OnsagerNet: Learning stable and interpretable dynamics using a generalized Onsager principle

Haijun Yu[*] and Xinyuan Tian

NCMIS & LSEC, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China and School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China

Weinan E

Department of Mathematics and the Program in Applied and Computational Mathematics, Princeton University, Princeton, New Jersey 08544, USA

Qianxiao Li[†]

*Department of Mathematics, National University of Singapore, Singapore 119077 and Institute of High Performance Computing, A*STAR, Singapore 138632*

(Received 24 February 2021; accepted 4 November 2021; published 23 November 2021)

We propose a systematic method for learning stable and physically interpretable dynamical models using sampled trajectory data from physical processes based on a generalized Onsager principle. The learned dynamics are autonomous ordinary differential equations parametrized by neural networks that retain clear physical structure information, such as free energy, diffusion, conservative motion, and external forces. For high-dimensional problems with a low-dimensional slow manifold, an autoencoder with metric-preserving regularization is introduced to find the low-dimensional generalized coordinates on which we learn the generalized Onsager dynamics. Our method exhibits clear advantages over existing methods on benchmark problems for learning ordinary differential equations. We further apply this method to study Rayleigh-Bénard convection and learn Lorenz-like low-dimensional autonomous reduced-order models that capture both qualitative and quantitative properties of the underlying dynamics. This forms a general approach to building reduced-order models for forced-dissipative systems.

DOI: 10.1103/PhysRevFluids.6.114402

I. INTRODUCTION

Discovering mathematical models from observed dynamical data is a scientific endeavor that dates back to the work of Ptolemy, Kepler, and Newton. With the recent advancements in machine learning, data-driven approaches have become viable alternatives to those relying purely on human insight. The former has become an active area of research with many proposed methodologies, which can be roughly classified into two categories. The first is *unstructured* approaches, where the learned dynamics are not *a priori* constrained to satisfy any physical principles. Instead, minimizing reconstruction error or model complexity is the dominant objective. For example, sparse symbolic regression methods try to find minimal models by searching in a big dictionary of candidate analytic

[*] hyu@lsec.cc.ac.cn

[†] qianxiao@nus.edu.sg

representations [1–3]; numerical discretization embedding methods attempt to learn hidden dynamics by embedding known numerical discretization schemes [4–6]; Galerkin-closure methods use neural networks to reduce the projection error of traditional Galerkin methods [7–11]. Another line of unstructured methods directly applies time-series modeling tools such as long short-term memory or gated recurrent units and reservoir computing [12–14] to model temporal evolution of physical processes. For example, it was demonstrated in [12] that reservoir computing can effectively capture the dynamics of high-dimensional chaotic systems, such as the Kuramoto-Sivashinsky equation. These unstructured methods can often achieve high predictive accuracy, but the learned dynamics may not have a clear physical structure and theoretical guarantee of temporal stability. This in turn limits their ability to capture qualitative properties of the underlying model on large time intervals. Such issues are significant in that often the goal of building reduced models is not to reproduce the original dynamics in its entirety but rather to extract some salient and physically relevant insights. This leads to the second class of *structured* approaches, where one imparts from the outset some physically motivated constraints to the dynamical system to be learned, e.g., gradient systems [15,16] and Hamiltonian systems [17–19]. These methods have the advantage that the learned models have predetermined structures and stability may be automatically ensured. However, works in this direction may have the limitation that the structures imposed may be too restrictive to handle complex problems beyond benchmark examples. We demonstrate this point in our benchmark examples to be presented later. It is worth noting that recent works address this issue by a combination of unstructured model fitting with information from an *a priori* known but imperfect model (see, e.g., [20]).

In this paper we propose a systematic method that overcomes the aforementioned limitations. The key idea is a neural network parametrization of a reduced dynamical model, which we call OnsagerNet, based on a highly general extension of the Onsager principle for dissipative dynamics [21,22]. We choose the Onsager principle, which builds on the second law of thermodynamics and microscopic reversibility, as the starting point due to its simplicity, generality, and physical motivation. It naturally gives rise to stable structures for dynamical systems in generalized coordinates and has been used extensively to derive mathematically well-posed models for complex systems in fluid mechanics, materials science, biological science, etc. (see, e.g., [23–29]). However, there are two challenges in using it to find dynamical models with high fidelity: (i) how to choose the generalized coordinates and (ii) how to determine the large amount of free coefficients in the structured dynamical system (some in-depth discussions could be found in [30], Chap. 1). One typical example is the modeling of liquid crystal systems [31,32]. If the underlying system is near equilibrium, it is possible to determine the coefficients of a reduced-order (macroscopic) model from mathematical analysis or quasiequilibrium approximations of the underlying microscopic model (see, e.g., [33–36]). Otherwise, this task becomes significantly harder, yet many interesting phenomena happen in this regime. Therefore, a method to derive high-fidelity macroscopic models that operate far from equilibrium without *a priori* scale information is much sought after.

We tackle these two tasks in a data-driven manner. For the first task we learn an approximately isometric embedding to find generalized coordinates. In the linear case this reduces to the principal component analysis (PCA) and in general we modify the autoencoder (AE) [37,38] with metric-regularization designed for trajectory data. For the second task, we avoid the explicit fitting of the prohibitively large number of response coefficients in linearization approaches. Instead, we parametrize nonlinear relationships between generalized coordinates and the physical quantities governing the dynamics (e.g., free energy and dissipation matrices) as neural networks and train them on observed dynamical data with an embedded Runge-Kutta method. The overall approach of OnsagerNet gives rise to stable and interpretable dynamics, yet retaining a degree of generality in the structure to potentially capture complex interactions. By interpretability, we mean that the structure of OnsagerNet parametrization has physical origins, and its learned components can thus be used for subsequent analysis (e.g., visualizing energy landscapes, as shown in Sec. IV). Moreover, we show that the network architecture that we used to parametrize the hidden dynamics has a more general hypothesis space than other existing structured approaches and can provably represent many dynamical systems with physically meaningful origins (e.g., systems described by generalized Poisson brackets). At the same time, the stability of OnsagerNet is ensured by its internal structure motivated by physical considerations, and this does not reduce its approximation capacity when applied to dynamical data arising from dissipative processes. We demonstrate the last point by applying the method to the well-known Rayleigh-Bénard convection (RBC) problem, based on which Lorenz discovered a minimal three-mode ordinary differential equation (ODE) system that exhibits chaos [39]. Lorenz used three dominant modes obtained from linear stability analysis as reduced coordinates and derived the governing equations by a Galerkin projection. As a severely truncated low-dimensional model, the Lorenz model has limited quantitative accuracy when the system is far from the linear stability region. In fact, it was shown in [40] that one needs to use more than 100 modes in a Fourier-Galerkin method to get quantitative accuracy for the two-dimensional RBC problem that is far from the linear region. In this paper we show that OnsagerNet is able to learn low-dimensional Lorenz-like autonomous ODE models with few modes yet having high quantitative fidelity to the underlying RBC problem. This validates, and improves upon, the basic approach of Lorenz to capture complex flow phenomena using low-dimensional nonlinear dynamics. Furthermore, this demonstrates the effectiveness of a principled combination of physics and machine learning when dealing with data from scientific applications.

II. FROM THE ONSAGER PRINCIPLE TO ONSAGERNET

The Onsager principle [21,22] is a well-known model for the dynamics of dissipative systems near equilibrium. Given generalized coordinates $h = (h_1, \dots, h_m)$, their dynamical evolution is modeled by

$$M\dot{h} = -\nabla V(h), \tag{1}$$

where $V : \mathbb{R}^m \to \mathbb{R}$ is a potential function, typically with a thermodynamic interpretation such as free energy or negated entropy. The matrix M models the energy dissipation of the system (or entropy production) and is positive semidefinite in the sense that $h \cdot Mh \geqslant 0$ for all h, owing to the second law of thermodynamics. An important result due to Onsager, known as the reciprocal relations, shows that if the system possesses microscopic time reversibility, then M is symmetric, i.e., $M_{ij} = M_{ji}$.

The dynamics (1) is simple, yet it can model a large variety of systems close to equilibrium in the linear response regime (see, e.g., [24,41]). However, many dynamical systems in practice are far from equilibrium, possibly sustained by external forces (e.g., the flow-induced transition in liquid crystals [33,34] and the kinetic equation with large Knudsen number [42]), and in such cases (1) no longer suffices. Thus, it is an important task to generalize or extend (1) to handle these systems.

The extension of known but limited physical principles to more general domains has been a continual exercise in the development of theoretical physics. In fact, (1) itself can be viewed as a generalization of Rayleigh's phenomenological principle of least dissipation for hydrodynamics [43]. While classical nonequilibrium statistical physics has been further developed in many aspects, including the study of transport phenomena via Green-Kubo relations [44,45] and the relations between fluctuations and entropy production [46], an extension of the dynamical model (1) to model general nonequilibrium systems remains elusive. Furthermore, whether a simple yet general extension with solid physical background exists at all is questionable.

This paper takes a rather different approach. Instead of the possibly arduous task of developing a general dynamical theory of nonequilibrium systems from first principles, we seek a data-driven extension of (1). In other words, given a dynamical process to be modeled, we posit that it satisfies some reasonable extension of the usual Onsager principle and determine its precise form from data. In particular, we define the generalized Onsager principle

$$[M(h) + W(h)]\dot{h} = -\nabla V(h) + g(h), \tag{2}$$

where the matrix valued function M (W) maps h to symmetric positive-semidefinite (antisymmetric) matrices. The last term $g : \mathbb{R}^m \to \mathbb{R}^m$ is a vector field that represents external forces to the otherwise closed system and may interact with the system in a nonlinear way. The antisymmetric term W models the conservative part of the system and together with g greatly extends the degree of applicability of the classical Onsager principle.

We will assume that $M(h) + W(h)$ is invertible everywhere, and hence $[M(h) + W(h)]^{-1}$ can be written as a sum of a symmetric positive-semidefinite matrix, denoted by $\tilde{M}(h)$, and a skew-symmetric matrix, denoted by $\tilde{W}(h)$ (see Theorem 5 in Appendix B for a proof). Thus we have an equivalent form for Eq. (2),

$$\dot{h} = -[\tilde{M}(h) + \tilde{W}(h)]\nabla V(h) + f(h), \tag{3}$$

where $f = (M + W)^{-1}g$. We will now work with (3) as it is more convenient.

We remark that the form of the generalized Onsager dynamics (2) is not an arbitrary extension of (1). In fact, the dissipative-conservative decomposition ($M + W$) and dependence on h are well motivated from classical dynamics. To arrive at the former, we make the crucial observation that a dynamical system defined by generalized Poisson brackets [30] has precisely this decomposition (see Appendix A). Moreover, such forms also appeared in partial differential equation (PDE) models for complex fluids [25,47]. Note that general Poisson brackets are required to satisfy the Jacobi identity (see, e.g., [30,48]), but in our approach we do not enforce such a condition. We refer to [49] for a neural network implementation based on the generalized Poisson brackets.

A. Generalization of the Onsager principle for model reduction

Here we show that the generalized Onsager principle (2) or its equivalent form (3) is invariant under coordinate transformations and is a suitable structured form for model reduction. In the original Onsager principle [21,22], the system is assumed to be not far from equilibrium, such that the system dissipation takes a quadratic form $\|\dot{h}\|_M^2$, i.e., the matrix M is assumed to be constant. In our generalization, we assume M is a general function that depends h. Here we give some arguments why this is necessary and how it can be obtained if the underlying dynamics is given from the viewpoint of model reduction.

To explain how nonlinear extension is necessary, we assume that the underlying high-dimensional dynamics which produces the observed trajectories satisfies the form of the generalized Onsager principle with constant M and W. The dynamics described by an underlying PDE after spatial semidiscretization takes the form

$$(M + W)\dot{u} = -\nabla_u V + g, \quad u \in \mathbb{R}^N, \tag{4}$$

where M is a symmetric positive-semidefinite *constant* matrix and W is an antisymmetric matrix. For the dynamics to make sense, we need $M + W$ to be invertible. Further, $\nabla_u V$ is a column vector of length N. By taking the inner product of (4) with $\dot{u}$, we have an energy dissipation relation of the form

$$\dot{V} = -\dot{u} \cdot M\dot{u} + g \cdot \dot{u}.$$

Now suppose that u has a solution in a low-dimensional manifold that could be well described by hidden variables $h(t) \in \mathbb{R}^m$ with $m \ll N$. Defining the low-dimensional solution as $u = u(h(t)) + \varepsilon$ and plugging it into (4), we obtain

$$(M + W)\nabla_h u\dot{h} = -\nabla_u V(u(h)) + g + O(\varepsilon).$$

Multiplying $J^T := (\nabla_h u)^T$ on both sides of the above equation, we have

$$J^T(M + W)J\,\dot{h} = J^T[-\nabla_u V(u(h)) + g] + O(\varepsilon).$$

So we obtain the ODE system with model reduction error $O(\varepsilon)$,

$$(\bar{M} + \bar{W})\dot{h} = -\nabla_h V + \bar{g}, \tag{5}$$

where

$$\bar{M} = J^T M J, \quad \bar{W} = J^T W J, \quad \bar{g} = J^T g. \tag{6}$$

Note that as long as $\nabla_h u$ has a full column rank, $\bar{M} + \bar{W}$ is invertible, so the ODE system makes sense. Now $\bar{M}$, $\bar{W}$, and $\bar{g}$ depend on h in general if $\nabla_h u$ is not a constant matrix. Moreover, if the solutions considered all exist exactly in a low-dimensional manifold, i.e., $\varepsilon = 0$ in the above derivation, then (5) is exact, which means the generalized Onsager principle is invariant to nonsingular coordinate transformations.

Remark 1. For the underlying dynamics given in the alternative form

$$\dot{u} = -(M + W)\nabla_u V + f, \quad u \in \mathbb{R}^N. \tag{7}$$

We first rewrite it in the form (4) as

$$(M + W)^{-1}\dot{u} = -\nabla_u V + (M + W)^{-1} f.$$

Then we use the same procedure as before to obtain

$$J^T (M + W)^{-1} J\, \dot{h} = J^T [-\nabla_u V(u(h)) + (M + W)^{-1} f] + O(\varepsilon),$$

from which we obtain a reduced model with error $O(\varepsilon)$,

$$\dot{h} = -(\tilde{M} + \tilde{W})\nabla_h V + \tilde{f}, \tag{8}$$

where $\tilde{M} = (G + G^T)/2$, $\tilde{W} = (G - G^T)/2$, $\tilde{f} = GJ^T (M + W)^{-1} f$, and $G = [J^T (M + W)^{-1} J]^{-1}$.

Remark 2. When M, W, and g in (4) are constant, if linear PCA is used for model reduction, in which $\nabla_h u$ is a constant matrix, then $\bar{M}$, $\bar{W}$, and $\bar{g}$ are constants. However, if we consider the incompressible Navier-Stokes equations written in the form (4), then M and W are not constant matrices. We obtain nonlinear coefficients in both formulations for the model reduction problem of incompressible Naiver-Stokes equations.

Note that the presence of a state h dependence on all the terms implies that we are not linearizing the system about some equilibrium state, as is usually done in linear response theory. Consequently, the functions W, M, g, and V may be complicated functions of h and we will learn them from data by parametrizing them as suitably designed neural networks. In this way, we preserve the physical intuition of nonequilibrium physics (dissipation term M, conservative term W, potential term V, and external fields g) yet exploit the flexibility of function approximation using data and learning.

In summary, we can view (2) and (3) as an extension of the classical Onsager principle to handle systems far from equilibrium or as a reduction of a high-dimensional dynamical system defined by generalized Poisson brackets. Both of these dynamics are highly general in their respective domains of application and serve as solid foundations on which we build our data-driven methodology.

B. Dissipative structure and temporal stability of OnsagerNet

From a mathematical point of view, modeling dynamics using (2) or (3) also has clear advantages in that the learned system is well behaved as it evolves through time, unlike unstructured approaches such as dynamic mode decomposition (see, e.g., [50,51]) and direct parametrization by neural networks, which may achieve a short-time trajectorywise accuracy, but cannot ensure mid- to long-time stability as the learned system evolves in time. In our case, we can show in the following result that under mild conditions, the dynamics described by (2) or (3) automatically ensures a dissipative structure and remains stable as the system evolves in time.

