communications physics

ARTICLE

https://doi.org/10.1038/s42005-022-00987-z OPEN

Discovering sparse interpretable dynamics from partial observations

Peter Y. Lu[1✉], Joan Ariño Bernad[1,2] & Marin Soljačić[1]

Identifying the governing equations of a nonlinear dynamical system is key to both understanding the physical features of the system and constructing an accurate model of the dynamics that generalizes well beyond the available data. Achieving this kind of interpretable system identification is even more difficult for partially observed systems. We propose a machine learning framework for discovering the governing equations of a dynamical system using only partial observations, combining an encoder for state reconstruction with a sparse symbolic model. The entire architecture is trained end-to-end by matching the higher-order symbolic time derivatives of the sparse symbolic model with finite difference estimates from the data. Our tests show that this method can successfully reconstruct the full system state and identify the equations of motion governing the underlying dynamics for a variety of ordinary differential equation (ODE) and partial differential equation (PDE) systems.

[1] Department of Physics, Massachusetts Institute of Technology, Cambridge, MA, USA. [2] Department of Physics, Universitat Politècnica de Catalunya, Barcelona, Spain. ✉email: lup@mit.edu

Analyzing data from a nonlinear dynamical system to understand its qualitative behavior and accurately predict future states is a ubiquitous problem in science and engineering. In many instances, this problem is further compounded by a lack of available data and only partial observations of the system state, e.g., forecasting fluid flow driven by unknown sources or predicting optical signal propagation without phase measurements. This means that, in addition to identifying and modeling the underlying dynamics, we must also reconstruct the hidden or unobserved variables of the system state. While traditional approaches to system identification have had significant success with linear systems, nonlinear system identification and state reconstruction is a much more difficult and open problem[1]. Moreover, modeling nonlinear dynamics in a way that provides interpretability and physical insight is also a major challenge.

Modern machine learning approaches have made significant strides in black box predictive performance on many tasks[2], such as data-driven prediction of nonlinear dynamics[3–5] including methods that only use partial observations[6–9]. However, because deep learning models often fail to take into account known physics, they require vast quantities of data to train and tend to generalize poorly outside of their training distribution. Standard deep learning models also lack the interpretability necessary for developing a detailed physical understanding of the system, although new approaches incorporating intrinsic dimensionality estimation[10] and recent unsupervised learning methods[11] can help mitigate this issue. Introducing physical priors and building physics-informed inductive biases, such as symmetries, into neural network architectures can significantly improve the performance of deep learning models and provide a greater degree of interpretability[11–14].

Recent data-driven nonlinear system identification methods based on Koopman operator theory offer a compelling alternative to deep learning approaches as well as a theoretical framework for incorporating neural networks into system identification methods[15–18]. However, these approaches still encounter barriers when dealing with certain types of nonlinear dynamics, such as chaos, which lead to a problematic continuous spectrum for the Koopman operator that cannot be modeled by a finite-dimensional linear system, although some progress has been made in addressing these limitations[18–20].

Instead, sparse symbolic identification approaches[21–24] work directly with the governing equations of motion, which are often sparse and provide a highly interpretable representation of the dynamical system that also generalizes well. By fitting a symbolic model, these methods capture the exact dynamics of the many physical systems in nature governed by symbolic equations. Previous work has shown that, by imposing a sparsity prior on the governing equations, it is possible to obtain interpretable and parsimonious models of nonlinear dynamics[21–23]. This sparsity prior, in combination with an autoencoder architecture, can also aid in extracting interpretable state variables from high-dimensional data[24].

In this work, we propose a machine learning framework for solving the common problem of partially observed system identification, where a portion of the system state is observed, but the remaining hidden states, as well as the underlying dynamics, are unknown. Unlike in the generic high-dimensional setting, this is a much more structured problem, and we take full advantage of this additional structure when designing our architecture. To deal with having only partial state information, our method combines an encoder for reconstructing the full system state and a sparse symbolic model, which directly learns the governing equations, providing a flexible framework for both system identification and state reconstruction (Fig. 1). The full architecture is trained by matching the higher order time derivatives of the symbolic model with finite difference estimates from the data. As illustrated in our numerical experiments on both ordinary differential equation (ODE) and partial differential equation (PDE) systems, this approach can be easily adapted for specific applications by incorporating known constraints into the architecture of the encoder and the design of the symbolic model.

Results

Problem formulation. Consider a nonlinear dynamical system defined by the first order ODE

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}). \tag{1}$$

The visible or observed state is given by a known "projection" function $\mathbf{x}_v = \mathbf{g}(\mathbf{x})$, while the hidden states $\mathbf{x}_h$ must be reconstructed such that $\mathbf{a}(\mathbf{x}_v, \mathbf{x}_h) = \mathbf{x}$, where $\mathbf{a}$ is a known aggregation function. The goal is to determine the governing equations defined by $\mathbf{F}(\mathbf{x})$ while simultaneously reconstructing the hidden state $\mathbf{x}_h$.

Without prior knowledge detailing the structure of the dynamical system, we can generically choose the visible state $\mathbf{x}_v = (x_1, x_2, \ldots, x_k)$ to be a subset of the full state $\mathbf{x} = (x_1, x_2, \ldots, x_k, x_{k+1}, \ldots, x_n)$, i.e., $\mathbf{g}$ is a simple projection of $\mathbf{x}$ onto the subset $\mathbf{x}_v$. The remaining components would then form the hidden state $\mathbf{x}_h = (x_{k+1}, x_{k+2}, \ldots, x_n)$, and the aggregation function $\mathbf{a}$ just concatenates the two states $\mathbf{x}_v$, $\mathbf{x}_h$. When additional information about the dynamical system is available, $\mathbf{g}$ and $\mathbf{a}$ can be chosen appropriately to reflect the structure of the dynamics (e.g., see our nonlinear Schrödinger phase reconstruction example).

