Journal of Computational Physics 513 (2024) 113157

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Learning about structural errors in models of complex dynamical systems

Jin-Long Wu [a,*,1], Matthew E. Levine [b,1], Tapio Schneider [c], Andrew Stuart [c]

[a] University of Wisconsin-Madison, 1513 University Ave, Madison, WI 53706, USA
[b] Eric and Wendy Schmidt Center, Broad Institute of MIT and Harvard, Cambridge, 02142, USA
[c] California Institute of Technology, 1200 E California Blvd, Pasadena, CA 91125, USA

ARTICLE INFO

Keywords:
Model error
Dynamical system
Machine learning
Inverse problem
Stochastic model
Non-local model

ABSTRACT

Complex dynamical systems are notoriously difficult to model because some degrees of freedom (e.g., small scales) may be computationally unresolvable or are incompletely understood, yet they are dynamically important. For example, the small scales of cloud dynamics and droplet formation are crucial for controlling climate, yet are unresolvable in global climate models. Semi-empirical closure models for the effects of unresolved degrees of freedom often exist and encode important domain-specific knowledge. Building on such closure models and correcting them through learning the structural errors can be an effective way of fusing data with domain knowledge. Here we describe a general approach, principles, and algorithms for learning about structural errors. Key to our approach is to include structural error models inside the models of complex systems, for example, in closure models for unresolved scales. The structural errors then map, usually nonlinearly, to observable data. As a result, however, mismatches between model output and data are only indirectly informative about structural errors, due to a lack of labeled pairs of inputs and outputs of structural error models. Additionally, derivatives of the model may not exist or be readily available. We discuss how structural error models can be learned from indirect data with derivative-free Kalman inversion algorithms and variants, how sparsity constraints enforce a "do no harm" principle, and various ways of modeling structural errors. We also discuss the merits of using non-local and/or stochastic error models. In addition, we demonstrate how data assimilation techniques can assist the learning about structural errors in non-ergodic systems. The concepts and algorithms are illustrated in two numerical examples based on the Lorenz-96 system and a human glucose-insulin model.

1. Introduction

Numerical simulation is at the heart of modeling, predicting, and understanding dynamical systems that are too complex to be amenable to analytical solution. Complex dynamical systems here extend from molecular dynamics with quantum effects to the planetary scales of weather and climate. The range of dynamically important scales in these systems can be vast, for example, in case of the atmosphere, extending over 13 orders of magnitude from the micrometers of cloud droplets and aerosols to the tens of

* Corresponding author.
E-mail address: jinlong.wu@wisc.edu (J.-L. Wu).
[1] Jin-Long Wu and Matthew Levine started to work on this research at California Institute of Technology.

https://doi.org/10.1016/j.jcp.2024.113157
Received 29 December 2023; Received in revised form 25 May 2024; Accepted 26 May 2024

Available online 31 May 2024
0021-9991/© 2024 Elsevier Inc. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

thousands kilometers of planetary waves. The number of degrees of freedom that would need to be resolved for a faithful simulation of such systems (e.g., $\gtrsim 10^{21}$ for a typical atmospheric boundary layer flow) often exceeds what will be computationally feasible for the foreseeable future [1].

Instead of direct numerical simulation, a variety of approaches has been devised to approximately resolve the most important degrees of freedom in numerical simulations. The degrees of freedom that remain unresolved but, because of nonlinear interactions, are still important for the resolved degrees of freedom are then represented by closure models, which link what is unresolved to what is resolved. The state X of the approximate system evolves according to dynamics of the form

$$\dot{X} = f(X; \theta_{\mathrm{P}}), \tag{1}$$

where f may depend on derivatives of the state X; hence, the system may represent partial differential equations. The system depends on empirical parameters θ_{P} that appear in closure models. For example, in large-eddy simulations of turbulent flows, the most energetic "large eddies" are explicitly resolved in the dynamics represented by f. The effect of the unresolved scales is modeled by subgrid-scale models, such as the classical Smagorinsky model [2], which depend on empirical parameters θ_{P} (e.g., the Smagorinsky coefficient). Similar semi-empirical models are used in many other fields. They encode domain-specific knowledge, and their parameters θ_{P} need to be calibrated with data.

Data y that are informative about the system come in a variety of forms, such as direct measurements of the time evolution of the state X or more indirect mappings of the state X onto observables, which may, for example, be statistical aggregates of state variables or convolutions of state variables with kernels. Convolutional data arise, for example, when representing the effect of a state variable such as temperature on the radiative energy fluxes that a satellite measures from space. Generally, we can write that the state maps to observables via an observation operator $\mathcal{H}$, such that

$$\hat{y} = \mathcal{H}[X]. \tag{2}$$

The challenge is that simulated observables $\hat{y}$ generally are biased estimates of actual data y. The actual data y are affected by measurement error, and the simulated data $\hat{y}$ are affected by structural errors in the approximate dynamical system (1); both y and $\hat{y}$ can also be affected by sampling error. For example, while a general feature of turbulence is to enhance mixing of conserved quantities, turbulent mixing is not always diffusive in character. Therefore, diffusive subgrid-scale models such as the Smagorinsky model are not always structurally correct, especially in convective situations with coherent flow structures [3]. This can lead to biases that, for example, adversely affect the calibration of model parameters θ_{P}.

The purpose of this paper is to summarize principles of, and algorithms for, learning about structural error models that correct semi-empirical closure models. Wholesale replacement of semi-empirical closure models with neural networks and other deep learning approaches promises to overcome the structural strictures of existing closure models through more expressive models; it has recently received much attention [4–11]. However, existing semi-empirical closure models encode valuable domain-specific knowledge. Learning flexible corrections to these models is often less data hungry, more interpretable, and potentially more generalizable than replacing them wholesale.

What follows is a distillation of experiences we gained in studying various complex dynamical systems. Our goal is to provide guidelines and algorithms that can lead to a broad-purpose computational framework for systematically learning about model error in dynamical systems. We focus on two important parts of error modeling: (i) how to construct an error model, and (ii) how to calibrate an error model.

Our approach to constructing error models builds upon but goes beyond the classical work of Kennedy and O'Hagan [12], who accounted for model error through an external bias correction term $\delta(X; \theta_{\mathrm{E}})$, parameterized by parameters θ_{E} and inserted at the boundary between output from a computer model $\hat{y} = \mathcal{G}(\theta_{\mathrm{P}})$ and data y:

$$y = \hat{y} + \delta(X; \theta_{\mathrm{E}}) + \eta. \tag{3}$$

Here, $\mathcal{G}(\theta_{\mathrm{P}}) = \mathcal{H}[X(\theta_{\mathrm{P}})]$ corresponds to solving (1) for the time series of the state X, which depends parametrically on θ_{P}, and then applying the observation operator (2); hence, $\mathcal{G}$ is a mapping from the space of model parameters θ_{P} to the space of observations y. The noise η represents additional (e.g., observation) errors, assumed to have zero mean. In this approach, the model parameters θ_{P} remain fixed (i.e., a property of $\mathcal{G}$) while parameters θ_{E} in the error model δ are tuned such that the residual $y - \hat{y}$ has a small magnitude and zero mean. This approach of externalizing model error for bias correction has been applied and further expanded in many subsequent papers (e.g., [13–17]). A key advantage of the external model error approach is that the model producing $\hat{y}$ can be treated as a black box, which facilitates use of this approach across different domains. State variables X and residuals $y - \hat{y}$ form input-output pairs from which the error model $\delta(X; \theta_{\mathrm{E}})$ can be learned, for example, with supervised learning[2] approaches, usually in an offline setting separate from learning about the model parameters θ_{P}.

However, the external model error approach has several drawbacks:

- It is difficult to incorporate physical (or other process-based) knowledge or constraints (e.g., conservation laws) in the error model $\delta(X; \theta_{\mathrm{E}})$ [15].
- It cannot improve predictions for quantities other than the observations y on which the error model $\delta(X; \theta_{\mathrm{E}})$ has been trained.

[2] Supervised learning refers to regression or interpolation of data.

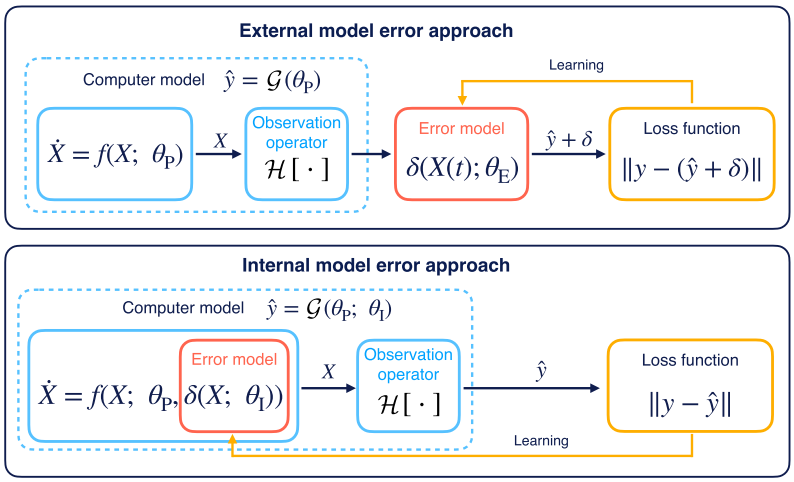


Fig. 1. External and internal approaches to modeling structural errors in complex dynamical systems. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

- It leads to interpretability challenges because $\delta(X;\theta_E)$ is a catch-all error term that typically represents the sum total of errors made in several, and often disparate, closure models.

To address these drawbacks, a few researchers have started to explore an approach that internalizes model error [18–20]. Such an internal model error approach embeds L error models $\delta_l(X;\ \theta_I^{(l)})$ $(l = 1, \dots, L)$ within the dynamical system, at the places (e.g., in closure models) where the errors are actually made. Let their collection be written as

$$\delta(X;\ \theta_I) := \left\{ \delta_l\left(X\ ;\ \theta_I^{(l)}\right) \right\}_{l=1}^{L} \tag{4}$$

so that we can write for the overall system

$$\dot{X} = f\Big(X;\ \theta_P, \delta(X;\ \theta_I)\Big). \tag{5}$$

The error models internal to the dynamical system are chosen so that the error-corrected computer model $\hat{y} = \mathcal{G}(\theta_P;\ \theta_I) = \mathcal{H}[X(\theta_P;\ \theta_I)]$ provides unbiased estimates of the data y:

$$y = \mathcal{G}(\theta_P;\ \theta_I) + \eta, \tag{6}$$

where the additional errors η are still assumed to have zero mean. Fig. 1 illustrates and contrasts the external and internal approaches to modeling structural errors.

Such an approach has found applications, for example, in turbulence modeling [21–27]. By incorporating the structural error models $\delta(X;\ \theta_I)$ inside the dynamical system, the error models can in principle lead to improved predictions even of quantities that were not used to train the error models. The error models can be learned alongside the model parameters θ_P in an online setting. They also are more amenable to interpretation because they are included in the places where errors are actually made. A potential downside of internalizing model error is that the effects of the structural errors δ map onto data y only through the resolved state X of the dynamical system, and residuals $y - \hat{y}$ are generally not directly informative about how the structural errors $\delta(X;\ \theta_I)$ depend on state variables X; thus, learning about structural errors $\delta(X;\ \theta_I)$ can generally not be accomplished with direct supervised learning approaches. Instead, residuals $y - \hat{y}$ only provide indirect information about structural errors $\delta(X;\ \theta_I)$. Additionally, if derivatives of the dynamical system f with respect to parameters are not easily available, or if the dynamical system is not differentiable, gradient-based methods for learning about the model errors $\delta(X;\ \theta_I)$ are difficult or impossible to use.

Here we show various ways of constructing models for structural errors and demonstrate how one can learn about the structural errors from direct or indirect data in the absence of derivatives of the dynamical system. As error models, we will consider:

- Gaussian processes, as in Kennedy and O'Hagan [12];
- Models assembled from dictionaries of terms (e.g., involving differential operators), as in the data-driven discovery of partial differential equations [28–33];
- Neural networks, for their expressivity [4,6,7,34–36];
- Stochastic models, because without a clear scale separation between resolved and unresolved degrees of freedom, theoretical analysis suggests that closure models generally should be stochastic (e.g., [37–40]);
- Non-local models, because structural errors may be non-local in space [41–45], in time [46–48], or in both [49].

We will discuss how to learn about such error models both from direct and indirect data. Supervised learning of various types of error models δ_l from direct data has been performed in a number of settings, for example, to discover terms in differential equations or neural network closure models from residual time tendencies that give direct information about the error model that is sought (e.g., [26,27,50,51]). However, data directly informative about error models, such as high-resolution time tendencies, are not always available. When training an error model on time tendencies, it can also be difficult to ensure both stability of the dynamical system including the error model and satisfaction of physical constraints (e.g., energy conservation) (e.g., [36]). Training an error model with indirect data, in an inverse problem rather than supervised learning setting (e.g., [24,52,53]), can be advantageous in those circumstances. We demonstrate how it can be accomplished.

Our main contributions are:

- Summarizing the internal model error approach and comparing and contrasting it with the external model error approach.
- Presenting a comprehensive study of the internal model error approach for various model forms, including dictionary-based models, Gaussian processes, and neural networks, including non-local and stochastic generalizations.
- Demonstrating various methods for calibrating the internal model error, including (i) learning from direct data, (ii) learning from indirect data (e.g., moments of the time series for ergodic systems) with ensemble Kalman inversion (EKI), and (iii) learning from the time series of non-ergodic systems and using EKI for data assimilation (DA).
- Illustrating the importance of constraints (i.e., sparsity and known physics) for improved generalization of the calibrated internal model error.