Theorem 1. The solutions to the system (3) satisfy an energy evolution law

$$\dot{V}(h) = -\|\nabla V(h)\|_{\tilde{M}}^2 + f(h)\cdot\nabla V(h). \tag{9}$$

If we assume further that there exist positive constants α and β and non-negative constants c_0 and c_1 such that $h\cdot\tilde{M}h \geqslant \alpha\|h\|^2$ (uniformly positive dissipation), $V(h) \geqslant \beta\|h\|^2$ (coercive potential), and $\|f(h)\| \leqslant c_0 + c_1\|h\|$ (external force has linear growth), then $\|h(t)\|, V(h(t)) < \infty$ for all t. In particular, if there is no external force, then $V(h(t))$ is nonincreasing in t.

Proof. Equation (9) can be obtained by pairing both sides of Eq. (3) with ∇V. We now prove boundedness. Using Young's inequality, we have, for any ε,

$$\begin{aligned} f(h)\cdot\nabla V(h) &\leqslant \varepsilon\|\nabla V(h)\|^2 + \frac{1}{4\varepsilon}\|f(h)\|^2 \\ &\leqslant \varepsilon\|\nabla V(h)\|^2 + \frac{1}{2\varepsilon}\big(c_0^2 + c_1^2\|h\|^2\big). \end{aligned}$$

Putting the above estimate with $\varepsilon = \alpha$ into (9), we get

$$\begin{aligned} \dot{V}(h) &= -\|\nabla V(h)\|_{\tilde{M}}^2 + \alpha\|\nabla V(h)\|^2 + \frac{1}{2\alpha}\big(c_0^2 + c_1^2\|h\|^2\big) \\ &\leqslant \frac{c_1^2}{2\alpha\beta}V(h) + \frac{c_0^2}{2\alpha}. \end{aligned}$$

By the Grönwall inequality, we obtain

$$V(h) \leqslant \begin{cases} e^{(c_1^2/2\alpha\beta)t}V_0 + \big(e^{(c_1^2/2\alpha\beta)t} - 1\big)\frac{c_0^2\beta}{c_1^2}, & c_1 > 0 \\ V_0 + \frac{c_0^2}{2\alpha}t, & c_1 = 0, \end{cases}$$

where $V_0 = V(h(0))$. Finally, by the assumption $\|h\|^2 \leqslant \frac{1}{\beta}V$, h is bounded. Note that when $f(h) \equiv 0$, we have $c_0 = c_1 = 0$, so the above inequality is reduced to $V(h) \leqslant V_0$, which can be obtained directly from (9) without the requirement of $\alpha > 0$ and $\beta > 0$. ■

Theorem 1 shows that the dynamics is physical under the assumed conditions and we will design our neural network parametrization so that these conditions are satisfied.

III. ONSAGERNET ARCHITECTURE AND LEARNING ALGORITHM

In this section we introduce the detailed network architecture for the parametrization of the generalized Onsager dynamics and discuss the details of the training algorithm.

A. Network architecture

We implement the generalized Onsager principle based on Eq. (3). Here $\tilde{M}(h)$, $\tilde{W}(h)$, $V(h)$, and $f(h)$ are represented by neural networks with shared hidden layers and are combined according to (3). The resulting composite network is named OnsagerNet. Accounting for (anti)symmetry, the numbers of independent variables in $\tilde{M}(h)$, $\tilde{W}(h)$, $V(h)$, and $f(h)$ are $(m+1)m/2$, $(m-1)m/2$, 1, and m, respectively.

One important fact is that $V(h)$, as an energy function, should be lower bounded. To ensure this automatically, one may define $V(h) = \frac{1}{2}U(h)^2 + C$, where C is some constant that is smaller than or equal to V's lower bound. Since the constant C does not affect the dynamics, we drop it in numerical implementation. The actual form we take is

$$V(h) = \frac{1}{2}\sum_{i=1}^{m}\left(U_i(h) + \sum_{j=1}^{m}\gamma_{ij}h_j\right)^2 + \beta\|h\|^2, \tag{10}$$

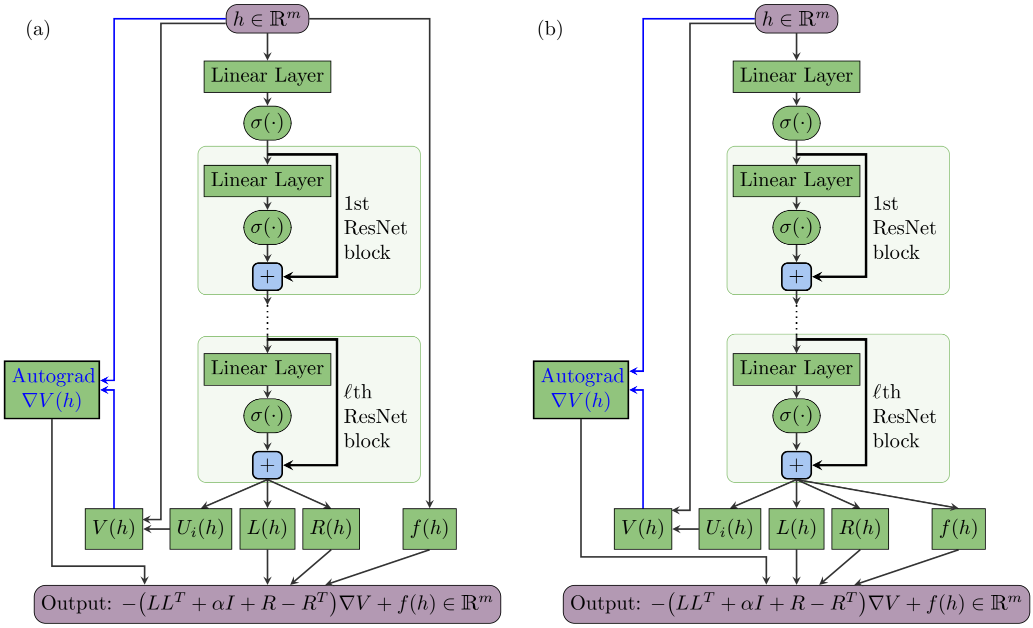


FIG. 1. Computational graph of OnsagerNet for (a) affine $f(h)$ and (b) nonlinear $f(h)$.

where $\beta \geqslant 0$ is a positive hyperparameter as assumed in Theorem 1 for forced systems. Here U_i has a structure similar to one component of $\tilde{W}(h)$. We use m terms in the form (10) to ensure that an original quadratic energy after a dimension reduction can be handled easily. To see this, suppose the potential function for the high-dimensional problem (with coordinates u) is defined as $V(u) = u^T A u$, with A symmetric positive definite. Further, let a linear PCA $u \approx u_0 + Jh$ be used for dimensionality reduction. Then $V(h) \approx (u_0 + Jh)^T A(u_0 + Jh) = \|u_0\|^2 + 2u_0^T AJh + h^T J^T AJh = \|v_0 + Gh\|^2 + \text{const}$, where $G^T G = J^T AJ$ and $v_0^T = u_0^T AJG^{-1}$. Hence, with γ_{ij} representing the matrix G, a constant U_i suffices to fully represent quadratic potentials. The autograd mechanism implemented in PYTORCH [52] is used to calculate $\nabla V(h)$.

To ensure the positive-semidefinite property of $\tilde{M}(h)$, we let $\tilde{M}(h) = L(h)L(h)^T + \alpha I$, where $L(h)$ is a lower triangular matrix, I is the identity matrix, and $\alpha \geqslant 0$. Note that the degree of freedom of $L(h)$ and $\tilde{W}(h)$ can be combined into one $m \times m$ matrix, whose upper triangular part $R(h)$ determines $\tilde{W}(h) = R(h) - R(h)^T$. A standard multilayer perception neural network with residual network structure (ResNet) [53] is used to generate adaptive bases, which takes $(h_1, \ldots, h_m)$ as input, and outputs $\{L(h), R(h), U_i(h)\}$ as linear combinations of those bases. The external force $f_i(h)$ is parametrized based on *a priori* information and should be of limited capacity so as not to dilute the physical structures imposed by the other terms. For the forced systems considered in this paper, we typically take f_i as affine functions of h. The final output of the OnsagerNet is given by

$$\dot{h}_i = \sum_{k=1}^{m} [L(h)L(h)^T + \alpha I + \tilde{W}(h)]_{i,k}[-\partial_{h_k} V(h)] + f_i(h), \quad i = 1, \ldots, m, \tag{11}$$

where $V(h)$ is defined by (10). Note that in an unforced system we have $\alpha = \beta = 0$, since they are only introduced in a forced system to ensure the stability of the learned system as required by Theorem 1. The computation procedure of OnsagerNet is described in Architecture I. The full architecture is summarized in Fig. 1.

Architecture 1 OnsagerNet($\alpha, \beta, l, n_l; h$).

```
Input: $h \in \mathbb{R}^m$, parameters $\alpha \geqslant 0$ and$\beta \geqslant 0$, activation function $\sigma_A$,number of hidden layers $l$, and number of nodes in each hidden layer $n_l$
Output: OnsagerNet($\alpha, \beta, l, n_l; h$) $\in \mathbb{R}^m$
1: Calculate the shared subnet output using an $l$-layer neural network $\phi = \mathrm{MLP}(h) \in \mathbb{R}^{n_l}$. Here MLP has $l-1$ hidden layers and one output layer; each layer has $n_l$ nodes. An activation function $\sigma_A$ is applied for all hidden layers and output layer. If $l > 1$, ResNet shortcuts are used.
2: Evaluate $U_i$ using a linear layer as $U_i = \sum_j \omega_{ij}^{(1)} \phi_j + b_i^{(1)}$; calculate $V$according to (10).
3: Use the autograd mechanism of PYTORCH to calculate the gradient $\nabla_{h_k} V(h)$, $k = 1, \dots, m$.
4: Evaluate $A \in \mathbb{R}^{m^2}$ as $A_i = \sum_j \omega_{ij}^{(2)} \phi_j + b_i^{(2)}$.
5: Reshape $A$ as an $m \times m$ matrix and take its lower triangular part including the main diagonal as $L$ and the upper triangular part without the main diagonal as $R$ to form $\tilde{W} = R - R^T$.
6: if the system is forced then
7:     calculate the external force or control $f_i$ using an a priori form.
8: else
9:     take $f_i = 0$.
10: Calculate the output of OnsagerNet using (11).
11: return OnsagerNet($\alpha, \beta, l, n_l; h$).
```

B. Training objective and algorithm

To learn the ODE model represented by OnsagerNet based on sampled ODE trajectory data, we minimize the loss function

$$\mathcal{L}_{\mathrm{ODE}} = \frac{1}{|S|} \sum_{(h(t), h(t+\tau)) \in S} \frac{1}{\tau^2} \left\| h(t+\tau) - \mathrm{RK2}\left(\mathrm{OnsagerNet}; h(t), \frac{\tau}{n_s}, n_s \right) \right\|^2. \tag{12}$$

Here τ is the time interval of sample data, S is the sample set, RK2 stands for a second-order Runge-Kutta method (Heun method), and n_s is the number of RK2 steps used to forward the solution of OnsagerNet from snapshots at t to $t + \tau$. Other Runge-Kutta methods can be used. For simplicity, we only present results using the Heun method in this paper. This Runge-Kutta embedding method has several advantages over the linear multistep methods [5,6], e.g., the variable-time-interval case and long-time-interval case can be easily handled by Runge-Kutta methods.

With the definition of the loss function and model architecture, we can then use standard stochastic gradient algorithms to train the network. Here we will use the Adam optimizer [54,55] to minimize the loss function with a learning rate scheduler that halves the learning rate if the loss is not decreasing in a certain number of iterations.

C. Reduced-order model via embedding

The preceding section described the situation when no dimensionality reduction is sought or required and the Onsager dynamics is learned directly on the original trajectory coordinates. On the other hand, if the data are generated from a numerical discretization of some PDE or a large ODE system and we want to learn a small reduced-order model, then a dimensionality reduction procedure is needed. One can use either linear principal component analysis or nonlinear embedding, e.g., the autoencoder, to find a set of good latent coordinates from the high-dimensional data. When PCA is used, we perform PCA and OnsagerNet training separately. When an autoencoder is used, we can train it either separately or together with the OnsagerNet. The end-to-end training loss function is taken as

$$\begin{aligned}\mathcal{L}_{\text{tot}} &= \mathcal{L}_{\text{AE}} + \mathcal{L}_{\text{ODE}} + \mathcal{L}_{\text{reg}} \\ &= \frac{1}{|S|}\sum_{(u(t),u(t+\tau))\in S}\Big\{\beta_{\text{ae}}\|u(t) - \psi\circ\varphi\circ u(t)\|^2 + \beta_{\text{ae}}\|u(t+\tau) - \psi\circ\varphi\circ u(t+\tau)\|^2 \\ &\quad + \frac{1}{\tau^2}\left\|\varphi\circ u(t+\tau) - \text{RK2}\left(\text{OnsagerNet}; \varphi\circ u(t), \frac{\tau}{n_s}, n_s\right)\right\|^2 \\ &\quad + \beta_{\text{isom}}[|\|u(t+\tau) - u(t)\|^2 - \|\varphi\circ u(t+\tau) - \varphi\circ u(t)\|^2| - \alpha_{\text{isom}}]_+\Big\},\end{aligned} \tag{13}$$

where φ and ψ stand for the encoder function and decoder function of the autoencoder, respectively. In the last term, $|\|u(t+\tau) - u(t)\|^2 - \|\varphi\circ u(t+\tau) - \varphi\circ u(t)\|^2|$ is an estimate of the isometric loss (i.e., deviation from φ being an isometry) of the encoder function based on trajectory data, with α_{isom} being a constant smaller than the average isometric loss of the PCA dimension reduction. Here $(\cdot)_+$ stands for positive part, β_{isom} is a penalty constant, and β_{ae} is a parameter to balance the autoencoder accuracy and OnsagerNet fitting accuracy.

The choice of autoencoder architecture follows from our observation that PCA performed respectably on a number of examples we studied. Thus, we build the autoencoder by extending the basic PCA. The resulting architecture, which we call PCAResNet, is a stacked architecture with each layer consisting of a fully connected autoencoder block with a PCA-type shortcut connection

$$h^{k+1} = \text{PCA}_{n_k,n_{k+1}}(h^k) + W_2^k\sigma\left(W_1^k h^k + b^k\right), \quad k = 0, \dots, L-1, \tag{14}$$

where $n_{k+1} < n_k$ and $\text{PCA}_{n_k,n_{k+1}}$ is a PCA transform from dimension n_k to dimension n_{k+1}. Here σ is a smooth activation function. The parameters W_2^k, W_1^k, and b^k in the encoder are initialized close to zero such that the encoder becomes a small perturbation of the PCA. On can regard such an autoencoder as a nonlinear extension of PCA. The decoder is designed similarly. We note that there was a similar but different autoencoder proposed to find slow variables [56].

IV. APPLICATIONS

In this section we present various applications of the OnsagerNet approach. We will start with benchmark problems and then proceed to investigate more challenging settings such as the Rayleigh-Bénard convection problem. The code that can be used to reproduce these experiments and downloadable links for the training data sets are provided in [57].

A. Benchmark problem 1: Deterministic damped Langevin equation

Here we use the OnsagerNet to learn a deterministic damped Langevin equation

$$\dot{x} = v, \quad \dot{v} = -\frac{\gamma}{m}\dot{x} - \frac{1}{m}\nabla_x U(x). \tag{15}$$

The friction coefficient γ may be a constant or a parameter that depends on v. For the potential $U(x)$, we consider two cases: the Hookean spring

$$U(x) = \frac{\kappa}{2}x^2 \tag{16}$$

and the pendulum model

$$U(x) = \frac{4\kappa}{\pi^2}\left[1 - \cos\left(\frac{\pi x}{2}\right)\right]. \tag{17}$$

Note that no dimensionality reduction is required here and the coordinate $h = (x, v)$ entails the full phase space. The goal of this toy example is to quantify the accuracy and stability of the OnsagerNet as well as to highlight the interpretability of the learned dynamics.

