the Markovian closure methods still qualitatively reproduce the invariant statistics even for large ε settings where we would expect substantial memory effects. Figure 6 also demonstrates this quantitatively using KL-divergence between invariant measures and mean-squared-error between ACFs. It seems that for this dissipative system, memory effects tend to average out in the invariant statistics. However, the improvements in validity time for trajectory-based forecasting deteriorate for $\varepsilon = 2^{-1}$.

To visualize this non-Markovian structure, and how it might be exploited, we examine the residuals from f_0 in Figure 7 and observe that there are discernible trajectories

walking around the Markovian closure term. For small ε, these trajectories oscillate rapidly around the closure term. For large ε (e.g. 2^{-1}), however, we observe a slow-moving residual trajectory around the Markovian closure term. This indicates the presence of a stronger memory component, and thus would benefit from a non-Markovian approach to closure modeling.

Jiang and Harlim [71] show that the memory component in this setting with $\varepsilon = 2^{-1}$ can be described using a closure term with a simple delay embedding of the previous state at lag 0.01. They learn the closure using a kernel method cast in an RKHS framework, for which random feature methods provide an approximation.

6.3. **Learning from partial, noisy observations.** In this section, we focus on the non-Markovian setting outlined in Section 2.3, and attempt to model the dynamics of the observable using (4.3), with f_1, f_2 given by two-layer, fully connected neural networks with GeLU activations [65], and perform the learning by minimizing (4.7) from Section 4.3.3, using 3DVAR for the data assimilation [87, 88], with the ADAM optimizer [80]. The learning rate was initialized at 0.01 and tuned automatically using a scheduler that halved the learning rate if the training error had not decreased over 10 (mini-batched) epochs. Data were sampled at $\Delta t = 0.01$ in all cases, and normalized to have mean zero and unit variance. Numerical integration was performed with the `torchdiffeq` implementation of the Dormand-Prince adaptive fifth-order Runge-Kutta method: for the L63 example, simple backpropagated autodifferentiation was performed through this solver; for the L96MS example, we used the adjoint method provided by [142].

6.3.1. *Lorenz '63.* We first consider modeling the dynamics of the first-component of the L63 system in (6.1), where we noisily observe the first-component – that is, we observe a noisy trajectory of u_x (i.i.d. additive zero-mean, variance-one Gaussian), but do not observe the remaining components u_y, u_z. We jointly trained on 100 trajectories, each of length $T = 10$, and randomly initialized from a box around the attractor; we chose this approach to ensure that we had data coverage both on and off the attractor although we note that similar success is obtained with a single trajectory of length $T = 1000$. The neural network had width 50. We chose an assimilation time of $\tau_1 = 3$ and a forecast time of $\tau_2 = 0.1$. The optimization ran for approximately 200 epochs, and took roughly 24hrs on a single GPU. Adequate results were obtained using a fixed 3DVAR gain matrix $K = [0.5, 0, 0]^T$. However, we present results using the algorithm in which $K = \theta_{\mathrm{DA}}$ is jointly learned along with parameters θ_{DYN}, as described in Section 4.3.3; this demonstrates that the gain need not be known *a priori*.

First, we present results using knowledge that the correct hidden dimension $d_r = 2$: in Figure 8a, we show an example of the trained model being assimilated (using 3DVAR with learnt K) during the first 3 time units, then predicting for another 7 time units; recall that training was performed using only a $\tau_2 = 0.1$ forecasting horizon, but we evaluate on a longer time horizon to provide a robust test metric. Observe that the learnt hidden dynamics in gray are synchronized with the data, then used to perform the forecast. In Figure 8b and 8c, we show that by solving the system for long time (here, $T = 10^4$), we are able to accurately reproduce invariant statistics (invariant measure and autocorrelation, resp.) for the true system. In Figure 8d, we show the evolution of the learnt K.

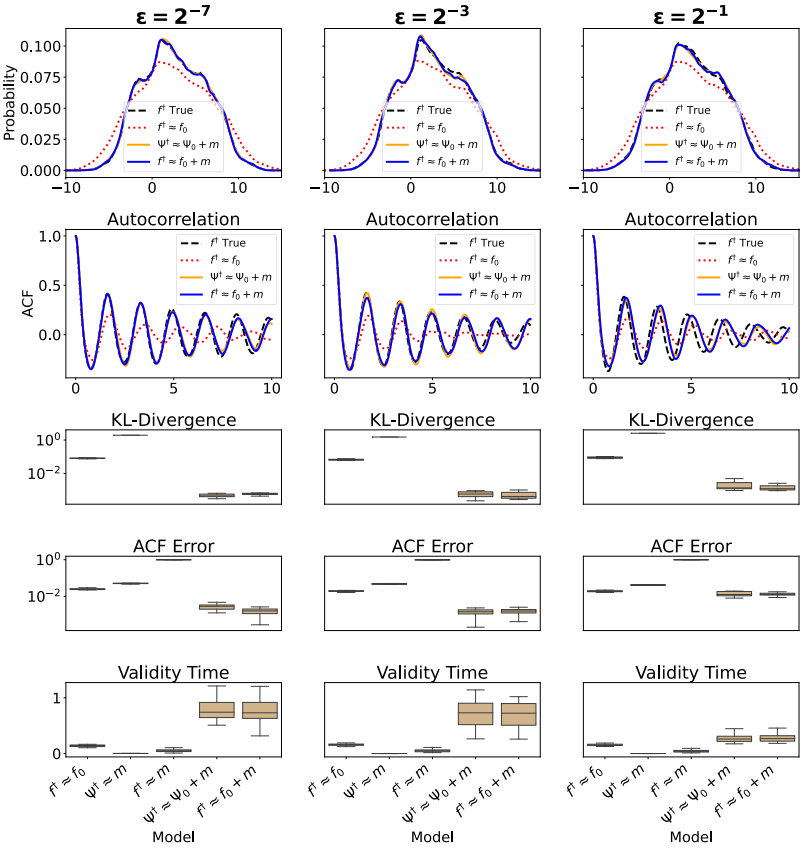


FIGURE 6. This figure shows the performance of different approaches to modeling the L96MS slow subsystem (6.4). In $f^\dagger \approx f_0$, we only use the nominal physics f_0. In $\Psi^\dagger \approx m$ and $f^\dagger \approx m$, we try to learn the entire right-hand-side using only data (in discrete- and continuous-time settings, respectively). In $\Psi^\dagger \approx \Psi_0 + m$ and $f^\dagger \approx f_0 + m$, we focus on learning Markovian residuals for the known physics (in discrete- and continuous-time settings, respectively). The residual-based correctors substantially outperform the nominal physics and purely data-driven methods according to all presented metrics: invariant measure (shown qualitatively in the first row and quantitatively in the third row), ACF (shown qualitatively in the second row and quantitatively in the fourth row), and trajectory forecasts (shown in the final row). The boxplots show the distributions of quantitative metrics (e.g. KL-divergence, squared errors, validity time), which come from different models, each trained on a different trajectory, and generated using an independent random feature set. Notably, the Markovian residual-based methods' performance deteriorates for small scale-separation ($\varepsilon = 2^{-1}$), where the Markovian assumption breaks down.

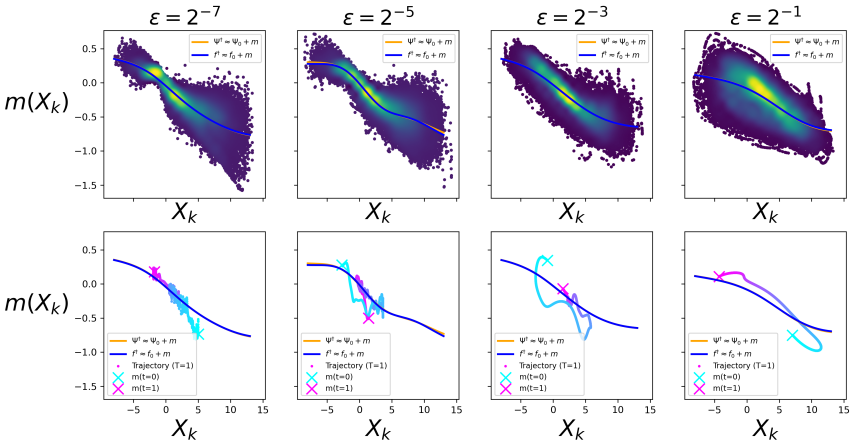


FIGURE 7. This figure shows the observed and estimated residuals of the nominal physics model f_0 for the L96MS slow subsystem (6.4) at different scale-separation factors. The first row shows the density of these residuals (yellow is high density, blue is low), as well as the fit of our closure terms in continuous- (blue) and discrete- (orange) time (the discrete model was normalized by dividing by Δt). The second row shows temporal structure in the errors of our residual fit by superimposing a short (T=1) one-dimensional trajectory (this represents $\sim 0.1\%$ of training data).

Next, we let $d_r = 10$, exceeding the true dimension of the hidden states; thus we are able to explore issues caused by learning an overly expressive (in terms of dimension of hidden states) dynamical model. Figure 9 shows dynamics for a learnt model in this setting; we found its reproduction of invariant statistics to be similar to the cases in Figure 8b and 8c, but omit the plots for brevity. This success aligns with the approximation theory, as discussed in Remark 5.5, and provides empirical reassurance that the methodology can behave well in situations where the dimension of the hidden variable is unknown and dimension d_r used in learning exceeds its true dimension. Nevertheless, we construct an example in Section 6.4 in which a specific embedding of the true dynamics in a system of higher dimension can lead to poor approximation; this is caused by an instability in the model which allows departure from the invariant manifold on which the true dynamics is accurately captured. However, we emphasize that this phenomenon is not observed empirically in the experiment reported here with $d_r = 10$. Nonetheless we also note expected decreases in efficiency caused by over-estimating the dimension of the hidden variable, during both model training and testing; thus determining the smallest choice for d_r, compatible with good approximation, is important. Recent research has addressed this challenge in the discrete-time setting by applying manifold learning to a delay-embedding space, then using the learnt manifold to inform initialization and dimensionality of LSTM hidden states [78].

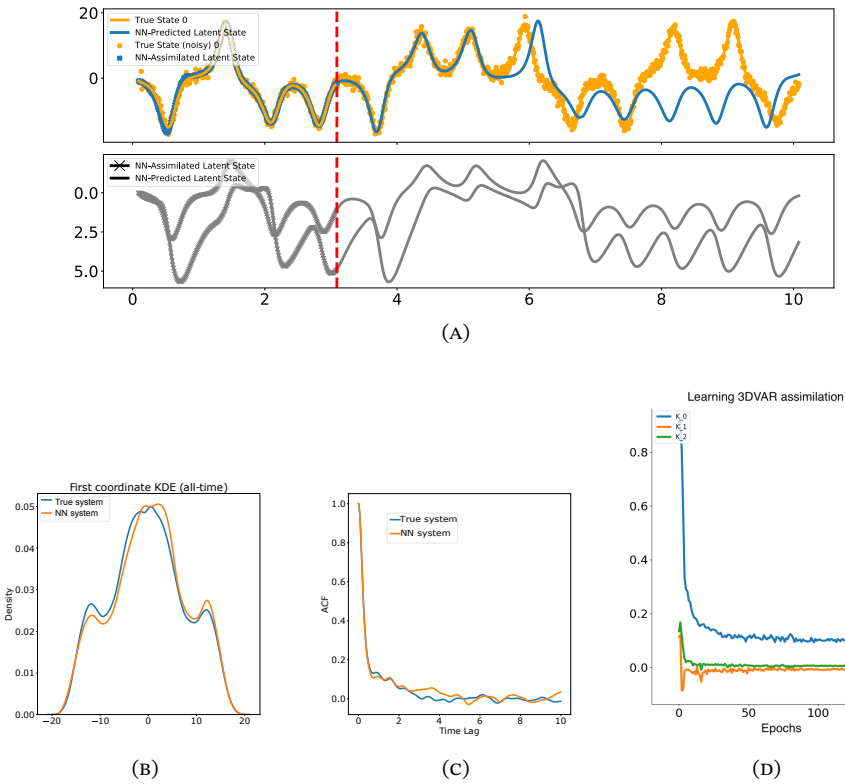
Note that our early attempts at achieving these numerical results, using the optimization ideas in Sections 4.3.1 and 4.3.2, yielded unstable models that exhibited blow-up on shorter time scales (e.g. $T < 1000$); however, by incorporating data assimilation as in [26], and further tuning the optimization to achieve lower training errors, we were able to obtain a model that, empirically, did not exhibit blow-up, even when solved for very long time (e.g. $T = 10^5$). We also note that we were unable to achieve such high-fidelity results using the methods of [118] on neural networks with nonlinear activation functions; this may be explained by noting that Ouala et al. [118] achieved their results using linear, identity-based activations, resulting in inference of polynomial models containing the true L63 model.

