Reservoir computing model of two-dimensional turbulent convection

Sandeep Pandey [1] and Jörg Schumacher [1,2]
[1]*Institute of Thermodynamics and Fluid Mechanics, Technische Universität Ilmenau, D-98684 Ilmenau, Germany*
[2]*Tandon School of Engineering, New York University, New York, New York 11201, USA*

(Received 29 January 2020; accepted 27 October 2020; published 19 November 2020)

Reservoir computing is an efficient implementation of a recurrent neural network that can describe the evolution of a dynamical system by supervised machine learning without solving the underlying mathematical equations. In this work, reservoir computing is applied to model the large-scale evolution and the resulting low-order turbulence statistics of a two-dimensional turbulent Rayleigh-Bénard convection flow at a Rayleigh number $\mathrm{Ra} = 10^7$ and a Prandtl number $\mathrm{Pr} = 7$ in an extended spatial domain with an aspect ratio of 6. Our data-driven approach, which is based on a long-term direct numerical simulation of the convection flow, comprises a two-step procedure: (1) reduction of the original simulation data by a proper orthogonal decomposition (POD) snapshot analysis and subsequent truncation to the first 150 POD modes which are associated with the largest total energy amplitudes; (2) setup and optimization of a reservoir computing model to describe the dynamical evolution of these 150 degrees of freedom and thus the large-scale evolution of the convection flow. The quality of the prediction of the reservoir computing model is comprehensively tested by a direct comparison of the results of the original direct numerical simulations and the fields that are reconstructed by means of the POD modes. We find a good agreement of the vertical profiles of mean temperature, mean convective heat flux, and root-mean-square temperature fluctuations. In addition, we discuss temperature variance spectra and joint probability density functions of the turbulent vertical velocity component and temperature fluctuation, the latter of which is essential for the turbulent heat transport across the layer. At the core of the model is the reservoir, a very large sparse random network characterized by the spectral radius of the corresponding adjacency matrix and a few further hyperparameters which are varied to investigate the quality of the prediction. Our work demonstrates that the reservoir computing model is capable of modeling the large-scale structure and low-order statistics of turbulent convection, which can open new avenues for modeling mesoscale convection processes in larger circulation models.

DOI: 10.1103/PhysRevFluids.5.113506

I. INTRODUCTION

The application of machine learning (ML) methods, in particular of deep neural networks (DNNs) [1–5], to fluid flows has transformed the way of processing and analyzing large amounts of data. ML methods are used to parametrize unresolved scales in Reynolds stresses and subgrid scale models for complex physical or geometrical flow configurations at high Reynolds numbers which still remain inaccessible to direct numerical simulations or even large eddy simulations [6–11]. They are also used for the detailed segmentation of complex images [12,13]. DNNs converted large-scale patterns in an extended three-dimensional turbulent convection flow [14,15] into a planar dynamical

network where the edges are regions of locally enhanced convective heat flux. Physics-informed and physics-constrained neural networks were developed as a substitute for solving the underlying partial differential equations of the fluid motion [16–18]. All supervised ML algorithms make use of the fact that it is often easier to train a DNN with a number of labeled training data of an intended input-output behavior, than to develop a specific numerical code to provide the correct answer for all possible input data. During the training, information propagates forward through the network while weights and biases are updated at each neuron in each hidden layer of the network backwards from the output to the input layer. This back-and-forth iteration (which is called epoch) has to be repeated multiple times until a minimum of the loss function is obtained. Typically, (stochastic) gradient descent algorithms are used for the minimum search of the loss function [5]. Such a training process requires typically big data records. One further disadvantage of DNNs is their limited capability to process unseen data at very different flow parameters, such as Reynolds or Rayleigh numbers. Possible solutions to overcome this limitation are to stick with comprehensive training sets from different cases or to switch to transfer learning methods [5] which continue the training to adapt a pretrained DNN to the new input data.

Turbulence problems are inherently highly chaotic with stochastic temporal variation of the involved fields. High-dimensional data due to the large grid sizes have to be processed by DNNs to predict statistical properties or the flow evolution; in some cases they fail completely, as for example discussed in detail in Refs. [19,20]. These methods suffer from the already mentioned large dimension of the input vector which leads to expensive training procedures of the model. Recurrent neural networks (RNNs) are better suited: Due to their "internal memory" and feedback mechanisms they are by construction more appropriate to learn the dynamics (and thus the resulting statistics of the flow). The long short-term memory (LSTM) network [21] is a specific subclass of RNNs, which provides a "gated mechanism" for information flow in a feedback loop. The internal memory state allows the sequential processing of the data, using a smaller layer size or less layers and processing the time series in smaller batches compared to other DNNs [22]. LSTMs have been applied recently with success for a small Galerkin 9-mode model of a turbulent shear flow [23] and compared with a standard DNN [22]. In order to circumvent expensive training procedures, echo state networks (ESNs) or reservoir computing models (RCMs) [24,25] seem to be a further alternative which is considered here. RCMs have received recently renewed attention as a method of equation-free modeling of nonlinear dynamical systems, such as for the already mentioned Galerkin 9-mode model of a turbulent shear flow [26], the Lorenz96 model [27], or the one-dimensional partial differential Kuramoto-Sivashinsky (KS) equation [28,29] on up to 512 grid points. However, none of the previous examples handles two-dimensional spatial fields and evaluates reservoir computing on the basis of turbulent statistics. We also mention here that larger numbers of degrees of freedom of these dynamical systems can been processed with parallel versions of RNNs, as in Ref. [27] for a LSTM.

The present work derives a reservoir computing model for a two-dimensional, fully turbulent Rayleigh-Bénard convection (RBC) flow at a relatively large Rayleigh number of $\mathrm{Ra} = 10^7$ and a Prandtl number of $\mathrm{Pr} = 7$ [30,31] in a domain $\Omega = L \times H$ with an aspect ratio $\Gamma = L/H = 6$. Here L is the horizontal length and H the height of the simulation domain. We set up a (Boussinesq) equation-free model to describe the large-scale dynamics of the flow and to reproduce low-order statistics, such as profiles of the temperature fluctuations and the convective heat flux across the layer as well as the joint statistics of velocity components and temperature. This requires the following subsequent steps:

(1) Data-driven reduction of the fully resolved direct numerical simulation (DNS) record to the most energetic degrees of freedom by a standard proper orthogonal decomposition (POD) based on the snapshot method [32–35], which is applied to the three-dimensional vector field composed of velocity field and temperature fluctuations. This truncation will contain here 83% of the mean total turbulent energy and cause a compression by 92%.

(2) Construction and training of a RCM that predicts the dynamical evolution of the most energetic degrees of freedom obtained in step (1) and thus of the large-scale flow and the result-

ing low-order statistics. We substitute here a Galerkin-truncated reduced-order model, which is obtainable by a projection of the Boussinesq equations of turbulent convection on the individual POD eigenspaces, by the simple dynamics on a large reservoir (which will be explained further below) that does not have any explicit information on the underlying RBC dynamics and is purely data driven.

The combined application of POD analysis with a small feedforward neural network on two-dimensional fields was reported by Krischer *et al.* [36]. The usage of RNNs in combination with POD has been reported by Wan *et al.* [37] for a two-dimensional Kolmogorov flow, by Vlachas *et al.* [38] for the Lorenz96 model and the KS equation, by Deng *et al.* [39] for the reconstruction of time-resolved turbulent flow measurement from discrete data obtained from the experiments, and by Mohan and Gaitonde [40] for turbulent flow control. Pawar *et al.* [41] combined POD with a standard DNN for predictions in a heated cavity flow.

The outline of this paper is as follows. Section II describes the Boussinesq model of turbulent Rayleigh-Bénard convection along with numerical procedure and some basic results. Section III illustrates the POD snapshot method. Section IV presents the RCM employed in this work and the variation of hyperparameters of the reservoir. Section V discusses the results which are obtained for the present application and provides a comparison with the original DNS and POD based data. We conclude with a summary and an outlook in Sec. VI.