We normalize the parameters γ and κ by m, i.e., we take $m = 1$. To generate data, we simulate the systems using a third-order strong-stability-preserving Runge-Kutta method [58] for a fixed period of time with initial conditions $\{x_0, v_0\}$ sampled from $\Omega_S = [-1, 1]^2$. Then we use OnsagerNet (11) to learn an ODE system by fitting the simulated data.

In particular, we will demonstrate that the energy V learned has physical meaning. Note that the energy function in the generalized Onsager principle need not be unique. For example, for the heat equation $\dot{u} = \Delta u$, both $\frac{1}{2}\|u\|^2$ and $\frac{1}{2}\|\nabla u\|^2$ can serve as an energy functional governing the dynamics, with diffusion operators (M matrix) being $-\Delta$ and the identity, respectively. The linear Hookean model is similar. Let $V(x, v) = \frac{1}{2}\kappa x^2 + \frac{1}{2}v^2$. Then

$$\begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\kappa & -\gamma \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} = -\begin{pmatrix} 0 & -1 \\ 1 & \gamma \end{pmatrix} \nabla V(x, v). \tag{18}$$

The eigenvalue of the matrix $A = \left(\begin{smallmatrix} 0 & 1 \\ -k & -\gamma \end{smallmatrix}\right)$ is $\lambda_{1,2} = -\frac{\gamma}{2} \pm \frac{1}{2}\sqrt{\gamma^2 - 4k}$. When $\gamma \geqslant 2\sqrt{k}$, we always obtain real negative eigenvalues and the system is overdamped. From Eq. (18) we may define another energy $\tilde{V}(x, v) = \frac{1}{2}x^2 + \frac{1}{2}v^2$ with the dissipative matrix and conservative matrix $-\frac{1}{2}(A + A^T)$ and $\frac{1}{2}(A - A^T)$, respectively. For this system, $\hat{V}(x, v) := \tilde{V}(F(x, v))$ with any nonsingular linear transform F could serve as energy, and the corresponding dynamics is

$$\begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} = A(F^T F)^{-1} \nabla \hat{V}.$$

Hence, we use this transformation to align the learned energy before making a comparison with the exact energy function.

Let us now present the numerical results. We test two cases: (i) the Hookean model with $k = 4$ and $\gamma = 3$ and (ii) the pendulum model with $k = 4$ and $\gamma(v) = 3|v|^2$. To generate sample data, we simulate the ODE systems to obtain 100 trajectories with uniform random initial conditions $(x, v) \in [-1, 1]^2$. For each trajectory, 100 pairs of snapshots at $(iT/100, iT/100 + 0.001)$, $i = 0, \ldots, 99$, are used as sample data. Here $T = 5$ is the chosen time period of sampled trajectories. Snapshots from the first 80 trajectories are taken as the training set, while the remaining snapshots are taken as the test set.

We test three different methods for learning dynamics: OnsagerNet, a symplectic dissipative ODE net (SymODEN [17]), and a simple multilayer perception ODE network (MLP-ODEN) with ResNet structure. To make the numbers of trainable parameters in three ODE nets comparable, we choose $l = 1$ and $n_l = 12$ for OnsagerNet, $n_l = 17$ for SymODEN, and MLP-ODEN with two hidden layers, each layer having nine hidden units, such that the total numbers of tunable parameters in OnsagerNet, SymODEN and MLP-ODEN are 120, 137, and 124, correspondingly.

To test the robustness of those networks paired with different activation functions, five C^1 activation functions are tested, including ReQU, ReQUr, softplus, sigmoid, and tanh. Here ReQU, defined as $\mathrm{ReQU}(x) := x^2$ if $x \geqslant 0$ and 0 otherwise, is the rectified quadratic unit studied in [59]. Since ReQU is not uniformly Lipschitz continuous, we introduce ReQUr as a regularization of ReQU, defined as $\mathrm{ReQUr}(x) := \mathrm{ReQU}(x) - \mathrm{ReQU}(x - 0.5)$.

The networks are trained using a new version of Adam [55] with minibatches of size 200 and initial learning rate 0.0256. The learning rate is halved if the loss is not decreasing in 25 epochs. The default number of iterations is set to 600 epochs.

For the Hookean case, the mean square error (MSE) loss on the testing set can be reduced to about 10^{-5}–10^{-8} for all three ODE nets, depending on different random seeds and activation functions used. For the nonlinear pendulum case, the MSE loss on the testing set can be reduced to about 10^{-4}–10^{-5} for OnsagerNet and 10^{-3}–10^{-5} for MLP-ODEN, but only 10^{-2} for SymODEN (see Fig. 2). The reason for the low accuracy of SymODEN is that in the SymODEN, the diffusion matrix is assumed to be a function of general coordinate x only [17], but here in the damped pendulum problem, the diffusion term depends on v. From the test results presented in Fig. 2(a), we see that the results of OnsagerNet are not sensitive to the nonlinear activation functions used.

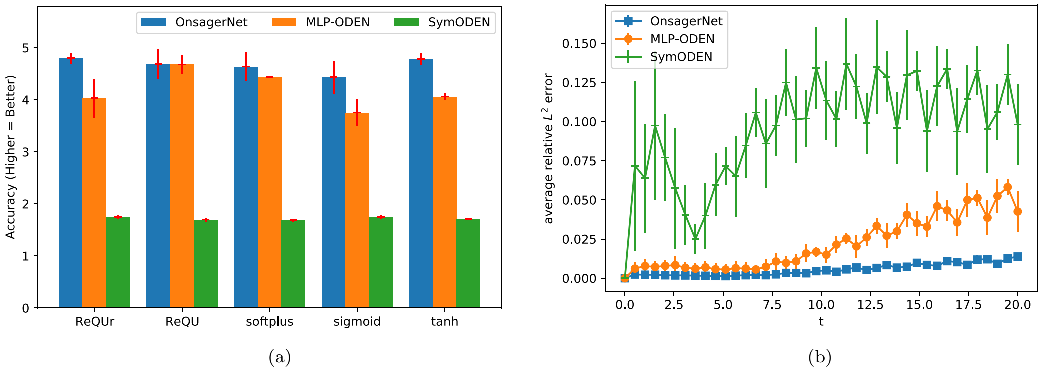


FIG. 2. Accuracy of learned dynamics by using three different ODE neural networks and five different activation functions for the nonlinear damped pendulum problem with $\kappa = 4$ and $\gamma = 3v^2$. (a) Testing MSE accuracy. The height of the bars stands for$-\log_{10}$ of the testing MSE. The results are averages from training with three different random seeds. The heights of the red crosses on top of the bars indicate standard deviations. (b) Relative error of predictions versus time for three ODE neural networks with the ReQUr activation function.

Moreover, Fig. 2(b) shows that OnsagerNet has much better long-time prediction accuracy. Since the nonlinearities in many practical dynamical systems are of polynomial type, we will mainly use ReQU and ReQUr as activation functions for other numerical experiments in this paper.

In Fig. 3 we plot the contours of learned energy functions using OnsagerNet and compare with the exact ground truth total energy function $U(x) + v^2/2$. We observe a clear correspondence up to rotation and scaling for the linear Hookean model case and a scaling of the nonlinear pendulum case. In all cases, the minimum ($x = 0$, $v = 0$) is accurately captured. Note that we used a linear transform to align the learned free energy. After the alignment, the relative L^2-norm errors between the learned and physical energy for the two tested cases are 6.3×10^{-3} and 8.6×10^{-2}, respectively. To verify that the OnsagerNet approach is able to learn nonquadratic potentials, we also test an example with an exact double-well-type potential $U(x) = (x^2 - 0.5)^2$ and $\gamma = 3$. The relative L^2-norm error between the learned and exact energy is 7.5×10^{-2}, where a simple min-max rescaling is used before calculating the numerical error for this example.

In Fig. 4 we plot the trajectories for exact dynamics and learned dynamics with initial values starting from four boundaries of the sample region. We see that the learned dynamics are quantitatively accurate, and the qualitative features of the phase space are also preserved over long times.

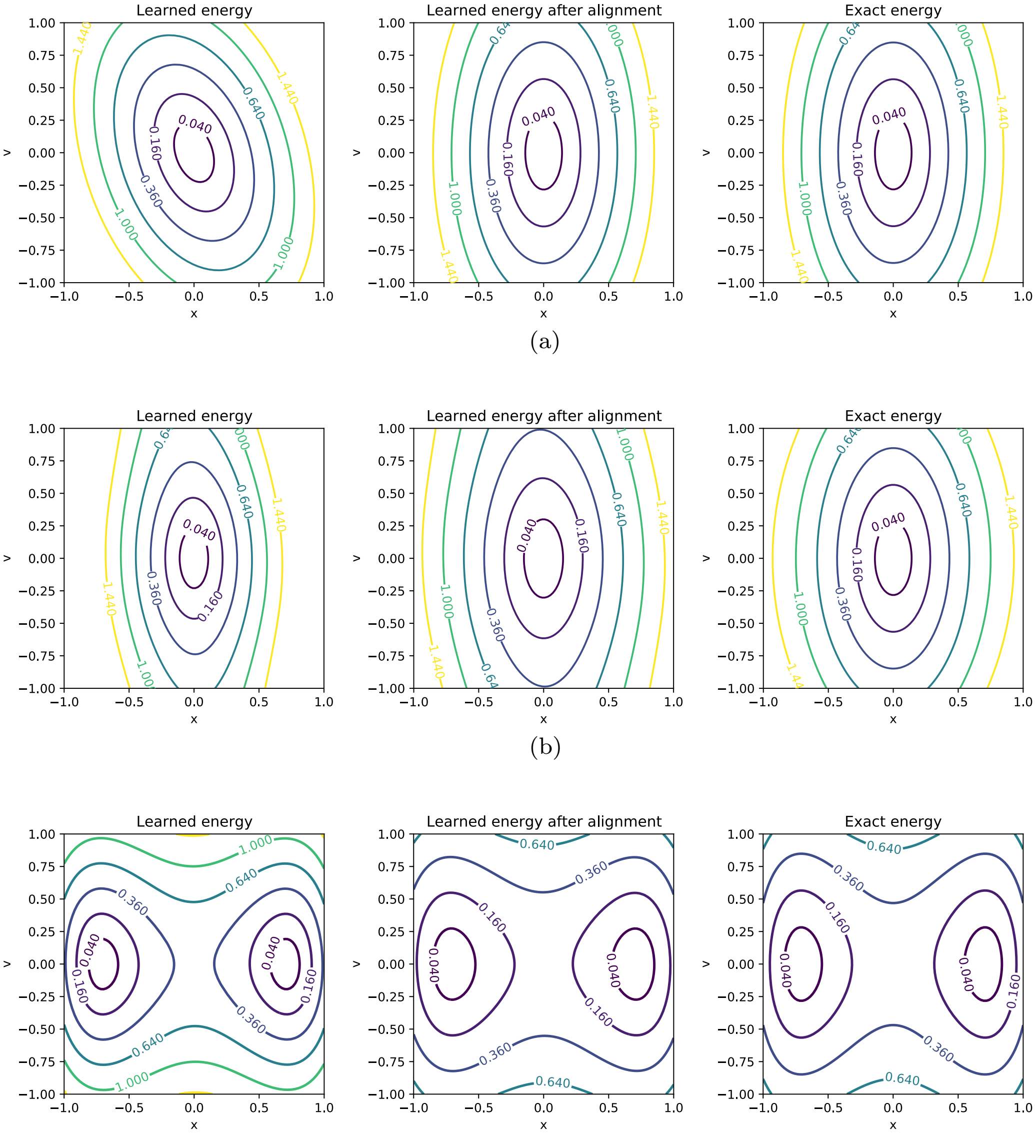
B. Benchmark problem 2: The Lorenz system

The previous oscillator models have simple Hamiltonian structures, so it is plausible that with some engineering one can obtain a special structured model that works well for such systems. In this section we consider a target nonlinear dynamical system that may not have such a simple structure. However, we will show that, owing to the flexibility of the generalized Onsager principle, OnsagerNet still performs well. Concretely, we consider the well-known Lorenz system [39]

$$\frac{dX}{d\tau} = -\sigma X + \sigma Y, \tag{19}$$

$$\frac{dY}{d\tau} = -XZ + rX - Y, \tag{20}$$

$$\frac{dZ}{d\tau} = XY - bZ, \tag{21}$$



(a)

(b)

(c)

FIG. 3. Learned energy functions and the exact energy functions in three problems: (a) ReQUr OnsagerNet results for the Hookean model with $k = 4$ and $\gamma = 3$, (b) ReQU OnsagerNet results for the pendulum model with $k = 4$ and $\gamma = 3|v|^2$, and (c) ReQU OnsagerNet results for the double-well potential $U(x) = (x^2 - 0.5)^2$ with $\gamma = 3$.

where $b > 0$ is a geometric parameter, σ is the Prandtl number, and r is the rescaled Rayleigh number.

The Lorenz system (19)–(21) is a simple autonomous ODE system that produces chaotic solutions, and its bifurcation diagram is well studied [60,61]. To test the performance of OnsagerNet, we fix $b = \frac{8}{3}$ and $\sigma = 10$ and vary r as commonly studied in the literature. For the $b = \frac{8}{3}$ and $\sigma = 10$ case, the first (pitchfork) bifurcation of the Lorenz system happens at $r = 1$, followed by a homoclinic explosion at $r \approx 13.92$ and then a bifurcation to the Lorenz attractor at $r \approx 24.06$.

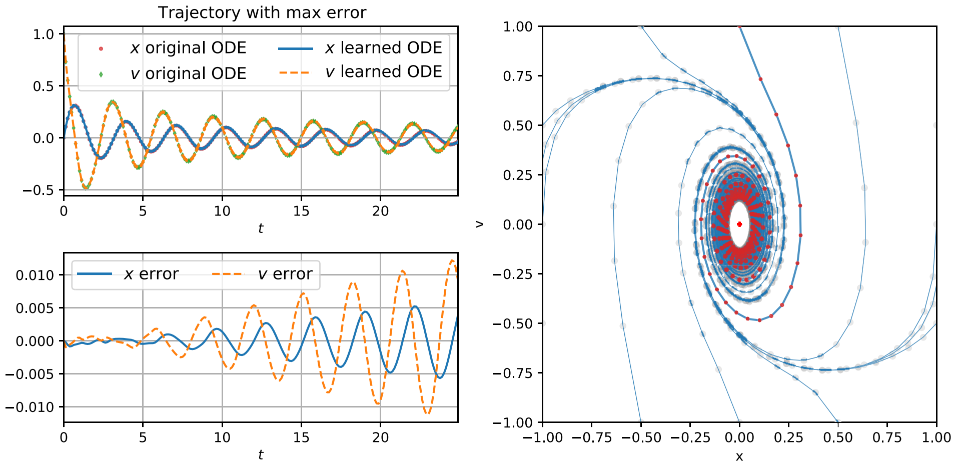


FIG. 4. Results of the learned ODE model by ReQU OnsagerNet for the nonlinear damped pendulum problem. In the right plot, the dots are trajectory snapshots generated from exact dynamics and the solid curves are trajectories generated from learned dynamics. The red cross is the fixed point numerically calculated for the learned ODE system. Note that the time period of sampled trajectories used to train the OnsagerNet is $T = 5$.

Soon after, the Lorenz attractor becomes the only attractor at $r \approx 24.74$ (see, e.g., [62]). To show that OnsagerNet is able to learn systems with different kinds of attractors and chaotic behavior, we present results for $r = 16$ and 28.

The procedure of generating sample data and training is similar to the previously discussed case of learning Langevin equations, except that here we set $\alpha = 0.1$, $\beta = 0.1$, and a linear representation for $f(h)$ in OnsagerNet (with $l = 1$ and $n_l = 20$). This is because the Lorenz system is a forced system. The results for the case $r = 16$ are presented in Fig. 5. We see that both the trajectories and the stable fixed points and unstable limit cycles can be learned accurately. The results for the case $r = 28$ are presented in Fig. 6. The largest Lyapunov indices of numerical trajectories (run for sufficient long times) are estimated using a method proposed in [63]. They are all positive, which suggests that the learned ODE system indeed has chaotic solutions.