6.3.2. *Lorenz '96 Multiscale* ($\varepsilon = 2^{-1}$). Recall that Markovian closures fail to capture autocorrelation statistics for the slow components of this model in the case of $\varepsilon = 2^{-1}$ (see top right panel of Figure 6). As evidenced by the slow-moving trajectory around the Markovian closure in Figure 7, this is a case ripe for non-Markovian modeling. We investigate the applicability of our continuous-time ODE formulation in (4.3), using a neural network of width of 1000. We applied the above described methodology for minimizing (4.7), under the data setting described in Section 6.2.4, to learn hidden dynamics. Similarly to the previous section, we jointly trained on 100 trajectories, each of length $T = 20$ and randomly initialized from a box around the attractor. We chose an assimilation time of $\tau_1 = 2$ and a forecast time of $\tau_2 = 1$; note that longer times can become quite costly, especially for high-dimensional systems; nevertheless, the assimilation time τ_1 appears intrinsically tied to the amount of memory present in the system.

In Figure 10a and 10b, we plot comparisons of the true and learnt (via (4.3)) ACF and invariant measure, and observe substantial improvement over the Markovian closure. However, this learnt model exhibited instabilities when solved for longer than $T = 500$. We expect that this can be remedied via further training (as was found for the L63 example); however, the incorporation of stability constraints into the model, as in [150], would be valuable. In order to train this larger model for longer time, further studies of efficient optimization must also be performed in this setting ([26] has begun highly relevant investigation in this direction).

In Figure 11, we visualize the learnt 3DVAR gain (which encodes the learnt model's covariance structure), in which each row corresponds to the gain for a given component of the learnt model as a function of observed components (indexed in the columns); trends are elucidated via hierarchical clustering and a row-based normalization of the learnt matrix K. It clearly learns a consistent diagonal covariance structure for the observables. More impressively, it illustrates cross-covariances between observed and hidden components that mirror the compartmentalized structure of the model in (6.4); note that each observed component has a distinct grouping of hidden variables which have high correlation (white) primarily with that component and low correlation (black) with other observables. This type of analysis may provide greater interpretability of learnt models of hidden dynamics.

6.4. **Initializing and stabilizing the RNN.** As mentioned in Remark 5.8 the RNN approximates an enlarged system which contains solutions of the original system as trajectories confined to the invariant manifold $m = m^\dagger(x, y)$; see identity (5.14). However,



(A) (B) (C) (D)

FIGURE 8. This figure concerns learning of a continuous-time RNN to model the L63 system (6.1), based on noisy observation of only the first component; it uses an augmented state space $d_r = 2$. Figure 8a shows how the trained model can be used for forecasting—by first synchronizing to data using 3DVAR, then forecasting into the future. The top-half depicts dynamics of the observed component (model-solutions in blue; observations in yellow); the bottom-half depicts the augmented state space (both hidden components are shown in gray). We observed a validity time of roughly 3 model time units. Figure 8b and 8c shows that long-time solutions of the learnt model accurately mirror invariant statistics (invariant measure and autocorrelation, resp.) for the true system. Figure 8d shows the learning process for estimating a 3DVAR gain K.

this invariant manifold may be unstable, either as a manifold within the continuous-time model (5.13) or as a result of numerical instability. We now demonstrate this with numerical experiments. This instability points to the need for data assimilation to be used with RNNs if prediction of the original system is desired, not only to initialize the system but also to stabilize the dynamics to remain near to the desired invariant manifold.

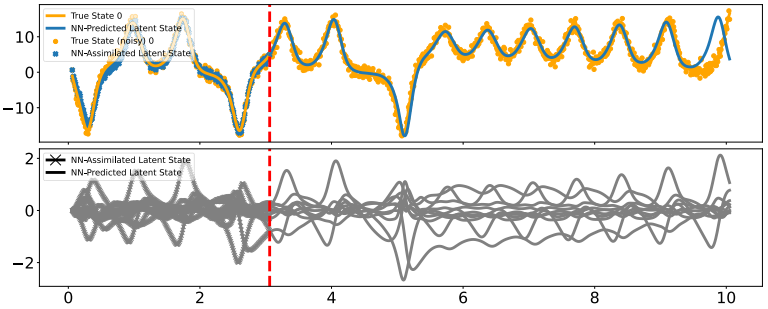


FIGURE 9. This figure concerns learning of a continuous-time RNN to model the L63 system (6.1), based on noisy observation of only the first component; it uses an augmented state space $d_r = 10$. The top-half depicts dynamics of the observed component (model-solutions in blue; observations in yellow); the bottom-half depicts the augmented state space (all 10 hidden components are shown in gray). In the first 3 time units, the model is assimilated to a sequence of observed data using 3DVAR, then in the subsequent 7 time units, a forecast is issued. We found this model to have similar short-term and long-term fidelity when compared to the model presented in Figure 8a, 8b, 8c, and 8d, which used the correct hidden dimension $d_r = 2$.

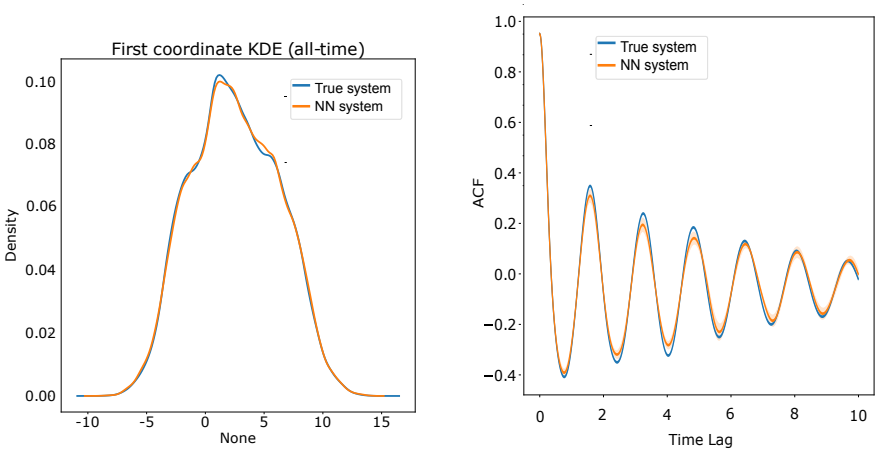
To illustrate these challenges, we consider the problem of modeling evolution of a single component of the L63 system (6.1). Consider this as variable x in (2.7). As exhibited in (5.13), model error may be addressed in this setting by learning a representation that contains the hidden states y in (2.7) (i.e. the other two unobserved components of (6.1)), but since the dimension of the hidden states is typically not known *a priori* the dimensions of the latent variables in the RNN (and the system it approximates) may be greater than those of y; in the specific construction we use to prove the existence of an approximating RNN we introduce a vector field for evolution of the error m as well as y. We now discuss the implications of embedding the true dynamics in a higher dimensional system in the specific context of the embedded system (5.13). However the observations apply to any embedding of the desired dynamics (2.7) (with $\epsilon = 1$) within any higher dimensional system.

We choose examples for which (5.14) implies that $m - m^\dagger$ is constant in time. Then, under (5.13),

$$\big(m - m^\dagger(x, y)\big)(t) = \text{constant};$$

that is, it is constant in time. The desired invariant manifold (where the constant is 0) is thus stable. However this stability only holds in a neutral sense: linearization about the manifold exhibits a zero eigenvalue related to translation of $m - m^\dagger$ by a constant. We now illustrate that this embedded invariant manifold can be unstable; in this case the instability is caused by numerical integration, which breaks the conservation of $m - m^\dagger$ in time.

Example 1. Consider equation (6.1) which we write in form (2.8) by setting $x = u_x$ and $y = (u_y, u_z)$. Then we let $f_0(u_x) := -au_x$ yielding $m^\dagger(u_y) = au_y$. Thus $f^\dagger = f_0 + m^\dagger$ is defined by the first component of the right-hand side of (6.1). The function



(A) (B)

FIGURE 10. This figure concerns learning of a continuous-time RNN to model the first 9 (slow) components of the L96MS system ($\varepsilon = 0.5$ in (6.4)), based on noisy observations of these slow components; it uses an augmented state space $d_r = 72$. We trained using noised observations (standard deviation 0.01) of only the first 9 components of the true 81−dimensional system. These plots show that this model can accurately reproduce both the invariant measure (Figure 10a) and ACF (Figure 10b) for these observed states. These statistics were calculated by running the learnt model for $T = 500$ model time units; longer runs encountered instabilities that caused trajectories to leave the attractor and blow-up.

$g^\dagger(u_y, u_z)$ is then given by the second and third components of the right-hand side of (6.1). Applying the methodology leading to (5.13) to (6.1) results in the following four dimensional system:

$$\dot{u}_x = f_0(u_x) + m, \qquad u_x(0) = x_0, \tag{6.5a}$$

$$\dot{u}_y = bu_x - u_y - u_x u_z, \qquad u_y(0) = y_0, \tag{6.5b}$$

$$\dot{u}_z = -cu_z + u_x u_y, \qquad u_z(0) = z_0, \tag{6.5c}$$

$$\dot{m} = a\big(bu_x - u_y - u_x u_z\big), \qquad m(0) = m^\dagger(y_0). \tag{6.5d}$$

Here we have omitted the u_y-dependence from equation (6.1) for u_x, and aim to learn this error term; we introduce the variable m in order to do so. This system, when projected into u_x, u_y, u_z, behaves identically to (6.1) when $m(0) = m^\dagger(y_0)$. Thus the 4−dimensional system in (6.5) has an embedded invariant manifold on which the dynamics is coincident with that of the 3−dimensional L63 system.

We numerically integrate the 4−dimensional system in (6.5) for 10000 model time units (initialized at $x_0 = 1, y_0 = 3, z_0 = 1, m_0 = ay_0 = 30$), and show in Figure 12 that the resulting measure for u_x (dashed red) is nearly identical to its invariant measure

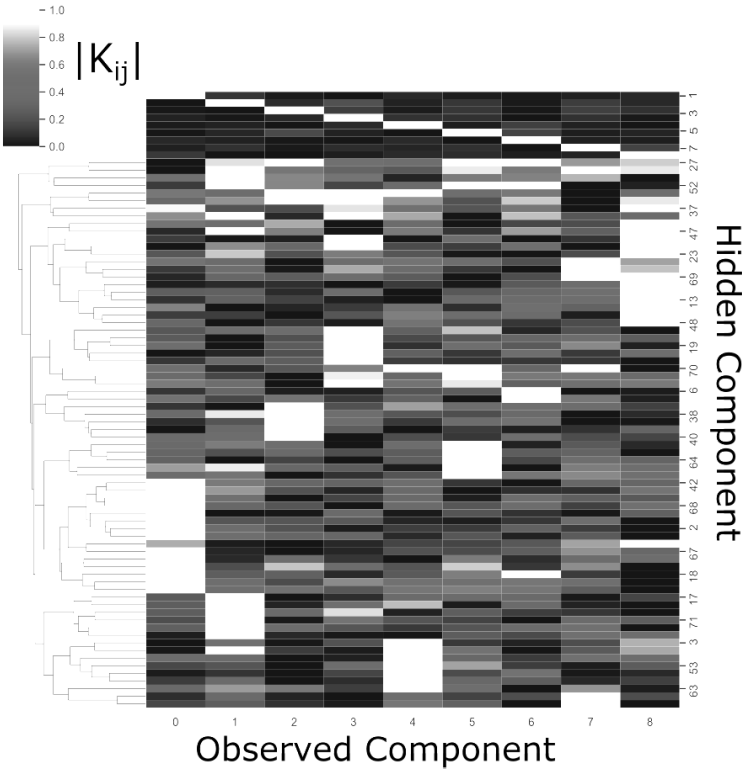


FIGURE 11. Here we visualize the learnt 3DVAR gain matrix K (81 x 9) (θ_{DA} in Section 4.3.3) associated with the non-Markovian learning of L96MS (6.4). We first compute entry-wise absolute values, then apply a row-normalization; white indicates highest correlation, and black indicates lowest correlation. The top 9 rows shown directly correspond to the first 9 rows of K. The bottom 72 rows are re-ordered (via hierarchical clustering) to illustrate associations between the 9 observed components and the 72 hidden variables.

in the traditional 3−dimensional L63 system in (6.1) (solid black). However, we rerun the simulation for a perturbed $m(0) = m^\dagger(y_0) + 1$, and see in Figure 12 (dotted blue) that this yields a different invariant measure for u_x. This result emphasizes the importance of correctly initializing an RNN not only for efficient trajectory forecasting, but also for accurate statistical representation of long-time behavior.