II. DIRECT NUMERICAL SIMULATION

Direct numerical simulations are applied to solve the Boussinesq equations for the two-dimensional case. These equations are made dimensionless by the height of the layer H and the free-fall velocity $U_f = \sqrt{g\alpha\Delta T H}$, where g is the acceleration due to gravity, α is the thermal expansion coefficient at constant pressure, and ΔT represents the imposed temperature difference between the bottom and the top. Times are expressed in units of the free-fall time $T_f = H/U_f$. The Rayleigh number is given by $\mathrm{Ra} = g\alpha\Delta T H^3/(\nu\kappa) = 10^7$, the Prandtl number by $\mathrm{Pr} = \nu/\kappa = 7$, and the aspect ratio is $\Gamma = L/H = 6$. Here, ν is the kinematic viscosity and κ is the thermal diffusivity of the fluid. The equations of motion in dimensionless form, which couple the two-dimensional velocity field $\mathbf{u} = (u_x, u_y)$, the pressure field p, and the temperature field T, are given by

$$\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} = 0, \tag{1}$$

$$\frac{\partial u_x}{\partial t} + u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y} = -\frac{\partial p}{\partial x} + \sqrt{\frac{\mathrm{Pr}}{\mathrm{Ra}}}\left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2}\right), \tag{2}$$

$$\frac{\partial u_y}{\partial t} + u_x\frac{\partial u_y}{\partial x} + u_y\frac{\partial u_y}{\partial y} = -\frac{\partial p}{\partial y} + \sqrt{\frac{\mathrm{Pr}}{\mathrm{Ra}}}\left(\frac{\partial^2 u_y}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2}\right) + T, \tag{3}$$

$$\frac{\partial T}{\partial t} + u_x\frac{\partial T}{\partial x} + u_y\frac{\partial T}{\partial y} = \frac{1}{\sqrt{\mathrm{PrRa}}}\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right). \tag{4}$$

No-slip boundary conditions are imposed at the bottom ($y = 0$) and top ($y = 1$) for the velocity field. The temperature field is constant, $T = 1$ at $y = 0$ and $T = 0$ at $y = 1$. The side walls obey periodic boundary conditions for all fields. Equations (1)–(4) are numerically solved by the NEK5000 spectral element method package [42]. We use $N_e = 48 \times 16$ spectral elements and apply Lagrangian interpolations polynomials of order $N = 11$ on each element and in each spatial direction. Further details pertaining to the numerical procedure can be found in Refs. [15,31]. We sampled the turbulent flow every $0.25T_f$ for the subsequent POD snapshot analysis.

We apply the standard Reynolds decomposition, which is given by

$$u_x(x, y, t) = \langle u_x(y)\rangle_{x,t} + u_x'(x, y, t) = u_x'(x, y, t), \tag{5}$$

$$u_y(x, y, t) = \langle u_y(y)\rangle_{x,t} + u_y'(x, y, t) = u_y'(x, y, t), \tag{6}$$

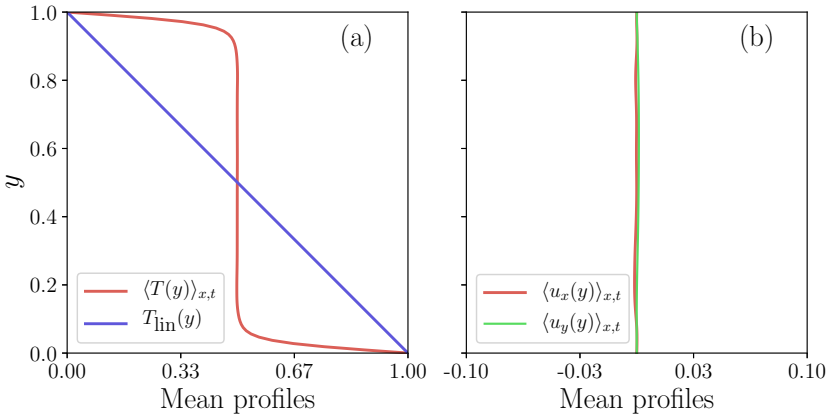

FIG. 1. Mean profiles of (a) temperature and (b) velocity components. The average quantities are obtained by averaging in time and in the homogeneous direction x. The diffusive equilibrium temperature profile $T_{\rm lin}(y)$ is added to panel (a).

$$T(x, y, t) = \langle T(y)\rangle_{x,t} + T'(x, y, t), \tag{7}$$

Mean profiles which have been obtained by a combined x-line and time average are displayed in Fig. 1. In panel (a) of this figure we show linear profile of diffusive heat transfer, $T_{\rm lin}(y) = 1 - y$, which exists for Rayleigh numbers below the instability threshold $\mathrm{Ra}_c = 1708$, together with the mean temperature profile $\langle T(y)\rangle_{x,t}$ for $\mathrm{Ra} \gg \mathrm{Ra}_c$. As expected, the magnitude of mean velocity is practically zero which is shown in Fig. 1(b). This means that $u_x = u'_x$ and $u_y = u'_y$, as indicated in Eqs. (5) and (6). The DNSs start from the diffusive equilibrium state and relax into the statistically stationary turbulent convection state after an initial period of $t \approx 100T_f$. Any subsequent snapshot analysis starts for $t > 100T_f$. For the data analysis, we interpolate all DNS snapshots spectrally onto a uniform, somewhat coarser two-dimensional mesh of $N_x \times N_y = 160 \times 108$ points. Our data base consists of $N_s = 2000$ equidistant simulation snapshots spanning a time range of $500T_f$. Figure 2 displays the time series of the Nusselt number, the dimensionless measure of the turbulent heat transfer, which is taken at the bottom and top plates as an average of the diffusive heat flux,

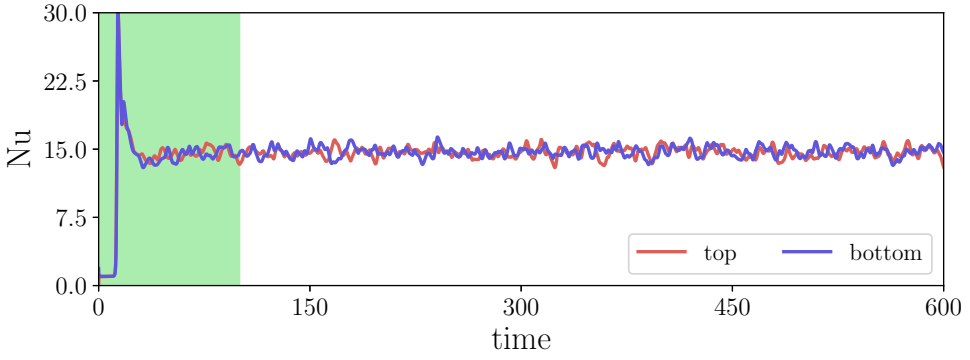

FIG. 2. Temporal variation of Nusselt number for bottom and top plates obtained in the spectral element DNS. We start sampling our data for times $t > 100T_f$ to exclude the initialization effects which are indicated by the green-shaded region.