Finally, we compare OnsagerNet with MLP-ODEN for learning the Lorenz system. The SymODEN method [17] cannot be applied since the Lorenz system is non-Hamiltonian. The OnsagerNet used here has one shared hidden layer with 20 hidden nodes, and the total number of trainable parameters is 356. The MLP-ODENs have two hidden layers with 16 hidden nodes in each layer, with a total 387 tunable parameters. In Fig. 7 we show the accuracy on the test data set for OnsagerNet and MLP-ODEN using ReQU and ReQUr as activation functions, from which we see that OnsagerNet performs much better. To ensure that these results hold across different model configurations, we also tested learning the Lorenz system with larger OnsagerNets and MLP-ODENs with three hidden layers with 100 hidden nodes each. Some typical training and testing loss curves are given in Fig. 8, from which we see that OnsagerNet performs better than MLP-ODEN and the activation function ReQUr performs better than tanh as the activation function.

C. Application to Rayleigh-Bénard convection with fixed parameters

We now discuss the main application of OnsagerNet in this paper, namely, learning reduced-order models for the two-dimensional Rayleigh-Bénard convection (RBC) problem, described by

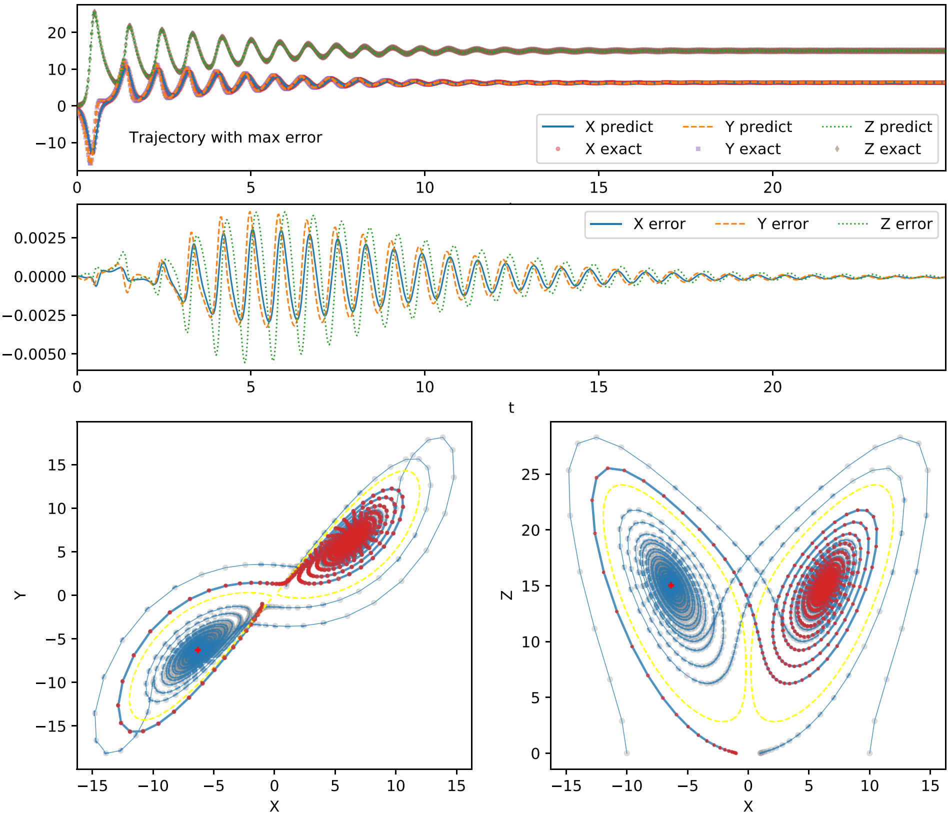


FIG. 5. Results of the learned ODE model by ReQUr OnsagerNet for the Lorenz system (19)–(21) for $b = \frac{8}{3}$, $\sigma = 10$, and $r = 16$. In the bottom plots, the dots are trajectory snapshots generated from exact dynamics, the solid curves are trajectories generated from learned dynamics, and the thick curve with red dots is the one with the largest numerical error. The small red crosses are the fixed points numerically calculated for the learned ODE system. The yellow closed curves are the unstable limit cycles calculated from the learned ODE system.

the coupled partial differential equations

$$\partial_t \vec{u} + (\vec{u}\cdot\nabla)\vec{u} = \nu\Delta\vec{u} - \nabla p + \alpha_0 g_0 \hat{y}\theta, \quad \nabla\cdot\vec{u} = 0, \tag{22}$$

$$\partial_t \theta + \vec{u}\cdot\nabla\theta = \kappa\Delta\theta + \Gamma v, \tag{23}$$

where $\vec{u} = (u, u, 0)$ is the velocity field, g_0 is the gravitational acceleration, $\hat{y}$ is the unit vector opposite to gravity, θ is the departure of the temperature from the linear temperature profile $\bar{\theta}(y) = \bar{\theta}_H - \Gamma y$, and $\Gamma = (\bar{\theta}_H - \bar{\theta}_L)/d$, with d the depth of the channel between two plates. The constants α_0, ν, and κ denote the coefficient of thermal expansion, the kinematic viscosity, and the thermal conductivity, respectively. The system is assumed to be homogeneous in the z direction, and periodic in the x direction, with period L_x. The dynamics depends on three nondimensional parameters: the Prandtl number $\mathrm{Pr} = \nu/\kappa$, the Rayleigh number $\mathrm{Ra} = d^4 \frac{g_0\alpha_0\Gamma}{\nu\kappa}$, and aspect ratio $a = 2d/L_x$. The critical Rayleigh number to maintain the stability of zero solution is $R_c = \pi^4(1+a^2)^3/a^2$. The terms $\alpha_0 g_0 \hat{y}\theta$ in (22) and Γv in (23) are external forcing terms. For given divergence-free velocity $\vec{u}$ with no-flux or periodic boundary conditions, the convection operator $\vec{u}\cdot\nabla$ is skew symmetric. Thus, the RBC equations satisfy the generalized Onsager principle with potential $\frac{1}{2}\|u\|^2 + \frac{1}{2}\|\theta\|^2$. Therefore, the OnsagerNet approach, which requires that the reduced system also satisfies such

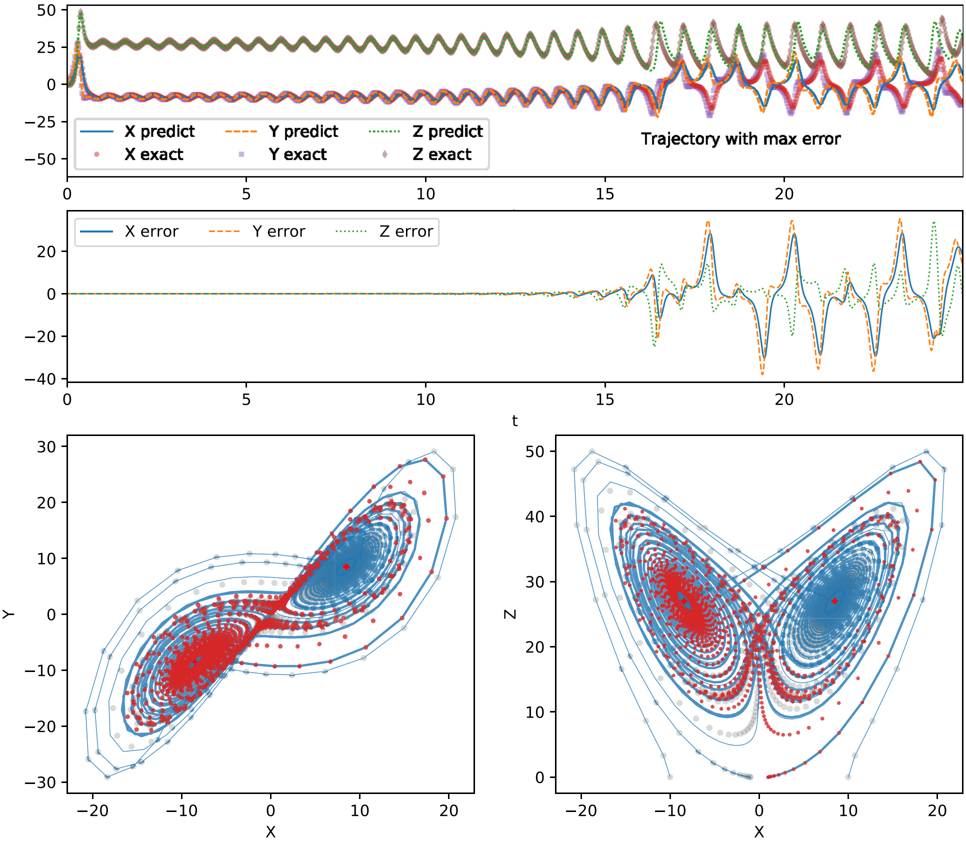


FIG. 6. Results of the learned ReQU OnsagerNet model for the Lorenz system (19)–(21) for $b = \frac{8}{3}$, $\sigma = 10$, and $r = 28$. In the bottom plots, the dots are trajectory snapshots generated from exact dynamics and the solid curves are trajectories generated from learned dynamics. The thick curve with red dots is the one with the largest numerical error. The two red crosses are the (unstable) fixed points numerically calculated for the learned ODE system.

a principle, is appropriate in this case. The velocity $\vec{u}$ can be represented by a stream function $\phi(x, y)$:

$$\vec{u} = (-\partial_y\phi, \partial_x\phi, 0). \tag{24}$$

By eliminating pressure, one gets the following equations for ϕ and θ:

$$\partial_t\Delta\phi - \partial_y\phi\partial_x(\Delta\phi) + \partial_x\phi\partial_y(\Delta\phi) = \nu\Delta^2\phi + g_0\alpha_0\partial_x\theta, \tag{25}$$

$$\partial_t\theta - \partial_y\phi\partial_x\theta + \partial_x\phi\partial_y\theta = \kappa\Delta\theta + \Gamma\partial_x\phi. \tag{26}$$

The solutions ϕ and θ to (25) and (26) have representations in Fourier sine series as

$$\theta(x, y) = \sum_{k_1=-\infty}^{\infty}\sum_{k_2=1}^{\infty}\theta_{k_1k_2}e^{2i\pi k_1x/L_x}\sin\left(\frac{\pi k_2 y}{d}\right),$$

$$\phi(x, y) = \sum_{k_1=-\infty}^{\infty}\sum_{k_2=1}^{\infty}\phi_{k_1k_2}e^{2i\pi k_1x/L_x}\sin\left(\frac{\pi k_2 y}{d}\right),$$

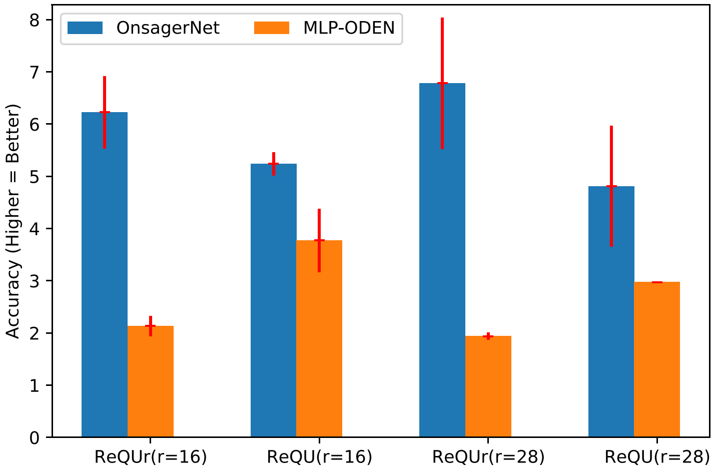


FIG. 7. Accuracy of learned dynamics by using OnsagerNets and MLP-ODEN with ReQU and ReQUr as activation functions for learning the Lorenz system with $r = 16, 28$. The height of the bars stands for $-\log_{10}$ of the testing MSE plus 3.5; the higher the better. The heights of the red crosses on top of the bars indicate standard deviations. The results are averages of trainings using three different random seeds.

where $\theta_{k_1k_2} = \bar{\theta}_{-k_1k_2}$ and $\phi_{k_1k_2} = \bar{\phi}_{-k_1k_2}$ since θ and ϕ are real. In the Lorenz system, only the three most important modes ϕ_{11}, θ_{11}, and θ_{02} are retained. In this case, the solution has the form

$$\phi(x, y, t) = \frac{(1+a^2)\kappa}{a}\sqrt{2}X(t)\sin\left(\frac{2\pi x}{L_x}\right)\sin\left(\frac{\pi y}{d}\right), \tag{27}$$

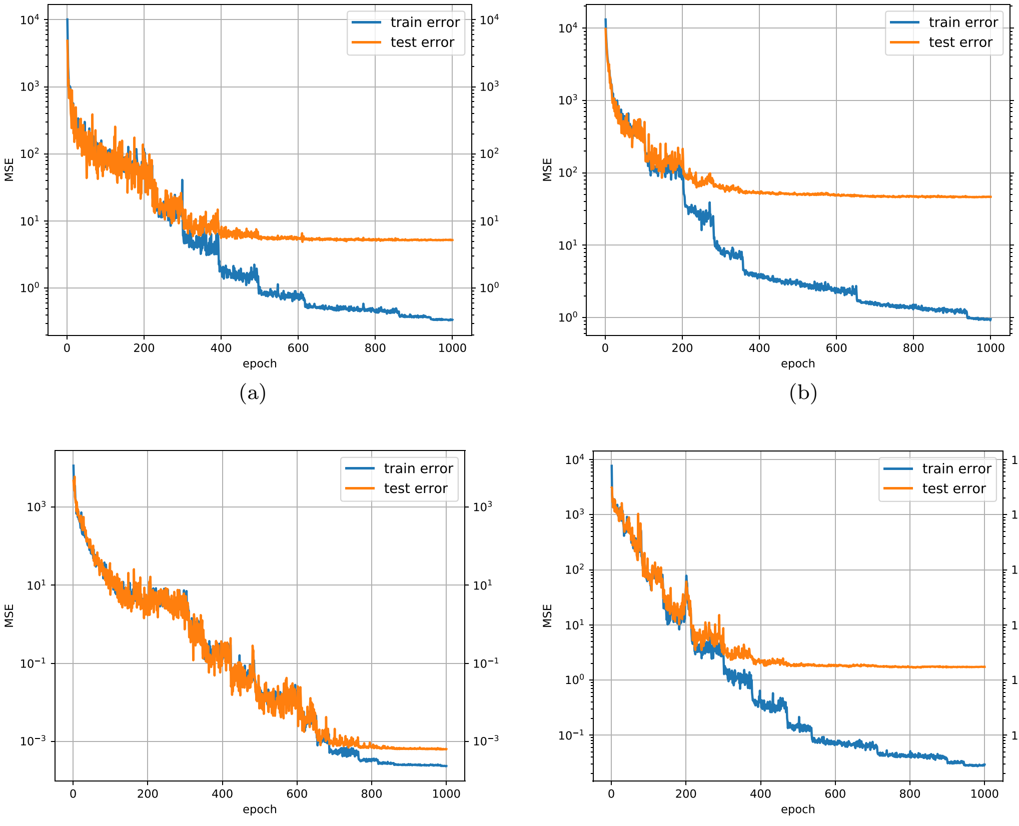
$$\theta(x, y, t) = \frac{R_c\Gamma d}{\pi\,\mathrm{Ra}}\left[\sqrt{2}Y(t)\cos\left(\frac{2\pi x}{L_x}\right)\sin\left(\frac{\pi y}{d}\right) - Z(t)\sin\left(\frac{2\pi y}{d}\right)\right], \tag{28}$$

The Lorenz equations (19)–(21) for the evolution of X, Y, and Z are obtained by a Galerkin approach [39] with time rescaling $\tau = \pi^2\frac{(1+a^2)}{d^2}\kappa t$, the rescaled Rayleigh number $r = \mathrm{Ra}/R_c$, $b = 4/(1+a^2)$, and the Prandtl number $\sigma = \nu/\kappa$.

Since Lorenz system is aggressively truncated from the original RBC problem, it is not expected to give a quantitatively accurate prediction of the dynamics of the original system when $r \gg 1$. Some extensions of the Lorenz system to dimensions higher than 3 have been proposed, e.g., Curry's 14-dimensional model [64], but numerical experiments have shown that much large numbers of spectral coefficients need to be retained to get quantitatively good results in a Fourier-Galerkin approach [40]. In fact, [40] used more than 100 modes to obtain convergent results for a parameter region similar to $b = \frac{8}{3}$, $\sigma = 10$, and $r = 28$ used by Lorenz. In the following, we show that by using OnsagerNet, we are able to directly learn reduced-order models from RBC solution data that are quantitatively accurate and require many fewer modes than the Galerkin projection method.