See the "Proposed Machine Learning Framework" subsection of the Methods for a detailed description our approach for reconstructing the hidden state $\mathbf{x}_h$ and the governing equations (Eq. (1)) given only the visible state $\mathbf{x}_v$.

ODE experiments. To demonstrate our method, we use data from two standard examples of chaotic nonlinear dynamics: the Rössler system (Fig. 2a) and the Lorenz system (Fig. 2b). Both systems have a three-dimensional phase space (u, v, w), and we take the first two dimensions (u, v) to be the visible state with the remaining dimension w as the hidden state. In both cases, we are able to accurately identify the governing equations—via the learned symbolic model—and reconstruct the hidden state w (Fig. 2). We achieve a hidden state reconstruction error of 4.6×10^{-4} (relative to the range of the hidden state) for the Rössler system and 1.7×10^{-3} for the Lorenz system.

Note that the hidden states discovered by our approach often differ from the ground truth hidden states by an affine transformation (e.g., $w' = aw + b$). In order to make a direct comparison, we fit the discovered hidden states to the true hidden states using linear regression and show the transformed result as the reconstructed governing equations and hidden states (Fig. 2).

PDE experiments. To test our method in a more challenging setting, we use data from two PDE systems: a 2D diffusion system with an exponentially decaying source term (Fig. 3a) and a 2D diffusive Lokta–Volterra predator–prey system (Fig. 3b)—commonly used for ecological modeling[25–27]. For the diffusion system, we observe a diffusing visible state $u(x, y, t)$ and must reconstruct the hidden dynamic source term $v(x, y, t)$. Similarly, for the diffusive Lokta–Volterra system, one of the two components is visible $u(x, y, t)$ while the other is hidden $v(x, y, t)$. We accurately identify the governing equations and reconstruct the hidden component for both systems (Fig. 3), achieving a relative error of 1.4×10^{-4} for the diffusion system and 1.0×10^{-3} for the diffusive Lokta–Volterra system. The neural network

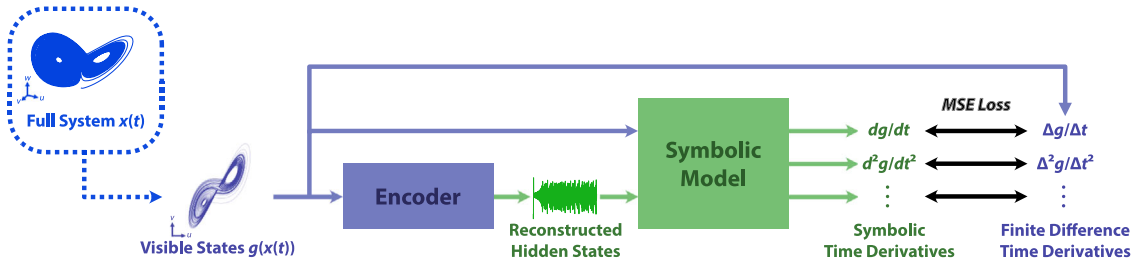

Fig. 1 A machine learning framework for simultaneous system identification and state reconstruction. With only a visible portion of the full state available $\mathbf{x}_v = \mathbf{g}(\mathbf{x})$, an encoder is first used to reconstruct the hidden states. The fully reconstructed state $\hat{\mathbf{x}}$, including the visible and hidden states, is then passed into a symbolic model of the governing equations. Using automatic differentiation, multiple symbolic time derivatives $d^p\mathbf{g}(\hat{\mathbf{x}})/dt^p$ of the visible states are generated from the symbolic model and compared with finite difference derivatives $\Delta^p\mathbf{g}(\mathbf{x})/\Delta t^p$ computed directly from the sequence of visible states. The entire architecture is trained end-to-end using the mean squared error (MSE) loss between the symbolic and finite difference derivatives. See the "Proposed Machine Learning Framework" subsection of the Methods for additional details.