The principles and algorithms we will discuss are broad purpose and applicable across a range of domains and model complexities. To illustrate their properties in a relatively simple setting, we use two versions of the Lorenz 96 [54] dynamical system: the basic version of the model and its multiscale generalization [55]. Section 2 discusses the calibration of internal error models with direct or indirect data and the enforcement of constraints such as conservation properties. Section 3 introduces various ways of constructing error models. Section 4 introduces the two Lorenz 96 systems and then proceeds to present various concepts and methods through numerical results for these systems. The Lorenz 96 systems serve as illustrative examples of ergodic systems, for which we avoid trajectory matching and learn about unknown parameters from time-averaged statistics, which approximate the expectation with respect to the invariant measure. Section 5 is organized similarly to the previous section, but deals with non-ergodic systems for which trajectory matching is needed to learn about the underlying unknown parameters; as an example, we study a model of the human glucose-insulin system. In Section 6, we state our conclusions.

2. Calibrating error models

We first summarize several important aspects of calibrating internal error models, including (i) direct or indirect data, (ii) gradient-based or derivative-free optimization, and (iii) enforcing constraints (e.g., sparsity or physical laws). We discuss these aspects with a generic internal error model $\delta(X;\theta_\mathrm{I})$, which may represent any one of the error model types we will introduce later.

2.1. Data availability

2.1.1. Direct data

Direct data to calibrate δ are defined as "labeled" input-output pairs $\{X(t_i), \delta(X(t_i))\}_{i=1}^N$, where i denotes a time index. Consider the additive error model

$$\dot{X} = f(X;\theta_\mathrm{p}) + \delta(X;\theta_\mathrm{I})$$

as an example. A fine temporal resolution of $X(t)$ is usually needed to approximate $\dot{X} - f(X;\theta_\mathrm{p})$ and obtain estimates of the error $\delta(X;\theta_\mathrm{I})$ as a residual. With this method, it becomes challenging to obtain reliable direct data when the trajectories $\dot{X}$ are noisy, for example, when the dynamical system is chaotic [50]. Furthermore it may not be possible to observe the entirety of X. This may be handled as a missing data problem [56], and could be handled by joint parameter-state estimation for example using data assimilation; see [57,58].

An additional complication with using direct data is ensuring the stability of the dynamical system with the calibrated error model $\delta(\cdot)$. Although with direct data, we can get more control of the accuracy of the error model itself, the calibrated error model often leads to unstable simulations of the dynamical system with the error model [59,60]. There are several ways to mitigate the instability introduced by the error model, e.g., adopting a structure that ensures physical constraints [61], enforcing physical constraints [62], ensuring stability by bounding the eigenvalues of the linearized operator, and limiting the Lipschitz constant of $\delta(\cdot)$ [63]. However, a systematic approach to ensure stability is lacking.

2.1.2. Indirect data

Instead of assuming access to direct data $\{X, \delta(X)\}$, the error model can also be calibrated with indirect data by solving an inverse problem associated with (6) (i.e., solve for the most likely parameters $\theta_\mathrm{p}, \theta_\mathrm{I}$ given the model $\mathcal{G}$ and data y). Using indirect data involves simulating the dynamical system with the error model as in (5); therefore, the calibration procedure with indirect data favors error models that lead to stable simulations, an important advantage over the direct methods. Typical examples of problems giving rise to indirect data include time-series of X for which the resolution is not fine enough to extract direct data for calibration

[64], time-averaged statistics of X [65] or dynamical systems that are partially observed [66]. More generally, indirect data can also be interpreted as constraints on X, and thus physical constraints can be enforced via augmenting indirect data.

2.2. Methods of calibration

Using direct data $\{X, \delta(X)\}$ for calibration leads to a regression problem, which can be solved with standard methods for a given parameterization of the error model (e.g., least squares fit for dictionary learning, gradient descent methods for neural network). By contrast, using indirect data for calibration leads to an inverse problem associated with Eq. (6). Indirect methods can be linked to direct methods by framing the inverse problem as a missing data problem [56] and alternating between updating the missing data and updating the calibration parameters using learned direct data, for example, using the expectation-maximization (EM) algorithm [67]. However, in this section we focus on the calibration in the inverse problem setting; we discuss gradient-based and derivative-free optimization methods and how to enforce constraints.

2.2.1. Gradient-based or derivative-free optimization

Eq. (6) defines a forward problem in which $\mathcal{G}, \theta_{\mathrm{P}}, \theta_{\mathrm{I}}$ and noise η can be used to generate simulated data. The associated inverse problem involves identifying the most likely parameters $\theta_{\mathrm{P}}, \theta_{\mathrm{I}}$ for $\mathcal{G}$, conditioned on observed data y. To formalize this, we first define a loss function

$$\mathcal{L}(\theta) = \frac{1}{2}\left|y - \mathcal{G}\left(\theta_{\mathrm{P}};\ \theta_{\mathrm{I}}\right)\right|_{\Sigma}^{2}, \tag{7}$$

where Σ denotes the covariance of the zero-mean noise η, which is assumed to be independent of θ.[3] The inverse problem

$$\theta^{*} = \arg\min_{\theta} \mathcal{L}(\theta)$$

can be solved by gradient descent methods once the gradient

$$\frac{d\mathcal{L}}{d\theta} = \frac{d\mathcal{G}^{T}}{d\theta}\Sigma^{-1}(y - \mathcal{G}) \tag{8}$$

is calculated, where $\theta = [\theta_{\mathrm{I}}, \theta_{\mathrm{P}}]$ collects all parameters. In practice, the action of the gradient $d\mathcal{G}^{T}/d\theta$ is often evaluated via adjoint methods for efficiency. Although the gradient-based optimization is usually more efficient when $\mathcal{G}$ is differentiable, the evaluation of $\mathcal{G}$ can be noisy (e.g., when using finite-time averages to approximate infinite-time averaged data [68]) or stochastic (e.g., when using stochastic processes to construct the error model). In these settings, gradient-based optimization may no longer be suitable, and derivative-free optimization becomes necessary. In this paper we focus on Kalman-based derivative-free optimization for solving the inverse problem; Appendix A briefly reviews a specific easily implementable form of ensemble Kalman inversion (EKI), to illustrate how the methodology works, and gives pointers to the broader literature in the field.

It is worth noting that EKI requires ensemble simulations, which may be less attractive when the system is not chaotic and gradient information is available. Even for chaotic systems, matching short trajectories can be achieved by gradient-based optimization [69], without the need to run ensemble simulations. For large datasets, the computational cost of EKI increases, e.g., because of the cost of the matrix inversions in the EKI formula. However, it is possible to make use of the sparse structure of the covariance matrix in data space and thus reduce the computational cost of the matrix inversion [70]. For a large dataset and a large number of parameters, a relatively small ensemble size may not suffice to approximate the covariance matrix; in that case, localization techniques [71–73] can be used to avoid spurious correlations due to the small ensemble size. In practice, localization techniques often lead to sparse covariance matrices, which allows users to employ standard efficient algorithms for the inverse and the multiplication of those covariance matrices in EKI.

2.2.2. Enforcing constraints

There are various types of constraints that can be enforced when calibrating an error model. Two common constraints are sparsity constraints and physical constraints (e.g., conservation laws). Here we present the general concept of enforcing these two types of constraints in calibration, and Appendix A presents more details about using EKI to solve the corresponding constrained optimization problems.

Sparsity is important to impose on error models to avoid adding parameters to an error model that have no or little measurable impact on the output $\hat{y}$ of the host model; that is, the first principle in modeling structural errors should be to do no harm that would result from unnecessary error model complexity. To impose sparsity on the solution of θ_{I}, we aim to solve the optimization problem [74]

$$\begin{aligned} \mathcal{L}(\theta;\lambda) &:= \frac{1}{2}\left|y - \mathcal{G}\left(\theta_{\mathrm{P}};\ \theta_{\mathrm{I}}\right)\right|_{\Sigma}^{2} + \lambda|\theta_{\mathrm{I}}|_{\ell_0}, \\ \theta^{*} &= \arg\min_{\theta\in\mathfrak{B}} \mathcal{L}(\theta;\lambda), \end{aligned} \tag{9}$$

[3] By $|\cdot|_{B}$, we denote the covariance-weighted norm defined by $|v|_{B} = v^{*}B^{-1}v$ for any positive definite B.

where $\mathfrak{B} = \{\theta : |\theta_{\mathrm{I}}|_{\ell_1} \leq \gamma\}$. The regularization parameters γ and λ can be determined via cross-validation. In practice, adding the ℓ_0 constraint is achieved by thresholding the results from ℓ_1-constrained optimization. The detailed algorithm was proposed in [74] and is summarized in Appendix A.

In many applications, it is important to find error models that satisfy physical constraints, such as energy conservation. To impose physical constraints on the solution of θ_{I} from EKI, we first generalize the constraint as:

$$\mathfrak{B} = \{\theta : \mathcal{R}(\theta_{\mathrm{I}}) \leq \gamma\}. \tag{10}$$

Here, $\mathcal{R}$ can be interpreted as a function that evaluates the residuals of certain physical constraints (typically, by solving Eq (5)). The constraint function $\mathcal{R}$ can be nonlinear with respect to θ_{I}. Taking the additive error model $\dot{X} = f(X) + \delta(X;\theta_{\mathrm{I}})$ as an example, the function $\mathcal{R}$ corresponding to the energy conservation constraint can be written explicitly as

$$\mathcal{R}(\theta_{\mathrm{I}}) = \left| \int_0^T \left(\langle \delta(X(t);\theta_{\mathrm{I}}), X(t) \rangle \right) dt \right|, \tag{11}$$

which constrains the total energy introduced into the system during the time interval $[0, T]$. Alternatively, a stronger constraint can be formulated as

$$\mathcal{R}(\theta_{\mathrm{I}}) = \int_0^T \left| \langle \delta(X(t);\theta_{\mathrm{I}}), X(t) \rangle \right| dt, \tag{12}$$

which constrains the additional energy introduced into the system at every infinitesimal time step within the time interval $[0, T]$. The notation $\langle \cdot, \cdot \rangle$ denotes the inner product. Both forms of constraint in (11) and (12) can be implemented by using augmented observations, i.e., including the accumulated violation of the energy constraint as a additional piece of observation data whose true mean value is zero.

2.2.3. Accounting for initial conditions

Evaluation of $\mathcal{G}$ typically requires an initial condition, $X(0)$, when solving Eq. (5). When we observe ergodic systems over sufficiently long time scales, we can treat $\mathcal{G}$ as a stochastic map over a distribution of initial conditions [75]. However, when we observe systems over shorter time scales (or if they are non-ergodic), it is often necessary to identify initial conditions that generated the observations. We highlight two basic approaches: (i) append $X(0)$ to the vector of learned parameters for $\mathcal{G}$, and (ii) implicitly identify $X(0)$ by applying data assimilation techniques. To achieve (i), we can minimize:

$$\mathcal{L}\big(\theta, X(0)\big) = \frac{1}{2} \Big| y - \mathcal{G}\big(\theta_{\mathrm{P}};\ \theta_{\mathrm{I}}, X(0)\big) \Big|_{\Sigma}^2. \tag{13}$$

To achieve (ii), we can minimize:

$$\begin{aligned} \mathcal{L}(\theta) =& \frac{1}{2} \Big| y - \mathcal{G}\big(\theta_{\mathrm{P}};\ \theta_{\mathrm{I}}, X(0)\big) \Big|_{\Sigma}^2, \\ \text{s.t. } X(0) =& \mathcal{DA}(\theta, y), \end{aligned} \tag{14}$$

where $\mathcal{DA}$ represents a data assimilation algorithm that performs state estimation given observations y and assumed parameters θ. While Eqs. (13) and (14) appear quite similar, the utilization of efficient state-estimation algorithms in (14) can help cope with sensitivities to initial conditions that may cause (13) to fail. Recent work has explored different variants of (14), often using gradient-based optimization approaches and sequential state-estimation schemes [51,58,76,77].

3. Constructing error models

We highlight three different approaches to representing structural errors: dictionary learning, Gaussian processes, and neural networks; however, other representations of structural error can also be considered within the overarching framework proposed here. For these three approaches, existing work mainly focuses on constructing deterministic error models that are locally dependent on state variables; however, the approaches can all be extended to the construction of stochastic error models or can be made non-locally dependent on state variables as described in Sections 3.4 and 3.5. For simplicity, we define the error models for the whole collection of structural errors $\delta(X, \theta_{\mathrm{I}})$; however, we can also define and learn them independently for each component of the structural error model $\delta_l(X, \theta_{\mathrm{I}}^l)$ for $l = 1, \ldots, L$.

3.1. Dictionary learning

If a set of candidate terms in error models is known or can be approximated, an error model can be constructed via learning from a dictionary of J candidate terms,

$$\delta(X;\theta_{\mathrm{I}}) = \sum_{j=1}^{J} \alpha_j \phi_j(X;\beta_j), \tag{15}$$

where $\theta_I = \{\alpha, \beta\}$ and $\phi_j(X; \beta_j)$ denote user-specified, parametric basis functions that can, for example, include differential operators [28–30,50]. In practice, it is difficult to know all suitable basis functions a priori, and thus it is common to include redundant basis functions in the dictionary. Basis functions can then be pruned based on data by imposing sparsity constraints on the coefficients α_j. Such sparsity constraints have proven to be beneficial in the construction of data-driven models of dynamical systems [28–30,50,74]. They are also commonly used in compressed sensing [78], where dictionary learning has been widely used.

An advantage of using dictionary learning is the potential interpretability of the constructed error model, arising because the error model is a linear combination of user-specified and hence interpretable basis functions. On the other hand, this approach can be overly restrictive when the dictionary of basis functions $\{\phi_j\}$ is misspecified, resulting in an insufficiently expressive error model.