Example 2. Now we consider (6.1) which we write in form (2.8) by setting $x = u_z$ and $y = (u_x, u_y)$. We let $f_0(u_z) \coloneqq -cu_z$ and $m^\dagger(u_x, u_y) \coloneqq u_x u_y$, so that $f^\dagger = f_0 + m^\dagger$ corresponds to the third component of the right-hand side of (6.1). Function $g^\dagger(u_x, u_y)$ is defined by the first two components of the right-hand side of (6.1). We again form a 4−dimensional system corresponding to (6.1) using the methodology that leads to

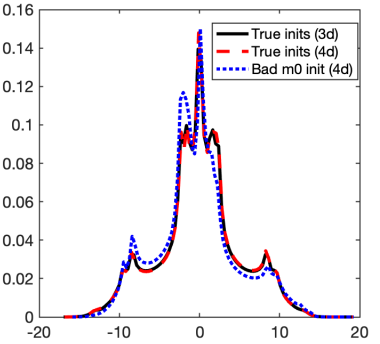


FIGURE 12. Here, we show that the invariant density for the first component of L63 (black) can be reproduced by a correctly initialized augmented 4−d system (dashed red) in (6.5). However, incorrect initialization of $m(0)$ in (6.5) (dotted blue) yields a different invariant density.

(5.13):

$$\text{(6.6a)} \qquad \dot{u}_x = a(u_y - u_x), \qquad u_x(0) = x_0,$$
$$\text{(6.6b)} \qquad \dot{u}_y = bu_x - u_y - u_x u_z, \qquad u_y(0) = y_0,$$
$$\text{(6.6c)} \qquad \dot{u}_z = f_0(u_z) + m, \qquad u_z(0) = z_0,$$
$$\text{(6.6d)} \qquad \dot{m} = u_x \dot{u}_y + u_y \dot{u}_x, \qquad m(0) = m^\dagger(x_0, y_0).$$

We integrate (6.6) for 3000 model time units (initialized at $x_0 = 1, y_0 = 3, z_0 = 1, m_0 = x_0 y_0 = 3$), and show in Figure 13 that the 3−dimensional Lorenz attractor is unstable with respect to perturbations in the numerical integration of the 4−dimensional system. The solutions for u_x, u_y, u_z eventually collapse to a fixed point after the growing discrepancy between $m(t)$ and $m^\dagger$ becomes too large. The time at which collapse occurs may be delayed by using smaller tolerances within the numerical integrator (we employ MATLAB RK45) demonstrating that the instability is caused by the numerical integrator. This collapse is very undesirable if prediction of long-time statistics is a desirable goal. On the other hand, Figure 14 shows short-term accuracy of the 4−dimensional system in (6.6) up to 12 model time units when correctly initialized ($m_0 = m^\dagger(x_0, y_0)$, dashed red), and accuracy up to 8 model time units when initialization of m_0 is perturbed ($m_0 = m^\dagger(x_0, y_0) + 1$, dotted blue). This result demonstrates the fundamental challenges of representing chaotic attractors in enlarged dimensions and may help explain observations of RNNs yielding good short-term accuracy, but inaccurate long-term statistical behavior. While empirical stability has been observed in some discrete-time LSTMs [62, 164], the general problem illustrated above is likely to manifest in any problems where the dimension of the learned model exceeds that of the true model; the issue of how to address initialization of such models, and its interaction with data assimilation, therefore merits further study.

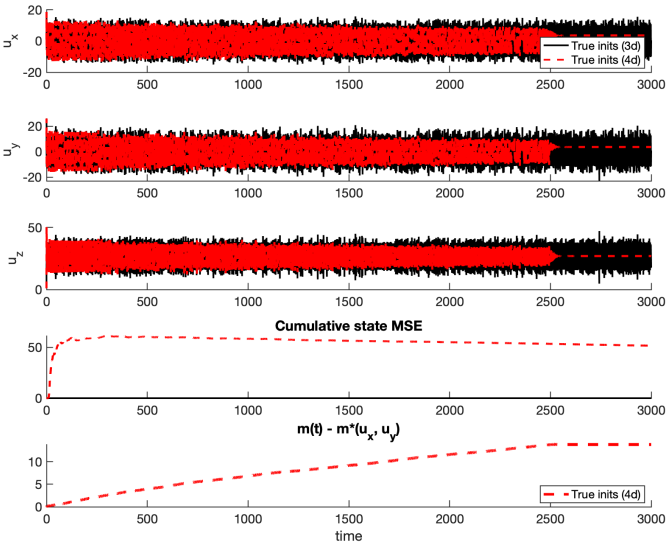


FIGURE 13. Here we show that the embedded 3−dimensional manifold of L63, within the 4−dimensional system given by (6.6), is unstable. Indeed the correctly initialized 4−dimensional system (dashed red) has solution which decays to a fixed point. The bottom figure shows divergence of the numerically integrated model error term $m(t)$ and the state-dependent term $m^{\dagger}$; this growing discrepancy is likely responsible for the eventual collapse of the 4−dimensional system.

7. CONCLUSIONS

In this work we evaluate the utility of blending mechanistic models of dynamical systems with data-driven methods, demonstrating the power of hybrid approaches. We provide a mathematical framework that is consistent across parametric and nonparametric models, encompasses both continuous- and discrete-time, and allows for Markovian and memory-dependent model error. We also provide basic theoretical results that underpin the adopted approaches. The unified framework elucidates commonalities between seemingly disparate approaches across various applied and theoretical disciplines. It would be desirable if the growing recognition of the need for hybrid modeling were to motivate flexible incorporation of mechanistic models into open-source software for continuous-time Markovian and non-Markovian modeling of error [2, 22, 42, 58, 118, 142].

Our work is focused on immutable mechanistic models (f_0 and Ψ_0), but these models themselves often have tunable parameters. In principle one can jointly learn parameters for the mechanistic model and closure term. However, the lack of identifiability between modifying the closure and modifying the physics brings up an interesting question in explainability. Future work might focus on decoupling the learning of parameters and closure terms so that maximal expressivity is first squeezed out of the mechanistic model [123, 124].

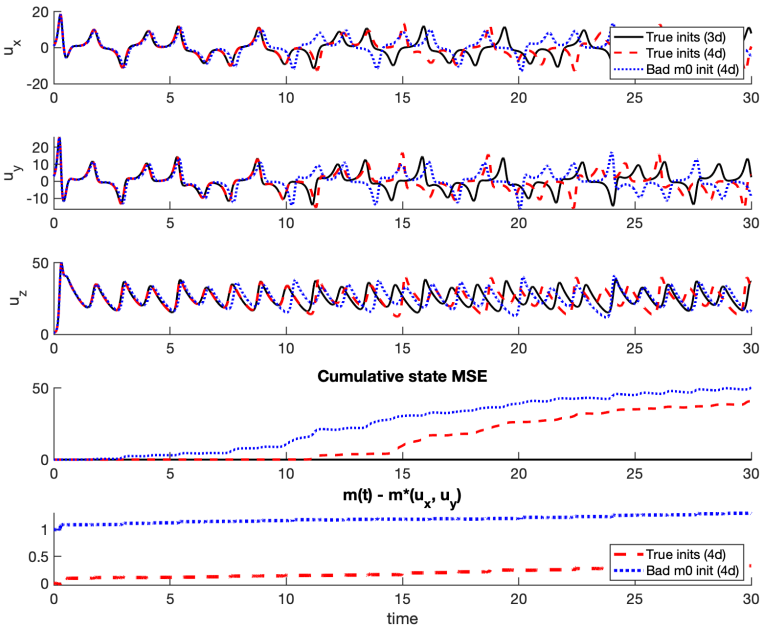


FIGURE 14. Here, we show short-term accuracy for the 4−dimensional system in (6.6). Predictions using the correct initialization of m_0 (dashed red) remain accurate for nearly twice as long as predictions that use a perturbed initialization ($m_0 = m^\dagger(u_x, u_y) + 1$). The bottom figure shows that $m(t)$ diverges from the state-dependent $m^\dagger$ more quickly for the poorly initialized model, but in both cases errors accumulate over time.

Our numerical results demonstrate the superiority of hybrid modeling over learning an entire system from scratch, even when the available mechanistic model has large infidelities. Hybrid modeling also showed surprisingly large performance gains over using mechanistic models with only small infidelities. We quantify these improvements in terms of data hunger, demands for model complexity, and overall predictive performance, and find that all three are significantly improved by hybrid methods in our experiments.

We establish bounds on the excess risk and generalization error that decay as $1/\sqrt{T}$ when learning model discrepancy from a trajectory of length T in an ergodic continuous-time Markovian setting. We make minimal assumptions about the nominal physics (i.e. $f_0 \in C^1$); thus, our result equivalently holds for learning the entire vector field $f^\dagger$ (i.e. $f_0 \equiv 0$). However the upper bounds on excess risk and generalization error scale with the size of the function being learned, hence going some way towards explaining the superiority of hybrid modeling observed in the numerical experiments. Future theoretical work aimed at quantifying the benefits of hybrid learning versus purely data-driven learning is of interest. We also note that the ergodic assumption underlying our theory will not be satisfied by many dynamical models, and alternate statistical learning theories need to be developed in such settings.

We illustrate trade-offs between discrete-time and continuous-time modeling approaches by studying their performance as a function of training data sample rate. We find that hybrid discrete-time approaches can alleviate instabilities seen in purely data-driven discrete-time models at small timesteps; this is likely due to structure in the integrator Ψ_0, which has the correct parametric dependence on timestep. In the continuous-time setting, we find that performance is best when derivatives can accurately be reconstructed from the data, and deteriorates in tandem with differentiation inaccuracies (caused by large timesteps); continuous-time hybrid methods appear to offer additional robustness to inaccurate differentiation when compared to purely data-driven methods. In cases of large timesteps and poorly resolved derivatives, ensemble-based data assimilation methods may still allow for accurate learning of residuals to the flow field for continuous-time modeling [53].

Finally, we study non-Markovian memory-dependent model error, through numerical experiments and theory, using RNNs. We prove universal approximation for continuous-time hybrid RNNs and demonstrate successful deployment of the methodology. Future work focusing on the effective training of these models, for more complex problems, would be of great value; ideas from data assimilation are likely to play a central role [26]. Further work on theoretical properties of reservoir computing (RC) variants on RNNs would also be of value: they benefit from convex optimization, and may be viewed as random feature methods between Banach spaces. These RNN and RC methods will benefit from constraining the learning to ensure stability of the latent dynamical model. These issues are illustrated via numerical experiments that relate RNNs to the question of stability of invariant manifolds representing embedded desired dynamics within a higher dimensional system.

8. Appendix

8.1. **Proof of excess risk/generalization error theorem.** Note that in both (5.7) and (5.8) $\varphi(\cdot)$ is only evaluated on (compact) $\mathcal{A}$ obviating the need for any boundedness assumptions on the functions $\{f_\ell\}_{\ell=0}^p$ and $m^\dagger$ in what follows.

Lemma 8.1. *Let Assumptions* A1 *and* A2 *hold. Then there is* Σ *positive semi-definite symmetric in* $\mathbb{R}^{p\times p}$ *such that* $\theta_T^* \to \theta_\infty^*$ *almost surely, and* $\sqrt{T}(\theta_T^* - \theta_\infty^*) \Rightarrow N(0,\Sigma)$ *with respect to* $x(0) \sim \mu$. *Furthermore, there is constant* $C \in (0,\infty)$ *such that, almost surely w.r.t.* $x(0) \sim \mu$,

$$\text{limsup}_{T\to\infty}\Big(\frac{T}{\log\log T}\Big)^{\frac{1}{2}} \|\theta_T^* - \theta_\infty^*\| \le C.$$

Proof. By rearranging the equation for θ_∞^* we see that

$$\begin{aligned} A_T\theta_T^* &= b_T, \\ A_T\theta_\infty^* &= b_\infty + (A_T - A_\infty)\theta_\infty^*. \end{aligned}$$

Thus, subtracting,

$$(\theta_T^* - \theta_\infty^*) = A_T^{-1}(b_T - b_\infty) - A_T^{-1}(A_T - A_\infty)\theta_\infty^*. \tag{8.1}$$

Because $\{f_\ell(\cdot)\}$ and $m^\dagger(\cdot)$ are Hölder (Assumption A2, and discussion immediately preceding it), so are $\langle f_i(\cdot), f_j(\cdot)\rangle$ and $\langle m^\dagger(\cdot), f_j(\cdot)\rangle$. Thus each entry of matrix A_T (resp. vector b_T) converges almost surely to its corresponding entry in A_∞ (resp. b_∞), by the ergodicity implied by Assumption A1, and the pointwise ergodic theorem. The almost

sure convergence of θ_T^* to θ_∞^* follows, after noting that A_∞ is invertible. Furthermore, also by Assumption A1, there are constants $\{\sigma_{ij}\}, \{\sigma_j\}$ such that

$$\sqrt{T}\Big((A_T)_{ij} - (A_\infty)_{ij}\Big) \Rightarrow N(0, \sigma_{ij}^2),$$
$$\sqrt{T}\Big((b_T)_j - (b_\infty)_j\Big) \Rightarrow N(0, \sigma_j^2).$$