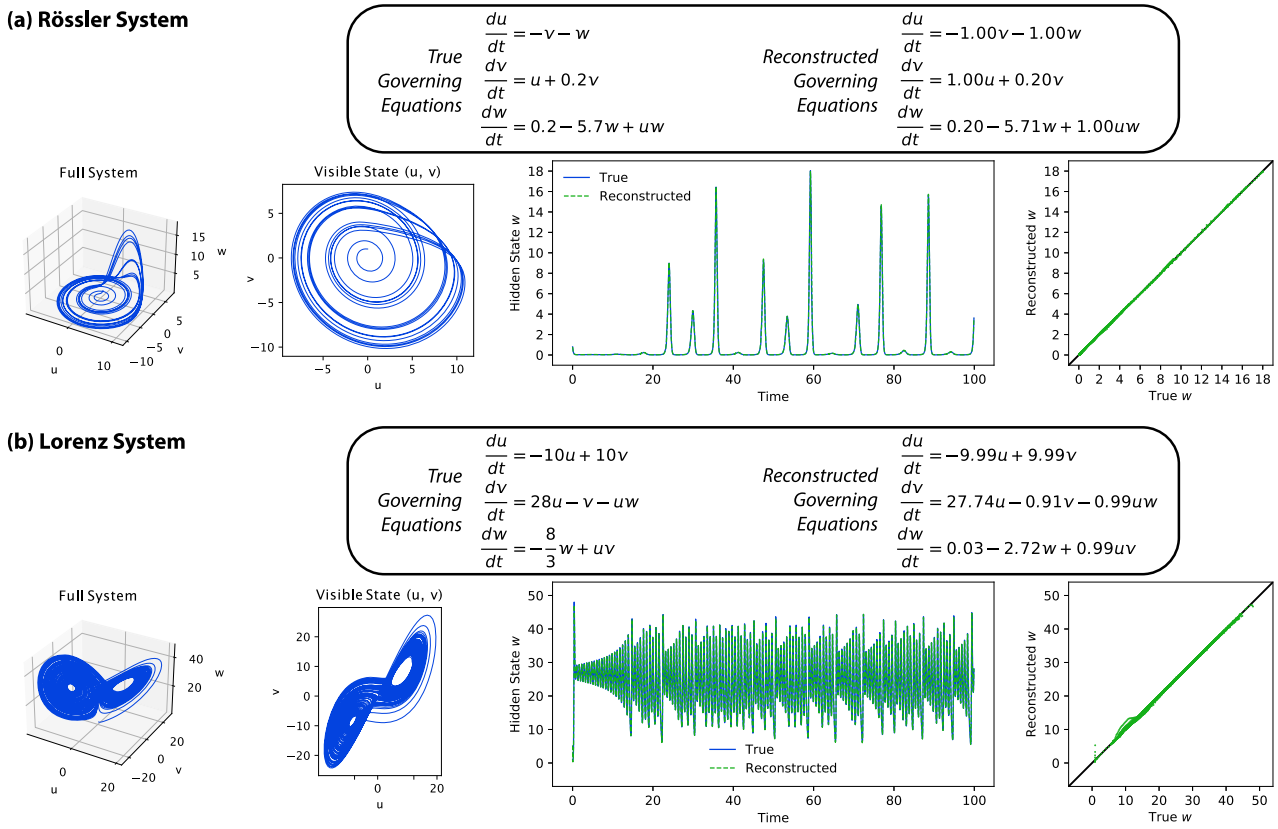

Fig. 2 System identification and hidden state reconstruction for the ODE systems. For both the **a** Rössler and **b** Lorenz systems, u and v components are visible while the w component is hidden. For each system, the full three-dimensional system dynamics (u, v, w) is plotted alongside the projection onto the visible states (u, v). Using only the visible states (u, v), our method reconstructs the governing equations as well as the hidden state w. The true (blue) and reconstructed (green, dashed) hidden states w are shown as a function of time and also plotted directly against each other (green, points) for comparison. Note that the reconstructed hidden states shown here are directly reconstructed by the encoder from the corresponding visible states at nearby times and are not based on an autonomous prediction of the learned model.

encoder has more difficulty with the more complex and nonlinear diffusive Lokta–Volterra system, resulting in a slightly blurry reconstruction.

Phase reconstruction. As a final example, we consider the phase reconstruction problem for the 1D nonlinear Schrödinger equation—a model for light propagation through a nonlinear fiber[28]—to demonstrate the breadth of our approach and its ability to handle a more difficult and structured problem. Using only visible amplitude data $|\psi(x, t)|$, we aim to identify the underlying dynamics and reconstruct the hidden phase $\varphi(x, t) = \arg(\psi(x, t))$. For this system, we also assume we have some prior knowledge about the structure of the dynamics: a complex wave equation with a global phase shift symmetry and only odd nonlinearities to model an optical material with inversion symmetry[28]. This allows us to limit the library of predefined terms used by our symbolic model. Our prior knowledge also informs our choice of projection $\mathbf{g}(\psi) = |\psi|$ and aggregation functions $\mathbf{a}(|\psi|, \varphi) = |\psi|e^{i\varphi}$.

Our method successfully identifies the governing equation for the nonlinear Schrödinger data and roughly captures the correct phase profile (Fig. 4). Although the overall phase reconstruction seems somewhat poor, with a relative error of 0.35, this also