3.2. Gaussian processes

Another option of constructing an error model is via Gaussian processes (GPs) [79],[4]

$$\delta(X; \theta_I) \sim \mathcal{GP}(m, \mathcal{K}), \tag{16}$$

where $m : \mathcal{X} \mapsto \mathbb{R}$ denotes the mean of δ, $\mathcal{K} : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ represents a kernel, and $\mathcal{X}$ is the input space of the structural error model. Given data at J different points $X^{(j)}$ for $j = 1, 2, ..., J$, the mean of the error model can be written as a linear combination of basis functions,

$$m(X) = \sum_j \alpha_j \mathcal{K}(X^{(j)}, X; \psi), \tag{17}$$

where ψ denotes the hyper-parameters of the kernel $\mathcal{K}$. Therefore, the parameters that characterize the error model become $\theta_I = \{\alpha, \psi\}$ if the mean of a GP is used to represent the model error term δ. The GP approach requires the choice of a kernel $\mathcal{K}$, which then determines the kernel functions $\mathcal{K}(X^{(j)}, \cdot)$ in Eq. (17). This may appear restrictive, but the hyper-parameters of $\mathcal{K}$ are learned from the data; thus, the set of functions in which the solution is sought is data-adapted. This confers a potential advantage over dictionary learning, in particular for problems lacking in strong prior knowledge about the functional form of the model $\delta(X; \theta_I)$ to be learned. In the case of indirect data, the locations $X^{(j)}$ must also be chosen a priori (or learned as additional parameters).

Because of the similar forms of Eqs. (17) and (15), the GP shares similar shortcomings as dictionary learning when the kernel $\mathcal{K}$ is misspecified, even in the presence of hyper-parameter learning. In practice, a more sophisticated kernel $\mathcal{K} = \sum_i \mathcal{K}_i$ is often constructed from some basic kernels $\mathcal{K}_i$ [80–82]. If a redundant set of basic kernels is used, sparsity constraints can be imposed in a similar way as in dictionary learning to prune the kernel set. A further limitation of using GPs is the computational cost, which grows exponentially with the dimension of $\mathcal{X}$. This pathology can be ameliorated by representing the GP as a linear combination of random Fourier features [83], which allows us to recast a GP as a dictionary-based approach in which the bases ϕ_j are drawn randomly from a special distribution known to reproduce a kernel of interest.

3.3. Neural networks

Compared to dictionary learning, neural networks are more expressive, and they are more scalable than GPs, as the latter suffer from the curse of dimensionality if the model has high-dimensional input. Neural networks can also be used to construct an error model,

$$\delta(X; \theta_I) = \mathcal{NN}(X; \theta_I), \tag{18}$$

where $\mathcal{NN}$ denotes a neural network and θ_I the coefficients (biases and weights) of the neural network. While neural networks are expressive and scalable, it is more difficult to enforce stability of a dynamical system with a neural network error model [60]. This is mainly because the nonlinearity introduced by a neural network is often more difficult to analyze compared with dictionary learning, for which we explicitly specify basis functions and thus can avoid using those that lead to instability; it is also more difficult to analyze than GP based learning because the latter is easier to interpret, as the kernels are hand-picked and then tuned to data. In Section 2.2.2, we discuss a general approach to enhancing stability by enforcing energy constraints in the context of learning from indirect data.

3.4. Stochastic extension

In the preceding sections, we briefly summarized commonly used tools for constructing error models. All of those models were deterministic, with fixed parameters θ_I. To quantify uncertainties, we can take a Bayesian perspective, view the unknown parameters as random variables, and infer the distributions of those parameters given the data. We can then propagate the uncertainties of those parameters to the simulated state X and predicted observations $\hat{y}$ via Monte Carlo simulations. Although this is a standard approach to quantifying uncertainties, it cannot directly account for the impact of neglected information of unresolved scales upon

[4] Here we use only the mean of the GP, and the methodology is simply a form of data-adapted regression; we are not using the uncertainty quantification that comes with GPs.

the resolved state X. The randomness of the unresolved state can have an order one impact upon X; this issue is particularly prevalent in applications without a clear scale separation, such as turbulence, but can also happen in scale-separated problems. In such a scenario, directly modeling this impact as randomness in the resolved state becomes more appropriate, and it can be achieved by further adding a stochastic process to the deterministic error model:

$$\delta\Big(X;\theta_{\mathrm{I}}\Big)\mathrm{d}t = \delta_{\mathrm{det}}(X;\theta_{\mathrm{det}})\mathrm{d}t + \sqrt{\sigma^2(X;\theta_{\mathrm{ran}})}\mathrm{d}W, \tag{19}$$

where det indicates a deterministic model, W denotes the Wiener process, and the overall unknown parameters are defined as $\theta_{\mathrm{I}} = \{\theta_{\mathrm{det}}, \theta_{\mathrm{ran}}\}$. In practice, the above formulation can be further generalized by using stochastic processes (e.g., with desired temporal correlations) other than the Wiener process.

Fitting stochastic models to time-series data has been explored in some previous works [84–87]; a common problem when applying these methods is the inconsistency between data and the incremental structure of the Gaussian noise driving the model as the time step is approaching zero [88–91]. A common practice to address this issue is the multi-scale use of data, e.g., via subsampling [66,92–96]. Some previous works also explored Kramers–Moyal averaging with finite sampling rate correction [97–99]. On the other hand, fitting a discretized version of stochastic processes to time-series data has been explored using autoregressive models [100–103]. For some dynamical systems, the unresolved state has conditional (with respect to the resolved state) Gaussian statistics [104,105], and then fitting the stochastic models can be achieved using analytically derived likelihoods.

In the absence of the whole trajectories of time series, some recent works started to explore fitting stochastic models to statistics of time-series data [52,106–108]. Using time-averaged data to estimate linear SDEs has been studied for decades to account for climate variability [109–112], and extension to nonlinear SDEs was discussed in [113].

3.5. Representing non-local effects

Spatial non-locality The states $X(t)$ for approximate models and their structural corrections typically consider $X(t)$ as a discretized spatial field. Most traditional closure models are formed locally; that is, they rely on the assumption of local dependence on $X(t,r)$, where $X(t,\cdot) : \mathbb{R}^p \mapsto \mathbb{R}$ is a spatial field, and $r \in \mathbb{R}^p$ represents the spatial coordinate. For some applications, it is useful to consider non-local effect in the error model. Indeed, our formulations of the approximate physical model in (1) and models for structural error in Sections 3.1 to 3.3 are well-specified for scalar (local, component-wise) or vector-valued (non-local) X. Moreover, we note that non-local functions of the state $X(t)$ are best conceptualized as function-valued *operators*—while they take as inputs a vector of neighboring coordinates from $X(t)$, this vector represents a discretized spatial function. Thus, when designing spatially non-local closures, it is often sensible to build them to be consistent across different spatial discretizations.

In the case of neural networks, we can build spatially non-local closures with convolutional neural networks (CNNs); the Fourier neural operator (FNO) [114] or deep operator network (DeepONet) [115] provide an extension to an operator limit. Similarly, GPs and random feature methods (a dictionary-based formulation of GPs) can be designed with spatially non-local vectorized inputs from $X(t)$. Recent theoretical work has also allowed these basic methods to be taken to a continuous operator limit [116,117].

As an emerging topic in the context of data-driven modeling, some recent works have explored non-local diffusion [43–45,118] and spatially non-local modeling [41,42,49]. In this work, we capture the spatially non-local dependence on X via a data-driven convolution kernel:

$$\delta\Big(X(t,r);\theta_{\mathrm{I}}\Big) = \int_{r'\in\Omega} \delta_{\mathrm{loc}}(X(t,r');\theta_{\mathrm{loc}})C(r-r';\theta_{\text{non-loc}})\,dr' \tag{20}$$

where $\Omega \subset \mathbb{R}^p$ represents a subset of $\mathbb{R}^p$ that contains r, and $C : \mathbb{R}^p \mapsto \mathbb{R}$ denotes a convolution kernel with hyper-parameters $\theta_{\text{non-loc}}$. The overall parameterization is defined by $\theta_{\mathrm{I}} = \{\theta_{\mathrm{loc}}, \theta_{\text{non-loc}}\}$, such that the unknown parameters in the local error model δ_{loc} and the convolutional kernel C can be jointly estimated.

Note that hyper-parameters can be made state-dependent, so that $\theta_{\text{non-loc}}(X(t,r);\kappa)$; in this case, the additional unknowns κ can be learned, appending them to θ_{I}. Similarly, learning a nonlinear integral kernel has been discussed in [119] and shown to be a continuous generalization of the transformer architecture [120].

The form of non-local closure in (20) draws inspiration from a series of works about non-local modeling [118], in which δ_{loc} corresponds to a local Laplace operator. Some mathematical foundations of non-local operators and calculus were summarized in [118], and the connection to fractional differential operators was illustrated in [121].

Temporal non-locality Non-locality in time (memory) is also important. Generically, any form of variable elimination or coarse-graining results in memory effects which require, at each current point in time, integration of the entire time-history from the initial condition up to the current time [37]. Such memory effects are undesirable as they lead to computational algorithms that scale poorly with respect to length of the time interval. Markovian models that encapsulate memory can be constructed, for example, by introducing a recurrent neural network [51], or by the use of delay embedding [122]; such Markovian models are more computationally expedient. Temporally non-local modeling has received significant recent attention [46–49]. If diffusion/advection mechanisms are present in the resolved system, memory effects of any state variable would manifest themselves in the state variables as a spatial non-locality. For this reason non-local models with a flexible enough kernel could potentially be used to capture memory effects, without significantly increasing the computational costs.

4. Lorenz 96 systems as illustrative examples

For our simulation studies of structural model error, we consider two variants of the celebrated Lorenz 96 model [54], described in what follows.

4.1. Lorenz models

We consider a multiscale Lorenz 96 model, together with a single-scale companion model, to illustrate the principles and algorithms described in the subsequent sections. In each case, we use an untruncated version of the model as the true data-generating model and a truncated version as the model in which structural error models are to be learned.

4.1.1. Multiscale Lorenz 96 model

The Lorenz 96 multi-scale system [54] describes the evolution of a simplified atmospheric flow, which is periodic along latitude circles (space). It does so through one set of slow variables, x_k ($k = 1, \dots, K$), coupled to a set of fast variables, $z_{j,k}$ ($j = 1, \dots, J$), whose indices label space coordinates:

$$\begin{aligned} \dot{x}_k &= -x_{k-1}(x_{k-2} - x_{k+1}) - x_k + F - hc\bar{z}_k, \\ \frac{1}{c}\dot{z}_{j,k} &= -bz_{j+1,k}(z_{j+2,k} - z_{j-1,k}) - z_{j,k} + \frac{h}{J}x_k. \end{aligned} \tag{21}$$

Reflecting the periodicity along latitude circles, the variables are periodic in their indices, with

$$x_{k+K} = x_k, \qquad z_{j,k+K} = z_{j,k}, \qquad z_{j+J,k} = z_{j,k+1}. \tag{22}$$

The coupling term $hc\bar{z}_k$ describes the impact of the fast dynamics on the slow dynamics, with only the average

$$\bar{z}_k = \frac{1}{J}\sum_{j=1}^{J} z_{j,k} \tag{23}$$

of the fast variables affecting the slow variables. To generate data, we work with the parameter choices $K = 36$, $J = 10$, and $F = b = 10$ [54,65]. The choices of h and c are summarized in Subsection 4.2 for different cases.

To study how to model structural errors, we consider a coarse-grained system in which we only simulate approximate versions X_k of the slow variables x_k, neglecting the fast variables. The approximate slow variables are governed by the system,

$$\begin{aligned} \dot{X}_k &= -X_{k-1}(X_{k-2} - X_{k+1}) - X_k + F + \delta(X_k, X_k^-; \theta_1), \\ X_{k+K} &= X_k, \end{aligned} \tag{24}$$

for $X_k^- = (X_{k-d}, \cdots, X_{k-1}, X_{k+1}, \cdots, X_{k+d})$. Here, $\delta(\cdot)$ is the error model that accounts for the missing multiscale interactions. If there is no dependence on X_k^-, the model is local; otherwise, we allow for non-local dependency with a stencil of width d on either side of X_k. If specified correctly, the model error ensures that the resolved variables X_k of the coarse-grained system (24) approximate the variables x_k of the full data generating system (21)–(23). We will use data generated with the full system (21)–(23) to learn about the error model $\delta(\cdot)$ in the coarse-grained system (24).

As data $y = \mathcal{H}[x(\cdot)]$ we consider

$$\mathcal{H}[x(\cdot)] = \mathbb{E}\mathcal{F}(x(\cdot)),$$

where $\mathbb{E}$ denotes expectation with respect to the stationary distribution of $x(t)$ and $\mathcal{F} : \mathcal{X}_{\text{traj}} \mapsto \mathbb{R}^q$ is a function on the space of solution trajectories (i.e., $\mathcal{X}_{\text{traj}} : 0^+ \mapsto \mathcal{X}$). In this work, we use moments of the vector x and the averaged auto-correlation function as data and employ a finite-time average to approximate the expectation $\mathbb{E}$:

(i) We will use m^{th}-moments of the vector x, i.e., $\mathcal{F}_m(x) = \Pi_{k\in M} x_k$:

$$\mathcal{H}[x(\cdot)] \approx \frac{1}{T}\int_0^T \Pi_{k\in M} x_k(t)\mathrm{d}t,$$

where x_k denotes the k^{th} element of vector x, and M is a subset of size m comprising indices (repetition allowed) from $\{1, \cdots, K\}$.

(ii) We will also use autocorrelation function $\mathcal{F}_{ac}(x(\cdot)) = x(t+\tau) \otimes x(t)$:

$$\mathcal{H}[x(\cdot)] \approx \frac{1}{T}\int_0^T x(t+\tau) \otimes x(t)\mathrm{d}t.$$

In this work, we only consider the autocorrelation of the same element in the vector x, i.e., $x_k(t+\tau)x_k(t)$.

4.1.2. Single-scale Lorenz 96 model

The single-scale Lorenz 96 system does not include the fast variables. We use it to illustrate the combined use of direct and indirect data and the advantage of enforcing conservation constraints in error models. The single-scale Lorenz 96 system describes the evolution of the x_k variables alone,

$$\begin{aligned} \dot{x}_k &= -x_{k-1}(x_{k-2} - x_{k+1}) - x_k + F, \\ x_{k+K} &= x_k, \end{aligned} \tag{25}$$

and we use it to generate data in the setting where $K = 36$ and $F = 10$.