1. Data generation and equation parameters

We use a Legendre-Fourier spectral method to solve the Rayleigh-Bénard convection problem (22) and (23) based on the stream function formulation (25) and (26) to generate sample data. To be consistent with the case considered by Lorenz, we use the free boundary condition for velocity. A RBC problem is chosen with the following parameters: $\mathrm{Re} = 10$, $L_x = 4\sqrt{2}$, $\alpha_0 = 0.1$, $g = 9.8$, $\kappa = 0.01$, and $\Gamma = 1.174\,13$. The corresponding Prandtl number and Rayleigh number are $\sigma = 10$ and $\mathrm{Ra} = 18\,410.3$, respectively. The relative Rayleigh number is $r = \mathrm{Ra}/\mathrm{R_c} = 28$. In this section we use OnsagerNet to learn one dynamical model for each fixed r value. The results of learning one parametric dynamical model for a wide range of r value are given in next subsection. Initial



(a)

(b)

(c)

(d)

FIG. 8. Typical training MSE curves to learn the Lorenz system (19)–(21) ($r = 28$) with overparametrized neural ODE networks: (a) MLP-ODEN with ReQUr, (b) MLP-ODEN with tanh, (c) OnsagerNet with ReQUr, and (d) OnsagerNet with tanh.

conditions of Lorenz form (27) and (28) are used, where X, Y, and Z are sampled from a Gaussian distribution and rescaled such that $X^2 + Y^2 + Z^2 \leqslant R_B$, with R_B a constant.

The semidiscretized Legendre-Fourier system is solved using a second-order implicit-explicit time marching scheme with time step size $\tau = 0.001$ for 100 time units. Here 128 real Fourier modes are used for the x direction and 49 Legendre modes for the y direction. To generate training and testing data, we simulated 100 initial values. The solver outputs two snapshots of (u, v, θ) at $(t, t + \tau)$ for each integer time t. The data from the first 80 trajectories are used as training data, while the last 20 trajectories are used as testing data.

To have an estimate of the effective dimension, we first apply PCA to the data. The result for $r = 28$ is shown in Fig. 9(a). We observe that 99.7% variance is captured by the first nine principal components, so we seek a reduced model of comparable dimensions.

2. *Structure of OnsagerNet*

The network parameters are similar to those in learning the Lorenz ODE system. Since the Rayleigh-Bénard convection system is not a closed system, we ensure the stability established in Theorem 1 by taking $\alpha = 0.1$ and $\beta = 0.1$. To make $f(h)$ Lipschitz continuous, we simply let $f(h)$ be an affine function of h. We use two common hidden layers (i.e., $l = 2$), with each layer having $n_1 = 2C^m_{m+2}$ hidden nodes for evaluating $L(h)$, $R(h)$, and $U_i(h)$ in OnsagerNet. We use ReQUr as

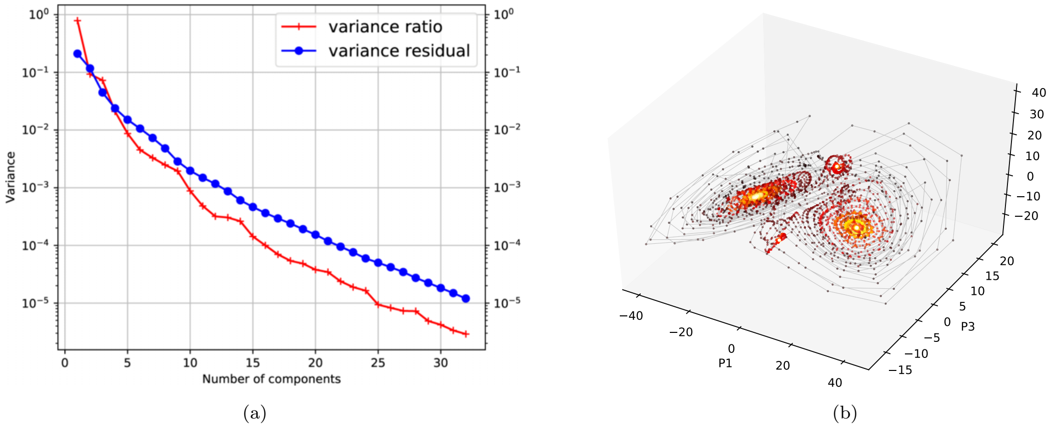


FIG. 9. (a) Relative variance of the first 32 principal components and (b) the sample trajectories for the Rayleigh-Bénard convection problem with $r = 28$ projected on the first three components. The trajectories are generated using Lorenz-type initial values with random amplitude.

the activation function in this application as it gives slightly better training results than ReQU. The ReQUr OnsagerNet is trained using the standard Adam optimizer [54] to minimize the loss with one embedded RK2 step. The embedding of multiple RK2 steps improves the accuracy only slightly since the time interval of the data set $\tau = 0.001$ is small and so the error of the RK2 integrator is not the major error in the total loss. The use of multiple hidden layers in OnsagerNet also improves accuracy. Since our purpose is to find small models for the dynamics, we only present numerical results of OnsagerNets with two common hidden layers.

3. *Numerical results for r* = 28

We now present numerical results of the RBC problem with $b = \frac{8}{3}$, $\sigma = 10$, and $r = 28$, a parameter set that when used in the Lorenz approximation leads to chaotic solutions but for the original RBC problems has only fixed points as attractors. We perform PCA for the sampled data and train OnsagerNets using three, five, and seven principal components. The OnsagerNets are trained with batch size 200 and initial learning rate 0.0064 and are reduced by half if the loss is not decreased in 25 epochs. After the reduced ODE models are learned, we simulate the learned equations using a third-order Runge-Kutta method [58] for one time unit with initial conditions taken from the PCA data at t_j, $j = 0, \dots 99$, and compare the results to the original PCA data at t_{j+1}. To show that the learned ODE system is stable, we also simulate the learned ODE models for 99 time units. We summarize the results in Table I. For comparison, we also present the results of MLP-ODEN. We see that OnsagerNets have better long-time stability than MLP-ODEN. The huge $t = 99$ prediction error in the MLP-ODEN PCA $m = 5$ model indicates that the MLP-ODEN model learned is not stable. From Table I we also observe that the OnsagerNets using only three variables ($m = 3$) can give a good prediction for the period of one time unit ($E^{\text{pred,rel}}_{t=1}$) but have a large relative L^2 error ($E^{\text{pred,rel}}_{t=99}$) for long times ($t = 99$). By increasing m gradually to 5 and 7, both the short-time and the long-time prediction accuracy increase.

Some detailed training and testing losses are given in Fig. 10. The gap between training error and testing error indicates that the training data have non-negligible sampling error. However, from the curve of the testing error shown in Fig 10(a), we see the model is not overfitting. We also observe that MLP-ODEN has a much larger training-testing error gap than OnsagerNet.

To clearly visualize the numerical errors with respect to time, we show numerical results of using learned OnsagerNets to simulate one selected trajectory in the training set and one in the testing set

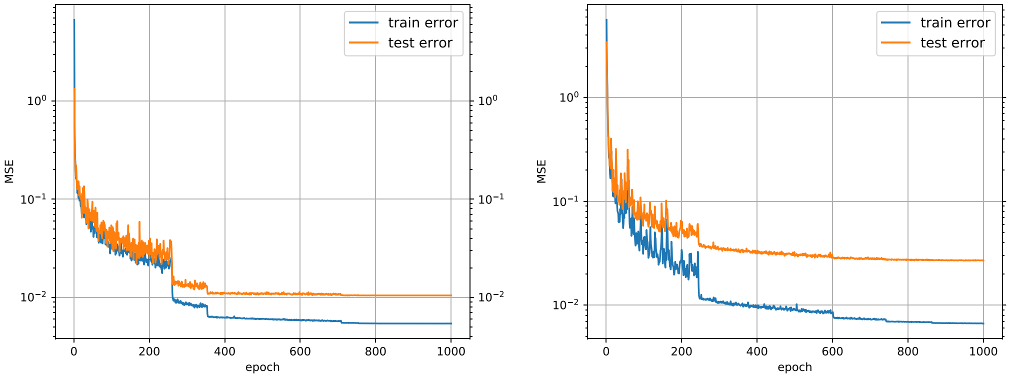
TABLE I. Numerical results of learning the reduced hidden dynamics for the RBC problem ($r = 28$). In the last column N_{fail} represents the average (over three random seeds) number of trajectories (out of 100) in the learned ODE systems that do not converge to the correct final steady states.

Method and dimension	No. of parameters	$\text{MSE}_{\text{train}}$	MSE_{test}	$E_{t=1}^{\text{pred,rel}}$	$E_{t=99}^{\text{pred, rel}}$	N_{fail}
MLP-ODEN PCA 3	983	2.63×10^{-1}	3.37×10^{-1}	2.32×10^{-2}	3.51×10^{-1}	62/3
MLP-ODEN PCA 5	4079	2.95×10^{-2}	7.84×10^{-2}	8.18×10^{-3}	4.12×10^{4}	16/3
MLP-ODEN PCA 7	11599	6.60×10^{-3}	2.68×10^{-2}	3.71×10^{-3}	4.79×10^{-2}	7/3
OnsagerNet PCA 3	776	3.17×10^{-1}	3.85×10^{-1}	2.54×10^{-2}	2.76×10^{-1}	53/3
OnsagerNet PCA 5	3408	3.88×10^{-2}	7.47×10^{-2}	8.40×10^{-3}	6.87×10^{-2}	13/3
OnsagerNet PCA 7	10032	6.71×10^{-3}	1.26×10^{-2}	2.68×10^{-3}	1.07×10^{-2}	2/3
OnsagerNet AE 3	2.5×10^{6} (AE) + 776	4.95×10^{-3}	1.97×10^{-2}	5.34×10^{-3}	9.64×10^{-2}	16/3
OnsagerNet AE 5	2.5×10^{6} (AE) + 3408	3.71×10^{-3}	1.19×10^{-2}	3.35×10^{-3}	4.63×10^{-2}	9/3
OnsagerNet AE 7	2.5×10^{6} (AE) + 10032	1.46×10^{-3}	4.88×10^{-3}	1.48×10^{-3}	1.93×10^{-2}	3/3

with relatively large errors in Fig. 11. We see that as more and more hidden variables are used, the ODE models learned by OnsagerNets are increasingly accurate, even for long times.

The results of training OnsagerNet together with a PCAResNet autoencoder and regularized by the isometric loss defined in Eq. (13) are also presented Table I. The autoencoder we used has two hidden encode layers and two hidden decode layers with ReQUr as activation functions. From Table I we see that the results are improved in this autoencoder plus OnsagerNet approach, especially for models with few hidden variables and short-time predictions.

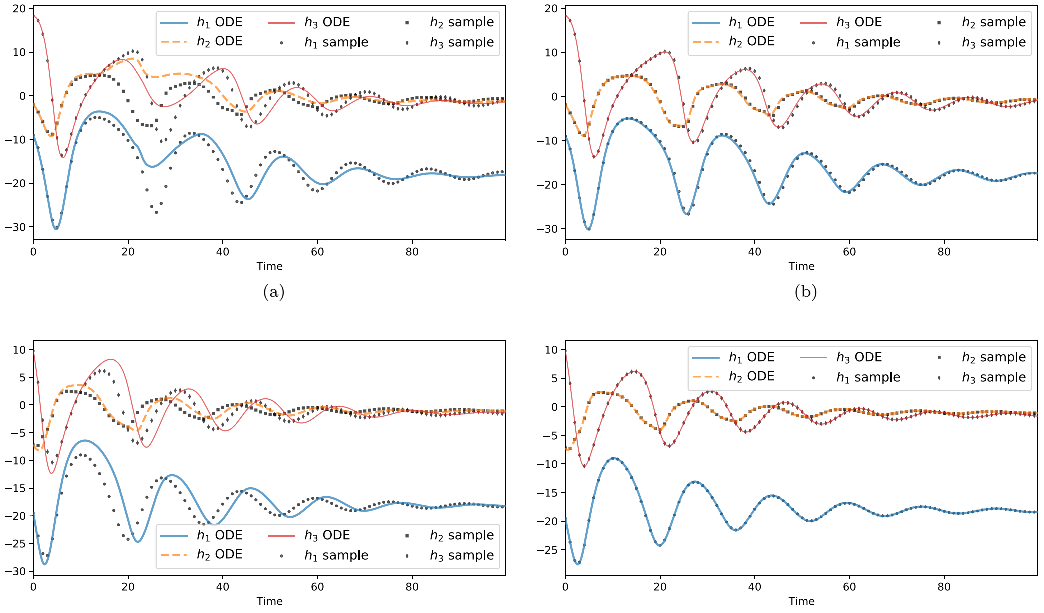
To demonstrate the advantage of PCAResNet for dimensionality reduction, we also carried out additional numerical experiments on using different autoencoders, including a standard stacked autoencoder (SAE), a stacked autoencoder with contractive regularization (CAE), which is a multilayer generalization of the original CAE [38], and the PCAResNet. All three autoencoders use two hidden layers with 128 and 32 nodes, respectively. The activation function for PCAResNet is ReQUr, while the activation function for SAE and CAE is elu. We tested other activation functions including ReLU, softplus, elu, tanh, and sigmoid for CAE and SAE and we found that elu produced the best training results. When training SAE and CAE with OnsagerNet, we first pretrain SAE and CAE, then train OnsagerNet with low-dimensional trajectories produced by pretrained SAE and CAE, and then train the autoencoder and OnsagerNet together in the final step. Note



(a)

(b)

FIG. 10. Training and testing loss of OnsagerNet and MLP-ODEN for learning the RBC problem with $r = 28$: (a) OnsagerNet + PCA with $m = 7$ and (b) MLP-ODEN + PCA with $m = 7$.



(a)

(b)

(c)

(d)

FIG. 11. First three principal components of one trajectory (trajectory 4) in the training set and one (trajectory 89) in the testing set for the RBC problem with $r = 28$ and the corresponding simulation results of learned reduced models by OnsagerNets with $m = 3$ and 7, respectively: (a) trajectory 4 with $m = 3$, (b) trajectory 4 with $m = 7$, (c) trajectory 89 with $m = 3$, and (d) trajectory 89 with $m = 7$.

that PCAResNet does not require a pretraining step, since it is initialized by a simple PCA. The performance of the three different autoencoders is reported in Table II. The β_{isom} and α_{isom} in Eq. (13) are related to the parameters $\beta^{\star}_{\text{isom}}$ and $\alpha^{\star}_{\text{isom}}$ in Table II by

$$\beta_{\text{isom}} = \beta^{\star}_{\text{isom}} \times \frac{\text{pretrained OnsagerNet MSE loss}}{\text{PCA isometric loss}}, \quad \alpha_{\text{isom}} = \alpha^{\star}_{\text{isom}} \times (\text{PCA isometric loss}). \quad (29)$$

TABLE II. Performance of three autoencoders for the RBC problem (with $r = 28$ and three principal components). In the last column N_{fail} is the number of trajectories (out of 100, averaged over five seeds) in the learned ODE system that failed to correct the final steady state.