Since arbitrary linear combinations of the $\{(A_T)_{ij}\}, \{(b_T)_j\}$ are time-averages of Hölder functions, it follows that $\sqrt{T}\{A_T - A_\infty, b_T - b_\infty\}$ converges in distribution to a Gaussian, by the Cramér-Wold Theorem [57]. Weak convergence of $\sqrt{T}(\theta_T^* - \theta_\infty^*)$ to a Gaussian follows from (8.1) by use of the Slutsky Lemma [57], since A_T converges almost surely to invertible A_∞. Matrix Σ cannot be identified explicitly in terms of only the $\{\sigma_{ij}\}, \{\sigma_i\}$ because of correlations between A_T and b_T. The almost sure bound on $\|\theta_T^* - \theta_\infty^*\|$ follows from (8.1) after multiplying by $(T/\log\log T)^{\frac{1}{2}}$, noting that $A_T \to A_\infty$ almost surely, and the almost sure bounds on $(T/\log\log T)^{\frac{1}{2}}\{\|A_T - A_\infty\|, \|b_T - b_\infty\|\}$, using Assumption A1. □

In what follows it is helpful to define

$$R_T^+ = (\theta_T^* - \theta_\infty^*)\big(\|\theta_T^*\| + \|\theta_\infty^*\| + 1\big),$$
$$G_T^+ = \mathcal{I}_T(m_\infty^*) - \mathcal{I}_\infty(m_\infty^*).$$

Lemma 8.2. *Let Assumption* A2 *hold. Then, assuming $x(0) \sim \mu$, there is constant $C > 0$ such that the excess risk R_T satisfies*

$$R_T \le C\|R_T^+\|.$$

Furthermore the generalization error satisfies

$$|G_T| \le 2C\|R_T^+\| + |G_T^+|.$$

Proof. For the bound on the excess risk we note that

$$\begin{aligned} R_T &= \mathcal{L}_\mu(m_T^*, m^\dagger) - \mathcal{L}_\mu(m_\infty^*, m^\dagger) \\ &= \int_{\mathbb{R}^{d_x}} \Big\langle (m_T^* - m_\infty^*)(x), (m_T^* + m_\infty^* - 2m^\dagger)(x) \Big\rangle \mu(dx) \\ &\le \Big(\int_{\mathbb{R}^{d_x}} \Big\| (m_T^* - m_\infty^*)(x) \Big\|^2 \mu(dx) \Big)^{\frac{1}{2}} \Big(\int_{\mathbb{R}^{d_x}} \Big\| (m_T^* + m_\infty^* - 2m^\dagger)(x) \Big\|^2 \mu(dx) \Big)^{\frac{1}{2}}. \end{aligned}$$

The first follows from the boundedness of the $\{f_\ell\}_{\ell=1}^p$ and $m^\dagger$, since the first term in the product above is bounded by a constant multiple of $\|\theta_T^* - \theta_\infty^*\|$ and the second term by a constant multiple of $\|\theta_T^*\| + \|\theta_\infty^*\| + \sup_{\mathcal{A}} \|m^\dagger\|$.

For the bound on the generalization error we note that

$$\begin{aligned} G_T &= \mathcal{I}_T(m_T^*) - \mathcal{I}_\infty(m_T^*) \\ &= \mathcal{I}_T(m_T^*) - \mathcal{I}_T(m_\infty^*) \\ &\qquad + \mathcal{I}_T(m_\infty^*) - \mathcal{I}_\infty(m_\infty^*) \\ &\qquad + \mathcal{I}_\infty(m_\infty^*) - \mathcal{I}_\infty(m_T^*) \\ &= \big(\mathcal{I}_T(m_T^*) - \mathcal{I}_T(m_\infty^*)\big) + G_T^+ - R_T. \end{aligned}$$

The third term in the final identity is the excess risk that we have just bounded; the first term may be bounded in the same manner that we bounded the excess risk, noting that integration with respect to μ is simply replaced by integration with respect to the empirical measure generated by the trajectory data which, by assumption, is confined to the attractor $\mathcal{A}$; the second term is simply G_T^+. Thus the result follows. □

Proof of Theorem 5.2. By Assumption A1, with choice of $\varphi(x) = \|m^\dagger(x) - m_\infty^*(x)\|^2$, $\sqrt{T}G_T^+$ converges in distribution to a scalar-valued centred Gaussian. By Lemma 8.1 and the Slutsky Lemma [57], $\sqrt{T}R_T^+$ converges in distribution to a centred Gaussian in $\mathbb{R}^p$. By the Cramer-Wold Theorem [57] $\sqrt{T}(R_T^+, G_T^+)$ converges in distribution to a centred Gaussian in $\mathbb{R}^{p+1}$.

The convergence in distribution results for excess risk R_T and generalization error $|G_T|$ then follow from Lemma 8.2, under Assumption A1. Furthermore, by Lemma 8.1, there is constant $C_1 > 0$ such that

$$\text{limsup}_{T\to\infty}\Big(\frac{T}{\log\log T}\Big)^{\frac{1}{2}}\|R_T^+\| \le C_1;$$

similarly, possibly by enlarging C_1, Assumption A1 gives

$$\text{limsup}_{T\to\infty}\Big(\frac{T}{\log\log T}\Big)^{\frac{1}{2}}|G_T^+| \le C_1.$$

The desired almost sure bound on $R_T + |G_T|$ follows from Lemma 8.2. □

8.2. **Proof of continuous-time ODE approximation theorem (general case).**

Proof. Recall equation (5.13). By Assumption A6, for any $\delta_o > 0$ there exist dimensions N_g and N_m and parameterizations $\theta_g \in \mathbb{R}^{N_g}, \theta_m \in \mathbb{R}^{N_m}$ such that for any $(x, y) \in B(0, 2\rho_T)$, and in the maximum norm,

$$\|g^\dagger(x,y) - f_2(x,y;\,\theta_g)\| \le \delta_o,$$
$$\|m^\dagger(x,y) - f_1(x,y;\,\theta_m)\| \le \delta_o.$$

By using these, we can rewrite (5.13) as

$$\begin{aligned}\dot{x} &= f_0(x) + f_1(x,y;\,\theta_m) + e_x(t),\\ \dot{y} &= f_2(x,y;\,\theta_g) + e_y(t),\end{aligned} \tag{8.2}$$

where, uniformly for $\big(x(0), y(0)\big) \in B(0, \rho_0)$,

$$\sup_{t\in[0,T]}\|e_y(t)\| \le \delta_o,$$
$$\sup_{t\in[0,T]}\|e_x(t)\| \le \delta_o.$$

By removing the bounded error terms, we obtain the approximate system:

$$\begin{aligned}\dot{x}_\delta &= f_0(x_\delta) + f_1(x_\delta, y_\delta;\,\theta_m),\\ \dot{y}_\delta &= f_2(x_\delta, y_\delta;\,\theta_g).\end{aligned} \tag{8.3}$$

Next, we obtain a stability bound on the discrepancy between the approximate system (8.3) and the true system (originally written as (2.8) and re-formulated as (8.2)). First, let $w = (x, y)$, $w_\delta = (x_\delta, y_\delta)$ and define F to be the concatenated right-hand-side of (8.3). Note that F is $L-$Lipschitz in the maximum norm on $B(0, 2\rho_T)$, for some L

related to the Lipschitz continuity of f_0, f_1, and f_2. Then we can write the true and approximate systems, respectively, as (using the maximum norm)

$$\dot{w} = F(w) + e_w(t), \tag{8.4a}$$

$$\dot{w}_\delta = F(w_\delta), \tag{8.4b}$$

where

$$\sup_{t\in[0,T]} \|e_w(t)\| \le \sup_{t\in[0,T]} \|e_y(t)\| + \sup_{t\in[0,T]} \|e_x(t)\| \le 2\delta_o.$$

Let $Pw = (x, y)$. Then, for any $t \in [0, T]$, and for all $Pw(0), Pw_\delta(0) \in B(0, \rho_0)$

$$\|w(t) - w_\delta(t)\| \le \Big\|w(0) - w_\delta(0)\Big\| + \int_0^t \Big\|e_w(s)\Big\| ds + \int_0^t \Big\|F\big(w(s)\big) - F(w_\delta(s))\Big\| ds.$$

This follows by writing (8.4) in integrated form, subtracting and taking norms. Using the facts that $\Big\|e_w(s)\Big\| \le 2\delta_o$ and F is $L-$Lipschitz we obtain, for $t \in [0, T]$,

$$\Big\|w(t) - w_\delta(t)\Big\| \le \Big\|w(0) - w_\delta(0)\Big\| + 2\delta_o T + L\int_0^t \Big\|w(s) - w_\delta(s)\Big\| ds.$$

By the integral form of the Gronwall Lemma, it follows that for all $t \in [0, T]$:

$$\Big\|w(t) - w_\delta(t)\Big\| \le \big[\|w(0) - w_\delta(0)\| + 2\delta_o T\big] \exp(Lt).$$

Thus,

$$\sup_{t\in[0,T]} \Big\|w(t) - w_\delta(t)\Big\| \le \big[\|w(0) - w_\delta(0)\| + 2\delta_o T\big] \exp(LT).$$

By choice of initial conditions and δ_o sufficiently small we can achieve a $\delta > 0$ approximation. Finally, we note that the approximate system (8.3) is a function of parameter $\theta_\delta = [\theta_m, \theta_g] \in \mathbb{R}^{N_\delta}$ with $n_\delta = N_g + N_m$. □

8.3. **Proof of continuous-time RNN approximation theorem (linear in observation).**

Proof. Recall equation (5.13). By approximation theory by means of two-layer feed-forward neural networks [34], for any $\delta_o > 0$ there exist embedding dimensions N_g and N_h and parameterizations

$$\theta_g = \{C_g \in \mathbb{R}^{d_y \times N_g},\ B_g \in \mathbb{R}^{N_g \times d_x},\ A_g \in \mathbb{R}^{N_g \times d_y},\ c_g \in \mathbb{R}^{N_g}\},$$
$$\theta_h = \{C_h \in \mathbb{R}^{d_x \times N_h},\ B_h \in \mathbb{R}^{N_h \times d_x},\ A_h \in \mathbb{R}^{N_h \times d_y},\ c_h \in \mathbb{R}^{N_h}\}$$

such that for any $(x, y) \in B(0, 2\rho_T)$, and in the maximum norm,

$$\|g^\dagger(x, y) - C_g\sigma(B_g x + A_g y + c_g)\| \le \delta_o,$$
$$\|h^\dagger(x, y) - C_h\sigma(B_h x + A_h y + c_h)\| \le \delta_o.$$

Without loss of generality we may assume that C_g and C_h have full rank since, if they do not, arbitrarily small changes can be made which restore full rank. By using these

parameterizations and embedding dimensions, we can rewrite (5.13) as

$$\begin{aligned} \dot{x} &= f_0(x) + m, \\ \dot{y} &= C_g\sigma(B_g x + A_g y + c_g) + e_y(t), \\ \dot{m} &= C_h\sigma(B_h x + A_h y + c_h) + e_{m^\dagger}(t). \end{aligned} \tag{8.5}$$

where, uniformly for $\big(x(0), y(0)\big) \in B(0, \rho_0)$,

$$\sup_{t\in[0,T]} \|e_y(t)\| \le \delta_o,$$
$$\sup_{t\in[0,T]} \|e_{m^\dagger}(t)\| \le \delta_o.$$

By removing the bounded error terms, we obtain the approximate system:

$$\begin{aligned} \dot{x}_\delta &= f_0(x_\delta) + m_\delta, \\ \dot{y}_\delta &= C_g\sigma(B_g x_\delta + A_g y_\delta + c_g), \\ \dot{m}_\delta &= C_h\sigma(B_h x_\delta + A_h y_\delta + c_h). \end{aligned} \tag{8.6}$$

Here $m_\delta(t)$ is initialized at $m^\dagger(x(0), y(0))$. Next, we obtain a stability bound on the discrepancy between the approximate system (8.6) and the true system (originally written as (2.8) and re-formulated as (8.5)). First, let $w = (x, y, m)$, $w_\delta = (x_\delta, y_\delta, m_\delta)$ and define F to be the concatenated right-hand-side of (8.6). Note that F is $L-$Lipschitz in the maximum norm, for some L related to the Lipschitz continuity of f_0, approximation parameterization θ_δ, and regularity of nonlinear activation function σ. Then we can write the true and approximate systems, respectively, as

$$\dot{w} = F(w) + e_w(t), \tag{8.7a}$$

$$\dot{w}_\delta = F(w_\delta), \tag{8.7b}$$

where

$$\sup_{t\in[0,T]} \|e_w(t)\| \le \sup_{t\in[0,T]} \|e_y(t)\| + \sup_{t\in[0,T]} \|e_{m^\dagger}(t)\| \le 2\delta_o.$$

Let $Pw = (x, y)$ and $P^\perp w = m$; recall that $P^\perp w(0)$ is defined in terms of $Pw(0)$. Then, for any $t \in [0, T]$, and for all $Pw(0), Pw_\delta(0) \in B(0, \rho_0)$

$$\|w(t) - w_\delta(t)\| \le \Big\|w(0) - w_\delta(0)\Big\| + \int_0^t \Big\|e_w(s)\Big\|ds + \int_0^t \Big\|F\big(w(s)\big) - F\big(w_\delta(s)\big)\Big\|ds.$$

By following the logic in Section 8.2, we have

$$\sup_{t\in[0,T]} \Big\|w(t) - w_\delta(t)\Big\| \le \big[\|w(0) - w_\delta(0)\| + 2\delta_o T\big]\exp(LT).$$

By choice of initial conditions and δ_o sufficiently small we can achieve a $\delta > 0$ approximation.