As the truncated system, we will only assume that we know the linearized part of the dynamics, resulting in an approximate model of the form

$$\begin{aligned} \dot{X}_k &= -X_k + F + \delta(X_{k-2}, X_{k-1}, X_{k+1}, X_{k+2}; \theta_1), \\ X_{k+K} &= X_k. \end{aligned} \tag{26}$$

Here, δ is the error model that models the missing quadratic terms; we note that we postulate the need to learn a single universal function δ to account for model error in each component of the equation, reflecting an *a priori* assumption about the homogeneity of the structural error with respect to k. Since the error model δ accounts for the unknown convection term and thus should not introduce additional energy, the state variable X_k is excluded from the inputs of δ in the k-th equation. We will use data generated from the untruncated system (25) to learn about the error δ in the truncated system (26). As data $y = \mathcal{H}[x(t)]$, we employ the same types of data (i.e., moments and autocorrelation of the vector x) as described in Section 4.2.2.

4.2. Numerical results for Lorenz models

Before presenting detailed numerical results for Lorenz systems, we summarize several highlights of our numerical results.

1. For a multiscale system with clear scale separation, local deterministic error models using either direct or indirect data lead to satisfactory model fits. Detailed results are presented in Figs. 2 and 3.
2. For a multiscale system with less clear scale separation, local deterministic error models using direct data or indirect data do not lead to satisfactory model fits. Detailed results are presented in Figs. 4 and 5. However non-local or stochastic error models do lead to satisfactory fits. Detailed results are presented in Figs. 6 to 8.
3. For the single-scale Lorenz model, we show how an energy constraint can be incorporated into the EKI learning framework and leads to enhanced calibration of the error model. Detailed results are presented in Figs. 9 to 11.

Here, we investigate the long-term behavior of the trained models for the Lorenz systems via the invariant measure. For the numerical examples, the invariant measure corresponds to the probability density function of a system state variable, when simulating the dynamical system for a long enough time so that the probability density function does not vary. For the numerical examples with indirect data, the invariant measure is employed as a qualitative indicator for the long-term performance of the trained models. Considering that the indirect data only contain partial information about the true invariant measure, the quantitative comparison of the ensemble mean for the invariant measures may lead to over-interpretation of the results; thus, we only present the ensemble of invariant measures.

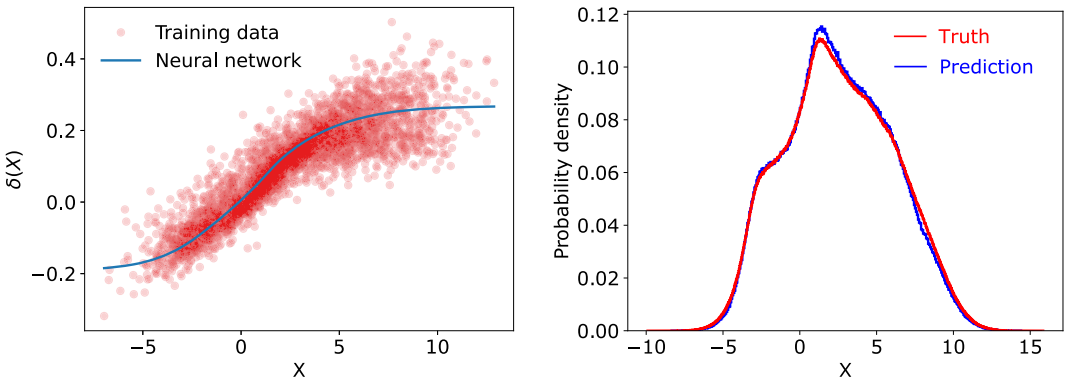
4.2.1. Lorenz 96 multi-scale model

We first study the multi-scale Lorenz 96 system. The numerical examples for the multi-scale Lorenz 96 system are as follows:

(i) For $c = 10$ and $h = 1$ in the multi-scale Lorenz 96 system, a fully-connected neural network is trained as a local deterministic error model $\delta(X_k)$ using direct data ($\{x_k, hc\overline{z}_k\}$), and a dictionary-learning-based model is trained using indirect data (first and second moments of the slow variable x_k). The direct data comprise 36000 data points. For the indirect data in this example, we assume partial observations of the first eight slow variables and include cross-terms of second moments (i.e., $\mathbb{E}(x_i x_j)$ for different i and j). The indirect data comprise 44 data points. The results are presented in Figs. 2 and 3.

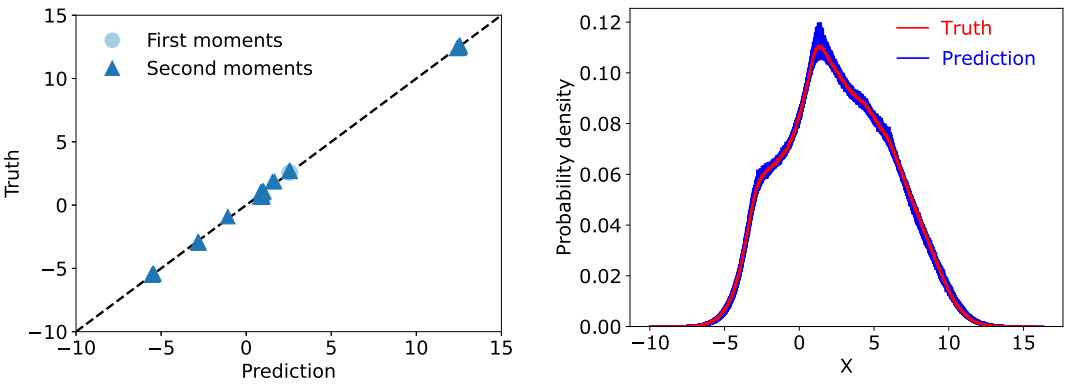
(ii) For $c = 3$ and $h = 10/3$ in the multi-scale Lorenz 96 system, the scale separation between fast and slow variables becomes smaller and thus leads to a more challenging case. In this case, a fully-connected neural network is trained as the local deterministic error model $\delta(X_k)$ using either direct data ($\{x_k, hc\overline{z}_k\}$) or indirect data (first to fourth moments of the slow variable x_k and the autocorrelation of x_k). For the indirect data in this and the next examples, we enable the full observation of all 36 slow variables and preclude the use of all cross-terms of second to fourth moments. The direct data comprise 36000 data points, and the indirect data 154 data points. The results are presented in Figs. 4 and 5.

(iii) For $c = 3$ and $h = 10/3$ in the multi-scale Lorenz 96 system, we train a non-local deterministic error model $\delta(X) = \sum_{k'} \delta(X_{k'}) C(k - k'; \theta_{\text{non-loc}})$, a local stochastic error model with additive noise $\delta(X_k) + \sqrt{\sigma^2}\dot{W}_k$, and a local stochastic error model with multiplicative noise $\delta(X_k) + \sqrt{\sigma^2(X_k)}\dot{W}_k$, using indirect data (first to fourth moments of the slow variable x_k and the autocorrelation of x_k). The results are presented in Figs. 6 to 8.



(a) Trained model (b) Invariant measure

Fig. 2. Direct training of the error model ($c = 10$) using a neural network, with results for (a) the trained error model and (b) the invariant measure.



(a) Data comparison (b) Invariant measure

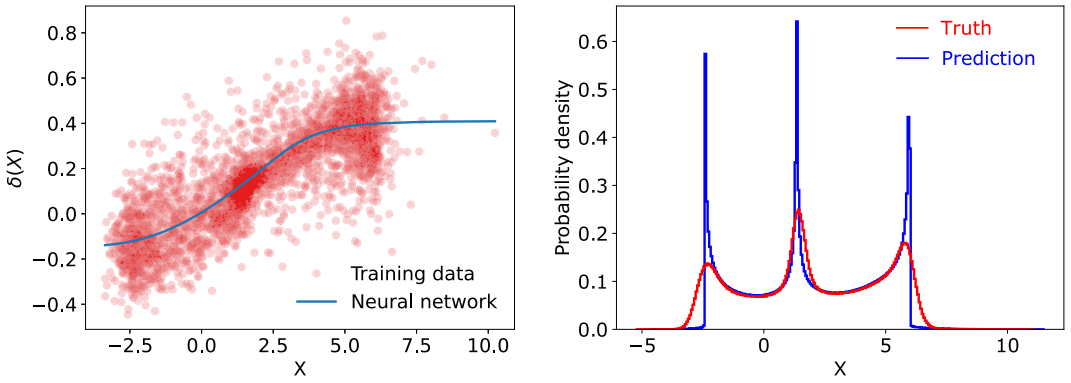
Fig. 3. Indirect training of the error model ($c = 10$) using dictionary learning, with the results for (a) first and second moments and (b) the invariant measure. The results with a GP and a neural network have similar performance and are omitted.

Fig. 2a presents the direct data $\{x_k, hc\bar{z}_k\}$. Based on direct data, a regression model $\delta(X_k)$ can be trained and then used to simulate the dynamical system of X_k in (24). In this work, we train such a regression model using a fully-connected neural network with two hidden layers (five neurons at the first hidden layer and one neuron at the second hidden layer). It can be seen in Fig. 2a that the trained model captures the general pattern of the training data. We also simulate the dynamical system in (24) for a long time trajectory and compare the invariant measure of X_k with the true system in (21). As shown in Fig. 2b, we obtain a good agreement between the invariant measures of the modeled system and the true system.

Because direct data $\{x_k, hc\bar{z}_k\}$ may not be accessible in some applications, we also explore the use of indirect data to calibrate the error model $\delta(X_k)$. In this example, the first and second moments of the first eight components of x_k are used for the calibration. We tested different approaches that parameterize the error model $\delta(X_k)$, including dictionary learning (Fig. 3), GPs and neural networks. The error model based on dictionary learning has the form $\delta(X_k) = \sum_{i=1}^{2} \alpha_i \phi_i(X_k)$, where we choose the basis function dictionary $\phi_i(X_k) \in \{\tanh(\beta_1 X_k), \tanh(\beta_2 X_k^2)\}$. Therefore, we have $\{\alpha_i, \beta_i\}_{i=1}^{2}$ as unknown parameters to be learned. Instead of using polynomial basis functions, we have introduced the hyperbolic tangent function $\tanh(\cdot)$ to enhance the numerical stability. The error model based on a GP has the form $\delta(X_k) = \sum_{j=1}^{7} \alpha_j \mathcal{K}(X_k^{(j)}, X_k;\ \psi)$, where we chose the $X_k^{(j)}$ as seven fixed points uniformly distributed in $[-15, 15]$, and $\mathcal{K}$ as a squared exponential kernel with unknown constant hyper-parameters $\psi = \{\sigma_{\mathrm{GP}}, \ell\}$, where σ_{GP} denotes the standard deviation and ℓ the length scale of the kernel. The results with a GP and with a neural network are similar to the ones with dictionary learning in Fig. 3 and are omitted here. The calibrated models in all three tests lead to good agreement in both data and invariant measure, and the performance of the calibrated model is not sensitive to the specific choice of parameterization approaches.

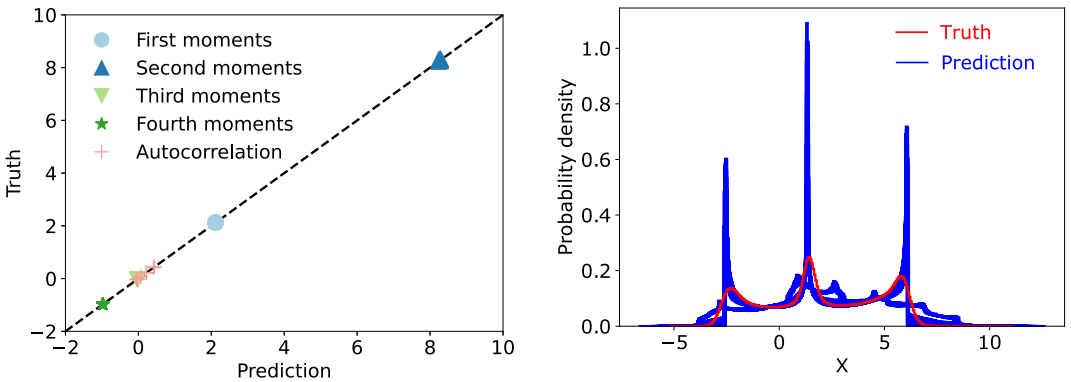
Although the performance of the calibrated error model is not sensitive to either the types of data or the parameterization approaches for this numerical example, the specific choices made in constructing and calibrating error models are still important, and even more so for more challenging scenarios, e.g., when the resolved and unresolved degrees of freedom have less scale separation.

To illustrate the advantage of using indirect data and stochastic/non-local error models, we study a more challenging scenario where the scale separation between x_k and $y_{j,k}$ in (21) is narrower, by setting $h = 10/3$ and $c = 3$. It can be seen in Fig. 4a that the general pattern of the direct data is still captured by the trained error model $\delta(X_k)$. However, the comparison of the invariant



(a) Trained model (b) Invariant measure

Fig. 4. Direct training of the error model ($c = 3$) using a neural network, with results for (a) the trained error model and (b) the invariant measure.



(a) Data comparison (b) Invariant measure

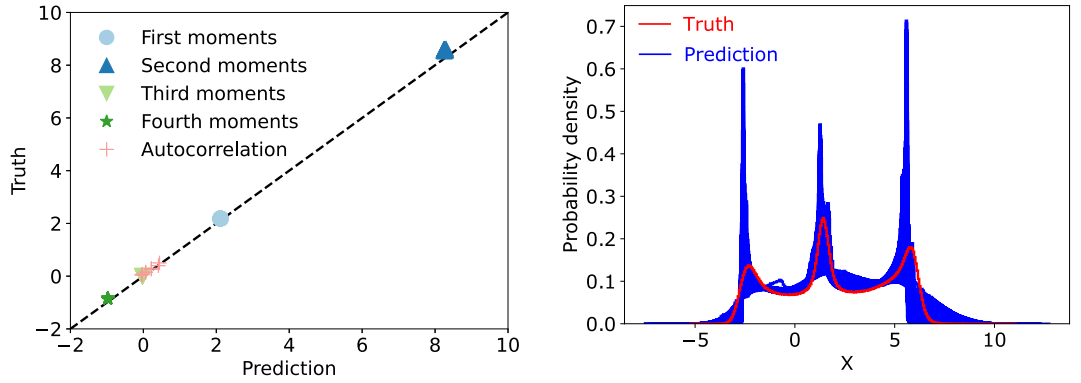
Fig. 5. Indirect training of the error model ($c = 3$) using a deterministic model (local), with the results for (a) the first four moments and autocorrelation, and (b) the invariant measure.

measures in Fig. 4b shows that the long-time behavior of the trained model does not agree with the true system, indicating the limitation of using only direct data for the calibration.