$\mathrm{Nu}(y=0,1) = -\partial\langle T\rangle_{x,t}/\partial y|_{y=0,1}$. The Nusselt number is obtained as an arithmetic average of both means, and is found to be $\mathrm{Nu} = 14.67 \pm 0.54$. This value agrees well with the results from three-dimensional simulations of RBC for the same parameters which were reported in Ref. [13]

III. PROPER ORTHOGONAL DECOMPOSITION OF SIMULATION DATA

Rather than using slices of DNS data as a direct input into a ML algorithm, as done for example by King *et al.* [19] in a semisupervised algorithm, we introduce an intermediate data reduction step. We take a standard POD analysis for simplicity, a prominent method to extract a subset of the energetically dominant degrees of freedom from the fully resolved turbulent convection data [32–35,43,44]. In detail, we will apply the snapshot method [45,46] which was developed by Sirovich and Park [32,33]. The input is the three-dimensional vector field $v_m = (v_1, v_2, v_3) = (u_x, u_y, T')$ with zero mean, $\langle v_m\rangle_{x,y,t} = 0$. A mean total turbulent energy or variance (which comprises turbulent kinetic energy and scalar fluctuation variance) is given by

$$E = \left\langle \int_\Omega \left(u_x^2 + u_y^2 + T'^2\right) d\Omega \right\rangle_t = \left\langle \int_\Omega v_i^2 d\Omega \right\rangle_t. \tag{8}$$

We want to determine POD modes $\Phi_m(x, y)$ that maximize the averaged projection of v_m onto Φ_m, i.e.,

$$\frac{\langle |(v_m, \Phi_m)|^2\rangle_t}{(\Phi_m, \Phi_m)} \to \max, \tag{9}$$

where $(\cdot, \cdot)$ denotes a scalar product on $L_2(\Omega, \mathbb{R}^2) \oplus L_2(\Omega)$ with $u_m = (u_x, u_y) \in L_2(\Omega, \mathbb{R}^2)$ and $T' \in L_2(\Omega)$ which induces a norm $\|v_m\| = (v_m, v_m)^{1/2}$. We use the Einstein summation convention and drop the summation symbols for same indices, e.g., (v_m, v_m) is the short notation for $\sum_{m=1}^{3}(v_m, v_m)$. Variational calculus translates (9) into a search of maxima of a constrained functional J [47], i.e.,

$$\left.\frac{dJ[\Phi_m + \varepsilon\Psi_m]}{d\varepsilon}\right|_{\varepsilon=0} = 0 \quad \text{with } J[\Phi_m] = \langle |(v_m, \Phi_m)|^2\rangle_t - \lambda(\|\Phi_m\|^2 - 1), \tag{10}$$

with the Lagrangian multiplier λ and $\varepsilon \in \mathbb{R}$. This leads to the following integral equation:

$$\begin{aligned}\int_\Omega \hat{K}_{mn}(x, y, x', y')\Phi_n^{(p)}(x', y')dx'dy' &= \int_\Omega \langle v_m(x, y, t) \otimes v_n(x', y', t)\rangle_t \Phi_n^{(p)}(x', y')dx'dy' \\ &= \lambda_p \Phi_m^{(p)}(x, y)\end{aligned} \tag{11}$$

with the Hermitean, non-negative kernel operator $\hat{K}_{ij}$. The tensor product symbol $\otimes$ is omitted for the rest of the paper. Indices $m, n = 1, 2, 3$ and the Einstein summation convection are again used. There is no summation over index $p = 1, 2, \ldots$, which stands for the POD modes that all solve (11). For the present case $p < \infty$, the kernel operator is a $3N_xN_y \times 3N_xN_y$ matrix with $N_x \times N_y$ being the size of the uniform data mesh. The number $3N_xN_y$ corresponds to the (finite) number of degrees of freedom, N_{dof}, of the flow, which is approximated on a computational grid and consists of three fields. In addition, we use a Fourier expansion in the homogeneous x direction which reduces the POD analysis to the vertical coordinate y only and uses the periodicity along x. Thus (11) translates into

$$\int_0^1 \langle F_k(n_x, y, t)F_l^*(n_x, y', t)\rangle_t \Phi_{l,n_x}^{(p)}(y')dy' = \lambda_{p,n_x}\Phi_{k,n_x}^{(p)}(y) \tag{12}$$

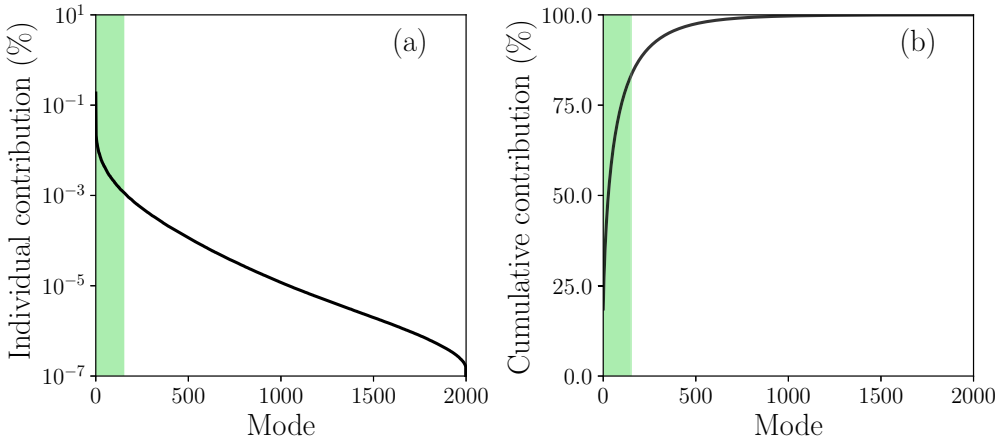

FIG. 3. Eigenvalue spectrum of POD modes as obtained from the analysis of 2000 DNS snapshots. (a) Individual contribution of each mode, and (b) cumulative contribution of modes. The shaded region shows the contribution of the first 150 modes which capture 83% of the total energy of the convection flow and thus provide a good approximation of the large-scale structure.

with $k, l = 1, 2, 3$ and $n_x \in \mathbb{Z}$ (see [35] for details). Here,

$$F_k(n_x, y, t) = \frac{1}{L}\int_{-L/2}^{L/2} v_k(x, y, t)\exp\left(-i\frac{2\pi n_x x}{L}\right)dx \quad \text{and} \quad \Phi_k^{(p)}(x, y) \to \Phi_{k,n_x}^{(p)}(y)\exp\left(i\frac{2\pi n_x x}{L}\right). \tag{13}$$

Thus the velocity components and temperature fluctuation are expanded in the following POD base:

$$v_k(x, y, t) = \sum_{p=1}^{N_{\rm dof}} \sum_{n_x=-N_x/2}^{N_x/2} a_{p,n_x}(t)\Phi_{k,n_x}^{(p)}(y)\exp\left(i\frac{2\pi n_x x}{L}\right), \tag{14}$$

with $p = 1, \ldots, 3N_xN_y = N_{\rm dof} = 51\,840$ (see Sec. II), $k = 1, 2, 3$, $n_x = -N_x/2, \ldots, N_x/2$, and the reality condition $a_{p,n_x}(t) = a^*_{p,-n_x}(t)$. Equation (11) [or (12))] would correspond to a large eigenvalue problem which has to be solved by a direct method. Following Sirovich [45], the complexity of this task can be reduced. The snapshot method converts the original eigenvalue problem (11) for the kernel operator into one for a $N_s \times N_s$ (snapshot) matrix with $N_s \ll N_{\rm dof}$. We therefore chose the following expansion ansatz for each POD mode:

$$\Phi_{k,n_x}^{(p)}(y) = \sum_{t_s=1}^{N_s} \beta_{n_x}^{(p)}(t_s)F_k(n_x, y, t_s). \tag{15}$$

We substitute the time average by an arithmetic average over the N_s snapshots. Expansion (15) is inserted into (12) and results in the following approximation:

$$\frac{1}{N_s}\sum_{t_s'=1}^{N_s}\int_0^1 F_k^*(n_x, y', t_s)F_k(n_x, y', t_s')\beta_{n_x}^{(p)}(t_s')\,dy' = \lambda_{p,n_x}\beta_{n_x}^{(p)}(t_s). \tag{16}$$

One has to find the N_s eigenvectors $\beta_{n_x}^{(p)}(t_s)$ and eigenvalues λ_{p,n_x}, which completes the procedure [35]. Here, t_s is the integer that stands for the DNS snapshot at time t_s. Note that the index p runs now from 1 to N_s only.

Figure 3 displays the spectrum of the eigenvalue analysis. The N_s eigenvalues are sorted with respect to their magnitude, which stands for the energy contained in the corresponding POD mode.