Finally, we note that the approximate system (8.6) may be written as a recurrent neural network of form (5.11) as follows. Consider the equations

$$\begin{aligned} \dot{x}_\delta &= f_0(x_\delta) + C_h n_\delta, \\ \dot{z}_\delta &= \sigma(B_g x_\delta + A_g C_g z_\delta + c_g), \\ \dot{n}_\delta &= \sigma(B_h x_\delta + A_h C_g z_\delta + c_h), \end{aligned} \tag{8.8}$$

where we have defined (z_δ, n_δ) in terms of (y_δ, m_δ) by $y_\delta = C_g z_\delta$ and $m_\delta = C_h n_\delta$. Now note that (8.8) is equivalent to (5.11), with recurrent state r_δ and parameters θ_δ given by:

- $r_\delta = \begin{bmatrix} z_\delta \\ n_\delta \end{bmatrix}$
- $C_\delta = \begin{bmatrix} 0 & C_h \end{bmatrix}$
- $B_\delta = \begin{bmatrix} B_g \\ B_h \end{bmatrix}$
- $A_\delta = \begin{bmatrix} A_g C_g & 0 \\ A_h C_g & 0 \end{bmatrix}$
- $c_\delta = \begin{bmatrix} c_g \\ c_h \end{bmatrix}$

Any initial condition on $(y_\delta(0), m_\delta(0))$ may be achieved by choice of initializations for $(z_\delta(0), n_\delta(0))$, since C_g, C_h are of full rank. □

8.4. **Random feature approximation.** Random feature methods lead to function approximation for mappings between Hilbert spaces $X \to Y$. They operate by constructing a probability space $(\Theta, \nu, \mathcal{F})$ with $\Theta \subseteq \mathbb{R}^p$ and feature map $\varphi : X \times \Theta \to Y$ such that $k(x, x') := \mathbb{E}^\vartheta[\varphi(x;\, \vartheta) \otimes \varphi(x';\, \vartheta)] \in \mathcal{L}(Y, Y)$ forms a reproducing kernel in an associated reproducing kernel Hilbert space (RKHS) K. Solutions are sought within $\text{span}\{\varphi(\cdot\,;\, \vartheta_l)\}_{l=1}^m$ where the $\{\vartheta_l\}$ are picked i.i.d. at random. Theory supporting the approach was established in finite dimensions by Rahimi and Recht [131]; the method was recently applied in the infinite dimensional setting in [113].

We now explain the precise random features setting adopted in Section 5, and hypothesis classes given by (5.1) and (5.6). We start with random feature functions $\varphi(\cdot\,;\, \vartheta) : \mathbb{R}^{d_x} \to \mathbb{R}$, with $\vartheta = [w, b]$,

$$\begin{aligned} w \in \mathbb{R}^{d_x} &\sim \mathcal{U}(-\omega, \omega), \\ b \in \mathbb{R} &\sim \mathcal{U}(-\beta, \beta), \\ \varphi(x;\, w, b) &:= \tanh(w^T x + b), \end{aligned} \tag{8.9}$$

and $\omega, \beta > 0$. We choose D i.i.d. draws of w, b, and stack the resulting random feature functions to form the map $\phi(x) : \mathbb{R}^{d_x} \to \mathbb{R}^D$ given by

$$\phi(x) := \begin{bmatrix} \varphi(x;\, w_1, b_w) & \dots & \varphi(x;\, w_D, b_D) \end{bmatrix}^T.$$

We define hypothesis class (5.1) by introducing matrix $C : \mathbb{R}^D \to \mathbb{R}^{d_x}$ and seeking approximation to model error in the form $m(x) = C\phi(x)$ by optimizing a least squares function over matrix C. This does not quite correspond to the random features model with $X = Y = \mathbb{R}^{d_x}$ because, when written as a linear span of vector fields mapping $\mathbb{R}^{d_x}$ into itself, the vector fields are not independent. Nonetheless we found this approach convenient in practice and employ it in our numerics.

To align with the random features model with $X = Y = \mathbb{R}^{d_x}$, we choose $D = d_x$ and draw p functions $\phi(\cdot)$, labelled as $\{f_\ell(\cdot)\}$ i.i.d. at random from the preceding construction, leading to hypothesis class (5.6): we then seek approximation to model error in the form $m(x) = \sum_{\ell=1}^p \theta_\ell f_\ell(x)$. We find this form of random features model most convenient to explain the learning theory perspective on model error.

8.5. **Derivation of Tikhonov-regularized linear inverse problem.** Here, we show that optimization of (5.2)

$$\mathcal{J}_T(C) = \frac{1}{2T}\int_0^T \left\|\dot{x}(t) - f_0(x) - C\phi\big(x(t)\big)\right\|^2 dt + \frac{\lambda}{2}\|C\|^2$$

reduces to a Tikhonov-regularized linear inverse problem. Since (5.5) is quadratic in C, there exists a unique global minimizer for C^* such that $\frac{\partial \mathcal{J}_T}{\partial C}(C^*) = 0$. The minimizer C^* satisfies:

$$(Z + \lambda I)C^T = Y,$$

where

$$Z = \overline{[\phi \otimes \phi]}_T,$$
$$Y = \overline{[\phi \otimes (\dot{x} - f_0)]}_T,$$

and

$$[A \otimes B]_t \coloneqq A(t)B^T(t),$$
$$\overline{A_T} \coloneqq \frac{1}{T}\int_0^T A(t)dt$$

for $A(t) \in \mathbb{R}^{m\times n}$, $B(t) \in \mathbb{R}^{m\times l}$.

To see this, observe that

$$\begin{aligned}\mathcal{J}_T(C) &= \frac{1}{2T}\int_0^T \|\dot{x}(t) - f_0\big(x(t)\big) - C\phi(x(t))\|^2 dt + \frac{\lambda}{2}\|C\|^2 \\ &= \frac{1}{2T}\int_0^T \|\dot{x}(t) - f_0\big(x(t)\big)\|^2, \\ &\qquad + \big\langle C\phi\big(x(t)\big), C\phi\big(x(t)\big)\big\rangle - 2\big\langle \dot{x}(t) - f_0\big(x(t)\big), C\phi\big(x(t)\big)\big\rangle dt + \frac{\lambda}{2}\langle C, C\rangle\end{aligned}$$

and

$$\frac{\partial \mathcal{J}_T(C)}{\partial C} = \frac{1}{2T}\int_0^T 2C\big[\phi\big(x(t)\big) \otimes \phi\big(x(t)\big)\big] - 2\big[\big(\dot{x}(t) - f_0\big(x(t)\big)\big) \otimes \phi\big(x(t)\big)\big]dt + \lambda C.$$

By setting the gradient to zero, we see that

$$C\Big[\frac{1}{T}\int_0^T \big[\phi\big(x(t)\big) \otimes \phi\big(x(t)\big)\big]dt + \lambda I\Big] = \frac{1}{T}\int_0^T \big[\big(\dot{x}(t) - f_0\big(x(t)\big)\big) \otimes \phi\big(x(t)\big)\big]dt.$$

Finally, we can take the transpose of both sides, apply our definitions of Y, Z, and use symmetry of Z to get

$$[Z + \lambda I]C^T = Y.$$

Acknowledgments

The authors are grateful to David Albers, Oliver Dunbar, Ian Melbourne, Fei Lu, Ivan D. Jimenez Rodriguez, and Yisong Yue for helpful discussions.

References

[1] Romeo Alexander and Dimitrios Giannakis, *Operator-theoretic framework for forecasting nonlinear time series with kernel analog techniques*, Phys. D **409** (2020), 132520, 24, DOI 10.1016/j.physd.2020.132520. MR4093838

[2] Ranjan Anantharaman, Yingbo Ma, Shashi Gowda, Chris Laughman, Viral Shah, Alan Edelman, and Chris Rackauckas, *Accelerating simulation of stiff nonlinear systems using continuous-time echo state networks*, `https://arxiv.org/abs/2010.04004v6`, 2020.

[3] Mark Asch, Marc Bocquet, and Maëlle Nodet, *Data assimilation*, Fundamentals of Algorithms, vol. 11, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2016. Methods, algorithms, and applications, DOI 10.1137/1.9781611974546.pt1. MR3602006

[4] Ibrahim Ayed, Emmanuel de Bézenac, Arthur Pajot, Julien Brajard, and Patrick Gallinari, *Learning dynamical systems from partial observations*, Second Workshop on Machine Learning and the Physical Sciences (NeurIPS 2019), Vancouver, Canada, February 2019.

[5] Yunhao Ba, Guangyuan Zhao, and Achuta Kadambi, *Blending diverse physical priors with neural networks*, `arXiv:1910.00201`, 2019.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, *Neural machine translation by jointly learning to align and translate*, `arXiv:1409.0473`, 2016.

[7] Wael Bahsoun, Ian Melbourne, and Marks Ruziboev, *Variance continuity for Lorenz flows*, Ann. Henri Poincaré **21** (2020), no. 6, 1873–1892, DOI 10.1007/s00023-020-00913-5. MR4100921

[8] Randall D. Beer, *On the dynamics of small continuous-time recurrent neural networks*, Adapt. Behav. **3** (1995), no. 4, 469–509, `http://journals.sagepub.com/doi/10.1177/105971239500300405`.

[9] A. Bensoussan, J.-L. Lions, and G. Papanicolaou, *Asymptotic analysis for periodic structures*, AMS Chelsea Publishing, Providence, RI, 2011. Corrected reprint of the 1978 original [MR0503330], DOI 10.1090/chel/374. MR2839402

[10] José Bento, Morteza Ibrahimi, and Andrea Montanari, *Information theoretic limits on learning stochastic differential equations*, 2011 IEEE International Symposium on Information Theory Proceedings, IEEE, 2011, pp. 855–859.

[11] Tom Beucler, Michael Pritchard, Stephan Rasp, Jordan Ott, Pierre Baldi, and Pierre Gentine, *Enforcing analytic constraints in neural networks emulating physical systems*, Phys. Rev. Lett. **126** (2021), no. 9, Paper No. 098302, 7, DOI 10.1103/physrevlett.126.098302. MR4231605

[12] Kaushik Bhattacharya, Bamdad Hosseini, Nikola B. Kovachki, and Andrew M. Stuart, *Model reduction and neural networks for parametric PDEs*, SMAI J. Comput. Math. **7** (2021), 121–157. MR4290514

[13] Marc Bocquet, Julien Brajard, Alberto Carrassi, and Laurent Bertino, *Bayesian inference of chaotic dynamics by merging data assimilation, machine learning and expectation-maximization*, Found. Data Sci. **2** (2020), no. 1, 55, `https://www.aimsciences.org/article/doi/10.3934/fods.2020004`.

[14] Francesco Borra, Angelo Vulpiani, and Massimo Cencini, *Effective models and predictability of chaotic multiscale systems via machine learning*, Phys. Rev. E **102** (2020), no. 5, 052203, 11, DOI 10.1103/physreve.102.052203. MR4189360

[15] Julien Brajard, Alberto Carrassi, Marc Bocquet, and Laurent Bertino, *Combining data assimilation and machine learning to emulate a dynamical model from sparse and noisy observations: a case study with the Lorenz 96 model*, J. Comput. Sci. **44** (2020), 101171, 11, DOI 10.1016/j.jocs.2020.101171. MR4117875

[16] Julien Brajard, Alberto Carrassi, Marc Bocquet, and Laurent Bertino, *Combining data assimilation and machine learning to infer unresolved scale parametrization*, Philos. Trans. Roy. Soc. A **379** (2021), no. 2194, Paper No. 20200086, 16, DOI 10.1098/rsta.2020.0086. MR4236154

[17] Leo Breiman, *Bagging predictors*, Mach. Learn., **24** (1996), no. 2, 123–140.

[18] N. D. Brenowitz and C. S. Bretherton, *Prognostic validation of a neural network unified physics parameterization*, Geophys. Res. Lett. **45**, no. 12, 6289–6298, 2018. `https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2018GL078510`.

[19] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz, *Discovering governing equations from data by sparse identification of nonlinear dynamical systems*, Proc. Natl. Acad. Sci. USA **113** (2016), no. 15, 3932–3937, DOI 10.1073/pnas.1517384113. MR3494081

[20] Dmitry Burov, Dimitrios Giannakis, Krithika Manohar, and Andrew Stuart, *Kernel analog forecasting: multiscale test problems*, Multiscale Model. Simul. **19** (2021), no. 2, 1011–1040, DOI 10.1137/20M1338289. MR4272899

[21] Kathleen Champion, Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton, *Data-driven discovery of coordinates and governing equations*, Proc. Natl. Acad. Sci. USA **116** (2019), no. 45, 22445–22451, DOI 10.1073/pnas.1906995116. MR4032517

[22] Bo Chang, Minmin Chen, Eldad Haber, and Ed H. Chi, *AntisymmetricRNN: A dynamical system view on recurrent neural networks*, arXiv:1902.09689, 2019.