We further investigate the use of indirect data. Specifically, the first four moments of X_k and ten points sampled from the averaged autocorrelation function of X_k are used as training data. Fig. 5 presents the results for the calibrated local model $\delta(X_k)$. It can be seen in Fig. 5a that the trained error model agrees with the training data, while the invariant measures in Fig. 5b still differ between the calibrated and the true systems, indicating overfitting of the training data.

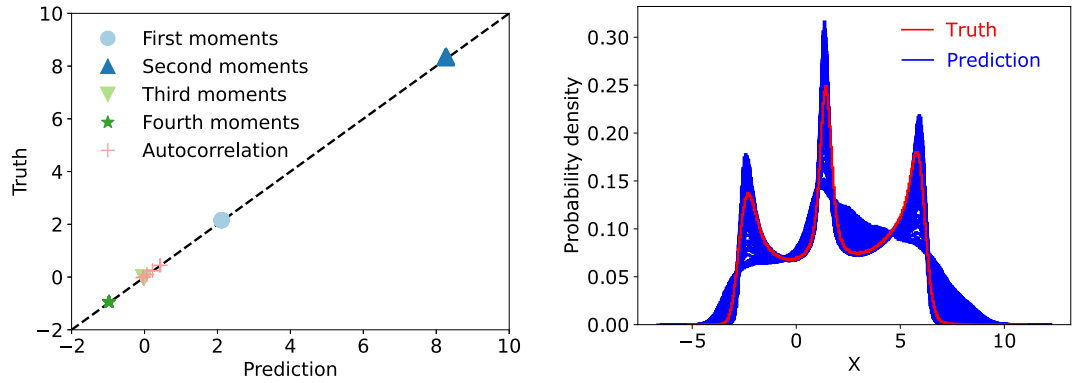
To avoid the overfitting in Fig. 5, we calibrate a non-local error model, as discussed in Section 3.5. Compared to the results for the local error model, it can be seen in Fig. 6 that the invariant measure of the calibrated system agrees better with the true system, which indicates that the closure model $\delta(\cdot)$ in (24) with non-local effects represents a better closure for unresolved scales if there is a less clear scale separation between resolved and unresolved scales. The non-local model tends to be more flexible than the local model. Fig. 5a demonstrates that the local model already achieves a good agreement in data space with the true system. It is likely that more than one possible non-local model can achieve a comparable performance of matching the indirect data. However, simulating those different non-local models for a much longer time can lead to more noticeable differences in their invariant measures, which provides a possible explanation for the larger variability within the ensemble in Fig. 6b. It should be noted that the key challenge of this example is the relatively large variability of the true closure term in Fig. 4a. Without accounting for such variability, the local and deterministic model does not correctly capture the invariant measure even in the standard regression setting with an abundance of training data as presented in Fig. 4b. In the non-local model, two local states with similar values may have noticeably different values of their neighbor states, and thus the non-local model with the local state and its neighbor states as the inputs can address some of the variability in Fig. 4a, which explains its improved performance relative to the local models. Considering that there are three peaks in the invariant measure of the true system, the non-local model has a greater chance of having some inputs from the left or right peak than merely having inputs from the central peak. Therefore, the trained model can be more in favor of the left or right peak of the invariant measure, which provides a possible explanation for the pattern of peaks presented in Fig. 6b.

We also explored learning a stochastic error model for this example. The neural networks used in the stochastic error models have 5 neurons in the first hidden layer and one neuron in the second layer. The hidden layer neurons have a sigmoid activation function, and the output layer neurons have no nonlinear activation function. For the stochastic error model with an additive noise term, 19



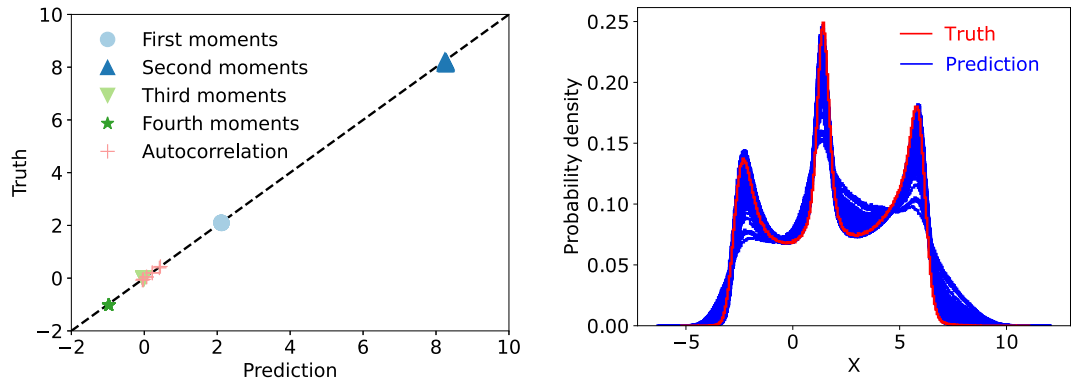
(a) Data comparison (b) Invariant measure

Fig. 6. Indirect training of the error model ($c = 3$) using a non-local deterministic model, with results for (a) the first four moments and autocorrelation, and (b) the invariant measure.



(a) Data comparison (b) Invariant measure

Fig. 7. Indirect training of the error model ($c = 3$) using stochastic model with additive noise, with the results for (a) the first four moments and autocorrelation, and (b) the invariant measure.



(a) Data comparison (b) Invariant measure

Fig. 8. Indirect training of the error model ($c = 3$) using a stochastic model with multiplicative noise, with the results for (a) the first four moments and autocorrelation, and (b) the invariant measure.

parameters need to be estimated. For the stochastic error model with a multiplicative noise term, 36 parameters need to be estimated. Fig. 7 presents the results of the calibrated system with an additive stochastic error model. Compared to the results for deterministic error models in Figs. 5 and 6, we can see that the invariant measure of the calibrated system agrees better with the true system in Fig. 7b. We further test the stochastic error model by also learning a state-dependent diffusion coefficient. As shown in Fig. 8b, the calibrated system achieves better agreement with the invariant measure of the true system, which confirms that increased flexibility in the stochastic error model can help achieve improved predictive performance via training against indirect data. It should be noted

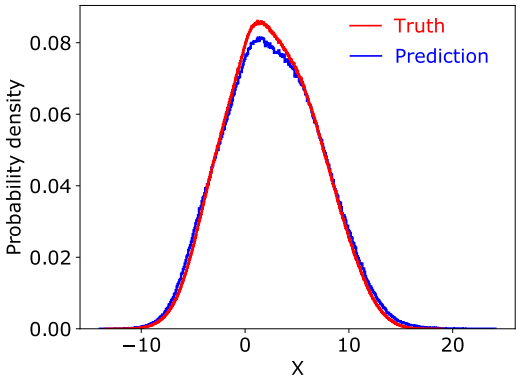
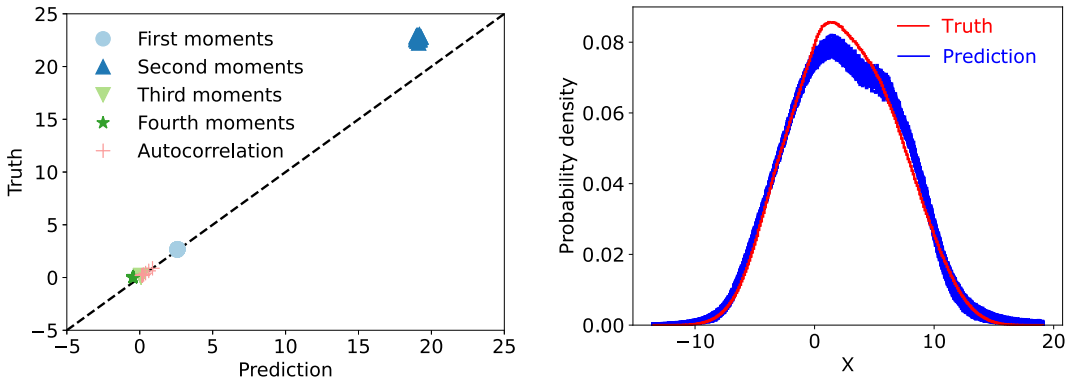


Fig. 9. Invariant measures via direct training of the error model ($c = 10$) for the single-scale Lorenz 96 system.



(a) Data comparison (b) Invariant measure

Fig. 10. Results from trained error model ($c = 10$) for the single-scale Lorenz 96 system *without* energy conservation constraint, including (a) first four moments and autocorrelation, and (b) invariant measures.

that Figs. 6 to 8 do not display a single converged invariant measure. This arises because the calibrated parameters vary across the ensemble; as a result, we obtain a family of structural error models that all fit the data with similar accuracy. We surmise that this is caused by the fact that the indirect data contain only partial information about the invariant measure. Nonetheless, the fits are all far superior to those obtained with the local deterministic model.

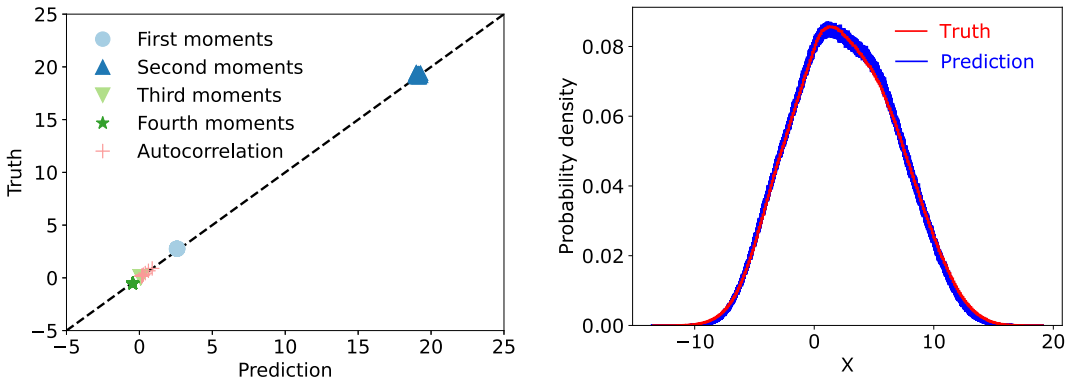
4.2.2. *Lorenz 96 single-scale model*

We studied the Lorenz 96 single-scale system to learn the quadratic term as an error model. Using this numerical example, we demonstrate the merit of combined use of direct and indirect data and the importance of enforcing physical constraints. More specifically, we assume that the quadratic term of the true system in (25) is unknown and then calibrate an error model $\delta(X_{k-2}, X_{k-1}, X_{k+1}, X_{k+2})$ as in (26). The size of the direct data is 36000, and the size of the indirect data is 154. The numerical examples for the single-scale Lorenz 96 system are as follows:

(i) We train a fully-connected neural network as an error model $\delta(\cdot)$ using time-series of x_k and the true quadratic term as direct data. The results are presented in Fig. 9.
(ii) We train a fully-connected neural network as an error model $\delta(\cdot)$ using indirect data (first to fourth moments of the state variable x_k and the autocorrelation of x_k). The results are presented in Fig. 10.
(iii) We train a fully-connected neural network as an error model $\delta(\cdot)$ using indirect data (first to fourth moments of the state variable x_k and the autocorrelation of x_k) and the energy conservation constraint in (12). The results are presented in Fig. 11.

Fig. 9 presents the comparison of invariant measures between the calibrated system and the true system when a fully connected neural network is used as an error model and is learned from direct data. As seen in Fig. 9, the calibrated system using direct data still differs noticeably in the invariant measure, indicating a difference from the long-time behavior of the true system.

In order to improve the results of Fig. 9, we incorporate indirect data about the x_k. Specifically, we employ the trained model using direct data as the prior mean of EKI, and we set the prior standard deviation as 30% of the mean values for each unknown coefficient of the error model. We then use EKI to calibrate the error model based on the first four moments of X_k and the ten sampled points from the autocorrelation function of X_k. Without enforcing energy conservation of the error model, we can see in



(a) Data comparison (b) Invariant measure

Fig. 11. Results from trained error model ($c = 10$) for the single-scale Lorenz 96 system *with* energy conservation constraint, including (a) first four moments and autocorrelation, and (b) invariant measures.

Fig. 10b that the performance of the calibrated model is similar to the calibrated system using direct data in Fig. 9. On the other hand, we also performed the calibration based on indirect data and enforced the energy conservation of the error model as discussed in Section 2.2.2. Considering that the true system conserves quadratic energy at every time, the energy constraint with the form in Eq. (12) is used. As shown in Fig. 11, the calibrated error model with energy conservation leads to a modeled system that better fits the training data and achieves a good agreement with the invariant measure of the true system.

5. Human glucose-insulin model as illustrative example

The examples based on Lorenz systems in Section 4 illustrate the construction and calibration of internal error models for chaotic and ergodic systems, which are representative of a wide range of applications, e.g., in weather forecasting and climate change prediction, design of hypersonic vehicles, or the control of unmanned vehicles in a turbulent environment. However, there is another large class of applications for which the systems are non-ergodic and the goal is to accurately predict the time series of the system evolution. To achieve this goal for non-ergodic systems, we can no longer neglect the impact of the initial condition. We use a human glucose-insulin model as an example of a non-ergodic system and illustrate how the internal error model can be calibrated based on time series data, using EKI combined with data assimilation techniques.