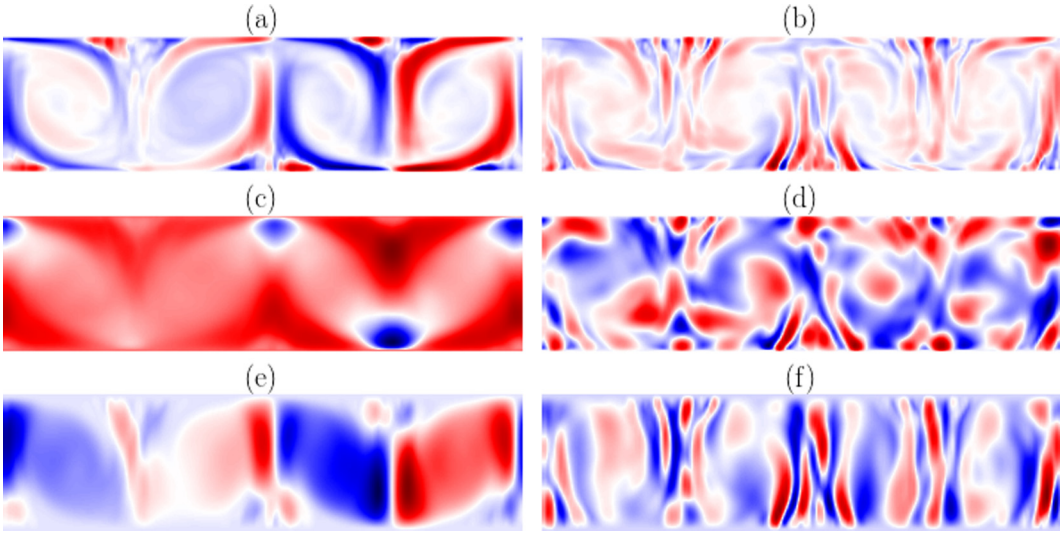

FIG. 4. Spatial structure of two POD modes. Temperature fluctuation field $T'(x, y, t_0)$ snapshot taken at time t_0 and resulting by projection onto (a) $\Phi_3^{(1)}(x, y)$ and (b) $\Phi_3^{(50)}(x, y)$. Horizontal velocity component $u_x(x, y, t_0)$ by projection onto (c) $\Phi_1^{(1)}(x, y)$ and (d) $\Phi_1^{(50)}(x, y)$. Wall-normal velocity component $u_y(x, y, t_0)$ by projection onto (e) $\Phi_2^{(1)}(x, y)$ and (f) $\Phi_2^{(50)}(x, y)$. These modes are shown in the simulation domain and were obtained by the inverse Fourier transform.

We denote these modes again by $\Phi_m^{(p)}$. As is typical for a turbulent flow, the spectrum falls off quickly, but shows a long tail. We will truncate the POD mode expansion to the $N_{\mathrm{POD}} = 150$ most energetic POD modes ($N_{\mathrm{POD}} < N_s$) which contain 83% of the total energy E as given by (8). This is shown in Fig. 3(b). Other truncation levels can be taken, but would not change the subsequent results qualitatively. Figure 4 illustrates the spatial structure of the three components of the modes $\Phi_m^{(1)}(x, y)$ and $\Phi_m^{(50)}(x, y)$. We obtain the time dependence of the expansion coefficients $a_p(t)$ of the most energetic POD modes $\Phi_m^{(p)}$ by

$$a_p(t) = \left(v_m(t), \Phi_m^{(p)}\right) = \left(\sum_{q,n_x} a_{q,n_x}(t)\Phi_{m,n_x}^{(q)} \exp\left(i\frac{2\pi n_x x}{L}\right), \sum_{n_x'} \Phi_{m,n_x'}^{(p)} \exp\left(i\frac{2\pi n_x' x}{L}\right)\right)$$
$$= L\sum_{n_x} a_{p,n_x}(t). \tag{17}$$

Here, $m = 1, 2, 3$ and $p, q = 1, \ldots, N_s$. Orthogonality of the POD modes has been used to obtain the final relation. The time series $a_m(t)$ are used to train the reservoir computing model, which is discussed in the following section. They are considered as the ground truth (GT). The first step of the RCM setup is now completed.

IV. RESERVOIR COMPUTING MODEL

A. Architecture and training of reservoir computing model

In the following, we summarize briefly the basics of the RCM, a special type of RNN [5]. The RCM architecture is inspired by the brain, where many neurons are randomly, recurrently, and sparsely connected. Figure 5 shows the general structure of a RCM. The input consists of training data in the form of time series of the POD expansion coefficients $a_i(t)$ with $i = 1, \ldots, N_{\mathrm{POD}}$. They are converted at each instant into a reservoir state vector $r_k(t) \in \mathbb{R}^N$ with $N \gg N_{\mathrm{POD}}$. This is done

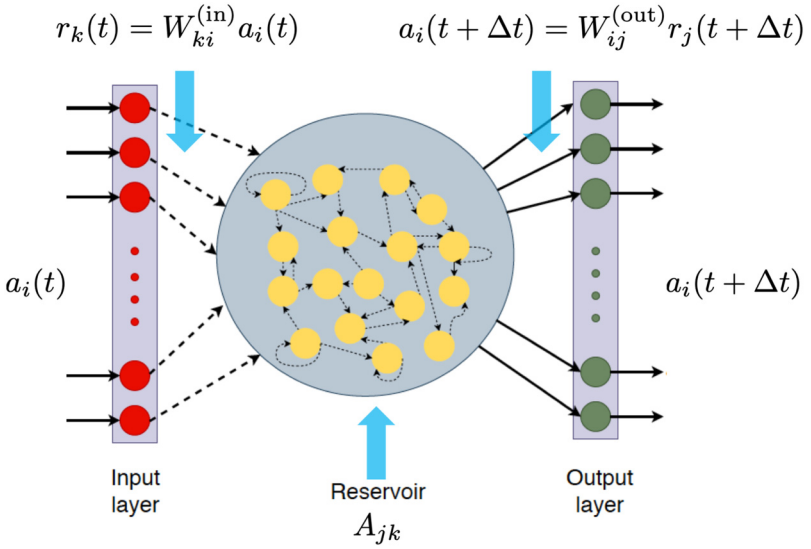

FIG. 5. The basic architecture of a reservoir computing model, RCM (or echo state network ESN), consisting of an input layer, a reservoir, and an output layer. Dotted arrows mark connections which remain fixed during the training. The reservoir consists of N nodes and both the input and output layers have $i = 1, \ldots, N_{\mathrm{POD}}$ units. The reservoir substitutes the stack of hidden layers in a deep neural network.

by means of the random weight matrix $W^{(\mathrm{in})} \in \mathbb{R}^{N \times N_{\mathrm{POD}}}$ which is determined at the beginning of the training,

$$r_k(t) = W_{ki}^{(\mathrm{in})} a_i(t). \tag{18}$$

Here N is the number of nodes in the reservoir, a big sparse random network which is described by a (symmetric) adjacency matrix $A \in \mathbb{R}^{N \times N}$. The initialization of A is again random, and different strategies for this initialization have been suggested which can improve the results, as shown by Strauss *et al.* [48]. Two important parameters of A are the reservoir density D of active nodes and the spectral radius $\rho(A)$, which is set by the largest absolute value of the eigenvalues. Across the reservoir nodes, a simple nonlinear dynamical system evolves which comprises the short-term memory of the network,

$$r_j(t + \Delta t) = (1 - \alpha) r_j(t) + \alpha \tanh\left[A_{jk} r_k(t) + W_{jm}^{(\mathrm{in})} a_m(t)\right], \tag{19}$$

with $j, k = 1, \ldots, N$ and $m = 1, \ldots, N_{\mathrm{POD}}$. The nonlinearity enters in the form of a typical activation function, here a hyperbolic tangent. A further parameter—the leakage rate α—enters the model, which blends linear and nonlinear contributions. Optimally, the reservoir should be operated close to an instability which implies a spectral radius $\rho(A) \lesssim 1$. The final element is the random output weight matrix $W^{(\mathrm{out})} \in \mathbb{R}^{N_{\mathrm{POD}} \times N}$ which maps the updated reservoir vector back to the POD expansion coefficients,

$$a_i(t + \Delta t) = W_{ij}^{(\mathrm{out})} r_j(t + \Delta t). \tag{20}$$

This procedure is repeated for $N_d = 700$ time steps and all reservoir states are saved. The big advantage of the RCM is that the training is performed with respect to the output layer only. Thus a back-propagation procedure that is required in case of DNNs is avoided (see, e.g., [28,29] for more details). As a consequence RCM obtains training input from the preceding time step only while other RNNs take a time step sequence from the past. The optimized output weight matrix, $W^{(\mathrm{out})*}$, is obtained by a minimization of a regularized quadratic cost function C. A regularization term is

added to C to tackle the overfitting problem [5]. The function is given by

$$C[W^{(\mathrm{out})}] = \sum_{k=1}^{N_d} \left| W_{ij}^{(\mathrm{out})} r_j(k\Delta t) - a_i(k\Delta t) \right|^2 + \gamma \, \mathrm{Tr}\left(W_{ij}^{(\mathrm{out})} W_{jm}^{T(\mathrm{out})} \right). \tag{21}$$