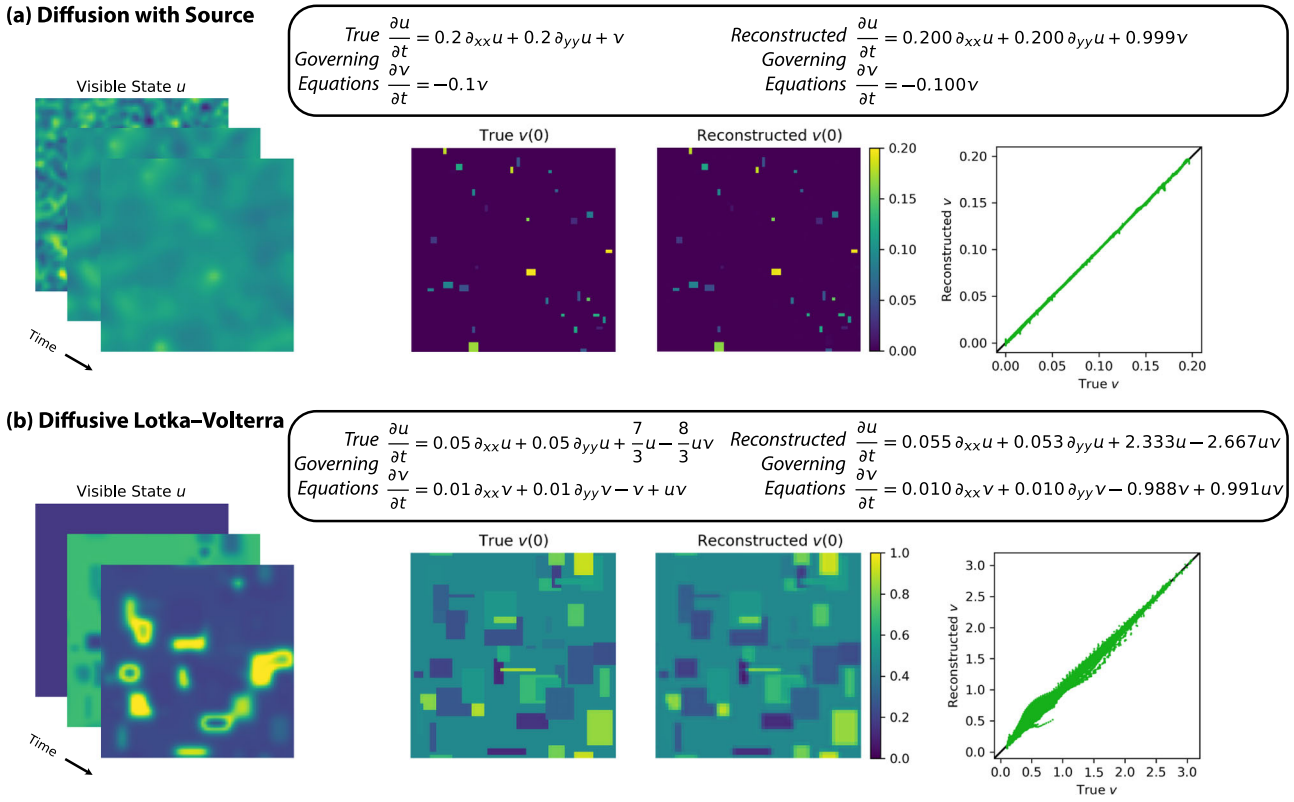

Fig. 3 System identification and hidden state reconstruction for the PDE systems. For both the **a** diffusion system with a decaying source term v and **b** diffusive Lokta–Volterra system, the u component is visible while the v component is hidden. Using only the visible state u, our method reconstructs the governing equations as well as the hidden state v. For each system, snapshots of the visible state u over time are shown alongside the true and reconstructed hidden states v at time $t = 0$. The true and reconstructed v are also plotted directly against each other for comparison.

includes an accumulated drift of the phase over time. The spatial derivative of the phase $\partial\varphi/\partial x$ has a much more reasonable relative error of 0.057. Furthermore, given the governing equations extracted by our method, other more specialized algorithms for nonlinear phase retrieval can be used as a post-processing step to significantly improve the quality of the phase reconstruction[29].

Discussion

On a wide variety of dynamical systems, we have demonstrated that our proposed machine learning framework can successfully identify sparse interpretable dynamics and reconstruct hidden states using only partial observations. By fitting symbolic models, we are able to discover the exact form of the symbolic equations governing the underlying physical systems, resulting in highly interpretable models and predictions (see Supplementary Note 1). Our method is also straightforward to implement and use, easily adapting to different levels of prior knowledge about the unknown hidden states and dynamics.

Compared with methods that require explicit integration[6,8], our approach can be significantly more computationally efficient since we only need to compute symbolic and finite difference derivatives. Methods that rely on explicit integration may also need to deal with stiffness and other issues that are relevant to choosing an appropriate integration scheme[12]. However, methods using explicit integration also have the advantage of being much more robust to noise. Because we require higher order finite difference time derivative estimates from data, our approach—like other derivative-based methods—is generally more susceptible to noise. Data smoothing techniques and careful tuning of our sparsity method help mitigate this to some extent (see Supplementary Note 2) in a similar fashion to methods like SINDy[21,22], and promising new methods for identifying the noise distribution alongside the dynamics[30] could be incorporated into our framework in the future.

Our framework offers a strong foundation for designing interpretable machine learning methods to deal with partial observations and solve the combined system identification and state reconstruction task. We hope to continue developing more robust encoders and more flexible symbolic models that will work within our proposed framework. For example, the encoder (see the "Phase Reconstruction" subsection of the Methods) used in our final experiment on phase reconstruction has similarities with variational approaches used for equation discovery[3,23,31], and we believe that these variational methods can be incorporated into our framework to provide a smoother encoding and improve robustness to noise. In future work, we will also study symbolic models that have multiple layers of composable units designed for symbolic regression tasks[32–34]. These alternative symbolic architectures provide more powerful and flexible models with a sparse symbolic prior, potentially addressing some current limitations of our implementation (see Supplementary Note 3) and allowing our framework to handle a wider range of governing equations—such as the Hill equations used in modeling gene expression[35]—without requiring large libraries of predefined terms.

Methods

Proposed machine learning framework. Our proposed framework consists of an encoder, which uses the visible states to reconstruct the corresponding hidden states, and an interpretable symbolic model, which represents the governing equations of the dynamical system. The encoder $\mathbf{e}_\eta$, typically a neural network architecture with learnable parameters η, takes as input the sequence of visible

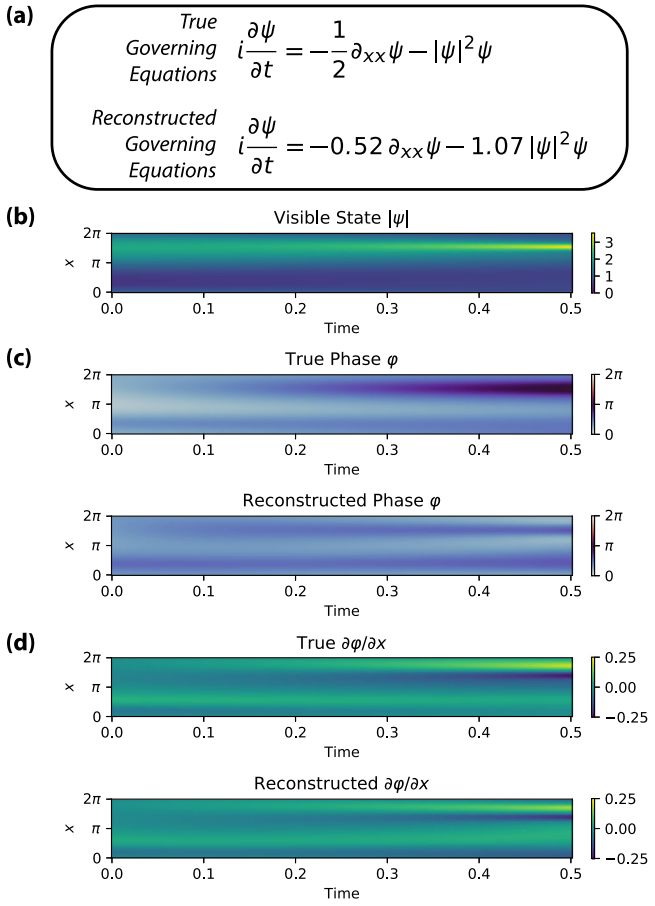

Fig. 4 System identification and phase reconstruction for the nonlinear Schrödinger system. a Our method successfully identifies the correct nonlinear Schrödinger equation. **b** The magnitude $|\psi|$ of the wave is visible while **c** the phase $\varphi = \arg(\psi)$ is hidden and must be reconstructed. **d** The spatial derivative of the phase $\partial\varphi/\partial x$ and its reconstruction are also shown.