5.1. Ultradian model

We consider the ultradian model of the glucose-insulin system proposed in [123]. Its primary state variables are the plasma glucose concentration, G, the plasma insulin concentration, I_p, and the interstitial insulin concentration, I_i. These three state variables are augmented with a three stage delay (h_1, h_2, h_3) which encodes a non-linear delayed hepatic glucose response to plasma insulin levels. The resulting ordinary differential equations have the form:

$$\frac{dI_p}{dt} = f_1(G) - E\left(\frac{I_p}{V_p} - \frac{I_i}{V_i}\right) - \frac{I_p}{t_p} \tag{27a}$$

$$\frac{dI_i}{dt} = E\left(\frac{I_p}{V_p} - \frac{I_i}{V_i}\right) - \frac{I_i}{t_i} \tag{27b}$$

$$\frac{dG}{dt} = -f_2(G) - f_3(I_i)G + f_4(h_3) + m_G(t) \tag{27c}$$

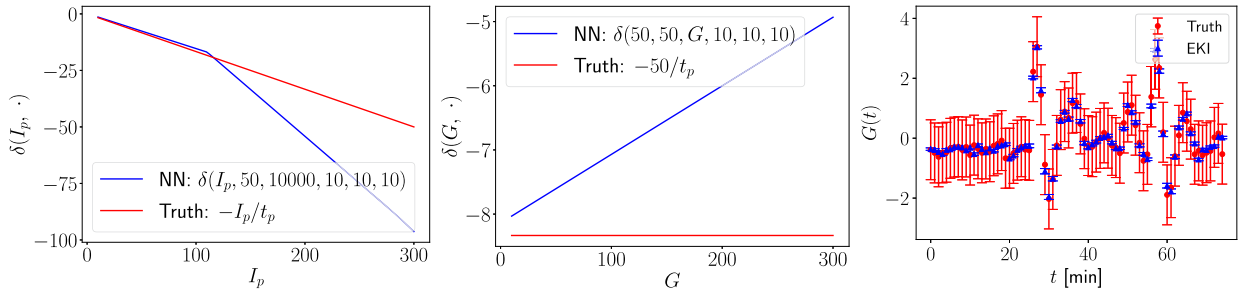
$$\frac{dh_1}{dt} = \frac{1}{t_d}(I_p - h_1) \tag{27d}$$

$$\frac{dh_2}{dt} = \frac{1}{t_d}(h_1 - h_2) \tag{27e}$$

$$\frac{dh_3}{dt} = \frac{1}{t_d}(h_2 - h_3) \tag{27f}$$

Here $m_G(t)$ represents a known rate of ingested carbohydrates appearing in the plasma, $f_1(G)$ represents the rate of glucose-dependent insulin production, $f_2(G)$ represents insulin-independent glucose utilization, $f_3(I_i)G$ represents insulin-dependent glucose utilization, and $f_4(h_3)$ represents delayed insulin-dependent hepatic glucose production. The functional forms of these parameterized processes are shown in Appendix B.

We represent a potential structural error in this model by removing the final linear term in (27a) (i.e., $t_p = \infty$). We then aim to identify a neural-network-based additive correction term for (27a), denoted by $\delta(I_p, I_i, G, h_1, h_2, h_3; \theta)$. We calibrate $\delta(\cdot\ ;\ \theta)$ using



(a) Trained model: δ vs I_p (b) Trained model: δ vs G (c) Glucose trajectory fits: $G(t)$

Fig. 12. Partial, noisy, transient timeseries data were used to calibrate model error. Results in (a-b) show the trained neural network error model for two specific input scenarios and (c) shows next-step predictions of G from sequential data assimilation using the trained model. Figures (a-b) plot the model δ (a function of all 6 state variables) for 5 fixed states while varying I_p and G, respectively.

partial, noisy observations of the physiologic state at times $\{t_k\}_{k=0}^K$. Specifically, we observe only the glucose state, $G(t)$, (it is the only state variable that is reliably measurable in patients) and add i.i.d. unit-Gaussian noise. Thus the observation operator $\mathcal{H}$ in Eq. (2) is defined through observation matrix $H \in \mathbb{R}^{1\times 6}$. To be specific $HX(t) = X_3(t)$ where $X(t) = [I_p(t), I_i(t), G(t), h_1(t), h_2(t), h_3(t)]^T$ and then $\mathcal{H}[X] = \{HX(t_k)\}_{k=0}^K$. It should be noted that the observation matrix H projects between the full system state space and the observable space and may be used in the definition of a gain matrix K for Kalman methods and generalizations such as 3DVAR (See Table 9.1 in [124]).

We refer to the observed data as $y = \left[y(t_1), \cdots, y(t_K)\right]$ and to the solution of the entire six dimensional dynamical system at the observation times as $\left[x(t_1), \cdots, x(t_K)\right]$. We infer parameters θ by minimizing a variant of the loss defined in Eq. (14) in which the forward model itself depends on y:

$$\mathcal{L}(\theta) = \frac{1}{2}\left|y - \mathcal{G}(\theta;\ y)\right|_\Sigma^2. \tag{28}$$

Specifically $\mathcal{G}_k(\theta;\ y)$ uses sequential data assimilation to combine previous observations $\left[y(t_1), \cdots, y(t_{k-1})\right]$ with parameters θ to produce a filtered estimate for $x(t_{k-1})$ and, from this, predict the observation $y(t_k)$; see section 2.2.3. The minimization of Eq. (28) is performed using EKI, where each evaluation of $\mathcal{G}(\theta; y)$ involves running a data assimilation sub-routine (3DVAR [124] with constant gain $K = [0, 0, 1, 0, 0, 0]$).

5.2. Numerical results for ultradian model

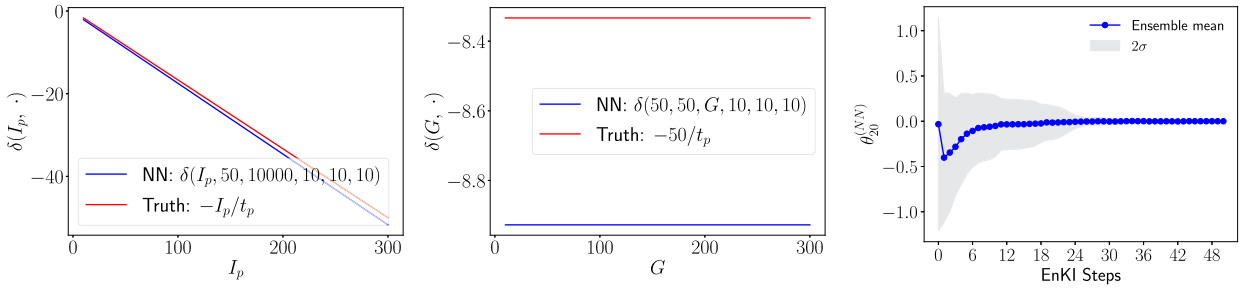
Through the presented numerical experiments in this section, we find that:

1. Model corrections can be learned for non-ergodic, non-stationary systems from noisy, partial observations of transient dynamics by applying DA as a sub-routine within the model inference pipeline (Fig. 12).
2. Key structural components of model error terms can be identified despite being unobserved (Fig. 12a).
3. Learned model error structure can exhibit inaccuracies outside of the observed data distribution and when models are insensitive to these inaccuracies (Fig. 12b).
4. Appropriate choice of regularization can improve the inference of otherwise un-identifiable model error structure (Fig. 13).

Fig. 12 presents results from a trained model obtained by minimizing (28) via EKI. Fig. 12c shows the transient trajectory data used to calibrate the model error, along with the obtained next-step predictions, which are seen to be highly accurate. Fig. 12a shows that the neural network captures the true missing linear term $-I_p/t_p$ quite accurately for $I_p \in [0, 110]$, but it is inaccurate for larger values of I_p. While only G is observed (not I_p), we note that indeed $I_p(t) < 110$ for solutions of the true model. In other words, under the true model, the error term is only identifiable within the domain $[0, 110]$. Fig. 12b shows the neural network as a function of an inactive input G (an input upon which the true missing term does not depend); instead, it learns a linear function of G with a small slope. Other results (not shown) indicate that the neural network learned substantial dependencies on inactive variables. This suggests that sparsity constraints may be useful in these experiments.

Fig. 13 presents results from a trained model obtained by minimizing (28) via sparse EKI [74]. Compared to standard EKI, sparse EKI incorporates sparsity into the coefficients of the trained model via ℓ_1-constrained optimization. It can be seen in Fig. 13a that the neural network captures the linear pattern of the true term $-I_p/t_p$ and achieves overall better agreement than the results in Fig. 12a. In addition, Fig. 13b shows that the neural network learns to correctly ignore G in all plotted scenarios. An example of sparse EKI driving initially non-zero error model coefficients to zero is presented in Fig. 13c, for which the standard EKI would lead to larger variability with even more EKI iterations. The sparse EKI also drives some redundant coefficients of the error model to values very close to zero.

This example highlights to importance of sparsity constraints in learning error models, to avoid learning nonzero error terms that may otherwise arise because the error has zero or small projection on the output data of the model.



(a) Trained model: δ vs I_p (b) Trained model: δ vs G (c) Inferred parameter $\theta_{20}^{(\mathrm{NN})}$

Fig. 13. Partial, noisy, transient timeseries data were used to calibrate model error with sparse EKI. Results in (a-b) show the trained neural network error model for two specific input scenarios and (c) shows an example of how sparse EKI drives initially non-zero error model coefficients to zero. Figures (a-b) plot the model δ (a function of all 6 state variables) for 5 fixed states while varying I_p and G, respectively.

6. Conclusions

Complex nonlinear dynamics and/or a large number of degrees of freedom are present in many systems, for example, in physiological models of the human body and turbulent flows around an airplane or in the wake of wind turbines. Typically, closure models are needed if some dynamically important degrees of freedom cannot be resolved by direct numerical simulations. Without a clear scale separation between resolved and unresolved degrees of freedom, most existing models, which are, semi-empirical, deterministic and local and are calibrated with only limited amounts of data, are not sophisticated enough to capture the true dynamics of the system. Incorporating models of structural errors that are informed by data can be an effective way to build on domain knowledge while using data more extensively.

We have summarized some key aspects of learning structural error models from data, including the construction of error models and their calibration. In doing so, we have provided guidelines about how to learn error models for complex dynamical systems, ranging from key insights about the incorporation of sparsity constraints ("do no harm") and physical constraints, to advanced aspects such as the combined use of direct and indirect data and the merit of using non-local/stochastic error models. By addressing these varied aspects in a systematic manner, our goal has been to inspire further applied, methodological and theoretical research in this area.

CRediT authorship contribution statement

Jin-Long Wu: Writing – review & editing, Writing – original draft, Software, Methodology, Investigation, Conceptualization. **Matthew E. Levine:** Writing – review & editing, Software, Methodology, Investigation, Conceptualization. **Tapio Schneider:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization. **Andrew Stuart:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the link to my data/code at the Attach File step.

Acknowledgements

TS, AMS, and JW were supported by Schmidt Sciences, LLC. AMS is also supported by a Department of Defense Vannevar Bush Faculty Fellowship, and by the SciAI Center, funded by the Office of Naval Research (ONR), under Grant Number N00014-23-1-2729. MEL was supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1745301 and by funding from the Eric and Wendy Schmidt Center at the Broad Institute of MIT and Harvard.

All codes are available at: https://github.com/jinlong83/Learning-Structural-Errors.git.

Appendix A. Ensemble Kalman inversion

The use of ensemble Kalman based methods for parameter calibration and the solution of inverse problems, and history of this subject, is overviewed in [Section 4] [125]. To be concrete we will concentrate on a particular variant of the methodology, sometimes termed Ensemble Kalman inversion (EKI). This is a specific ensemble-based, gradient-free optimization scheme that was proposed

and studied in [126]; we emphasize that other ensemble Kalman based methods share the core desirable attributes of EKI, namely that it is derivative-free, is effective with relatively few evaluations of the forward model $\mathcal{G}$ and is robust to the presence of noise in the evaluations of $\mathcal{G}$.

The core task of EKI is equivalent to a quadratic optimization problem, which facilitates adding linear equality and inequality constraints [127]. To explain the details of EKI, we first introduce a new variable $w = \mathcal{G}(\theta)$ and variables v and $g(v)$:

$$\begin{aligned} v &= (\theta, w)^\top, \\ g(v) &= (\theta, \mathcal{G}(\theta))^\top. \end{aligned} \tag{A.1}$$

Using these variables we formulate the following noisily observed dynamical system:

$$\begin{aligned} v_{m+1} &= g(v_m) \\ y_{m+1} &= H v_{m+1} + \eta_{m+1}. \end{aligned} \tag{A.2}$$

Here $H = [0, I], H^\perp = [I, 0]$, and hence $Hv = w, H^\perp v = \theta$. In this setting, $\{v_m\}$ is the state and $\{y_m\}$ are the data. The objective is to estimate $H^\perp v_m = \theta_m$ from $\{y_\ell\}_{\ell=1}^m$ and to do so iteratively with respect to m. In practice we only have one data point y and not a sequence y_m; we address this issue in what follows below.

The EKI methodology creates an ensemble $\{v_m^{(j)}\}_{j=1}^J$ defined iteratively in m as follows:

$$\begin{aligned} L_m^{(j)}(v) &:= \frac{1}{2}\left|y_{m+1}^{(j)} - Hv\right|_\Gamma^2 + \frac{1}{2}\left|v - g\left(v_m^{(j)}\right)\right|_{C_m^{gg}}^2, \\ v_{m+1}^{(j)} &= \arg\min_v L_m^{(j)}(v). \end{aligned} \tag{A.3}$$

The matrix C^{gg} is the empirical covariance of $\{g(v_m^{(j)})\}_{j=1}^J$. The data $y_{m+1}^{(j)}$ is either fixed so that $y_{m+1}^{(j)} \equiv y$ or created by adding random draws to y from the distribution of the η, independently for all m and j. At each step, m ensemble parameter estimates indexed by $j = 1, \cdots, J$ are found from $\theta_m^{(j)} = H^\perp v_m^{(j)}$.