During this training process, the number of nodes N, the reservoir density D, the spectral radius $\rho(A)$, the leakage rate α, and the prefactor $\gamma > 0$ of the regularization term of the quadratic cost function are hyperparameters to tune additionally. Prefactor γ in (21) is the ridge regression parameter. The result of this optimization procedure is the matrix $W^{(\mathrm{out})*}$ (as already said) which completes the training.

The prediction mode of the RCM after the training with given POD coefficient time series, $a_i(t)$, as an initial condition is given by

$$r_j(t + \Delta t) = (1 - \alpha) r_j(t) + \alpha \tanh\left[A_{jk} r_k(t) + W_{jk}^{(\mathrm{in})} W_{km}^{(\mathrm{out})*} r_m(t) \right]. \tag{22}$$

Now the reservoir dynamics is closed and no further input is required. The dynamics of Eq. (22) are used to examine the large-scale evolution and low-order statistics of a two-dimensional turbulent convection flow without using the underlying Boussinesq equations (1)–(3). The translation back to the turbulent fields goes via $W_{km}^{(\mathrm{out})*}$ to the $a_p(t)$ and the subsequent reconstruction by the corresponding POD modes $\Phi_k^{(p)}(x, y)$.

B. Implementation of reservoir computing model

Previous studies that apply reservoir computing were frequently performed for dynamical systems with a smaller number of degrees of freedom, as we discussed in the Introduction. Here, we want to take the RCM approach to a new level of complexity by an application to a fully turbulent flow in an extended domain. This will result in additional challenges that start with the architecture and training. The database consists of the last 1400 (out of the originally 2000) snapshots of the turbulent velocity and temperature fields as reconstructed from the 150 POD modes. These most energetic 150 POD modes account for 83% of the mean total turbulent energy E [see Eq. (8)]. By selecting 150 modes, we were able to compress our data by 92%, while losing 17% of the information. We also checked that the separate variances of the two-dimensional velocity field and the temperature fluctuations have a similar magnitude.

We use a 50%-50% split of this dataset for training and testing. The current problem is related to time-series forecasting; therefore a random splitting of the data is not considered. The first $N_d = 700$ snapshots are used for the training of our RCM and the remaining 700 snapshots are exclusively used for (blind) testing of our framework. As already mentioned earlier, there are a few tunable configuration parameters, commonly known as hyperparameters. All these parameters represent the dynamics of the reservoir A and thus affect the RCM [24,25]. These hyperparameters have to be tuned carefully. Bayesian optimization [49,50] is one of the sophisticated methods to find out the optimized hyperparameters in any machine learning task. Here, we applied a simpler grid search as a favorable option which resulted in 100 different hyperparameter settings (N, $\rho(A)$, α). We note here that RCM can become linearly unstable once $\rho(A) > 1$, which sets an upper bound for this parameter. On top of this grid search, the ridge regression parameter γ had to be tuned.

We use two measures to determine the quality of the RCM output in relation to GT, the training data. Note again that ground truth in our case comprises the reconstructions of the large-scale convection flow as obtained from the first 150 POD modes. Therefore, we monitor the following mean square error (MSE), which is given by

$$\mathrm{MSE} = \frac{1}{N_d} \sum_{n=1}^{N_d} \mathrm{MSE}(n) = \frac{1}{N_d} \sum_{n=1}^{N_d} \frac{1}{N_{\mathrm{POD}}} \sum_{j=1}^{N_{\mathrm{POD}}} \left| a_j^{\mathrm{GT}}(n) - a_j^{\mathrm{RCM}}(n) \right|^2. \tag{23}$$

TABLE I. Performance of the reservoir computing model for different hyperparameters triples $(N, \rho(A), \alpha)$ quantified by the mean square error (MSE) for training and test phases. The triples consist of the number of reservoir nodes N, the spectral radius $\rho(A)$, and the leakage rate α. A total of $5 \times 5 \times 4 = 100$ triples were investigated for three different ridge regression parameters γ. The first two columns display results for (N, ρ^*, α^*), the second pair are for $(N^*, \rho(A), \alpha^*)$, and the third pair of columns are for (N^*, ρ^*, α). Here, $N^* = 2100$, $\rho^* = 0.95$, and $\alpha^* = 0.95$ (see also Table II).

N	MSE Training/Test	$\rho(A)$	MSE Training/Test	α	MSE Training/Test
1000	$1.5 \times 10^{-5}/1.3 \times 10^{-3}$	0.1	$2.0 \times 10^{-6}/6.8 \times 10^{-4}$	0.1	$4.5 \times 10^{-4}/6.8 \times 10^{-4}$
1500	$1.3 \times 10^{-5}/8.4 \times 10^{-4}$	0.4	$1.0 \times 10^{-6}/7.2 \times 10^{-4}$	0.4	$9.0 \times 10^{-5}/6.4 \times 10^{-4}$
2100	$1.2 \times 10^{-5}/9.1 \times 10^{-4}$	0.7	$4.0 \times 10^{-6}/8.8 \times 10^{-4}$	0.7	$2.4 \times 10^{-5}/7.5 \times 10^{-4}$
3000	$1.3 \times 10^{-5}/7.9 \times 10^{-4}$	0.95	$1.2 \times 10^{-5}/9.1 \times 10^{-4}$	0.95	$1.2 \times 10^{-5}/9.1 \times 10^{-4}$
10000	$1.6 \times 10^{-5}/8.1 \times 10^{-4}$	1.0	$1.9 \times 10^{-5}/9.8 \times 10^{-4}$		

where $N_{\mathrm{POD}} = 150$ and $N_d = 700$ for training and test data, respectively. We are, however, also interested in the actual flow quantities, such as mean and fluctuation profiles across the convection layer, and thus take furthermore the normalized average relative error (NARE) to ground truth, following [22] in this respect. For example, this error is given for the mean temperature profile by

$$E[\langle T(y)\rangle_{x,t}] = \frac{1}{2\max_{y\in[0,1]}\left(\left|\langle T(y)\rangle_{x,t}^{\mathrm{GT}}\right|\right)} \int_0^1 \left|\langle T(y)\rangle_{x,t}^{\mathrm{GT}} - \langle T(y)\rangle_{x,t}^{\mathrm{RCM}}\right| dy \times 100\%. \tag{24}$$

Table I summarizes the MSE for both training and test phases, for different values of reservoir node number N (five different values), spectral radii $\rho(A)$ (five different values), and leakage rates α (four different values). In order to obtain the prescribed spectral radius, an adjacency matrix $\tilde{A}$ with the prescribed reservoir density is initialized randomly first. The spectral radius $\rho(\tilde{A})$ is determined subsequently and the desired spectral radius $\rho(A)$ is enforced afterwards by a rescaling $A = \tilde{A}\rho(A)/\rho(\tilde{A})$. The hyperparameter tuning proceeds as follows: A hyperparameter grid $(N, \rho(A), \alpha)$ with $5 \times 5 \times 4 = 100$ triples is investigated for three different ridge regression parameters γ. The MSE is determined together with the three NAREs for mean temperature, temperature fluctuations, and convective heat flux for each of the 300 cases. The three NARE values follow typically the same trend in the parameter grid search. These NARE magnitudes are calculated after the training or test runs by reconstructing the turbulence fields and performing the combined line-time average as given in (24). The tables for the three NARE, which would correspond to Table I, are not shown here but have been evaluated. It is observed that the reservoir dynamics show high sensitivity on all these hyperparameters. The joint evaluation by means of MSE and NARE avoids overfitting. For example, the MSE shows its lowest training value at $\rho(A) = 0.4$ (see columns 3 and 4 of Table I), but the reconstructed flow results in a high NARE amplitudes. This supports our earlier argument regarding the necessity of additional NARE monitoring of flow quantities rather than solely relying on the MSE measure of the POD expansion coefficients.