[23] Ashesh Chattopadhyay, Pedram Hassanzadeh, and Devika Subramanian, *Data-driven predictions of a multiscale Lorenz 96 chaotic system using machine-learning methods: reservoir computing, artificial neural network, and long short-term memory network*, Nonlinear Process. Geophys. **27** (2020), no. 3, 373–389, https://npg.copernicus.org/articles/27/373/2020/.

[24] Ashesh Chattopadhyay, Adam Subel, and Pedram Hassanzadeh, *Data-driven super-parameterization using deep learning: experimentation with multiscale Lorenz 96 systems and transfer learning*. J. Adv. Model. Earth Sys. **12** (2020), no. 11, e2020MS002084, https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020MS002084.

[25] Yifan Chen, Bamdad Hosseini, Houman Owhadi, and Andrew M. Stuart, *Solving and learning nonlinear PDEs with Gaussian processes*, J. Comput. Phys. **447** (2021), Paper No. 110668, 29, DOI 10.1016/j.jcp.2021.110668. MR4311012

[26] Yuming Chen, Daniel Sanz-Alonso, and Rebecca Willett, *Autodifferentiable ensemble Kalman filters*, SIAM J. Math. Data Sci. **4** (2022), no. 2, 801–833, DOI 10.1137/21M1434477. MR4443668

[27] Oksana A. Chkrebtii, David A. Campbell, Ben Calderhead, and Mark A. Girolami, *Bayesian solution uncertainty quantification for differential equations*, Bayesian Anal. **11** (2016), no. 4, 1239–1267, DOI 10.1214/16-BA1017. MR3577378

[28] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio, *On the properties of neural machine translation: encoder-decoder approaches*, arXiv:1409.1259, 2014.

[29] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, *Learning phrase representations using rnn encoder-decoder for statistical machine translation*, arXiv:1406.1078, 2014.

[30] Alexandre J. Chorin and Fei Lu, *Discrete approach to stochastic parametrization and dimension reduction in nonlinear dynamics*, Proc. Natl. Acad. Sci. **112** (2015), no. 32, 9804–9809, https://www.pnas.org/content/112/32/9804.

[31] Alexandre J. Chorin, Ole H. Hald, and Raz Kupferman, *Optimal prediction and the Mori-Zwanzig representation of irreversible processes*, Proc. Natl. Acad. Sci. USA **97** (2000), no. 7, 2968–2973, DOI 10.1073/pnas.97.7.2968. MR1750741

[32] David Colton and Rainer Kress, *Inverse acoustic and electromagnetic scattering theory*, 3rd ed., Applied Mathematical Sciences, vol. 93, Springer, New York, 2013, DOI 10.1007/978-1-4614-4942-3. MR2986407

[33] Grace Wahba, *Smoothing noisy data with spline functions*, Numer. Math. **24** (1975), no. 5, 383–393, DOI 10.1007/BF01437407. MR405795

[34] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, Math. Control Signals Systems **2** (1989), no. 4, 303–314, DOI 10.1007/BF02551274. MR1015670

[35] Eric Darve, Jose Solomon, and Amirali Kia, *Computing generalized Langevin equations and generalized Fokker–Planck equations*, Proc. Natl. Acad. Sci. **106** (2009), no. 27, 10884–10889.

[36] Ronald A. DeVore and George G. Lorentz, *Constructive approximation*, Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], vol. 303, Springer-Verlag, Berlin, 1993. MR1261635

[37] Jonathan Dong, Ruben Ohana, Mushegh Rafayelyan, and Florent Krzakala, *Reservoir computing meets recurrent kernels and structured transforms*, arXiv:2006.07310, 2020.

[38] J. R. Dormand and P. J. Prince, *A family of embedded Runge-Kutta formulae*, J. Comput. Appl. Math. **6** (1980), no. 1, 19–26, DOI 10.1016/0771-050X(80)90013-3. MR568599

[39] Qiang Du, Yiqi Gu, Haizhao Yang, and Chao Zhou, *The discovery of dynamics via linear multistep methods and deep learning: error estimation*, SIAM J. Numer. Anal. **60** (2022), no. 4, 2014–2045, DOI 10.1137/21M140691X. MR4464472

[40] Karthik Duraisamy, Gianluca Iaccarino, and Heng Xiao, *Turbulence modeling in the age of data*, Annual review of fluid mechanics. Vol. 51, Annu. Rev. Fluid Mech., vol. 51, Annual Reviews, Palo Alto, CA, 2019, pp. 357–377. MR3965623

[41] Weinan E, Chao Ma, and Lei Wu, *A priori estimates of the population risk for two-layer neural networks*, Commun. Math. Sci. **17** (2019), no. 5, 1407–1425, DOI 10.4310/CMS.2019.v17.n5.a11. MR4044196

[42] N. Benjamin Erichson, Omri Azencot, Alejandro Queiruga, Liam Hodgkinson, and Michael W. Mahoney, *Lipschitz recurrent neural networks*, arXiv:2006.12070, 2020.

[43] Alban Farchi, Patrick Laloyaux, Massimo Bonavita, and Marc Bocquet, *Using machine learning to correct model error in data assimilation and forecast applications*, arXiv:2010.12605, 2021.

[44] Ibrahim Fatkullin and Eric Vanden-Eijnden, *A computational strategy for multiscale systems with applications to Lorenz 96 model*, J. Comput. Phys. **200** (2004), no. 2, 605–638, DOI 10.1016/j.jcp.2004.04.013. MR2095278

[45] Brian A. Freno and Kevin T. Carlberg, *Machine-learning error models for approximate solutions to parameterized systems of nonlinear equations*, Comput. Methods Appl. Mech. Engrg. **348** (2019), 250–296, DOI 10.1016/j.cma.2019.01.024. MR3911069

[46] Roger Frigola, Yutian Chen, and Carl Edward Rasmussen, *Variational Gaussian process state-space models*, Adv. Neural Inform. Process. Systems **27** (2014), `https://proceedings.neurips.cc/paper/2014/hash/139f0874f2ded2e41b0393c4ac5644f7-Abstract.html`.

[47] Ken-ichi Funahashi, *Approximation theory, dynamical systems, and recurrent neural networks*, Proceedings of the 5th International Colloquium on Differential Equations, Vol. 2 (Plovdiv, 1994), Sci. Cult. Technol. Publ., Singapore, 1995, pp. 51–58. MR1656699

[48] Daniel J. Gauthier, Erik Bollt, Aaron Griffith, and Wendson A. S. Barbosa, *Next generation reservoir computing*, Nat. Comm. **12** (2021), no. 1, 5564, ISSN 2041-1723.

[49] Faheem Gilani, Dimitrios Giannakis, and John Harlim, *Kernel-based prediction of non-Markovian time series*, Phys. D **418** (2021), Paper No. 132829, 16, DOI 10.1016/j.physd.2020.132829. MR4204505

[50] R. González-García, R. Rico-Martínez, and I. G. Kevrekidis, *Identification of distributed parameter systems: a neural net based approach*, Comput. Chem. Eng. **22** (1998), S965–S968, `https://linkinghub.elsevier.com/retrieve/pii/S0098135498001914`.

[51] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep learning*, Adaptive Computation and Machine Learning, MIT Press, Cambridge, MA, 2016. MR3617773

[52] Georg A. Gottwald and Sebastian Reich, *Combining machine learning and data assimilation to forecast dynamical systems from noisy partial observations*, Chaos **31** (2021), no. 10, Paper No. 101103, 8, DOI 10.1063/5.0066080. MR4323018

[53] Georg A. Gottwald and Sebastian Reich, *Supervised learning from noisy observations: combining machine-learning techniques with data assimilation*, Phys. D **423** (2021), Paper No. 132911, 15, DOI 10.1016/j.physd.2021.132911. MR4249157

[54] Ayoub Gouasmi, Eric J. Parish, and Karthik Duraisamy, *A priori estimation of memory effects in reduced-order models of nonlinear systems using the Mori-Zwanzig formalism*, Proc. A. **473** (2017), no. 2205, 20170385, 24, DOI 10.1098/rspa.2017.0385. MR3710327

[55] Wojciech W. Grabowski, *Coupling cloud processes with the large-scale dynamics using the cloud-resolving convection parameterization* (*CRCP*), J. Atmos. Sci. **58** (2001), no. 9, 978–997, `https://journals.ametsoc.org/view/journals/atsc/58/9/1520-0469_2001_058_0978_ccpwtl_2.0.co_2.xml`.

[56] Lyudmila Grigoryeva and Juan-Pablo Ortega, *Echo state networks are universal*, `arXiv:1806.00797`, 2018.

[57] Geoffrey R. Grimmett and David R. Stirzaker, *Probability and random processes*, Oxford University Press, Oxford, 2020. Fourth edition [of 0667520]. MR4229142

[58] Abhinav Gupta and Pierre F. J. Lermusiaux, *Neural closure models for dynamical systems*, Proc. A. **477** (2021), no. 2252, Paper No. 20201004, 29. MR4312509

[59] Eldad Haber and Lars Ruthotto, *Stable architectures for deep neural networks*, Inverse Problems **34** (2018), no. 1, 014004, 22, DOI 10.1088/1361-6420/aa9a90. MR3742361

[60] Franz Hamilton, Alun L. Lloyd, and Kevin B. Flores, *Hybrid modeling and prediction of dynamical systems*, PLOS Comput. Biol. **13** (2017), no. 7, e1005655, `https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005655`.

[61] Boumediene Hamzi and Houman Owhadi, *Learning dynamical systems from data: a simple cross-validation perspective, part I: Parametric kernel flows*, Phys. D **421** (2021), Paper No. 132817, 10, DOI 10.1016/j.physd.2020.132817. MR4233447

[62] John Harlim, Shixiao W. Jiang, Senwei Liang, and Haizhao Yang, *Machine learning for prediction with missing dynamics*, J. Comput. Phys. **428** (2021), Paper No. 109922, 22, DOI 10.1016/j.jcp.2020.109922. MR4199367

[63] Fabrício P. Härter and Haroldo Fraga de Campos Velho, *Data assimilation procedure by recurrent neural network*, Eng. Appl. Comput. Fluid Mech. **6** (2012), 224–233, `https://doi.org/10.1080/19942060.2012.11015417`.

[64] Simon Haykin, José C. Príncipe, Terrence J. Sejnowski, and John McWhirter (eds.), *New directions in statistical signal processing: from systems to brain*, Neural Information Processing Series, MIT Press, Cambridge, MA, 2007. MR2590576