states $\{\mathbf{x}_v(t_0), \mathbf{x}_v(t_0+\Delta t), \ldots, \mathbf{x}_v(t_N)\}$ and reconstructs the hidden states $\{\hat{\mathbf{x}}_h(t_0), \hat{\mathbf{x}}_h(t_0+\Delta t), \ldots, \hat{\mathbf{x}}_h(t_N)\}$. This should, in general, be possible for hidden states that are sufficiently coupled to the visible states due to Takens' embedding theorem[36]. We can then obtain a reconstruction of the full state by applying the aggregation function $\hat{\mathbf{x}} = \mathbf{a}(\mathbf{x}_v, \hat{\mathbf{x}}_h)$. The fully reconstructed state $\hat{\mathbf{x}}$ allows us to compute symbolic time derivatives defined by a symbolic model of the governing equations

$$\frac{d\hat{\mathbf{x}}}{dt} = \hat{\mathbf{F}}_\theta(\hat{\mathbf{x}}) := \theta_1\mathbf{f}_1(\hat{\mathbf{x}}) + \theta_2\mathbf{f}_2(\hat{\mathbf{x}}) + \cdots + \theta_m\mathbf{f}_m(\hat{\mathbf{x}}), \tag{2}$$

where $\theta_1, \theta_2, \ldots, \theta_m$ are learnable coefficients and $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_m$ are predefined terms, such as monomial expressions or linear combinations representing spatial derivatives (for PDE systems). If the dimensionality of the system state $\mathbf{x}$ is unknown, we can treat it as a hyperparameter, tuned to achieve an optimal trade-off between high accuracy (e.g., low loss) and parsimony (e.g., fewer hidden states and model sparsity), or use recently suggested intrinsic dimensionality estimation methods[10].

To jointly train the encoder and symbolic model using only partial observations, we match higher order time derivatives of the visible states with finite difference estimates from the data. These time derivatives are implicitly defined by the symbolic model (Eq. (2)), so we develop and use an algorithmic trick that allows standard automatic differentiation methods[37] to compute higher order symbolic time derivatives of the *reconstructed* visible states $\mathbf{g}(\hat{\mathbf{x}})$ (see the "Automatic Computation of Symbolic Derivatives" subsection of the Methods). These symbolic derivatives can then be compared with finite difference time derivatives $\Delta^p\mathbf{g}(\mathbf{x})/\Delta t^p = \Delta^p\mathbf{x}_v/\Delta t^p$ computed directly from the visible states $\mathbf{x}_v$.

We train the entire architecture in an end-to-end fashion by optimizing the mean squared error (MSE) loss

$$\mathcal{L}(\eta,\theta) = \frac{1}{N}\sum_{i=1}^{N}\sum_{p=1}^{M}\frac{\alpha_p}{\sigma_p^2}\left(\frac{d^p\mathbf{g}(\hat{\mathbf{x}}(t_i))}{dt^p} - \frac{\Delta^p\mathbf{x}_v(t_i)}{\Delta t^p}\right)^2, \tag{3}$$

where σ_p^2 is the empirical variance of the pth order finite difference derivative $\Delta^p\mathbf{x}_v(t_i)/\Delta t^p$, and the α_p are hyperparameters that determine the importance of each derivative order in the loss function. This loss implicitly depends on the encoder $\mathbf{e}_\eta$ through the reconstructed state $\hat{\mathbf{x}}$ and the symbolic model $\hat{\mathbf{F}}_\theta$ through the symbolic time derivatives. To achieve sparsity in the symbolic model, we use a simple thresholding approach—commonly used in sparse linear regression applications[21]—which sets a coefficient θ_i to zero if its absolute value falls below a chosen threshold θ_{thres}. We implement this sparsification at regular intervals during training. While developing our approach, we also experimented with L_1 regularization but found that it tends to strongly degrade the performance of the learned model when applied with enough strength to achieve sparsity.

The code for implementing our framework and reproducing our results is available at https://github.com/peterparity/symder.

Dataset, architecture, and training details. The data and architecture requirements for using our approach are dependent on the properties of the dynamical system, the fraction of visible states, and the chosen symbolic model. For example, a more constrained symbolic model with a smaller library of terms will likely be more data efficient due to having a stronger inductive bias on the model. A smaller library also makes it easier for the sparse optimization to discover the correct sparsity pattern and thus accurate governing equations, so it is often better to start with a more constrained library of terms and then slowly expand it if model performance remains poor. In general, the trajectories from the data need to be long and varied enough in order to differentiate among the terms provided by the symbolic model, although we have found that a single trajectory is often sufficient for accurate state reconstruction and system identification. For the encoder architecture, we generally use small and relatively shallow neural networks, which already provide good hidden state reconstruction performance. This also means that our approach trains reasonably quickly, taking ~2.5 min for the ODE systems on a single consumer GPU (GeForce RTX 2080 Ti) and ~2 h for the much larger PDE systems on four GPUs. However, it is certainly possible that more structured encoders may help in certain cases requiring more complex reconstructions. In addition, we are able to obtain accurate results in our tests by matching the first and second order time derivatives, although higher-order derivatives will be necessary for datasets with a larger fraction of hidden states.

ODE Experiments. Each system is sampled for 10,000 time steps of size $\Delta t = 10^{-2}$, and the resulting time series data is normalized to unit variance.