Table II depicts the final optimal values from the grid search. These values are obtained by a cross-validation procedure, i.e., by monitoring both defined measures (23) and (24). We model a dynamical system that comprises $N_{\mathrm{POD}} = 150$ modes as the degrees of freedom. On the one hand, this requires a large number of nodes in the reservoir A; as a rule of thumb N should exceed N_{POD} by an order of magnitude. On the other hand, N should not be chosen too large which can cause an overfitting because irrelevant statistical fluctuations in the training data will be learned by the model [51]. The leakage rate $\alpha^* = 0.95$ is kept close to 1, which indicates that the reservoir evolves slowly [25,28]. This value gave the smallest training MSE error, as seen in two last columns of Table I. The spectral radius of adjacency matrix of the reservoir is the central parameter to tune [51]. A spectral radius close to 1, which is eventually taken here, drives the reservoir dynamics close to

TABLE II. Optimized hyperparameters which result from the whole grid search study (indicated with an asterisk). We list the number of reservoir nodes N, the spectral radius $\rho(A)$, the leakage parameter α, the reservoir density D (not varied), and the prefactor of the regularization term γ. Also displayed are the four measures: Mean square error MSE [see Eq. (23)] and the normalized average relative error (NARE) to ground truth as given by Eq. (24) for temperature fluctuations, mean temperature, and convective heat flux. Optimal hyperparameters are found with respect to all of these four measures.

N^*	$\rho^*(A)$	α^*	D^*	γ^*	MSE Training/Test	$E[\langle(T')^2\rangle]$ Training/Test	$E[\langle T\rangle]$ Training/Test	$E[\langle u_y' T'\rangle]$ Training/Test
2100	0.95	0.95	0.20	5×10^{-2}	1.2×10^{-5} / 9×10^{-4}	0.0071/0.1774	0.22/3.70	1.77/6.22

an instability as discussed in [5]. The training phases are relatively short and inexpensive when compared to standard DNN or LSTM networks since RCMs do not require back-propagation. With the present data size, it took less than a minute when performed on 24 CPUs.

V. PREDICTION OF TWO-DIMENSIONAL TURBULENT CONVECTION

Figure 6 shows examples of the temporal variation of four selected POD modes—the GT—together with their predicted time series from our RCM. The predicted time series closely follow the general trends of the POD data, such as amplitudes and frequencies. For instance, the time series of $a_1(t)$ varies on a much smaller frequency as the one for $a_{150}(t)$, which is seen in Figs. 6(a)

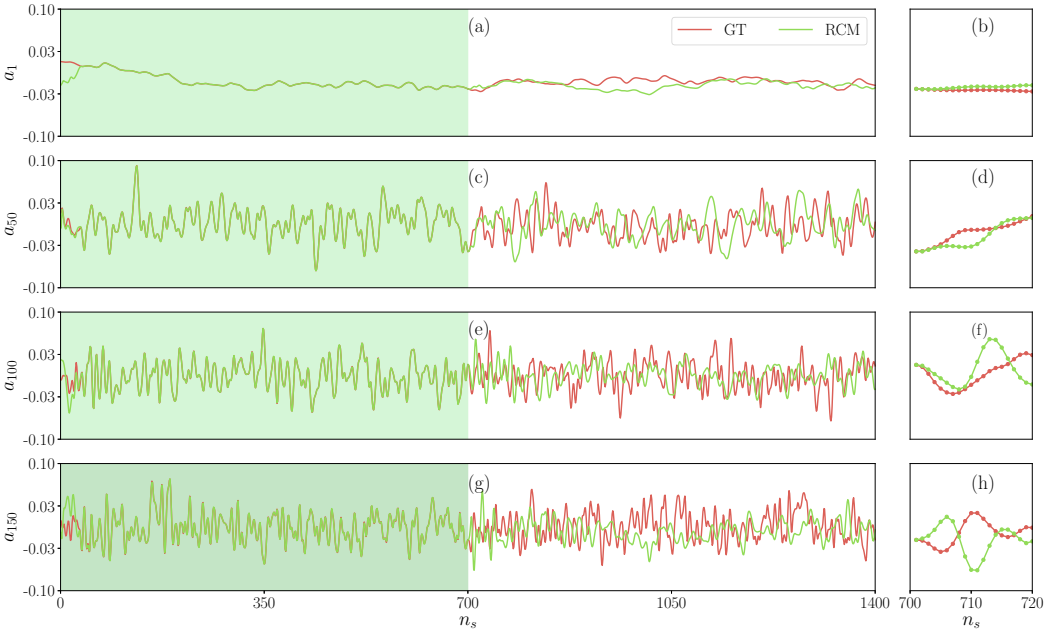

FIG. 6. Temporal evolution of four POD modes coefficients. From top to bottom: (a), (b) $a_1(t)$; (c), (d) $a_{50}(t)$; (e), (f) $a_{100}(t)$; and (g), (h) $a_{150}(t)$. The green shaded range shows the training phase while the other range depicts the test phase. Here $n_s = 1, \ldots, N_s = 1400$ is the snapshot index which translates into the time $t = (n_s/4)T_f$ in a time instant. The corresponding panels in the right column show the initial phase of the forecast. The distance between two snapshots corresponds to 50 integration time steps of the original DNS model.

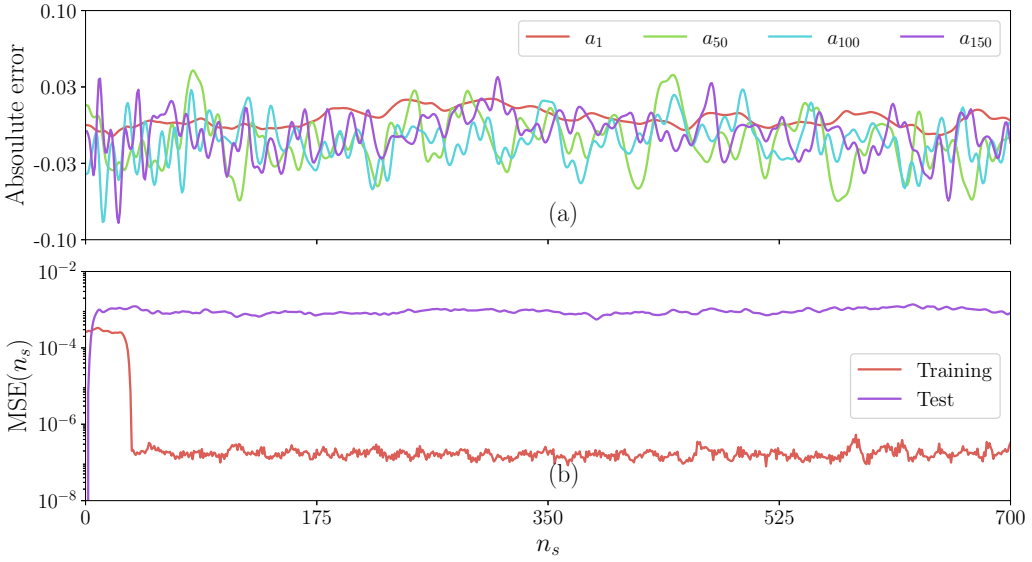

FIG. 7. Temporal evolution of the error for RCM and GT. (a) Absolute error $|a_j^{\mathrm{GT}} - a_j^{\mathrm{RCM}}|$ shown for four individual POD mode coefficients with the mode number indicated by the legend. (b) Mean square error MSE(n_s) versus number of snapshots n_s [see also Eq. (23)].

and 6(g). A comparison of the exact time evolution of four POD modes is displayed in the panels (b), (d), (f), and (h) of Fig. 6. The modes with a larger energy content match the GT for a longer period while the modes with a smaller content deviate quickly. We recall that the turbulent flow modeled here is a highly nonlinear dynamical system. Roundoff errors in the time integration are amplified quickly and lead to an exponentially fast separation of neighboring trajectories in phase space. We conclude that the forecast skills of our RCM model are limited. The GT is a sequence of snapshots which are separated by $0.25T_f$ from each other. This corresponds to 50 integration time steps $dt = 5 \times 10^{-3}$ in the original DNS—an additional source of roundoff errors.