Autoencoders	$\beta^{\star}_{\text{isom}}$	$\alpha^{\star}_{\text{isom}}$	$\text{MSE}^{\text{ODE}}_{\text{train}}$	$\text{MSE}^{\text{ODE}}_{\text{test}}$	$\frac{\text{MSE}^{\text{ODE}}_{\text{test}}}{\text{MSE}^{\text{ODE}}_{\text{train}}}$	$E^{\text{pred,rel}}_{t=1}$	$E^{\text{pred,rel}}_{t=99}$	N_{fail}
PCAResNet	1	0.0	1.99×10^{-2}	5.25×10^{-2}	2.64	7.53×10^{-3}	3.21×10^{-1}	85/5
PCAResNet	1	0.4	1.27×10^{-2}	3.40×10^{-2}	2.68	6.49×10^{-3}	2.67×10^{-1}	72/5
PCAResNet	1	0.8	4.91×10^{-3}	1.99×10^{-2}	4.05	5.38×10^{-3}	1.01×10^{-1}	27/5
PCAResNet	1	1.2	4.00×10^{-3}	1.70×10^{-2}	4.25	5.14×10^{-3}	$\mathbf{8.64 \times 10^{-2}}$	24/5
PCAResNet	1	1.6	3.78×10^{-3}	1.39×10^{-2}	3.68	5.41×10^{-3}	1.31×10^{-1}	37/5
PCAResNet	0		1.07×10^{-4}	3.37×10^{-3}	31.50	2.24×10^{-3}	9.17×10^{-2}	49/5
SAE	1	0.8	6.64×10^{-3}	3.62×10^{-2}	5.45	4.13×10^{-3}	2.39×10^{-1}	92/5
SAE	0		3.30×10^{-4}	2.26×10^{-2}	68.49	6.60×10^{-3}	1.86×10^{-1}	96/5
CAE	1	0.8	5.58×10^{-3}	2.80×10^{-2}	4.09	3.99×10^{-3}	1.62×10^{-1}	67/5
CAE	0		3.45×10^{-4}	6.41×10^{-2}	185.80	1.37×10^{-2}	3.26×10^{-1}	135/5

TABLE III. PCA trajectory prediction error on the testing set for the RBC problem ($r = 28$) using OnsagerNet and the nearest neighbor method.

Method	Dimension	$E_{t=1}^{\text{pred,rel}}$	$E_{t=10}^{\text{pred,rel}}$	$E_{t=20}^{\text{pred,rel}}$	$E_{t=99}^{\text{pred,rel}}$
OnsagerNet	$m = 3$	2.65×10^{-2}	8.47×10^{-2}	1.30×10^{-1}	2.63×10^{-1}
OnsagerNet	$m = 5$	9.29×10^{-3}	3.73×10^{-2}	6.91×10^{-2}	8.46×10^{-2}
OnsagerNet	$m = 7$	3.24×10^{-3}	1.28×10^{-2}	1.73×10^{-2}	3.23×10^{-3}
NN	$m = 3$	9.76×10^{-2}	1.40×10^{-1}	1.93×10^{-1}	1.02×10^{-2}
NN	$m = 5$	1.09×10^{-1}	1.676×10^{-1}	2.10×10^{-1}	1.75×10^{-1}
NN	$m = 7$	1.13×10^{-1}	1.74×10^{-1}	2.20×10^{-1}	2.69×10^{-1}

From this table we observe that PCAResNet produce better long-time results than SAE and CAE both with and without isometric regularization. From the numerical results presented in Table II, we clearly see the benefits of isometric regularization: It reduces overfitting (the ratio between testing MSE and training MSE is much smaller when isometric regularization is used) and improves the long-time performance for all three tested autoencoders. Note that $\beta_{\text{isom}}^{\star} = 1$ and $\alpha_{\text{isom}}^{\star} = 0.8$ are used to produce the autoencoder results presented in Table I.

We also carry out a comparison of the PCA trajectory prediction error between the OnsagerNet, long short-term memory (LSTM) recurrent neural network, and nearest-neighbor (NN) method. Since LSTM is a discrete-time process, it cannot learn an underlying ODE model. Rather, we focus on predictions over fixed time intervals ($\Delta t = 0.01$). On the other hand, given a test initial condition, the NN method simply selects the closest initial condition from the training set and use its associated trajectory as a prediction. The numerical results are given in Tables III and IV. We see that OnsagerNet outperforms the NN method for all cases except for $m = 3$ and prediction time $t = 99$. By using more PCs, OnsagerNet gives better results, but the NN method does not. From Table IV we also observe that LSTM does not give better results by using more PCs. We note that both LSTM and NN methods can only serve as predictive tools for trajectories but do not give rise to a reduced dynamical model in the form of differential equations.

To get an overview of the vector field that drives the learned ODE system represented by OnsagerNet, we draw two-dimensional projections of phase portraits following several representative trajectories in Fig. 12, from which we see that there are four stable fixed points, two which have relatively larger attraction basins and two which have very small attraction basins. We numerically verified they are all linearly stable by checking the eigenvalues of Jacobian matrices of the learned system at those points. The two fixed points with a larger attraction basin are very similar to those appearing in the Lorenz system with $r < 24.06$, which corresponds to the two fixed points resulting from the first pitchfork bifurcation (see, e.g., q_+, q_- in Fig. 2.1 in [62] and Fig. 5 herein).

We also test the learned dynamics using OnsagerNet by simulating 100 new numerical trajectories with random initial conditions in the phase space. We observe that they all have negative

TABLE IV. PCA trajectory prediction MSE for the RBC problem with $r = 28$ using LSTM. Since LSTM requires fixing the time step (resolution) and specifying sequence lengths for training, different models have to be trained for different predictive horizons or resolutions for best performance. The architecture of the LSTM is chosen so that for each PCA embedding dimension (Dimension), its number of trainable parameters is similar to that of the OnsagerNet.

Dimension	No. of parameters	$E_{t=1}^{\text{pred,rel}}$	$E_{t=99}^{\text{pred,rel}}$
$m = 3$	1131	4.15×10^{-1}	2.09×10^{-1}
$m = 5$	4245	1.23×10^{-1}	3.82×10^{-1}
$m = 7$	15653	1.02×10^{-1}	3.73×10^{-1}

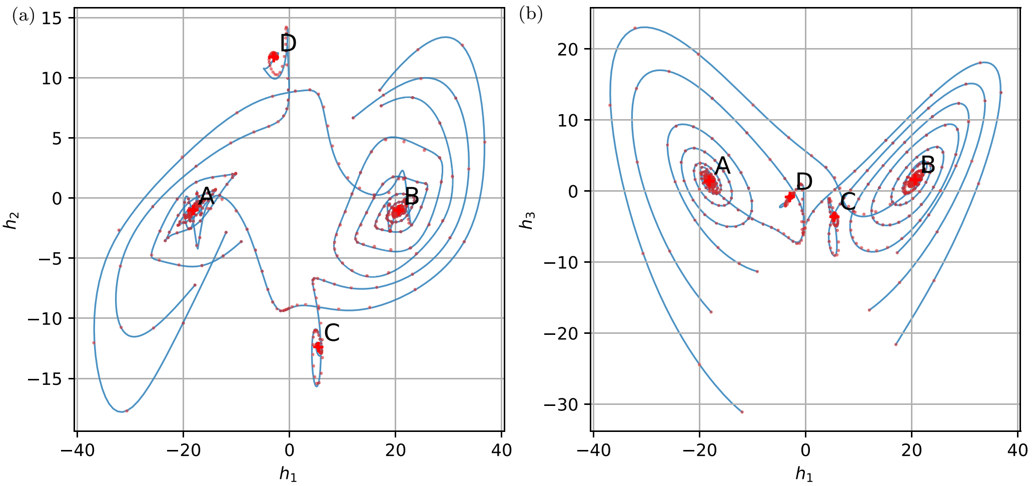


FIG. 12. Six representative trajectories from the sample data (red dots) and the corresponding ODE solutions (solid blue curves) evolved from the same initial values using the learned ODE system for the RBC problem with $r = 28$ and $m = 7$ for (a) projection to the (h_1, h_2) plane and (b) projection to the (h_1, h_3) plane. The four stable critical points (red crosses) of the learned ODE system are numerically calculated and plotted.

Lyapunov exponents, which suggests that the learned dynamics has no chaotic solutions. This result is consistent with the original RBC system, which also does not exhibit chaos at these parameter values. In this sense, OnsagerNet also preserves the qualitative properties of the full dynamics, despite being embedded in a much-lower-dimensional space.

The learned free-energy function. In Fig. 13 we show the isosurfaces of the learned free-energy function in the reduced OnsagerNet model for the RBC problem. When PCA data are used, with three hidden variables, the free energy learned is irregular, which is very different from a physical free-energy function one expects for a smooth dynamical system. As the number of hidden variables is increased, the isosurfaces of the learned free-energy function become more and more ellipsoidal, which means the free energy is more akin to a positive-definite quadratic form. This is consistent with the exact PDE model. The use of AE in the low-dimensional case also helps in learning a better free-energy function. This again highlights the preservation of physical principles in the OnsagerNet approach.

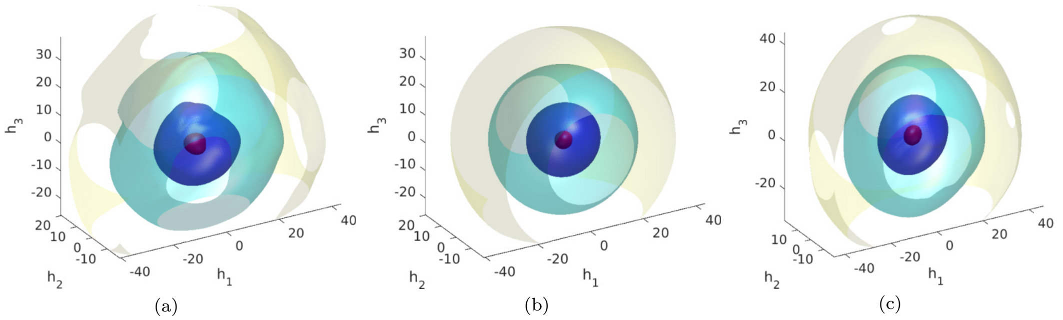


FIG. 13. Learned free energy by OnsagerNet with PCA and an autoencoder for the RBC problem with $r = 28$. The dependence of the energy function on the first three principal components are shown: (a) PCA with $m = 3$, (b) PCA with $m = 7$, and (c) autoencoder + OnsagerNet with $m = 3$.

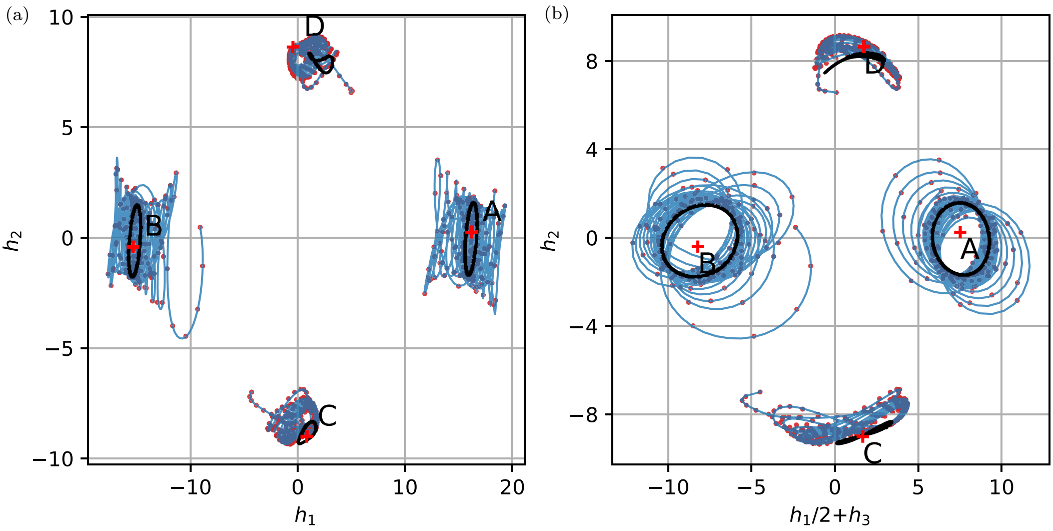


FIG. 14. Four representative trajectories from the sample data (red dots) and the corresponding ODE solutions (solid blue curves) evolved from the initial values using the learned ODE system ($m = 11$) for the case $r = 84$ for (a) projection to the (h_1, h_2) plane and (b) projection to the $(h_1/2 + h_3, h_2)$ plane. The four unstable critical points (red crosses) and four stable limit cycles (black curves) of the learned ODE system are also plotted.

4. Higher-Rayleigh-number case $r = 84$

For $r = 28$, the RBC problem has only fixed points as attractors. To check the robustness of OnsagerNet to approximate dynamics with different properties, we present the results of a higher-Rayleigh-number case with $r = 84$. The corresponding Rayleigh number is 55 230.95. In this case, the RBC problem has four stable limit cycles. However, starting from initial conditions considered by Lorenz's reduction, the solution needs to evolve for a very long time to get close to the limit cycles. Thus, whether or not the limit sets are limit cycles is not very clear based on only observations of the sample data. An overview of the learned dynamics for $m = 11$ is shown in Fig. 14, where four representative trajectories close to the limit cycles together with critical points (saddles) and limit cycles calculated from the learned model are plotted. As before, from this figure we see that limit cycles are accurately predicted by the reduced-order model learned using OnsagerNet. Meanwhile, the saddles can be calculated from the learned OnsagerNet.

A typical trajectory in the test set and corresponding results by the learned dynamics are plotted in Fig. 15. These results support the observation that OnsagerNet both achieves quantitative trajectory accuracy and faithfully captures the asymptotic behaviors of the original high-dimensional system.

To check whether the learned reduced-order model has chaotic solutions, we carried out a numerical estimate of the largest Lyapunov indices of the trajectories in very-long-time simulations. Here we did not find any trajectory with a positive Lyapunov index, which suggests that for the parameter setting used, the RBC problem has no chaotic solution, at least for the type of initial conditions considered in this paper.

D. Application to Rayleigh-Bénard convection in a wide range of Rayleigh number

In this section we build a reduced model that operates over a wide range of Rayleigh numbers, and hence we sample trajectories corresponding to ten different r values $\{8, 16, 22, 28, 42, 56, 70, 76, 80, 84\}$ in the range $[8, 84]$. The parameters ν, κ, and Γ are fixed and we vary α_0 to obtain different r values. In all cases, the Prandtl number is fixed at $\mathrm{Pr} = \frac{10}{3}$. The data

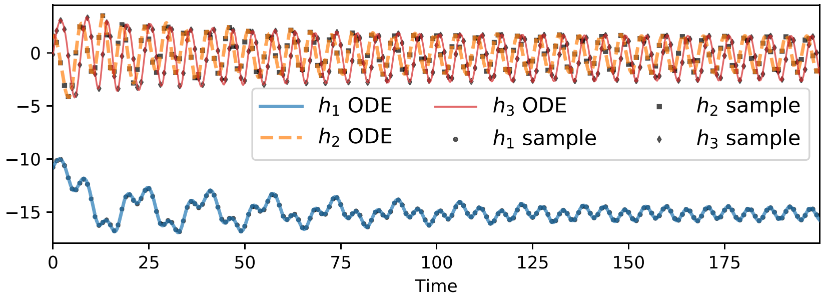


FIG. 15. First three principal components of an exact trajectory and the corresponding simulation results of learned reduced models by OnsagerNets trained using 11 principal components for the RBC problem with $r = 84$.

generating procedure is the same for the fixed r case, but since for the multiple r case the dimension of the fast manifold is relatively high, we do not use the first 39 pairs of solution snapshots in the trajectories.

The first step is to construct the low-dimensional generalized coordinates using PCA. To account for varying Rayleigh numbers, we perform a common PCA on trajectories with different r values. Consequently, we seek an OnsagerNet parametrization with quantities V, $\tilde{M}$, and $\tilde{W}$ independent of the Rayleigh number r. Examining (22) and (23), the r dependence may be regarded as the strength of the external force, which is linear. Thus, we seek the external force f of OnsagerNet as an affine function of r; however, different from the fixed r case, here we assume that f is a nonlinear function of h.

1. Quantitative trajectory accuracy

We first show that despite being low dimensional, the learned OnsagerNet preserves trajectorywise accuracy when compared with the full RBC equations. We summarize the MSE between trajectories generated from the learned OnsagerNet and the true RBC equations for different times in Table V. Observe that the model using only seven coordinates ($m = 7$) gives a good short-time ($t = 1$) prediction but has a relatively large error for the long-time ($t = 60$) prediction. By increasing m to 9 and 11, the accuracy of both the short-time and long-time predictions increases. We also tested generalization in terms of Rayleigh numbers by withholding data for $r = 28, 84$ and testing the learned models in these regimes (Fig. 16). In all cases, the OnsagerNet dynamics remains a quantitatively accurate reduction of the RBC system.