Using the fact that $v = (\theta, w)^T$, the minimizer $v_{m+1}^{(j)}$ in (A.3) decouples to give the update formula

$$\theta_{m+1}^{(j)} = \theta_m^{(j)} + C_m^{\theta\mathcal{G}}\left(C_m^{\mathcal{G}\mathcal{G}} + \Gamma\right)^{-1}\left(y_{m+1}^{(j)} - \mathcal{G}(\theta_m^{(j)})\right); \tag{A.4}$$

here the matrix $C_m^{\mathcal{G}\mathcal{G}}$ is the empirical covariance of $\{\mathcal{G}(\theta_m^{(j)})\}_{j=1}^J$, while matrix $C_m^{\theta\mathcal{G}}$ is the empirical cross-covariance of $\{\theta_m^{(j)}\}_{j=1}^J$ with $\{\mathcal{G}(\theta_m^{(j)})\}_{j=1}^J$.

To impose sparsity on the solution of θ from EKI, we solve the following constrained optimization problem after each EKI update step:

$$\begin{aligned} L_m^{(j)}(v, \lambda) &:= \frac{1}{2}\left|y_{m+1}^{(j)} - Hv\right|_\Gamma^2 + \frac{1}{2}\left|v - g\left(v_m^{(j)}\right)\right|_{C_m^{gg}}^2, \\ v_{m+1}^{(j)} &= \arg\min_{v\in\mathfrak{B}} L_m^{(j)}(v), \end{aligned} \tag{A.5}$$

where

$$\mathfrak{B} = \{v : |H^\perp v|_{\ell_1} \le \gamma\}. \tag{A.6}$$

We also employ the thresholding function $\mathcal{T}$ on vectors defined by

$$\mathcal{T}(\theta_i) = \begin{cases} 0, & \text{if } |\theta_i| < \sqrt{2\lambda} \\ \theta_i, & \text{otherwise,} \end{cases} \tag{A.7}$$

to threshold those θ_i with values close to zero, after having solved the constrained optimization problem in (A.5). Such a thresholding step after the ℓ_1-constrained optimization in (A.5) is equivalent to adding ℓ_0 constraint. More details about imposing sparsity into EKI can be found in [74].

For the multiscale Lorenz 96 example with $c = 10$ and the single scale Lorenz 96 example, 100 ensembles are used and the number of EKI iterations is 20. The multiscale Lorenz 96 example with $c = 3$ is more challenging to train, for which 200 ensembles are used and the number of EKI iterations is 30. The single-scale Lorenz 96 example uses 200 ensembles and the number of EKI iterations is 10. The ultradian model example is not chaotic while the convergence of standard EKI is relatively slow, for which 50 ensembles are used and the number of EKI iterations is 50. The noise level η is estimated by running ensemble simulations of the true system and calculating the covariance of the data y. In practice, only the diagonal of the estimated covariance matrix is kept.

Appendix B. Application to a glucose-insulin model

The functional forms of the parameterized processes in Eq. (27) are as follows:

$$f_1(G) = \frac{R_m}{1 + \exp\left(-\frac{G}{V_g c_1} + a_1\right)} \qquad \text{(rate of insulin production)} \tag{B.1}$$

$$f_2(G) = U_1\left(1 - \exp\left(-\frac{G}{C_2 V_g}\right)\right) \qquad \text{(insulin-independent glucose utilization)} \tag{B.2}$$

$$f_3(I_i) = \frac{1}{C_3 V_g}\left(U_0 + \frac{U_m - U_0}{1 + (\kappa I_i)^{-\beta}}\right) \qquad \text{(insulin-dependent glucose utilization)} \tag{B.3}$$

$$f_4(h_3) = \frac{R_g}{1 + \exp\left(\alpha\left(\frac{h_3}{C_5 V_p} - 1\right)\right)} \qquad \text{delayed insulin-dependent glucose utilization} \tag{B.4}$$

$$\kappa = \frac{1}{C_4}\left(\frac{1}{V_i} - \frac{1}{E t_i}\right). \tag{B.5}$$

The uptake of carbohydrates is modelled by the function

$$m_G(t) = \sum_{j=1}^{N(t)} \frac{m_j k}{60\ \text{min}} \exp(k(t_j - t)), \quad N(t) = \#\{t_j < t\} \tag{B.6}$$

in which N meals occur at times $\{t_j\}_{j=1}^{N}$ (in minutes), with carbohydrate composition $\{m_j\}_{j=1}^{N}$ (note that these typically differ from observation times).

References

[1] T. Schneider, J. Teixeira, C.S. Bretherton, F. Brient, K.G. Pressel, C. Schär, A.P. Siebesma, Climate goals and computing the future of clouds, Nat. Clim. Change 7 (2017) 3–5.
[2] J. Smagorinsky, General circulation experiments with the primitive equations. I. The basic experiment, Mon. Weather Rev. 91 (1963) 99–164.
[3] A. Holtslag, C.-H. Moeng, Eddy diffusivity and countergradient transport in the convective atmospheric boundary layer, J. Atmos. Sci. 48 (1991) 1690–1698.
[4] J. Ling, A. Kurzawski, J. Templeton, Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, J. Fluid Mech. 807 (2016) 155–166.
[5] J. Weatheritt, R. Sandberg, A novel evolutionary algorithm applied to algebraic modifications of the RANS stress–strain relationship, J. Comput. Phys. 325 (2016) 22–37.
[6] S. Rasp, M.S. Pritchard, P. Gentine, Deep learning to represent subgrid processes in climate models, Proc. Natl. Acad. Sci. (2018).
[7] R. Maulik, O. San, A. Rasheed, P. Vedula, Subgrid modelling for two-dimensional turbulence using neural networks, J. Fluid Mech. 858 (2019) 122–144.
[8] K. Duraisamy, G. Iaccarino, H. Xiao, Turbulence modeling in the age of data, Annu. Rev. Fluid Mech. 51 (2019) 357–377.
[9] Y. Zhao, H.D. Akolekar, J. Weatheritt, V. Michelassi, R.D. Sandberg, RANS turbulence model development using CFD-driven machine learning, J. Comput. Phys. 411 (2020) 109413.
[10] M. Schmelzer, R.P. Dwight, P. Cinnella, Discovery of algebraic Reynolds-stress models using sparse symbolic regression, Flow Turbul. Combust. 104 (2020) 579–603.
[11] S.L. Brunton, B.R. Noack, P. Koumoutsakos, Machine learning for fluid mechanics, Annu. Rev. Fluid Mech. 52 (2020) 477–508.
[12] M.C. Kennedy, A. O'Hagan, Bayesian calibration of computer models, J. R. Stat. Soc., Ser. B, Stat. Methodol. 63 (2001) 425–464.
[13] D. Higdon, M. Kennedy, J.C. Cavendish, J.A. Cafeo, R.D. Ryne, Combining field data and computer simulations for calibration and prediction, SIAM J. Sci. Comput. 26 (2004) 448–466.
[14] S. Wang, W. Chen, K.-L. Tsui, Bayesian validation of computer models, Technometrics 51 (2009) 439–451.
[15] J. Brynjarsdòttir, A. O'Hagan, Learning about physical parameters: the importance of model discrepancy, Inverse Probl. 30 (2014) 114007.
[16] S. Trehan, K.T. Carlberg, L.J. Durlofsky, Error modeling for surrogates of dynamical systems using machine learning, Int. J. Numer. Methods Eng. 112 (2017) 1801–1827.
[17] D. Calvetti, M. Dunlop, E. Somersalo, A. Stuart, Iterative updating of model error for Bayesian inversion, Inverse Probl. 34 (2018) 025008.
[18] M. Strong, J.E. Oakley, When is a model good enough? Deriving the expected value of model improvement via specifying internal model discrepancies, SIAM/ASA J. Uncertain. Quantificat. 2 (2014) 106–125.
[19] Y. He, D. Xiu, Numerical strategy for model correction using physical constraints, J. Comput. Phys. 313 (2016) 617–634.
[20] K. Sargsyan, X. Huan, H.N. Najm, Embedded model error representation for Bayesian model calibration, Int. J. Uncertain. Quantificat. 9 (2019).
[21] M. Emory, R. Pecnik, G. Iaccarino, Modeling structural uncertainties in Reynolds-averaged computations of shock/boundary layer interactions, in: 49th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, 2011, p. 479.
[22] T.A. Oliver, R.D. Moser, Bayesian uncertainty quantification applied to RANS turbulence models, J. Phys. Conf. Ser. 318 (2011) 042032.
[23] S.H. Cheung, T.A. Oliver, E.E. Prudencio, S. Prudhomme, R.D. Moser, Bayesian uncertainty analysis with applications to turbulence modeling, Reliab. Eng. Syst. Saf. 96 (2011) 1137–1149.
[24] H. Xiao, J.-L. Wu, J.-X. Wang, R. Sun, C. Roy, Quantifying and reducing model-form uncertainties in Reynolds-averaged Navier–Stokes simulations: a data-driven, physics-informed Bayesian approach, J. Comput. Phys. 324 (2016) 115–136.
[25] P. Pernot, F. Cailliez, A critical review of statistical calibration/prediction models handling data inconsistency and model inadequacy, AIChE J. 63 (2017) 4642–4665.
[26] J.-X. Wang, J.-L. Wu, H. Xiao, Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data, Phys. Rev. Fluids 2 (2017) 034603.
[27] J.-L. Wu, H. Xiao, E. Paterson, Physics-informed machine learning approach for augmenting turbulence models: a comprehensive framework, Phys. Rev. Fluids 3 (2018) 074602.