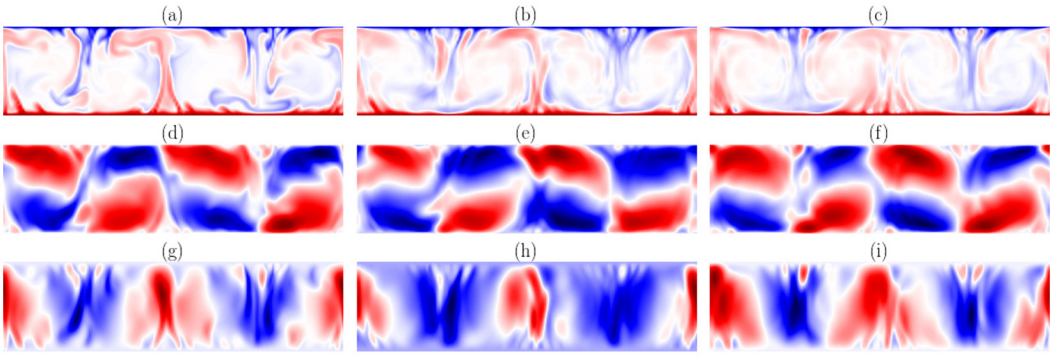

FIG. 8. Comparison of the turbulence fields in the test phase for $n_s = 201$. (a), (d), (g) Original DNS snapshot. (b), (e), (h) POD model reconstruction with the 150 most energetic modes which serves as the ground truth for our machine learning problem. (c), (f), (i) Predicted field from the reservoir computing model. In panels (a), (b), (c) the instantaneous temperature field $T(x, y, t_0)$ is shown, in (d), (e), (f) the velocity component $u_x(x, y, t_0)$, and in (g), (h), (i) the wall-normal velocity component $u_y(x, y, t_0)$.

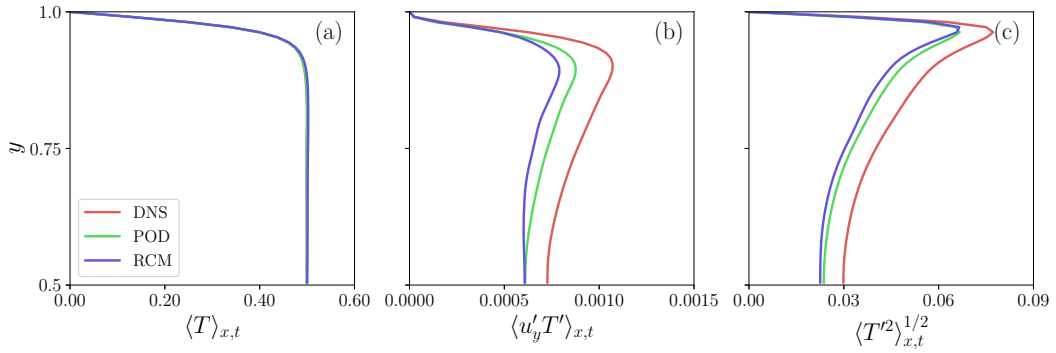

FIG. 9. Comparison line- and time-averaged mean profiles obtained from original DNS, truncated POD, and RCM. (a) Mean temperature, (b) mean turbulent convective heat flux, and (c) root-mean-square fluctuations of temperature. The profiles in case of the RCM were obtained from the test data set only and thus not used in the training phase.

Figure 7(a) depicts the absolute error of the POD modes coefficients. The error fluctuates around zero mean which suggests that trained model does not suffer from any bias in a specific direction. The MSE as a function of time is also illustrated in Fig. 7(b) for both training and testing phases. In correspondence with Fig. 6, the MSE quickly grows from 0 to 0.0005 with time, but saturates again. The RCM model in its present form is limited in view to a detailed forecast, but can reproduce the low-order statistics as we will see further below. A detailed determination of the forecast quality requires one to determine the Lyapunov spectrum of the turbulent flow, as done by Pathak *et al.* [52] and Vlachas *et al.* [27] for other dynamical systems. This is beyond the scope of the present work, which focuses on a reproduction of statistical properties.

Figure 8 depicts an instantaneous snapshot of all three fields from the original DNS and compares the data with the reconstructed field from the 150 POD modes as well as with the prediction from our RCM. We present results only for the blind test data to show the generalization ability of the trained network in predicting all three variables, i.e., the temperature and the two velocity components. It can be observed that the RCM can capture all the large-scale features of the turbulent convection flow with a very good quality even though the forecast skills are limited. This includes a reproduction of the large-scale motion as seen in Figs. 8(b) and 8(c). Thermal plumes, which are

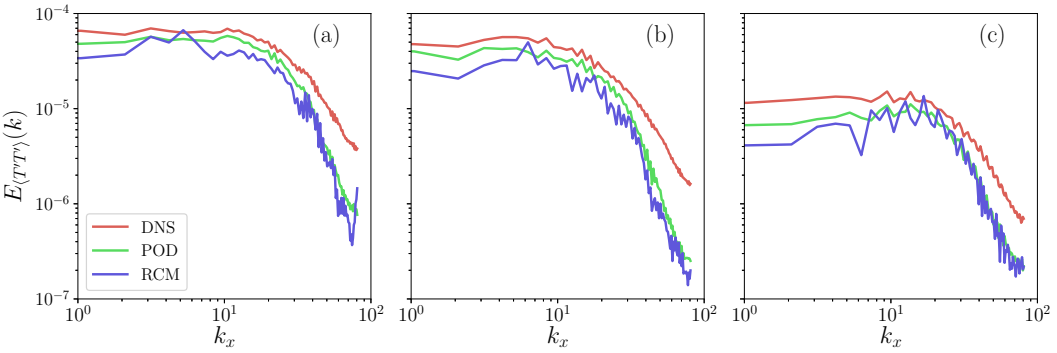

FIG. 10. Comparison of energy spectra for temperature variance at different distances from the wall: (a) $y_0 = 0.07$, (b) $y_0 = 0.15$, and (c) $y_0 = 0.50$. The legend indicates the three data sets taken for the calculation. Line colors are the same for all three panels.

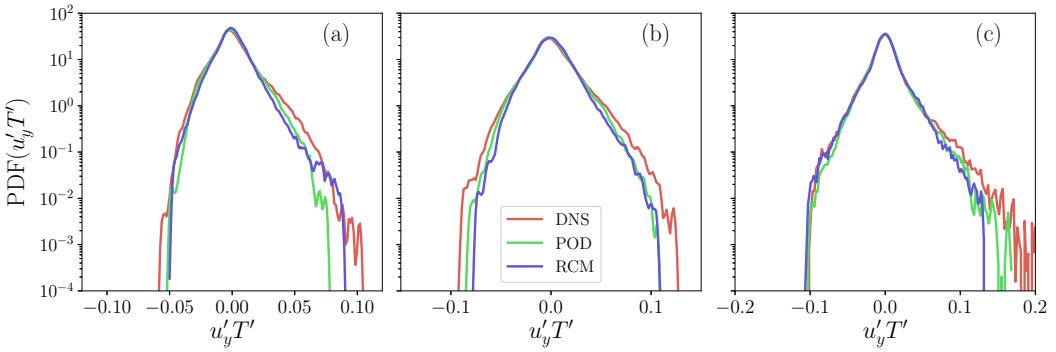

FIG. 11. Comparison of probability density functions (PDFs) of the local convective heat flux ($u_y'T'$) at (a) $y_0 = 0.07$, (b) $y_0 = 0.15$, and (c) $y_0 = 0.50$. The legend indicates the three data sets taken for the calculation. Line colors are the same for all three panels.

a result of thermal boundary layer instabilities [53], can also be captured by the RCM, which is a challenging task at such high Rayleigh number [see Fig. 8(a)].