The encoder takes a set of nine visible states $\{\mathbf{x}_v(t-4\Delta t), \mathbf{x}_v(t-3\Delta t), \ldots, \mathbf{x}_v(t+4\Delta t)\}$ as input to reconstruct each hidden state $\hat{\mathbf{x}}_h(t)$ and is implemented as a sequence of three 1D time-wise convolutional layers with kernel sizes 9–1–1 and layer sizes 128–128–1. This architecture enforces locality in time, allowing the neural network to learn a simpler and more interpretable mapping. The predefined terms of the symbolic model consist of constant, linear, and quadratic monomial terms, i.e., 1, u, v, w, u^2, v^2, w^2, uv, uw, and vw, for each governing equation.

We also scale the effective time step of the symbolic model by a factor of 10 to improve training by preconditioning the model coefficients. We then train for 50000 steps using the AdaBelief optimizer[38] with a learning rate of 10^{-3} and with hyperparameters $\alpha_1 = \alpha_2 = 1$ to equally weight the first two time derivative terms in the loss function ($\alpha_p = 0$ for $p > 2$). Every 5000 training steps, we sparsify the symbolic model, setting coefficients to zero if their absolute value is below $\theta_{\text{thres}} = 10^{-3}$.

PDE experiments. Each system is sampled on a 64×64 spatial mesh with grid spacing $\Delta x = \Delta y = 1$ for 1000 time steps of size $\Delta t = 5 \times 10^{-2}$, and the resulting data is normalized to unit variance.

The encoder is a sequence of three 3D spatiotemporal convolutional layers with kernel sizes 5–1–1 and layer sizes 64–64–1, which enforces locality in both time and space. The predefined terms of the symbolic model consist of constant, linear, and quadratic terms as well as up to second order spatial derivative terms, e.g. $\partial_x u$, $\partial_y u$, $\partial_{xx} u$, $\partial_{yy} u$, $\partial_{xy} u$, and similarly for v.

We scale the effective time step and spatial grid spacing of the symbolic model by a factor of 10 and $\sqrt{10}$, respectively, to precondition the model coefficients. For the diffusion system, we train for 50,000 steps with a learning rate of 10^{-4} and hyperparameters $\alpha_1 = 1$ and $\alpha_2 = 10$, and we sparsify the symbolic model every 1000 training steps with $\theta_{\text{thres}} = 5 \times 10^{-3}$. For the diffusive Lokta–Volterra system, we train for 100,000 steps with a learning rate of 10^{-3} and hyperparameters $\alpha_1 = \alpha_2 = 1$, and we sparsify the symbolic model every 1000 training steps with $\theta_{\text{thres}} = 2 \times 10^{-3}$.

Phase reconstruction. The system is sampled on a size 64 mesh with spacing $\Delta x = 2\pi/64$ for 500 time steps of size $\Delta t = 10^{-3}$.

Using the available prior knowledge, we allow the symbolic model to use spatial derivative terms $\partial_x^p\psi$ for $p \in \{1, 2, 3, 4\}$ and nonlinearity terms $|\psi|^q\psi$ for $q = \{2, 4, 6, 8\}$. We scale the effective time step by a factor of 10 to precondition the model coefficients and train for 100,000 time steps with a learning rate of 10^{-4} and hyperparameters $\alpha_1 = \alpha_2 = 1$ and $\beta = 10^3$. We sparsify the symbolic model every 10,000 training steps with $\theta_{\text{thres}} = 10^{-3}$.

Unlike the previous examples, reconstructing the phase is a much trickier problem that cannot be done using a local spatiotemporal encoder. Instead of using a neural network mapping, we use a direct embedding of the phase as a function of time, i.e., for each point in the spatiotemporal grid of the original data, we learn a parameter for the phase $(\hat{\varphi}x, t)$. This simple approach has the advantage of being incredibly flexible but also more difficult to train, requiring an additional encoder regularization term

$$\mathcal{R}_{\text{enc}} = \beta\left(\frac{\partial\hat{\psi}}{\partial t} - \frac{\Delta\hat{\psi}}{\Delta t}\right)^2 \tag{4}$$

that ensures the symbolic time derivatives match the finite difference time derivatives of the reconstructed state $\hat{\psi} = \mathbf{a}(|\psi|, \hat{\varphi})$. Unlike the compact neural network encoders from the previous experiments, this encoder also must scale with the dataset size and does not provide a useful mapping that can be used for future hidden state reconstruction.

Automatic computation of symbolic derivatives. The time derivatives can be derived by repeated differentiation of the symbolic model (Eq. (2)), substituting back in previously computed derivatives to obtain expressions only in terms of the reconstructed state $\hat{\mathbf{x}}$. For example, the first and second-time derivatives can be written in index notation as

$$\frac{dg_i}{dt} = \sum_j \frac{dg_i}{d\hat{x}_j}\frac{d\hat{x}_j}{dt} = \sum_j \frac{dg_i}{d\hat{x}_j}\hat{F}_{\theta j} \tag{5}$$

$$\frac{d^2g_i}{dt^2} = \sum_{j,k} \frac{d^2g_i}{d\hat{x}_j d\hat{x}_k}\hat{F}_{\theta j}\hat{F}_{\theta k} + \frac{dg_i}{d\hat{x}_j}\frac{d\hat{F}_{\theta j}}{d\hat{x}_k}\hat{F}_{\theta k}. \tag{6}$$

The expressions for the symbolic time derivatives (Eqs. (5) and (6)) quickly grow more and more unwieldy for higher order derivatives. Implementing these expressions by hand is likely to be both time-consuming and error-prone, especially for more complex symbolic models such as those used in our PDE experiments. To address this issue, we develop an automated approach that takes advantage of powerful modern automatic differentiation software (in our case, the JAX library[39]).

Automatic differentiation is the algorithmic backbone of modern deep learning[37], and a new generation of source-to-source automatic differentiation libraries are quickly becoming available[39,40]. Automatic differentiation uses a library of custom derivative rules defined on a set of primitive functions, which can then be arbitrarily composed to form more complex expressions. The algorithm normally requires a forward evaluation of a function that sets up a backward pass which computes the gradient of the function. In our case, the appropriate forward step is integrating the symbolic model (Eq. (2)) using an ODE solver, which makes the time variable and its derivatives explicit rather than being implicitly defined by the governing equations. This, however, would introduce significant overhead and would not produce the exact expressions that we derived earlier. In fact, integration should not be necessary at all for efficiently implementing symbolic differentiation. Instead, we propose a simple algorithmic trick that allows standard automatic differentiation to compute symbolic time derivatives without explicit integration.