2. Qualitative reproduction of phase-space structure

Next we show that besides quantitative trajectory accuracy, the qualitative aspects of the RBC, such as long-time stability and the nature of the attractor sets, are also adequately captured. This highlights the fact that the approximation power of OnsagerNet does not come at a cost of physical stability and relevance.

TABLE V. Accuracy of learned models for the RBC problem $r \in [8, 84]$.

Dimension	$\text{MSE}_{\text{train}}$	MSE_{test}	$E_{t=1}^{\text{pred,rel}}$	$E_{t=60}^{\text{pred,rel}}$
$m = 7$	1.55×10^{-2}	3.43×10^{-2}	7.07×10^{-4}	6.04×10^{-2}
$m = 9$	2.63×10^{-3}	4.36×10^{-3}	6.84×10^{-4}	1.05×10^{-2}
$m = 11$	2.14×10^{-3}	4.12×10^{-3}	5.12×10^{-4}	8.45×10^{-3}

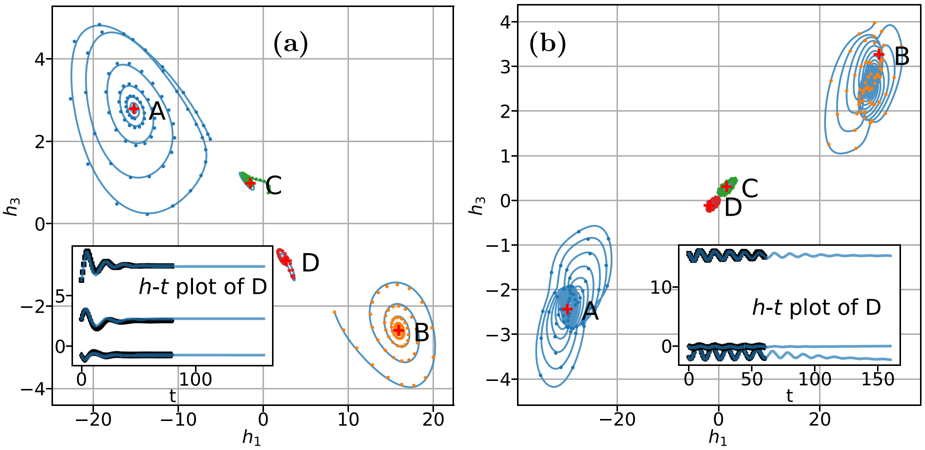


FIG. 16. Some representative trajectories from the sample data (colored dots) and the corresponding solutions of learned 11-dimensional OnsagerNet ODEs (solid blue curves) from the same initial values for the RBC problem with (a) $r = 28$ and (b) $r = 84$. The red crosses are fixed points calculated from the learned ODE systems.

To get an overview of the vector field that drives the learned ODE system, we draw two-dimensional projections of phase portraits for several representative trajectories at $r = 28, 84$ in Fig. 16. The data for these Rayleigh numbers are not included in the training set. For the $r = 28$ case, we observe four stable fixed points. The two fixed points with larger attraction basins are similar to those appearing in the Lorenz system with $r < 24.06$, which corresponds to the two fixed points resulting from the first pitchfork bifurcation (see, e.g., q_+, q_- in Fig. 2.1 in [62]). Note that the Lorenz model with $r = 28$ is chaotic, but the original RBC problem and our learned model have only fixed points as attractors. For $r = 84$, the four attractors in the RBC problem become more complicated. Starting from Lorenz-type initial conditions, the solutions need to evolve for a very long time to get close to these attractors. The results in Fig. 16 show that the learned low-dimensional models can accurately capture those complicated behaviors. Due to the fact that the A, B attractors have larger attraction regions than C, D, we have more A, B-type trajectories than C, D-type ones. Thus, the vector field near fixed points A, B is learned with higher quantitative accuracy (see Fig. 17), while the accuracy of the C, D-type trajectory holds only for short times. Nevertheless, in all cases, the asymptotic qualitative behaviors of the trajectories and the attractor sets are faithfully captured.

As determined earlier, each term in the OnsagerNet parametrization has a clear physical origin, and hence once we learn an accurate model, we can study the behavior of each term to infer some

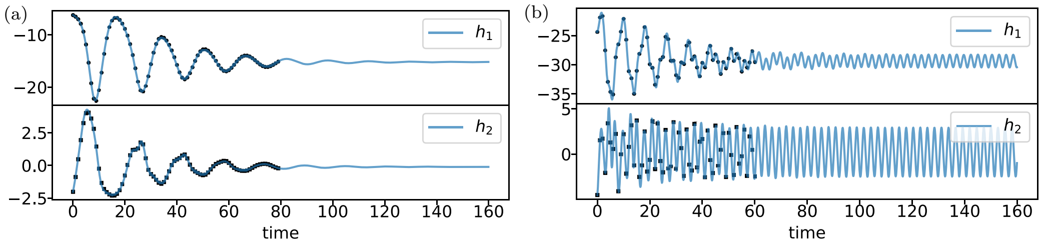


FIG. 17. Long time comparison of trajectories originating from an initial condition converging to A for (a) $r = 28$ and (b) $r = 84$. Black dots are RBC data and colored curves are predictions by the learned OnsagerNet.

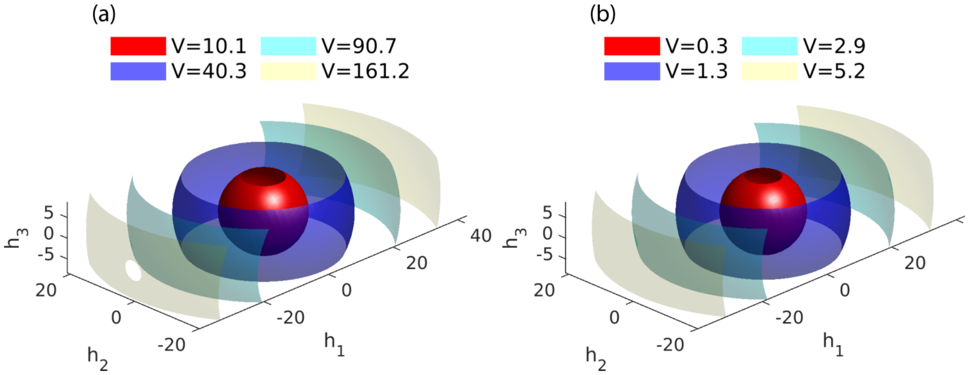


FIG. 18. (a) Learned potential (with $m = 11$) and (b) underlying potential by OnsagerNet for the RBC problem. The isosurfaces in the first three principal component dimensions are shown.

physical characteristics of the reduced dynamics. In Fig. 18 we plot the free energy V in the learned OnsagerNet model and compare it to the RBC energy function $\frac{1}{2}\|u\|^2 + \frac{1}{2}\|\theta\|^2$ projected to the same reduced coordinates. We observe that the shapes are similar with ellipsoidal isosurfaces, but the scales vary. This highlights the preservation of physical structure in our approach.

Besides the learned potential, we can also study the diffusion matrix M, which measures the rate of energy dissipation as a function of the learned coordinates h. In Fig. 19 we plot the eigenvalues of M (which characterize dissipation rates along dynamical modes) along the line from point A to B at $r = 28$, where we observe strong dependence on h. This verifies that the OnsagerNet approach is different from linear Galerkin-type approximations where the diffusive matrix M is constant.

In summary, using OnsagerNet to learn a reduced model of the RBC equations, we gained two main insights. First, it is possible, under the considered conditions, to obtain an ODE model of relatively low dimension (3–11) that can already capture the qualitative properties (energy functions and attractor sets) while maintaining quantitative reconstruction accuracy for a large range of Rayleigh numbers. This is in line with Lorenz's intuition in building his model. Second, unlike

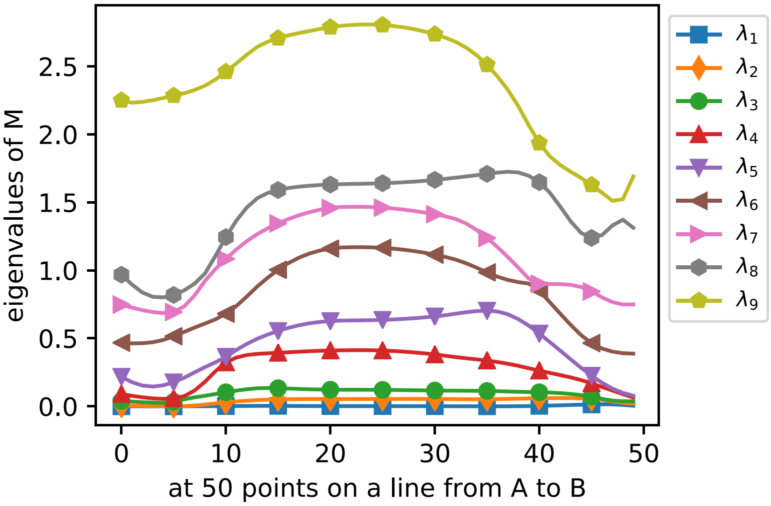


FIG. 19. Eigenvalues of the diffusion matrix $M(h)$ representing local dissipation rates along a line from A to B in the $r = 28$ and $m = 9$ case. We can see a clear deviation from the linear model where M is constant.

Lorenz's highly truncated model, we show that under the parameter settings investigated, the learned OnsagerNet, while having complex dynamics, does not have chaotic behavior.

V. CONCLUSION

We have presented a systematic method to construct (reduced) stable and interpretable ODE systems by a machine learning approach based on a generalized Onsager principle. In comparison to existing methods in the literature, our method has several distinct features.

(i) Compared to the nonstructured machine learning approaches (e.g., [1,3,5]), the dynamics learned by our approach have precise physical structure, which not only ensures the stability of the learned models automatically, but also gives physically interpretable quantities, such as free energy, diffusive, and conservative terms.

(ii) Compared to the existing structured approach, e.g., the Lyapunov function approach [15] and symplectic structure approach [18,19], our method, which relies on a principled generalization of the already general Onsager principle, has a more flexible structure incorporating both conservation and dissipation in one equation, which make it suitable for a large class of problems involving forced dissipative dynamics. In particular, the OnsagerNet structure we impose does not sacrifice short-term trajectory accuracy, but still achieves long-term stability and qualitative agreement with the underlying dynamics.

(iii) Different from the linear multistep method embedded training [5,6] and recurrent neural networks [8,9], we use multiple Runge-Kutta steps embedded in the loss function for training. This can handle large and variable data sampling intervals in a straightforward manner.

(iv) We proposed an isometry-regularized autoencoder to find the slow manifold for given trajectory data for dimension reduction, which is different from traditional feature extraction methods such as proper orthogonal decomposition [65,66], dynamic mode decomposition, and Koopman analysis [50,51,67,68], which only find important *linear* structures. Our method is also different from the use of an autoencoder with sparse identification of nonlinear dynamics proposed in [69] where the sparsity of the dynamics to be learned is used to regularize the autoencoder. The isometric regularization allows the autoencoder to be trained separately or jointly with ODE nets. Moreover, it is usually not easy to ensure long-time stability using these types of methods, especially when the dimension of the learned dynamics increases.

(v) As a model reduction method, different from the closure approximation approach [7] based on the Mori-Zwanzig theory [70], which needs to work with an explicit mathematical form of the underlying dynamical system and usually leads to nonautonomous differential equations, our approach uses only sampled trajectory data to learn autonomous dynamical systems; the learned dynamics can have very good quantitative accuracy and can readily be analyzed further with traditional ODE analysis methods or be used as surrogate models for fast online computation.

We have shown the versatility of OnsagerNet by applying it to identify closed and nonclosed ODE dynamics, i.e., the nonlinear pendulum and the Lorenz system with a chaotic attractor, and learn reduced-order models with high fidelity for the classical meteorological dynamical systems, i.e., the Rayleigh-Bénard convection problem.

While versatile, the proposed method can be further improved or extended in several ways. For example, in the numerical results of the RBC problem, we see that the accuracy is limited by the given sample data when we use enough hidden dimensions. One can use an active sampling strategy (see, e.g., [42,71]) together with OnsagerNet to further improve the accuracy, especially near saddle points. Furthermore, for problems where transient solutions are of interest but lie in a very-high-dimensional space, directly learning PDEs might be a better approach than learning ODE systems. The learning of PDEs using OnsagerNet can be accomplished by incorporating the differential operator filter constraints proposed in [4] into its components, e.g., the diffusive and conservative terms. Finally, for problems where sparsity (e.g., memory efficiency) is more important than accuracy or a balance between these two is needed, one can either add an l^1 regularization on the weights of OnsagerNets into the training loss function or incorporate sparse identification methods in the OnsagerNet approach.

ACKNOWLEDGMENTS

We would like to thank Prof. Chun Liu, Weiguo Gao, Tiejun Li, Qi Wang, and Dr. Han Wang for helpful discussions. The work of H.Y. and X.T. was supported by NNSFC Grants No. 91852116 and No. 11771439 and China Science Challenge Project No. TZ2018001. The work of W.E. was supported in part by a gift from iFlytek to Princeton University. The work of Q.L. was supported by a start-up grant through the NUS PYP program. The computations were partially performed on the LSSC4 PC cluster of State Key Laboratory of Scientific and Engineering Computing.

APPENDIX A: CLASSICAL DYNAMICS AS A GENERALIZED ONSAGER PRINCIPLE

Previously we showed how the h dependence arises from projecting high-dimensional dynamics to the dynamics on reduced coordinates. There it was assumed that the high-dimensional dynamics obeys the generalized Onsager principle with constant diffusive and conservative terms. Here we show how this assumption is sensible, by giving a general classical dynamical system that has such a representation.

Theorem 2. The classical Hamilton system with the Hamilton given by

$$H(x, q) = U(x) + \frac{1}{2m}q^2$$

can be written in the form of Eq. (2).

Proof. The Hamilton equation for given H is

$$\dot{x} = \frac{\partial H}{\partial q} = \frac{q}{m},$$

$$\dot{q} = -\frac{\partial H}{\partial x} = -\nabla_x U.$$

Taking $M = 0$ and $W = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$, we have

$$W\begin{pmatrix} \dot{x} \\ \dot{q} \end{pmatrix} = -\begin{pmatrix} \nabla_x H \\ \nabla_q H \end{pmatrix}. \qquad \blacksquare$$

Theorem 3. The deterministic damped Langevin dynamics

$$m\frac{d^2x}{dt^2} = -\gamma \dot{x} - \nabla_x U(x) \tag{A1}$$

can be written in the form of Eq. (2).

Proof. Defining $v = \dot{x}$, we then have

$$\dot{x} = v,$$

$$m\dot{v} + \gamma\dot{x} = -\nabla_x U(x).$$

Setting $M = \begin{pmatrix} \gamma & 0 \\ 0 & 0 \end{pmatrix}$ and $W = \begin{pmatrix} 0 & m \\ -m & 0 \end{pmatrix}$, we have

$$(M + W)\begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} = -\begin{pmatrix} \nabla_x U(x) \\ mv \end{pmatrix}.$$

This is of the form (2) with a new energy (total energy)

$$V(x, v) = U(x) + \frac{m}{2}v^2. \qquad \blacksquare$$

Theorem 4. The dynamics described by generalized Poisson brackets, defined below, can be written in the form (3). In the Poisson bracket approach, the system is described by generalized coordinates $(q_1, \dots, q_n)$ and generalized momenta $(p_1, \dots, p_n)$. Denoting the Hamiltonian of the system by $H(q_1, \dots, q_n; p_1, \dots, p_n)$, then the dynamics of the system is described by the equation [30]

$$F_t = \{F, H\} - [F, H], \tag{A2}$$

where F is an arbitrary functional depending on the system variables. The reversible and irreversible contributions to the system are represented by the Poisson bracket $\{\cdot, \cdot\}$ and the dissipation bracket $[\cdot, \cdot]$, respectively, which are defined as

$$\{F, H\} = \sum_{i=1}^{n} \frac{\partial F}{\partial q_i}\frac{\partial H}{\partial p_i} - \frac{\partial H}{\partial q_i}\frac{\partial F}{\partial p_i},$$

$$[F, H] = J_F M J_H^T, \quad J_\Phi = \left[\frac{\partial}{\partial q_1}, \dots, \frac{\partial}{\partial q_n}, \frac{\partial}{\partial p_1}, \dots \frac{\partial}{\partial p_n}\right]\Phi, \text{ for } \Phi = (F, H),$$

where M is symmetric positive semidefinite.