[65] Dan Hendrycks and Kevin Gimpel, *Gaussian error linear units* (*gelus*), https://arxiv.org/abs/1606.08415, 2016.
[66] Carmen Hijón, Pep Español, Eric Vanden-Eijnden, and Rafael Delgado-Buscalioni, *Mori–Zwanzig formalism as a practical computational tool*, Faraday Discuss. **144** (2010), 301–322.
[67] Sepp Hochreiter and Jürgen Schmidhuber. *Long short-term memory*, Neural Comput. **9** (1997), 1735–1780.
[68] Mark Holland and Ian Melbourne, *Central limit theorems and invariance principles for Lorenz attractors*, J. Lond. Math. Soc. (2) **76** (2007), no. 2, 345–364, DOI 10.1112/jlms/jdm060. MR2363420
[69] Herbert Jaeger, *The "echo state" approach to analysing and training recurrent neural networks-with an erratum note'*, German National Research Center for Information Technology GMD Technical Report, Bonn, Germany, January 2001, p. 148.
[70] Xiaowei Jia, Jared Willard, Anuj Karpatne, Jordan S. Read, Jacob A. Zwart, Michael Steinbach, and Vipin Kumar, *Physics-guided machine learning for scientific discovery: an application in simulating lake temperature profiles*, ACM/IMS Trans. Data Sci. **2** (2021), no. 3, 20:1–20:26, https://doi.org/10.1145/3447814.
[71] Shixiao W. Jiang and John Harlim, *Modeling of missing dynamical systems: deriving parametric models using a nonparametric framework*, Res. Math. Sci. **7** (2020), no. 3, Paper No. 16, 25, DOI 10.1007/s40687-020-00217-4. MR4123391
[72] Kadierdan Kaheman, Eurika Kaiser, Benjamin Strom, J. Nathan Kutz, and Steven L. Brunton, *Learning discrepancy models from experimental data*, arXiv:1909.08574, 2019.
[73] Kadierdan Kaheman, Steven L. Brunton, and J. Nathan Kutz, *Automatic differentiation to simultaneously identify nonlinear dynamics and extract noise probability distributions from data*, Mach. Learn. Sci. Technol. **3** (2022), no. 1, 015031, https://doi.org/10.1088/2632-2153/ac567a.
[74] Jari Kaipio and Erkki Somersalo, *Statistical and computational inverse problems*, Applied Mathematical Sciences, vol. 160, Springer-Verlag, New York, 2005. MR2102218
[75] J. Nagoor Kani and Ahmed H. Elsheikh, *DR-RNN: a deep residual recurrent neural network for model reduction*, arXiv:1709.00939, 2017.
[76] K. Kashinath, M. Mustafa, A. Albert, and et al., *Physics-informed machine learning: case studies for weather and climate modelling*, Philos. Trans. Roy. Soc. A **379** (2021), no. 2194, Paper No. 20200093, 36, DOI 10.1098/rsta.2020.0093. MR4236152
[77] Rachael T. Keller and Qiang Du, *Discovery of dynamics using linear multistep methods*, SIAM J. Numer. Anal. **59** (2021), no. 1, 429–455, DOI 10.1137/19M130981X. MR4216874
[78] Felix P. Kemeth, Tom Bertalan, Nikolaos Evangelou, Tianqi Cui, Saurabh Malani, and Ioannis G. Kevrekidis, *Initializing LSTM internal states via manifold learning*, Chaos **31** (2021), no. 9, Paper No. 093111, 14, DOI 10.1063/5.0055371. MR4311431
[79] Marat F. Khairoutdinov and David A. Randall, *A cloud resolving model as a cloud parameterization in the NCAR community climate system model: preliminary results*, Geophys. Res. Lett. **28** (2001), no. 18, 3617–3620, https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2001GL013552.
[80] Diederik P. Kingma and Jimmy Ba, *Adam: A method for stochastic optimization*, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, Conference Track Proceedings, arXiv:1412.6980, 2015.
[81] Juš Kocijan, *Modelling and control of dynamic systems using Gaussian process models*, Advances in Industrial Control, Springer, Cham, 2016, DOI 10.1007/978-3-319-21021-6. MR3410992
[82] Milan Korda, Mihai Putinar, and Igor Mezić, *Data-driven spectral analysis of the Koopman operator*, Appl. Comput. Harmon. Anal. **48** (2020), no. 2, 599–629, DOI 10.1016/j.acha.2018.08.002. MR4047538
[83] K. Krischer, R. Rico-Martínez, I. G. Kevrekidis, H. H. Rotermund, G. Ertl, and J. L. Hudson, *Model identification of a spatiotemporally varying catalytic reaction*, AIChE J. **39** (1993), no. 1, 89–98, January 1993, http://doi.wiley.com/10.1002/aic.690390110.
[84] S. Kullback and R. A. Leibler, *On information and sufficiency*, Ann. Math. Statistics **22** (1951), 79–86, DOI 10.1214/aoms/1177729694. MR39968
[85] Yury A. Kutoyants, *Statistical inference for ergodic diffusion processes*, Springer Series in Statistics, Springer-Verlag London, Ltd., London, 2004, DOI 10.1007/978-1-4471-3866-2. MR2144185
[86] I. E. Lagaris, A. Likas, D. I. Fotiadis, and C. V. Massalas, *A hardware implementable non-linear method for the solution of ordinary, partial and integrodifferential equations*, Mathematical methods in scattering theory and biomedical technology (Metsovo, 1997), Pitman Res. Notes Math. Ser., vol. 390, Longman, Harlow, 1998, pp. 110–126. MR1795139

[87] Kody Law, Abhishek Shukla, and Andrew Stuart, *Analysis of the 3DVAR filter for the partially observed Lorenz'63 model*, Discrete Contin. Dyn. Syst. **34** (2014), no. 3, 1061–1078, DOI 10.3934/dcds.2014.34.1061. MR3094561

[88] Kody Law, Andrew Stuart, and Konstantinos Zygalakis, *Data assimilation*, Texts in Applied Mathematics, vol. 62, Springer, Cham, 2015. A mathematical introduction, DOI 10.1007/978-3-319-20325-6. MR3363508

[89] Youming Lei, Jian Hu, and Jianpeng Ding, *A hybrid model based on deep LSTM for predicting high-dimensional chaotic systems*, `arXiv:2002.00799`, 2020.

[90] Zhen Li, Hee Sun Lee, Eric Darve, and George Em Karniadakis, *Computing the non-Markovian coarse-grained interactions derived from the Mori–Zwanzig formalism in molecular systems: application to polymer melts*, J. Chem. Phys. **146**, no. 1, 014104.

[91] Zhong Li, Jiequn Han, Weinan E, and Qianxiao Li, *On the curse of memory in recurrent neural networks: approximation and optimization analysis*, `arXiv:2009.07799`, 2020.

[92] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar, *Fourier neural operator for parametric partial differential equations*, `arXiv:2010.08895`, 2021.

[93] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar, *Markov neural operators for learning chaotic systems*, `arXiv:2106.06898`, 2021.

[94] Kevin K. Lin and Fei Lu, *Data-driven model reduction, Wiener projections, and the Koopman-Mori-Zwanzig formalism*, J. Comput. Phys. **424** (2021), Paper No. 109864, 33, DOI 10.1016/j.jcp.2020.109864. MR4165611

[95] Ori Linial, Neta Ravid, Danny Eytan, and Uri Shalit, *Generative ODE modeling with known unknowns*, Proceedings of the Conference on Health, Inference, and Learning, CHIL '21, New York, NY, USA, Association for Computing Machinery, April 2021, pp. 79–94, `https://doi.org/10.1145/3450439.3451866`.

[96] E. Lorenz, *Predictability—a problem partly solved*, Proc. Seminar on Predictability, Reading, UK, ECMWF, 1996. `https://ci.nii.ac.jp/naid/10015392260/en/`.

[97] Edward N. Lorenz, *Deterministic nonperiodic flow*, J. Atmospheric Sci. **20** (1963), no. 2, 130–141, DOI 10.1175/1520-0469(1963)020⟨0130:DNF⟩2.0.CO;2. MR4021434

[98] Robert J. Lovelett, José L. Avalos, and Ioannis G. Kevrekidis, *Partial observations and conservation laws: gray-box modeling in biotechnology and optogenetics*, Ind. Eng. Chem. Res. **59** (2020), no. 6, 2611–2620, `https://doi.org/10.1021/acs.iecr.9b04507`.

[99] Fei Lu, *Data-driven model reduction for stochastic Burgers equations*, Entropy **22** (2020), no. 12, Paper No. 1360, 22, DOI 10.3390/e22121360. MR4222910

[100] Fei Lu, Kevin K. Lin, and Alexandre J. Chorin, *Comparison of continuous and discrete-time data-based modeling for hypoelliptic systems*, Commun. Appl. Math. Comput. Sci. **11** (2016), no. 2, 187–216, DOI 10.2140/camcos.2016.11.187. MR3606402

[101] Fei Lu, Kevin K. Lin, and Alexandre J. Chorin, *Data-based stochastic model reduction for the Kuramoto-Sivashinsky equation*, Phys. D **340** (2017), 46–57, DOI 10.1016/j.physd.2016.09.007. MR3582762

[102] Lu Lu, Pengzhan Jin, and George Em Karniadakis, *DeepONet: learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators*, `arXiv:1910.03193`, 2020.

[103] Zhixin Lu, Brian R. Hunt, and Edward Ott, *Attractor reconstruction by machine learning*, Chaos **28** (2018), no. 6, 061104, 9, DOI 10.1063/1.5039508. MR3816936

[104] Mantas Lukoševičius and Herbert Jaeger, *Reservoir computing approaches to recurrent neural network training*, Comput. Sci. Rev. **3** (2009), no. 3, 127–149, August 2009, `https://www.sciencedirect.com/science/article/pii/S1574013709000173`.

[105] Chao Ma, Jianchun Wang, and Weinan E, *Model reduction with memory and the machine learning of dynamical systems*, Commun. Comput. Phys. **25** (2019), no. 4, `http://www.global-sci.com/intro/article_detail/cicp/12885.html`.

[106] Romit Maulik, Bethany Lusch, and Prasanna Balaprakash, *Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders*, Phys. Fluids **33** (2021), no. 3, 037106, March 2021, `https://aip.scitation.org/doi/abs/10.1063/5.0039986`.

[107] Kevin McGoff, Sayan Mukherjee, Andrew Nobel, and Natesh Pillai, *Consistency of maximum likelihood estimation for some dynamical systems*, Ann. Statist. **43** (2015), no. 1, 1–29, DOI 10.1214/14-AOS1259. MR3285598

[108] Xiao-Li Meng and David van Dyk, *The EM algorithm—an old folk-song sung to a fast new tune*, J. Roy. Statist. Soc. Ser. B **59** (1997), no. 3, 511–567, DOI 10.1111/1467-9868.00082. With discussion and a reply by the authors. MR1452025

[109] Andrew C. Miller, Nicholas J. Foti, and Emily Fox, *Learning insulin-glucose dynamics in the wild*, arXiv:2008.02852, 2020.

[110] Andrew C. Miller, Nicholas J. Foti, and Emily B. Fox, *Breiman's two cultures: you don't have to choose sides*, arXiv:2104.12219, 2021.

[111] Jonas Mockus, *Bayesian approach to global optimization*, Mathematics and its Applications (Soviet Series), vol. 37, Kluwer Academic Publishers Group, Dordrecht, 1989. Theory and applications; With a 5.25-inch IBM PC floppy disk, DOI 10.1007/978-94-009-0909-0. MR1111483

[112] Kumpati S. Narendra and Kannan Parthasarathy, *Neural networks and dynamical systems*, Internat. J. Approx. Reason. **6** (1992), no. 2, 109–131, https://www.sciencedirect.com/science/article/pii/0888613X9290014Q.

[113] Nicholas H. Nelsen and Andrew M. Stuart, *The random feature model for input-output maps between Banach spaces*, SIAM J. Sci. Comput. **43** (2021), no. 5, A3212–A3243, DOI 10.1137/20M133957X. MR4313849

[114] Duong Nguyen, Said Ouala, Lucas Drumetz, and Ronan Fablet, *EM-like learning chaotic dynamics from noisy and partial observations*, arXiv:1903.10335, 2019.

[115] Murphy Yuezhen Niu, Lior Horesh, and Isaac Chuang, *Recurrent neural networks in the eye of differential equations*, arXiv:1904.12933, 2019.

[116] Fernando Nogueira, *Bayesian optimization: open source constrained global optimization tool for Python*, 2014. https://github.com/fmfn/BayesianOptimization.

[117] Paul A. O'Gorman and John G. Dwyer, *Using machine learning to parameterize moist convection: potential for modeling of climate, climate change, and extreme events*, J. Adv. Model. Earth Syst. **10** (2018), no. 10, 2548–2563, https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2018MS001351.

[118] S. Ouala, D. Nguyen, L. Drumetz, B. Chapron, A. Pascual, F. Collard, L. Gaultier, and R. Fablet, *Learning latent dynamics for partially observed chaotic systems*, Chaos **30** (2020), no. 10, 103121, 17, DOI 10.1063/5.0019309. MR4164381

[119] Eric J. Parish and Karthik Duraisamy, *A dynamic subgrid scale model for large eddy simulations based on the Mori-Zwanzig formalism*, J. Comput. Phys. **349** (2017), 154–175, DOI 10.1016/j.jcp.2017.07.053. MR3695240

[120] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott, *Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach*, Phys. Rev. Lett. **120** (2018), no. 2, 024102, https://link.aps.org/doi/10.1103/PhysRevLett.120.024102.

[121] Jaideep Pathak, Alexander Wikner, Rebeckah Fussell, Sarthak Chandra, Brian R. Hunt, Michelle Girvan, and Edward Ott, *Hybrid forecasting of chaotic processes: using machine learning in conjunction with a knowledge-based model*, Chaos **28** (2018), no. 4, 041101, 9, DOI 10.1063/1.5028373. MR3787327

[122] Grigorios A. Pavliotis and Andrew M. Stuart, *Multiscale methods*, Texts in Applied Mathematics, vol. 53, Springer, New York, 2008. Averaging and homogenization. MR2382139

[123] Matthew Plumlee, *Bayesian calibration of inexact computer models*, J. Amer. Statist. Assoc. **112** (2017), no. 519, 1274–1285, DOI 10.1080/01621459.2016.1211016. MR3735376

[124] Matthew Plumlee and V. Roshan Joseph, *Orthogonal Gaussian process models*, Statist. Sinica **28** (2018), no. 2, 601–619. MR3791080

[125] Manuel Pulido, Pierre Tandeo, Marc Bocquet, Alberto Carrassi, and Magdalena Lucini, *Stochastic parameterization identification using ensemble Kalman filtering combined with maximum likelihood methods*, Tellus A Dyn. Meteorology Oceanogr. **70** (2018), no. 1, 1–17.