Consider a function $\mathcal{I}(\hat{\mathbf{x}}, \epsilon)$ that propagates the state $\hat{\mathbf{x}}$ forward by a time ϵ according to the governing equations (Eq. (2)), i.e.

$$\mathcal{I}(\hat{\mathbf{x}}(t), \epsilon) = \hat{\mathbf{x}}(t+\epsilon). \tag{7}$$

As $\epsilon \to 0$, $\mathcal{I}(\hat{\mathbf{x}}(t), 0) = \hat{\mathbf{x}}(t)$ reduces to the identity. Taking a derivative with respect to ϵ, we find that

$$\begin{aligned}\frac{\partial\mathcal{I}(\hat{\mathbf{x}}(t),\epsilon)}{\partial\epsilon} &= \frac{d\hat{\mathbf{x}}(t+\epsilon)}{d\epsilon}\\ &= \hat{\mathbf{F}}_\theta(\hat{\mathbf{x}}(t+\epsilon))\\ &= \hat{\mathbf{F}}_\theta(\mathcal{I}(\hat{\mathbf{x}}(t),\epsilon)),\end{aligned} \tag{8}$$

which reduces to $\partial\mathcal{I}(\hat{\mathbf{x}}, \epsilon)/\partial\epsilon|_{\epsilon=0} = d\hat{\mathbf{x}}/dt = \hat{\mathbf{F}}_\theta(\hat{\mathbf{x}})$ as $\epsilon \to 0$. This generalizes to higher order derivatives, allowing us to compute time derivatives of $\hat{\mathbf{x}}$ as

$$\frac{d^p\hat{\mathbf{x}}}{dt^p} = \left.\frac{\partial^p\mathcal{I}(\hat{\mathbf{x}},\epsilon)}{\partial\epsilon^p}\right|_{\epsilon=0}. \tag{9}$$

Since we only ever evaluate at $\epsilon = 0$, this formulation makes the time variable explicit without having to integrate the governing equations. To implement this trick using an automatic differentiation algorithm, we define a wrapper function $\mathcal{I}_0(\hat{\mathbf{x}}, \epsilon) := \hat{\mathbf{x}}$ that acts as the identity on the state $\hat{\mathbf{x}}$ but has a custom derivative rule

$$\frac{\partial\mathcal{I}_0(\hat{\mathbf{x}},\epsilon)}{\partial\epsilon} := \hat{\mathbf{F}}_\theta(\mathcal{I}_0(\hat{\mathbf{x}},\epsilon)). \tag{10}$$

This allows standard automatic differentiation to correctly compute exact symbolic time derivatives of our governing equations, including higher order derivatives. Our code for implementing this algorithmic trick and for reproducing the rest of our results is available at https://github.com/peterparity/symder.

The proposed algorithmic trick for computing higher order time derivatives, which exploits modern automatic differentiation, further simplifies the implementation of our method and allows the user to focus on designing an appropriate encoder and choosing a reasonable library of predefined terms for the sparse symbolic model.

Data availability

All the datasets used in this study can be generated using the publicly available data generation scripts provided at https://github.com/peterparity/symder.

Code availability

All the code necessary for reproducing our results is publicly available at https://github.com/peterparity/symder.

Received: 29 March 2022; Accepted: 28 July 2022;
Published online: 12 August 2022

References

1. Ljung, L. *System Identification: Theory for the User* 2nd edn. (Pearson, 1999)
2. Goodfellow, I., Bengio, Y. & Courville, A. *Deep Learning* http://www.deeplearningbook.org (MIT Press, 2016).
3. Raissi, M., Perdikaris, P. & Karniadakis, G. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **378**, 686–707 (2019).
4. Berg, J. & Nyström, K. Data-driven discovery of PDEs in complex datasets. *J. Comput. Phys.* **384**, 239–252 (2019).
5. Raissi, M. Deep hidden physics models: deep learning of nonlinear partial differential equations. *J. Mach. Learn. Res.* **19**, 1–24 (2018).
6. Ayed, I., Bézenac, E.d., Pajot, A. & Gallinari, P. *ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing* https://doi.org/10.1109/ICASSP40776.2020.9053035 (*ICASSP*) (IEEE, 2020).
7. Ouala, S. et al. Learning latent dynamics for partially observed chaotic systems. *Chaos* **30**, 103,121 (2020).
8. Chen, R. T. Q., Rubanova, Y., Bettencourt, J. & Duvenaud, D. K. In *Advances in Neural Information Processing Systems* (eds. Bengio, S. et al.) (Curran Associates, Inc., 2018)
9. Saha, P., Dash, S. & Mukhopadhyay, S. Physics-incorporated convolutional recurrent neural networks for source identification and forecasting of dynamical systems. *Neural Netw.* **144**, 359–371 (2021).
10. Chen, B. et al. Automated discovery of fundamental variables hidden in experimental data. *Nat. Comput. Sci.* **2**, 433–442 (2022).
11. Lu, P. Y., Kim, S. & Soljačić, M. Extracting interpretable physical parameters from spatiotemporal systems using unsupervised learning. *Phys. Rev. X* **10**, 031,056 (2020).
12. Rackauckas, C. et al. Universal differential equations for scientific machine learning. Preprint at arXiv https://doi.org/10.48550/ARXIV.2001.04385 (2020).
13. Yin, Y. et al. Augmenting physical models with deep networks for complex dynamics forecasting. *J. Stat. Mech.* **2021**, 124, 012 (2021).
14. Bronstein, M. M., Bruna, J., Cohen, T. & Veličković, P. Geometric deep learning: grids, groups, graphs, geodesics, and gauges. Preprint at arXiv https://doi.org/10.48550/ARXIV.2104.13478 (2021).
15. Mauroy, A., Susuki, Y. & Mezić, I. *The Koopman Operator in Systems and Control: Concepts, Methodologies, and Applications* https://doi.org/10.1007/978-3-030-35713-9_1 (Springer International Publishing, 2020).
16. Folkestad, C. et al. *2020 American Control Conference (ACC)* (IEEE, 2020).
17. Takeishi, N., Kawahara, Y. & Yairi, T. In *Advances in Neural Information Processing Systems* (eds. Guyon, I. et al.) (Curran Associates, Inc., 2017).
18. Brunton, S. L., Budišić, M., Kaiser, E. & Kutz, J. N. Modern Koopman theory for dynamical systems. *SIAM Review* **64**, 229–340 (2022).
19. Brunton, S. L., Brunton, B. W., Proctor, J. L., Kaiser, E. & Kutz, J. N. Chaos as an intermittently forced linear system. *Nat. Commun.* **8**, 19 (2017).
20. Lusch, B., Kutz, J. N. & Brunton, S. L. Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.* **9**, 4950 (2018).
21. Brunton, S. L., Proctor, J. L. & Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl Acad. Sci. USA* **113**, 3932–3937 (2016).
22. Kaheman, K., Kutz, J. N. & Brunton, S. L. SINDy-PI: a robust algorithm for parallel implicit sparse identification of nonlinear dynamics. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **476**, 20200,279 (2020).
23. Chen, Z., Liu, Y. & Sun, H. Physics-informed learning of governing equations from scarce data. *Nat. Commun.* **12**, 6136 (2021).