Proof. Define $(h_1, \dots, h_m) = (q_1, \dots q_{n,} p_1, \dots, p_n)$, with $m = 2n$. Equation (A2) can be written as

$$F_t = (\nabla_h F)^T \dot{h} = (\nabla_h F)^T \begin{pmatrix} 0 & I_n \\ -I_n & 0 \end{pmatrix} \nabla_h H - (\nabla_h F)^T M \nabla_h H.$$

By taking $F = (h_1, \dots, h_m)$ such that $\nabla_h F = I_m$, we obtain immediately

$$\dot{h} = -W \cdot \nabla_h H - M \cdot \nabla_h H = (M + W)(-\nabla_h H),$$

where $W = \left(\begin{smallmatrix} 0 & -I_n \\ I_n & 0 \end{smallmatrix}\right)$ is an antisymmetric matrix and M is a symmetric positive-semidefinite matrix, which is of the form (3). ■

APPENDIX B: BASIC PROPERTIES OF NONSYMMETRIC POSITIVE-DEFINITE MATRICES

We present some basic properties of real nonsymmetric positive-definite matrices required to show our results.

An $n \times n$ real matrix is called positive definite if there exists a constant $\sigma > 0$ such that $x^T A x \geqslant \sigma x^T x$ for any $x \in \mathbb{R}^n$. If $\sigma = 0$ in the above inequality, we call it positive semidefinite.

Any real matrix A can be written as a sum of a symmetric part $\frac{A+A^T}{2}$ and a skew-symmetric part $\frac{A-A^T}{2}$. For the skew-symmetric part, we have $x^T \frac{A-A^T}{2} x = 0$ for any $x \in \mathbb{R}^n$. So a nonsymmetric matrix is positive (semi)definite if and only if its symmetric part is positive (semi)definite.

The following theorem is used in deriving alternative forms of the generalized Onsager dynamics.

Theorem 5. (a) Suppose A is a positive-semidefinite real matrix; then the real parts of its eigenvalues are all non-negative. (b) The inverse of a nonsingular positive semidefinite is also positive semidefinite.

Proof. Suppose that $\lambda = \eta + i\mu$ is an eigenvalue of A and the corresponding eigenvector is $z = x + iy$. Then

$$Ax + iAy = (\eta x - \mu y) + i(\eta y + \mu x).$$

So

$$\begin{pmatrix} A & 0 \\ 0 & A \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \eta I & 0 \\ 0 & \eta I \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 & -\mu I \\ \mu I & 0 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}.$$

Left multiply the equation by (x^T, y^T) to get (this is the real part of $\bar{z}^T Az = \bar{z}^T \eta z$)

$$x^T Ax + y^T Ay = \eta x^T x + \eta y^T y = \eta(x^T x + y^T y).$$

If A is positive semidefinite, then we obtain $\eta \geqslant 0$, and (a) is proved.

Now suppose that A is positive semidefinite and invertible. Then for any $x \in \mathbb{R}^n$ we can define y by $Ay = x$ to get

$$x^T A^{-1} x = y^T A^T A^{-1} Ay = y^T A^T y = y^T Ay \geqslant 0.$$

Thus (b) is proved. ■

Note that the converse of (a) in Theorem 5 is not true. A simple counterexample is $A = \left(\begin{smallmatrix} 3 & 2 \\ -2 & -1 \end{smallmatrix}\right)$, whose eigenvalues are $\lambda_{1,2} = 1$. However, the eigenvalues of its symmetric part are $\lambda_{1,2} = -1, 3$.

[1] J. Bongard and H. Lipson, Automated reverse engineering of nonlinear dynamical systems, Proc. Natl. Acad. Sci. USA **104**, 9943 (2007).

[2] M. Schmidt and H. Lipson, Distilling free-form natural laws from experimental data, Science **324**, 81 (2009).

[3] S. L. Brunton, J. L. Proctor, and J. N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proc. Natl. Acad. Sci. USA **113**, 3932 (2016).

[4] Z. Long, Y. Lu, X. Ma, and B. Dong, PDE-Net: Learning PDEs from data, in *Proceedings of the 35th International Conference on Machine Learning, Stockholm, 2018*, edited by J. G. Dy and A. Krause (JMLR, Brookline, 2018), pp. 5067–5078.

[5] M. Raissi, P. Perdikaris, and G. E. Karniadakis, Multistep neural networks for data-driven discovery of nonlinear dynamical systems, arXiv:1801.01236.

[6] X. Xie, G. Zhang, and C. G. Webster, Non-intrusive inference reduced order model for fluids using linear multistep neural network, Mathematics **7**, 757 (2019).

[7] Z. Y. Wan, P. R. Vlachas, P. Koumoutsakos, and T. P. Sapsis, Data-assisted reduced-order modeling of extreme events in complex dynamical systems, PLoS One **13**, e0197704 (2018).

[8] S. Pan and K. Duraisamy, Data-driven discovery of closure models, SIAM J. Appl. Dyn. Syst. **17**, 2381 (2018).

[9] Q. Wang, N. Ripamonti, and J. S. Hesthaven, Recurrent neural network closure of parametric POD-Galerkin reduced-order models based on the Mori-Zwanzig formalism, J. Comput. Phys. **410**, 109402 (2020).

[10] C. Ma, J. Wang, and W. E, Model reduction with memory and the machine learning of dynamical systems, Commun. Comput. Phys. **25**, 947 (2019).

[11] C. Xie, K. Li, C. Ma, and J. Wang, Modeling subgrid-scale force and divergence of heat flux of compressible isotropic turbulence by artificial neural network, Phys. Rev. Fluids **4**, 104605 (2019).

[12] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach, Phys. Rev. Lett. **120**, 024102 (2018).

[13] P. R. Vlachas, J. Pathak, B. R. Hunt, T. P. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos, Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics, Neural Netw. **126**, 191 (2020).

[14] T. Arcomano, I. Szunyogh, J. Pathak, A. Wikner, B. R. Hunt, and E. Ott, A machine learning-based global atmospheric forecast model, Geophys. Res. Lett. **47**, e2020GL087776 (2020).

[15] J. Z. Kolter and G. Manek, in *Learning Stable Deep Dynamics Models*, Advances in Neural Information Processing Systems 32, Vancouver, 2019, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran, Red Hook, 2019), pp. 11126–11134.

[16] P. Giesl, B. Hamzi, M. Rasmussen, and K. Webster, Approximation of Lyapunov functions from noisy data, J. Comput. Dyn. **7**, 57 (2020).

[17] Y. D. Zhong, B. Dey, and A. Chakraborty, Symplectic ODE-Net: Learning Hamiltonian Dynamics with Control, in *ICLR Workshop on Integration of Deep Neural Models and Differential Equations*, edited by T. K. Nguyen, R. Baraniuk, A. Garg, S. J. Osher, A. Anandkumar, and B. Wang (ICLR, La Jolla, 2020).

[18] P. Jin, A. Zhu, G. E. Karniadakis, and Y. Tang, Symplectic networks: Intrinsic structure-preserving networks for identifying Hamiltonian systems, Neural Netw. **132**, 166 (2020).

[19] Y. D. Zhong, B. Dey, and A. Chakraborty, *Dissipative SymODEN: Encoding Hamiltonian Dynamics with Dissipation and Control into Deep Learning*, in ICLR Workshop on Integration of Deep Neural Models and Differential Equations, Addis Ababa, 2020 (Ref. [17]).

[20] A. Wikner, J. Pathak, B. Hunt, M. Girvan, T. Arcomano, I. Szunyogh, A. Pomerance, and E. Ott, Combining machine learning with knowledge-based modeling for scalable forecasting and subgrid-scale closure of large, complex, spatiotemporal systems, Chaos **30**, 053111 (2020).

[21] L. Onsager, Reciprocal relations in irreversible processes. I, Phys. Rev. **37**, 405 (1931).

[22] L. Onsager, Reciprocal relations in irreversible processes. II, Phys. Rev. **38**, 2265 (1931).

[23] T. Qian, X.-P. Wang, and P. Sheng, A variational approach to moving contact line hydrodynamics, J. Fluid Mech. **564**, 333 (2006).

[24] M. Doi, Onsager's variational principle in soft matter, J. Phys.: Condens. Matter **23**, 284118 (2011).

[25] X. Yang, J. Li, M. Forest, and Q. Wang, Hydrodynamic theories for flows of active liquid crystals and the generalized Onsager principle, Entropy **18**, 202 (2016).

[26] M.-H. Giga, A. Kirshtein, and C. Liu, in *Handbook of Mathematical Analysis in Mechanics of Viscous Fluids*, edited by Y. Giga and A. Novotny (Springer International, Cham, 2017), pp. 1–41.

[27] W. Jiang, Q. Zhao, T. Qian, D. J. Srolovitz, and W. Bao, Application of Onsager's variational principle to the dynamics of a solid toroidal island on a substrate, Acta Mater. **163**, 154 (2019).

[28] M. Doi, J. Zhou, Y. Di, and X. Xu, Application of the Onsager-Machlup integral in solving dynamic equations in nonequilibrium systems, Phys. Rev. E **99**, 063303 (2019).

[29] X. Xu and T. Qian, Generalized Lorentz reciprocal theorem in complex fluids and in non-isothermal systems, J. Phys.: Condens. Matter **31**, 475101 (2019).

[30] A. N. Beris and B. Edwards, *Thermodynamics of Flowing Systems* (Oxford Science, New York, 1994).

[31] M. Doi and S. F. Edwards, *The Theory of Polymer Dynamics* (Oxford University Press, New York, 1986).

[32] P.-G. de Gennes and J. Prost, *The Physics of Liquid Crystals*, 2nd ed. (Clarendon, Oxford, 1993).

[33] M. Kröger and P. Ilg, Derivation of Frank-Ericksen elastic coefficients for polydomain nematics from mean-field molecular theory for anisotropic particles, J. Chem. Phy. **127**, 034903 (2007).

[34] H. Yu, G. Ji, and P. Zhang, A nonhomogeneous kinetic model of liquid crystal polymers and its thermodynamic closure approximation, Commun. Comput. Phy. **7**, 383 (2010).

[35] W. E, *Principles of Multiscale Modeling* (Cambridge University Press, New York, 2011).

[36] J. Han, Y. Luo, W. Wang, P. Zhang, and Z. Zhang, From microscopic theory to macroscopic theory: A systematic study on modeling for liquid crystals, Arch. Ration. Mech. Anal. **215**, 741 (2015).

[37] G. E. Hinton and R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science **313**, 504 (2006).

[38] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, Contractive Auto-Encoders: Explicit Invariance During Feature Extraction, in *Proceedings of the 28th International Conference on Machine Learning, Bellevue, 2011*, edited by L. Getoor and T. Scheffer (Omnipress, Madison, 2011), pp. 833-840.

[39] E. N. Lorenz, Deterministic nonperiodic flow, J. Atmos. Sci. **20**, 130 (1963).

[40] J. H. Curry, J. R. Herring, J. Loncaric, and S. A. Orszag, Order and disorder in two- and three-dimensional Bénard convection, J. Fluid Mech. **147**, 1 (1984).

[41] S. R. De Groot and P. Mazur, *Non-Equilibrium Thermodynamics* (Dover, New York, 1962).

[42] J. Han, C. Ma, Z. Ma, and W. E, Uniformly accurate machine learning-based hydrodynamic models for kinetic equations, Proc. Natl. Acad. Sci. USA **116**, 21983 (2019).

[43] Lord Rayleigh, F.R.S., On the instability of jets, Proc. Lond. Math. Soc. **s1-10**, 4 (1878).

[44] M. S. Green, Markoff random processes and the statistical mechanics of time-dependent phenomena. II. Irreversible processes in fluids, J. Chem. Phys. **22**, 398 (1954).

[45] R. Kubo, Statistical-mechanical theory of irreversible processes. I. General theory and simple applications to magnetic and conduction problems, J. Phys. Soc. Jpn. **12**, 570 (1957).

[46] D. J. Evans and D. J. Searles, The fluctuation theorem, Adv. Phys. **51**, 1529 (2002).

[47] J. Zhao, X. Yang, Y. Gong, X. Zhao, X. Yang, J. Li, and Q. Wang, A general strategy for numerical approximations of non-equilibrium models—Part I: Thermodynamical systems, Int. J. Numer. Anal. Model. **15**, 884 (2018).

[48] H. C. Ottinger, *Beyond Equilibrium Thermodynamics* (Wiley, New York, 2005).

[49] Q. Hernández, A. Badias, D. Gonzalez, F. Chinesta, and E. Cueto, Structure-preserving neural networks, J. Comput. Phys. **426**, 109950 (2021).

[50] P. J. Schmid, Dynamic mode decomposition of numerical and experimental data, J. Fluid Mech. **656**, 5 (2010).

[51] N. Takeishi, Y. Kawahara, and T. Yairi, Learning Koopman Invariant Subspaces for Dynamic Mode Decomposition, in *Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, 2017*, edited by U. von Luxburg, I. Guyon, S. Bengio, H. Wallach, and R. Fergus (Curran, Red Hook, 2017), pp. 1130–1140.

[52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, in PyTorch: An Imperative Style, High-Performance Deep Learning Library, *Advances in Neural Information Processing Systems 32* (Ref. [15]), pp. 8026–8037.

[53] K. He, X. Zhang, S. Ren, and J. Sun, Deep Residual Learning for Image Recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, Piscataway, 2016), pp. 770–778.

[54] D. P. Kingma and J. Ba, Adam: A Method for Stochastic Optimization, in *3rd International Conference for Learning Representations, San Diego, 2015*, edited by Y. Bengio and Y. LeCun (ICLR, La Jolla, 2015).

[55] S. J. Reddi, S. Kale, and S. Kumar, On the Convergence of Adam and Beyond, *6th International Conference on Learning Representations, Vancouver, 2018* (ICLR, La Jolla, 2018).

[56] A. J. Linot and M. D. Graham, Deep learning to discover and predict dynamics on an inertial manifold, Phys. Rev. E **101**, 062209 (2020).

[57] https://github.com/yuhj1998/OnsagerNet.git.

[58] C.-W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, J. Comput. Phys. **77**, 439 (1988).

[59] B. Li, S. Tang, and H. Yu, Better approximations of high dimensional smooth functions by deep neural networks with rectified power units, Commun. Comput. Phys. **27**, 379 (2020).

[60] C. Sparrow, *The Lorenz Equations: Bifurcations, Chaos, and Strange Attractors* (Springer, New York, 1982).

[61] R. Barrio and S. Serrano, A three-parametric study of the Lorenz model, Physica D **229**, 43 (2007).

[62] X. Zhou and W. E, Study of noise-induced transitions in the Lorenz system using the minimum action method, Commun. Math. Sci. **8**, 341 (2010).

[63] M. T. Rosenstein, J. J. Collins, and C. J. De Luca, A practical method for calculating largest Lyapunov exponents from small data sets, Physica D **65**, 117 (1993).

[64] J. H. Curry, A generalized Lorenz system, Commun. Math. Phys. **60**, 193 (1978).

[65] J. Lumley, *Stochastic Tools in Turbulence* (Academic, New York, 1970).

[66] P. Holmes, J. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry* (Cambridge University Press, Cambridge, 1996).

[67] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson, Spectral analysis of nonlinear flows, J. Fluid Mech. **641**, 115 (2009).

[68] Q. Li, F. Dietrich, E. M. Bollt, and I. G. Kevrekidis, Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator, Chaos **27**, 103111 (2017).

[69] K. Champion, B. Lusch, J. N. Kutz, and S. L. Brunton, Data-driven discovery of coordinates and governing equations, Proc. Natl. Acad. Sci. USA **116**, 22445 (2019).

[70] A. J. Chorin, O. H. Hald, and R. Kupferman, Optimal prediction and the Mori-Zwanzig representation of irreversible processes, Proc. Natl. Acad. Sci. USA **97**, 2968 (2000).

[71] L. Zhang, D.-Y. Lin, H. Wang, R. Car, and Weinan E., Active learning of uniformly accurate interatomic potentials for materials simulation, Phys. Rev. Materials **3**, 023804 (2019).