[28] H. Schaeffer, R. Caflisch, C.D. Hauck, S. Osher, Sparse dynamics for partial differential equations, Proc. Natl. Acad. Sci. 110 (2013) 6634–6639.
[29] S.H. Rudy, S.L. Brunton, J.L. Proctor, J.N. Kutz, Data-driven discovery of partial differential equations, Sci. Adv. 3 (2017) e1602614.
[30] H. Schaeffer, Learning partial differential equations via data discovery and sparse optimization, Proc. R. Soc. A, Math. Phys. Eng. Sci. 473 (2017) 20160446.
[31] L. Zanna, T. Bolton, Data-driven equation discovery of ocean mesoscale closures, Geophys. Res. Lett. 47 (2020) e2020GL088376.
[32] B.M. de Silva, D.M. Higdon, S.L. Brunton, J.N. Kutz, Discovery of physics from data: universal laws and discrepancies, Front. Artif. Intell. 3 (2020) 25.
[33] P. Lemos, N. Jeffrey, M. Cranmer, S. Ho, P. Battaglia, Rediscovering orbital mechanics with machine learning, 2022.
[34] N.D. Brenowitz, C.S. Bretherton, Prognostic validation of a neural network unified physics parameterization, Geophys. Res. Lett. 45 (2018) 6289–6298.
[35] L. Zanna, T. Bolton, Deep learning of unresolved turbulent ocean processes in climate models, in: Deep Learning for the Earth Sciences: A Comprehensive Approach to Remote Sensing, Climate Science, and Geosciences, 2021, pp. 298–306.
[36] C.S. Bretherton, B. Henn, A. Kwa, N.D. Brenowitz, O. Watt-Meyer, J. McGibbon, W.A. Perkins, S.K. Clark, L. Harris, Correcting coarse-grid weather and climate models by machine learning from global storm-resolving simulations, J. Adv. Model. Earth Syst. 14 (2022) e2021MS002794.
[37] R. Zwanzig, Nonequilibrium Statistical Mechanics, Oxford University Press, 2001.
[38] V. Lucarini, R. Blender, C. Herbert, F. Ragone, S. Pascale, J. Wouters, Mathematical and physical ideas for climate science, Rev. Geophys. 52 (2014) 809–859.
[39] C.L. Franzke, T.J. O'Kane, J. Berner, P.D. Williams, V. Lucarini, Stochastic climate theory and modeling, Wiley Interdiscip. Rev.: Clim. Change 6 (2015) 63–78.
[40] T. Palmer, Stochastic weather and climate models, Nat. Rev. Phys. 1 (2019) 463–471.
[41] X.-H. Zhou, J. Han, H. Xiao, Learning nonlocal constitutive models with neural networks, Comput. Methods Appl. Mech. Eng. 384 (2021) 113927.
[42] J. Zhuang, D. Kochkov, Y. Bar-Sinai, M.P. Brenner, S. Hoyer, Learned discretizations for passive scalar advection in a two-dimensional turbulent flow, Phys. Rev. Fluids 6 (2021) 064605.
[43] Q. Du, M. Gunzburger, R.B. Lehoucq, K. Zhou, Analysis and approximation of nonlocal diffusion problems with volume constraints, SIAM Rev. 54 (2012) 667–696.
[44] M. D'Elia, M. Gunzburger, Identification of the diffusion parameter in nonlocal steady diffusion problems, Appl. Math. Optim. 73 (2016) 227–249.
[45] G. Pang, M. D'Elia, M. Parks, G.E. Karniadakis, nPINNs: nonlocal physics-informed neural networks for a parametrized nonlocal universal Laplacian operator. Algorithms and applications, J. Comput. Phys. 422 (2020) 109760.
[46] C. Ma, J. Wang, et al., Model reduction with memory and the machine learning of dynamical systems, arXiv preprint, arXiv:1808.04258, 2018.
[47] Q. Wang, N. Ripamonti, J.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.
[48] K.K. Lin, F. Lu, Data-driven model reduction, Wiener projections, and the Koopman-Mori-Zwanzig formalism, J. Comput. Phys. 424 (2021) 109864.
[49] A.-T.G. Charalampopoulos, T.P. Sapsis, Machine-learning energy-preserving nonlocal closures for turbulent fluid flows and inertial tracers, arXiv preprint, arXiv:2102.07639, 2021.
[50] S.L. Brunton, J.L. Proctor, J.N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proc. Natl. Acad. Sci. 113 (2016) 3932–3937.
[51] M. Levine, A. Stuart, A framework for machine learning of model error in dynamical systems, Commun. Am. Math. Soc. 2 (2022) 283–344.
[52] T. Schneider, A.M. Stuart, J.-L. Wu, Learning stochastic closures using ensemble Kalman inversion, Trans. Math. Appl. 5 (2021) tnab003.
[53] X.-L. Zhang, H. Xiao, X. Luo, G. He, Ensemble Kalman method for learning turbulence models from indirect observation data, J. Fluid Mech. 949 (2022) A26.
[54] E.N. Lorenz, Predictability: a problem partly solved, in: Proc. Seminar on Predictability, vol. 1, 1996.
[55] I. Fatkullin, E. Vanden-Eijnden, A computational strategy for multiscale systems with applications to Lorenz 96 model, J. Comput. Phys. 200 (2004) 605–638.
[56] R.J. Little, D.B. Rubin, Statistical Analysis with Missing Data, vol. 793, John Wiley & Sons, 2019.
[57] M. Bocquet, J. Brajard, A. Carrassi, L. Bertino, Data assimilation as a learning tool to infer ordinary differential equation representations of dynamical models, Nonlinear Process. Geophys. 26 (2019) 143–162.
[58] Y. Chen, D. Sanz-Alonso, R. Willett, Autodifferentiable ensemble Kalman filters, SIAM J. Math. Data Sci. 4 (2022) 801–833.
[59] A. Beck, D. Flad, C.-D. Munz, Deep neural networks for data-driven LES closure models, J. Comput. Phys. 398 (2019) 108910.
[60] N.D. Brenowitz, T. Beucler, M. Pritchard, C.S. Bretherton, Interpreting and stabilizing machine-learning parametrizations of convection, J. Atmos. Sci. 77 (2020) 4357–4375.
[61] J. Yuval, P.A. O'Gorman, C.N. Hill, Use of neural networks for stable, accurate and physically consistent parameterization of subgrid atmospheric processes with good performance at reduced precision, Geophys. Res. Lett. 48 (2021) e2020GL091363.
[62] T. Beucler, M. Pritchard, S. Rasp, J. Ott, P. Baldi, P. Gentine, Enforcing analytic constraints in neural networks emulating physical systems, Phys. Rev. Lett. 126 (2021) 098302.
[63] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, S.-J. Chung, Neural lander: stable drone landing control using learned dynamics, in: 2019 International Conference on Robotics and Automation (ICRA), IEEE, 2019, pp. 9784–9790.
[64] R.J. Boys, D.J. Wilkinson, T.B. Kirkwood, Bayesian inference for a discretely observed stochastic kinetic model, Stat. Comput. 18 (2008) 125–135.
[65] T. Schneider, S. Lan, A. Stuart, J. Teixeira, Earth system modeling 2.0: a blueprint for models that learn from observations and targeted high-resolution simulations, Geophys. Res. Lett. 44 (2017) 12–396.
[66] Y. Pokern, A.M. Stuart, P. Wiberg, Parameter estimation for partially observed hypoelliptic diffusions, J. R. Stat. Soc., Ser. B, Stat. Methodol. 71 (2009) 49–73.
[67] X.-L. Meng, D. Van Dyk, The EM algorithm—an old folk-song sung to a fast new tune, J. R. Stat. Soc., Ser. B, Stat. Methodol. 59 (1997) 511–567.
[68] O.R. Dunbar, A.B. Duncan, A.M. Stuart, M.-T. Wolfram, Ensemble inference methods for models with noisy and expensive likelihoods, SIAM J. Appl. Dyn. Syst. 21 (2022) 1539–1572.
[69] C. Chen, J.-L. Wu, Operator learning for continuous spatial-temporal model with a hybrid optimization scheme, arXiv preprint, arXiv:2311.11798, 2023.
[70] M. Bollhofer, A. Eftekhari, S. Scheidegger, O. Schenk, Large-scale sparse inverse covariance matrix estimation, SIAM J. Sci. Comput. 41 (2019) A380–A401.
[71] S.J. Greybush, E. Kalnay, T. Miyoshi, K. Ide, B.R. Hunt, Balance and ensemble Kalman filter localization techniques, Mon. Weather Rev. 139 (2011) 511–522.
[72] X. Tong, M. Morzfeld, Localized ensemble Kalman inversion, Inverse Probl. 39 (2023) 064002.
[73] S. Liu, S. Reich, X.T. Tong, Dropout ensemble Kalman inversion for high dimensional inverse problems, arXiv preprint, arXiv:2308.16784, 2023.
[74] T. Schneider, A.M. Stuart, J.-L. Wu, Ensemble Kalman inversion for sparse learning of dynamical systems from time-averaged data, J. Comput. Phys. 470 (2022) 111559.
[75] E. Cleary, A. Garbuno-Inigo, S. Lan, T. Schneider, A.M. Stuart, Calibrate, emulate, sample, J. Comput. Phys. 424 (2021) 109716.
[76] H. Ribera, S. Shirman, A. Nguyen, N. Mangan, Model selection of chaotic systems from data with hidden variables using sparse data assimilation, Chaos, Interdiscip. J. Nonlinear Sci. 32 (2022).
[77] J. Brajard, A. Carrassi, M. Bocquet, L. Bertino, Combining data assimilation and machine learning to infer unresolved scale parametrization, Philos. Trans. R. Soc. A 379 (2021) 20200086.
[78] D.L. Donoho, Compressed sensing, IEEE Trans. Inf. Theory 52 (2006) 1289–1306.
[79] C.K. Williams, C.E. Rasmussen, Gaussian Processes for Machine Learning, vol. 2, MIT Press, Cambridge, MA, 2006.
[80] B. Hamzi, H. Owhadi, Learning dynamical systems from data: a simple cross-validation perspective, part I: parametric kernel flows, Phys. D: Nonlinear Phenom. 421 (2021) 132817.
[81] M. Darcy, B. Hamzi, J. Susiluoto, A. Braverman, H. Owhadi, Learning dynamical systems from data: a simple cross-validation perspective, part II: nonparametric kernel flows, preprint, 2021.

[82] J. Lee, E. De Brouwer, B. Hamzi, H. Owhadi, Learning dynamical systems from data: a simple cross-validation perspective, part III: irregularly-sampled time series, Phys. D: Nonlinear Phenom. (2022) 133546.
[83] A. Rahimi, B. Recht, Random features for large-scale kernel machines, in: J. Platt, D. Koller, Y. Singer, S. Roweis (Eds.), Advances in Neural Information Processing Systems, vol. 20, Curran Associates, Inc., 2008.
[84] D.J. Wilkinson, Stochastic modelling for quantitative description of heterogeneous biological systems, Nat. Rev. Genet. 10 (2009) 122–133.
[85] G.O. Roberts, O. Stramer, On inference for partially observed nonlinear diffusion models using the Metropolis–Hastings algorithm, Biometrika 88 (2001) 603–621.
[86] G.O. Roberts, O. Stramer, Langevin diffusions and Metropolis-Hastings algorithms, Methodol. Comput. Appl. Probab. 4 (2002) 337–357.
[87] O. Papaspiliopoulos, G.O. Roberts, O. Stramer, Data augmentation for diffusions, J. Comput. Graph. Stat. 22 (2013) 665–688.
[88] L. Zhang, P.A. Mykland, Y. Aït-Sahalia, A tale of two time scales: determining integrated volatility with noisy high-frequency data, J. Am. Stat. Assoc. 100 (2005) 1394–1411.
[89] G.A. Pavliotis, A. Stuart, Parameter estimation for multiscale diffusions, J. Stat. Phys. 127 (2007) 741–781.
[90] A. Papavasiliou, G. Pavliotis, A. Stuart, Maximum likelihood drift estimation for multiscale diffusions, Stoch. Process. Appl. 119 (2009) 3173–3210.
[91] L. Boninsegna, F. Nüske, C. Clementi, Sparse learning of stochastic dynamical equations, J. Chem. Phys. 148 (2018) 241723.
[92] L. Zhang, Efficient estimation of stochastic volatility using noisy observations: a multi-scale approach, Bernoulli 12 (2006) 1019–1043.
[93] O. Papaspiliopoulos, Y. Pokern, G.O. Roberts, A.M. Stuart, Nonparametric estimation of diffusions: a differential equations approach, Biometrika 99 (2012) 511–531.
[94] G.A. Pavliotis, Y. Pokern, A.M. Stuart, Parameter estimation for multiscale diffusions: an overview, in: Statistical Methods for Stochastic Differential Equations, 2012, p. 429.
[95] P. Batz, A. Ruttor, M. Opper, Approximate Bayes learning of stochastic differential equations, Phys. Rev. E 98 (2018) 022109.
[96] A. Abdulle, G. Garegnani, G.A. Pavliotis, A.M. Stuart, A. Zanoni, Drift estimation of multiscale diffusions based on filtered data, Found. Comput. Math. (2021) 1–52.
[97] C. Honisch, R. Friedrich, Estimation of Kramers-Moyal coefficients at low sampling rates, Phys. Rev. E 83 (2011) 066701.
[98] S.J. Lade, Finite sampling interval effects in Kramers–Moyal analysis, Phys. Lett. A 373 (2009) 3705–3709.
[99] J.L. Callaham, J.-C. Loiseau, G. Rigas, S.L. Brunton, Nonlinear stochastic modelling with Langevin regression, Proc. R. Soc. A 477 (2021) 20210092.
[100] A. Neumaier, T. Schneider, Estimation of parameters and eigenmodes of multivariate autoregressive models, ACM Trans. Math. Softw. 27 (2001) 27–57.
[101] T. Schneider, A. Neumaier, Algorithm 808: ARfit — a Matlab package for the estimation of parameters and eigenmodes of multivariate autoregressive models, ACM Trans. Math. Softw. 27 (2001) 58–65.
[102] H. Arnold, I. Moroz, T. Palmer, Stochastic parametrizations and model uncertainty in the Lorenz'96 system, Philos. Trans. R. Soc. A, Math. Phys. Eng. Sci. 371 (2013) 20110479.
[103] F. Lu, Data-driven model reduction for stochastic Burgers equations, Entropy 22 (2020) 1360.
[104] N. Chen, A.J. Majda, Filtering nonlinear turbulent dynamical systems through conditional Gaussian statistics, Mon. Weather Rev. 144 (2016) 4885–4917.
[105] N. Chen, A.J. Majda, Conditional Gaussian systems for multiscale nonlinear stochastic systems: prediction, state estimation and uncertainty quantification, Entropy 20 (2018) 509.
[106] S. Krumscheid, G.A. Pavliotis, S. Kalliadasis, Semiparametric drift and diffusion estimation for multiscale diffusions, Multiscale Model. Simul. 11 (2013) 442–473.
[107] S. Krumscheid, M. Pradas, G. Pavliotis, S. Kalliadasis, Data-driven coarse graining in action: modeling and prediction of complex systems, Phys. Rev. E 92 (2015) 042139.
[108] S. Kalliadasis, S. Krumscheid, G.A. Pavliotis, A new framework for extracting coarse-grained models from time series with multiscale structure, J. Comput. Phys. 296 (2015) 314–328.
[109] K. Hasselmann, Stochastic climate models part I. Theory, Tellus 28 (1976) 473–485.
[110] C. Frankignoul, K. Hasselmann, Stochastic climate models, part ii application to sea-surface temperature anomalies and thermocline variability, Tellus 29 (1977) 289–305.
[111] C. Penland, T. Magorian, Prediction of Niño 3 sea surface temperatures using linear inverse modeling, J. Climate 6 (1993) 1067–1076.
[112] T. Schneider, S.M. Griffies, A conceptual framework for predictability studies, J. Climate 12 (1999) 3133–3155.
[113] K. Hasselmann, PIPs and POPs: the reduction of complex dynamical systems using principal interaction and oscillation patterns, J. Geophys. Res., Atmos. 93 (1988) 11015–11021.
[114] Z. Li, N.B. Kovachki, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, A. Anandkumar, et al., Fourier neural operator for parametric partial differential equations, in: International Conference on Learning Representations, 2020.
[115] L. Lu, P. Jin, G. Pang, Z. Zhang, G.E. Karniadakis, Learning nonlinear operators via deeponet based on the universal approximation theorem of operators, Nat. Mach. Intell. 3 (2021) 218–229.
[116] N.H. Nelsen, A.M. Stuart, The random feature model for input-output maps between Banach spaces, SIAM J. Sci. Comput. 43 (2021) A3212–A3243.
[117] Y. Chen, B. Hosseini, H. Owhadi, A.M. Stuart, Solving and learning nonlinear PDEs with Gaussian processes, 2021.
[118] Q. Du, Nonlocal Modeling, Analysis, and Computation: Nonlocal Modeling, Analysis, and Computation, SIAM, 2019.
[119] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: learning maps between function spaces with applications to PDEs, J. Mach. Learn. Res. 24 (2023) 1–97.
[120] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[121] C. Bucur, E. Valdinoci, Nonlocal Diffusion and Applications, vol. 20, Springer, 2016.
[122] T. Sauer, J.A. Yorke, M. Casdagli, Embedology, J. Stat. Phys. 65 (1991) 579–616.
[123] J. Sturis, K.S. Polonsky, E. Mosekilde, E. Van Cauter, Computer model for mechanisms underlying ultradian oscillations of insulin and glucose, Am. J. Physiol: Endocrinol. Metab. 260 (1991) E801–E809.
[124] D. Sanz-Alonso, A. Stuart, A. Taeb, Inverse Problems and Data Assimilation, vol. 107, Cambridge University Press, 2023.
[125] E. Calvello, S. Reich, A.M. Stuart, Ensemble Kalman methods: a mean field perspective, arXiv preprint, arXiv:2209.11371, 2022.
[126] M.A. Iglesias, K.J. Law, A.M. Stuart, Ensemble Kalman methods for inverse problems, Inverse Probl. 29 (2013) 045001.
[127] D.J. Albers, P.-A. Blancquart, M.E. Levine, E.E. Seylabi, A. Stuart, Ensemble Kalman methods with constraints, Inverse Probl. 35 (2019) 095007.