[126] Ryan Pyle, Nikola Jovanovic, Devika Subramanian, Krishna V. Palem, and Ankit B. Patel, *Domain-driven models yield better predictions at lower cost than reservoir computers in Lorenz systems*, Philos. Trans. Roy. Soc. A **379** (2021), no. 2194, Paper No. 20200246, 22, DOI 10.1103/physrevlett.120.024102. MR4236153

[127] Zhaozhi Qian, William R. Zame, Lucas M. Fleuren, Paul Elbers, and Mihaela van der Schaar, *Integrating expert ODEs into neural ODEs: pharmacology and disease progression*, arXiv:2106.02875, 2021.

[128] Alejandro F. Queiruga, N. Benjamin Erichson, Dane Taylor, and Michael W. Mahoney, *Continuous-in-depth neural networks*, arXiv:2008.02389, 2020.

[129] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman, *Universal differential equations for scientific machine learning*, arXiv:2001.04385, 2020.

[130] Christopher Rackauckas, Roshan Sharma, and Bernt Lie, *Hybrid mechanistic + neural model of laboratory helicopter*, pages 264–271, March 2021, pp. 264–271, https://ep.liu.se/en/conference-article.aspx?series=ecp&issue=176&Article_No=37.

[131] Ali Rahimi and Benjamin Recht, *Random features for large-scale kernel machines*, Adv. Neural Inform. Process. Syst. **20** (2008), Curran Associates, Inc., https://proceedings.neurips.cc/paper/2007/file/013a006f03dbc5392effeb8f18fda755-Paper.pdf.

[132] Ali Rahimi and Benjamin Recht, *Uniform approximation of functions with random bases*, 2008 46th Annual Allerton Conference on Communication, Control, and Computing, IEEE, 2008, pp. 555–561.

[133] Ali Rahimi and Benjamin Recht, *Weighted sums of random kitchen sinks: replacing minimization with randomization in learning*, Nips, Citeseer, 2008, pp. 1313–1320.

[134] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, J. Comput. Phys. **378** (2019), 686–707, DOI 10.1016/j.jcp.2018.10.045. MR3881695

[135] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, J. Comput. Phys. **378** (2019), 686–707, DOI 10.1016/j.jcp.2018.10.045. MR3881695

[136] Carl Edward Rasmussen and Christopher K. I. Williams, *Gaussian processes for machine learning*, Adaptive Computation and Machine Learning, MIT Press, Cambridge, MA, 2006. MR2514435

[137] Stephan Rasp, Michael S. Pritchard, and Pierre Gentine, *Deep learning to represent subgrid processes in climate models*, Proc. Natl. Acad. Sci. USA **115** (2018), no. 39, 9684–9689, https://www.pnas.org/content/115/39/9684.

[138] Sebastian Reich and Colin Cotter, *Probabilistic forecasting and Bayesian data assimilation*, Cambridge University Press, New York, 2015, DOI 10.1017/CBO9781107706804. MR3242790

[139] R. Rico-Martines, I. G. Kevrekidis, M. C. Kube, and J. L. Hudson, *Discrete- vs. continuous-time nonlinear signal processing: Attractors, transitions and parallel implementation issues*, 1993 American Control Conference, June 1993, pp. 1475–1479, San Francisco, CA, USA, IEEE, ISBN 978-0-7803-0860-2, https://ieeexplore.ieee.org/document/4793116/.

[140] R. Rico-Martínez, K. Krischer, I. G. Kevrekidis, M. C. Kube, and J. L. Hudson. *Discrete- vs. continuous-time nonlinear signal processing of Cu electrodissolution data*, Chemical Engineering Communications, 118 (1): 25–48, November 1992. https://www.tandfonline.com/doi/full/10.1080/00986449208936084.

[141] R. Rico-Martinez, J. S. Anderson, and I. G. Kevrekidis, *Continuous-time nonlinear signal processing: a neural network based approach for gray box identification*, Proceedings of IEEE Workshop on Neural Networks for Signal Processing, Ermioni, Greece, IEEE, 1994, pp. 596–605. ISBN 978-0-7803-2026-0, http://ieeexplore.ieee.org/document/366006/.

[142] Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud, *Latent ODEs for irregularly-sampled time series*, arXiv:1907.03907, 2019.

[143] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams, *Learning representations by back-propagating errors*, Nature **323** (1986), no. 6088, 533–536.

[144] Matteo Saveriano, Yuchao Yin, Pietro Falco, and Dongheui Lee, *Data-efficient control policy search using residual dynamics learning*, 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), September 2017, pp. 4709–4715, ISSN 2153-0866

[145] Hayden Schaeffer, Giang Tran, and Rachel Ward, *Learning dynamical systems and bifurcation via group sparsity*, Preprint, arXiv:1709.01558, 2017.

[146] Hayden Schaeffer, Giang Tran, and Rachel Ward, *Extracting sparse high-dimensional dynamics from limited data*, SIAM J. Appl. Math. **78** (2018), no. 6, 3279–3295, DOI 10.1137/18M116798X. MR3885764

[147] Hayden Schaeffer, Giang Tran, Rachel Ward, and Linan Zhang, *Extracting structured dynamical systems using sparse optimization with very few samples*, Multiscale Model. Simul. **18** (2020), no. 4, 1435–1461, DOI 10.1137/18M1194730. MR4164069

[148] Anton Maximilian Schäfer and Hans-Georg Zimmermann, *Recurrent neural networks are universal approximators*, International Journal of Neural Systems **17**, no. 4, 253–263, 2007.

[149] Robert E. Schapire, *The strength of weak learnability*, Mach. Learn. **5** (1990), no. 2, 197–227.

[150] Tapio Schneider, Andrew M. Stuart, and Jin-Long Wu, *Learning stochastic closures using ensemble Kalman inversion*, Trans. Math. Appl. **5** (2021), no. 1, Paper No. tnab003, 31, DOI 10.1093/imatrm/tnab003. MR4356471

[151] Skipper Seabold and Josef Perktold, *statsmodels: econometric and statistical modeling with python*, 9th Python in Science Conference, 2010.

[152] Alex Sherstinsky, *Fundamentals of recurrent neural network* (*RNN*) *and long short-term memory* (*LSTM*) *network*, Phys. D **404** (2020), 132306, 28, DOI 10.1016/j.physd.2019.132306. MR4057560

[153] Guanya Shi, Xichen Shi, Michael O'Connell, Rose Yu, Kamyar Azizzadenesheli, Animashree Anandkumar, Yisong Yue, and Soon-Jo Chung, *Neural Lander: stable drone landing control using learned dynamics*, arXiv:1811.08027, 2018.

[154] Jonathan D. Smith, Kamyar Azizzadenesheli, and Zachary E. Ross, *EikoNet: solving the eikonal equation with deep neural networks*, IEEE Transactions on Geoscience and Remote Sensing, 2020, pp. 1–12.

[155] Peter D. Sottile, David Albers, Peter E. DeWitt, Seth Russell, J. N. Stroh, David P. Kao, Bonnie Adrian, Matthew E. Levine, Ryan Mooney, Lenny Larchick, Jean S. Kutner, Matthew K. Wynia, Jeffrey J. Glasheen, and Tellen D. Bennett, *Real-time electronic health record mortality prediction during the COVID-19 pandemic: A prospective cohort study*, medRxiv, Cold Spring Harbor Laboratory Press, January 2021, p. 2021.01.14.21249793, https://www.medrxiv.org/content/10.1101/2021.01.14.21249793v1.

[156] Langxuan Su and Sayan Mukherjee, *A large deviation approach to posterior consistency in dynamical systems*, arXiv:2106.06894, 2021.

[157] Floris Takens, *Detecting strange attractors in turbulence*, Dynamical systems and turbulence, Warwick 1980 (Coventry, 1979/1980), Lecture Notes in Math., vol. 898, Springer, Berlin-New York, 1981, pp. 366–381. MR654900

[158] Zhihong Tan, Colleen M. Kaul, Kyle G. Pressel, Yair Cohen, Tapio Schneider, and João Teixeira, *An extended eddy-diffusivity mass-flux scheme for unified representation of subgrid-scale turbulence and convection*, J. Adv. Model. Earth Sys. **10** (2010), no. 3, 770–800.

[159] Giang Tran and Rachel Ward, *Exact recovery of chaotic systems from highly corrupted data*, Multiscale Model. Simul. **15** (2017), no. 3, 1108–1129, DOI 10.1137/16M1086637. MR3678496

[160] Jonathan H. Tu, Clarence W. Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz, *On dynamic mode decomposition: theory and applications*, J. Comput. Dyn. **1** (2014), no. 2, 391–421, DOI 10.3934/jcd.2014.1.391. MR3415261

[161] Eric Vanden-Eijnden, *Numerical techniques for multi-scale dynamical systems with stochastic effects*, Commun. Math. Sci. **1** (2003), no. 2, 385–391. MR1980483

[162] Vladimir N. Vapnik, *The nature of statistical learning theory*, Springer-Verlag, New York, 1995, DOI 10.1007/978-1-4757-2440-0. MR1367965

[163] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, Ilhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, and Paul van Mulbregt, *SciPy 1.0: fundamental algorithms for scientific computing in Python*, Nat. Methods **17** (2020), no. 3, 261–272, https://www.nature.com/articles/s41592-019-0686-2.

[164] P. R. Vlachas, J. Pathak, B. R. Hunt, T. P. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos, *Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics*, Neural Netw. **126** (2020), 191–217, https://linkinghub.elsevier.com/retrieve/pii/S0893608020300708.

[165] Jack Wang, Aaron Hertzmann, and David J. Fleet, *Gaussian process dynamical models*, Adv. Neural Inform. Process. Syst. **18** (2005), https://papers.nips.cc/paper/2005/hash/ccd45007df44dd0f12098f486e7e8a0f-Abstract.html.

[166] Qian Wang, Nicolò Ripamonti, and Jan S. Hesthaven, *Recurrent neural network closure of parametric POD-Galerkin reduced-order models based on the Mori-Zwanzig formalism*, J. Comput. Phys. **410** (2020), 109402, 32, DOI 10.1016/j.jcp.2020.109402. MR4079581

[167] Peter A. G. Watson, *Applying machine learning to improve simulations of a chaotic dynamical system using empirical error correction*, arXiv:1904.10904, 2019.

[168] Alexander Wikner, Jaideep Pathak, Brian Hunt, Michelle Girvan, Troy Arcomano, Istvan Szunyogh, Andrew Pomerance, and Edward Ott, *Combining machine learning with knowledge-based modeling for scalable forecasting and subgrid-scale closure of large, complex, spatiotemporal systems*, Chaos **30** (2020), no. 5, 053111, 16, DOI 10.1063/5.0005541. MR4094693

[169] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar, *Integrating scientific knowledge with machine learning for engineering and environmental systems*, arXiv:2003.04919, 2021.

[170] J. A. Wilson and L. F. M. Zorzetto, *A generalised approach to process state estimation using hybrid artificial neural network/mechanistic models*, Comput. Chem. Eng. **21** (1997), no. 9, 951–963, http://linkinghub.elsevier.com/retrieve/pii/S0098135496003365.
[171] Armand Wirgin, *The inverse crime*, arXiv:math-ph/0401050, 2004.
[172] David H. Wolpert, *Stacked generalization*, Neural Netw. **5** (1992), no. 2, 241–259, https://www.sciencedirect.com/science/article/pii/S0893608005800231.
[173] Zhong Yi Wan, Petr Karnakov, Petros Koumoutsakos, and Themistoklis P. Sapsis, *Bubbles in turbulent flows: data-driven, kinematic models with history terms*, Int. J. Multiph. Flow **129** (2020), 103286, 11, DOI 10.1016/j.ijmultiphaseflow.2020.103286. MR4096711
[174] Yuan Yin, Vincent Le Guen, Jérémie Dona, Emmanuel de Bézenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari, *Augmenting physical models with deep networks for complex dynamics forecasting*, J. Stat. Mech. Theory Exp. **12** (2021), Paper No. 124012, 30, DOI 10.1088/1742-5468/ac3ae5. MR4412840
[175] He Zhang, John Harlim, Xiantao Li, *Estimating linear response statistics using orthogonal polynomials: an RKHS formulation*, Found. Data Sci. **2** (2020), no. 4, 443–485, http://aimsciences.org//article/doi/10.3934/fods.2020021.
[176] He Zhang, John Harlim, and Xiantao Li, *Error bounds of the invariant statistics in machine learning of ergodic Itô diffusions*, Phys. D **427** (2021), Paper No. 133022, 28, DOI 10.1016/j.physd.2021.133022. MR4311551
[177] Jian Zhu and Masafumi Kamachi, *An adaptive variational method for data assimilation with imperfect models*, Tellus A Dyn. Meteorology Oceanogr. **52**, (2000), no. 3, 265–279, https://doi.org/10.3402/tellusa.v52i3.12265.
[178] Yuanran Zhu, Jason M. Dominy, and Daniele Venturi, *On the estimation of the Mori-Zwanzig memory integral*, J. Math. Phys. **59** (2018), no. 10, 103501, 39, DOI 10.1063/1.5003467. MR3856552

Department. of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, California 91125
Email address: mlevine@caltech.edu

Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, California 91125
Email address: astuart@caltech.edu