24. Champion, K., Lusch, B., Kutz, J. N. & Brunton, S. L. Data-driven discovery of coordinates and governing equations. *Proc. Natl Acad. Sci. USA* **116**, 22,445–22,451 (2019).
25. Dubois, D. M. A model of patchiness for prey-predator plankton populations. *Ecol. Model.* **1**, 67–80 (1975).
26. Comins, H. N. & Blatt, D. W. Prey-predator models in spatially heterogeneous environments. *J. Theor. Biol.* **48**, 75–83 (1974).
27. Kmet', T. & Holčík, J. The diffusive Lotka-Volterra model as applied to the population dynamics of the German carp and predator and prey species in the Danube River basin. *Ecol. Model.* **74**, 277–285 (1994).
28. Ablowitz, M. J. *Nonlinear Dispersive Waves: Asymptotic Analysis and Solitons* (Cambridge Univ. Press, 2011).
29. Lu, C. H., Barsi, C., Williams, M. O., Kutz, J. N. & Fleischer, J. W. Phase retrieval using nonlinear diversity. *Appl. Opt.* **52**, D92–D96 (2013).
30. Kaheman, K., Brunton, S. L. & Kutz, J. N. Automatic differentiation to simultaneously identify nonlinear dynamics and extract noise probability distributions from data. *Mach. Learn. Sci. Technol.* **3**, 015,031 (2022).
31. Ribera, H., Shirman, S., Nguyen, A. V. & Mangan, N. M. Model selection of chaotic systems from data with hidden variables using sparse data assimilation *Chaos* **32**, 063,101 (2022).
32. Kim, S. et al. Integration of neural network-based symbolic regression in deep learning for scientific discovery. In *IEEE Transactions on Neural Networks and Learning Systems* 1–12 (IEEE, 2020).
33. Costa, A. et al. Fast neural models for symbolic regression at scale. Preprint at arXiv https://doi.org/10.48550/ARXIV.2007.10784 (2020).
34. Udrescu, S. M. & Tegmark, M. AI Feynman: a physics-inspired method for symbolic regression. *Sci. Adv.* **6**, eaay2631 (2020).
35. Alon, U. *An Introduction to Systems Biology: Design Principles of Biological Circuits* (CRC Press, 2019)
36. Takens, F. In *Dynamical Systems and Turbulence, Warwick 1980* (eds. Rand, D. & Young, L. S.) (Springer, 1981).
37. Baydin, A. G., Pearlmutter, B. A., Radul, A. A. & Siskind, J. M. Automatic differentiation in machine learning: a survey. *J. Mach. Learn. Res.* **18**, 1–43 (2018).
38. Zhuang, J. et al. *Advances in Neural Information Processing Systems* (eds. Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F. & Lin, H.) (Curran Associates, Inc., 2020).
39. Bradbury, J. et al. JAX: composable transformations of Python+NumPy programs. http://github.com/google/jax (2018).
40. Innes, M. Don't unroll adjoint: differentiating SSA-form programs. Preprint at arXiv https://doi.org/10.48550/ARXIV.1810.07951 (2018).

Acknowledgements

We would like to acknowledge useful discussions with Samuel Kim, Rumen Dangovski, Charlotte Loh, Andrew Ma, and Ileana Rugina. This research is supported in part by the US Department of Defense through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program. This work is further supported in part by the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/). It is also based upon work supported in part by the US Army Research Office through the Institute for Soldier Nanotechnologies at MIT, under Collaborative Agreement Number W911NF-18-2-0048. This material is also based in part upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00111890042. This work is also supported in part by the Air Force Office of Scientific Research under the award number FA9550-21-1-0317. The research was also sponsored in part by the United States Air Force Research Laboratory and the United States Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the US Government. The US Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein.

Author contributions

All three authors contributed to the conception, design, and development of the proposed machine learning framework. P.Y.L. and J.A.B. created and tested an initial implementation of the framework, which was subsequently refactored, refined, and further tested by P.Y.L. The manuscript was written by P.Y.L. with support from J.A.B. and M.S.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42005-022-00987-z.

Correspondence and requests for materials should be addressed to Peter Y. Lu.

Peer review information *Communications Physics* thanks the anonymous reviewers for their contribution to the peer review of this work.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.