Next, we report mean and fluctuations profiles of the flow in Fig. 9. Again we show a comparison between the DNS, the projection of the snapshots on the POD modes, and the prediction from RCM. In the Boussinesq case, the flow has an up-down symmetry with respect to the midline at $y = 0.5$ such that we show the upper half only and mirror the data from the lower half. This symmetry, which is in line with a transformation $(x, y, u_x, u_y, T') \to (x, 1 - y, u_x, -u_y, -T')$, could possibly be embedded as a physical constraint in the model, a task which we leave for future work on the three-dimensional case. We also refer here to Ref. [54] that describes how to embed such a symmetry in a neural network for a Hamiltonian system.

Figure 9(a) compares the mean temperature profiles $\langle T(y)\rangle_{x,t}$, which are perfectly reproduced by the POD as well as the reservoir computing model. Figure 9(b) displays the mean profiles of the turbulent convective heat flux, $\langle u_y'T'\rangle_{x,t}$. It can be seen that the RCM result is in good agreement with the POD data, both close to the wall and to the center of the layer. The deviation between the POD data and the DNS data is caused by the truncation at 83% of the total energy, while the deviation between the POD data and the RCM is caused by the model error. Figure 9(c) shows the temperature fluctuations $\langle (T')^2\rangle_{x,t}^{1/2}$. Again the agreement is very good. To summarize,

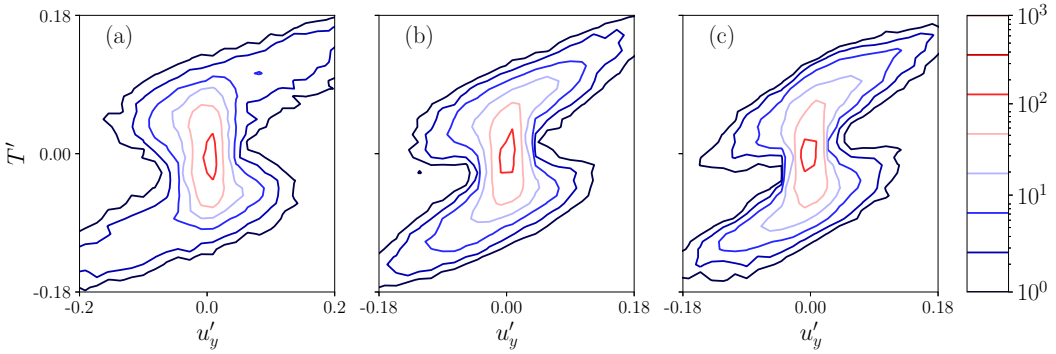

FIG. 12. Comparison of joint probability density functions of the turbulent vertical velocity component and the turbulent temperature fluctuation, $P(u_y', T')$, at the midline at $y_0 = 0.50$: (a) DNS, (b) POD, and (c) RCM. The isocontour levels in the legend are logarithmically spaced and hold for all three panels.

the RCM successfully predicts first and second-order statistics as demonstrated by these mean profiles.

To have a thorough comparison, we add results obtained for the Fourier spectra. Figure 10 depicts the spectrum of the temperature variance at three different locations y_0 in the wall-normal direction. As expected, both POD and RCM spectra deviate from the DNS spectra, particularly in the high-wave-number tail representing the small eddies. The reason is the truncation to 150 POD modes. However, our statistics prediction from the RCM follows closely the trend of the POD-based data.

Furthermore, we show a comparison of probability density functions (PDFs) of local convective heat flux at three different vertical locations in our turbulent convection domain. Figure 11 displays the PDFs, and we can again observe a very good agreement, even for most of the tails of the distributions. After having found such a good agreement of the PDFs of turbulent heat flux, we assess the joint probability density function of the two individual fluctuating fields of the turbulent heat flux. We have therefore sampled instantaneous fluctuations of the wall-normal velocity component u_y' and the temperature fluctuation T' at the midline. With these data, we determined the joint probability density functions, and their contours are shown in Fig. 12. The RCM is capable of reproducing the plume ejections from the bottom and the top which correspond to the skewed contours in the first and third quadrants, respectively.

VI. CONCLUSIONS AND OUTLOOK

In the present work, we have discussed a machine-learning-based approach to convective turbulence that aimed at a simple modeling of the large-scale evolution and low-order statistics. We applied a specific recurrent neural network with an efficient design: The reservoir computing model, also known as the echo state network. Since it is not possible to feed the direct numerical simulation data directly into such a model, in particular in view of future applications to three-dimensional cases, we had to add an intermediate data reduction procedure. Therefore, in this work we applied a standard and well-established proper orthogonal decomposition snapshot analysis which extracted the most energetic degrees of freedom in the form of empirical orthogonal modes of the fully developed turbulent flow at hand. Turbulence fields which are reconstructed from this finite number of POD modes are the ground truth and serve as training and testing data for the reservoir computing model. The quality of the prediction of the reservoir computing model is comprehensively tested in the final part of the work. This is done by a direct comparison of the RCM results with both the original direct numerical simulations and the fields reconstructed with the POD modes. In detail, we find a good agreement of the vertical profiles of mean temperature, mean convective heat flux, and root-mean-square temperature. In addition, we discuss temperature variance spectra and joint probability density functions of the turbulent vertical velocity component and temperature fluctuation, the latter of which is essential for the turbulent heat transport across the layer.

Our presented work should be considered as a first step and a proof-of-concept investigation which certainly can be extended into several directions, which we want to discuss briefly now. (1) Although we report a rather comprehensive analysis of the reservoir parameters, a full Bayesian hyperparameter optimization might improve the performance of the RCM even further. (2) For the present case, it was not necessary to augment the training data since we could generate a long-term DNS record for the two-dimensional case with a reasonable computational effort. In a three-dimensional application this will not be the case anymore as discussed for example in Refs. [13,15]. Data augmentation can be done for example by using symmetries of the corresponding turbulent flow, an idea that goes back to Sirovich and Park in their application of POD to convection [32,33]. Similar aspects of physics-informed ML were recently discussed in another context by Mattheakis *et al.* [54] for the symplectic nature of Hamiltonian systems which were modeled by neural networks. (3) First efforts have been discussed to generalize the RCM or ESN to a

deep-ESN which consists of multiple reservoirs for a possible direct simulation data processing [55]. The successful application of this network architecture to turbulence is yet to be accomplished.

To summarize, our presented reservoir computing model could successfully capture essential properties of the dynamics of the larger scales of a turbulent convection flow. It is a recurrent neural network that describes the turbulent convection flow without explicit knowledge of the Boussinesq equations, which can lead to interesting applications, e.g., for global circulation models where mesoscale (moist) convection processes [56] have to be parametrized [57]. Our presented efforts will be extended to the three-dimensional RBC case and to subsequent tests of the performance of the RCM with respect to variations of Ra and Pr.

ACKNOWLEDGMENTS

The work is supported by the Deutsche Forschungsgemeinschaft through Grant No. SCHU 1410/30-1 and in part by the project "DeepTurb–Deep Learning in and of Turbulence" which is funded by the Carl Zeiss Foundation. The generation of the long-term DNS database was possible due to supercomputing resources which were provided by the project grant HIL12 of the John von Neumann Institute for Computing. We thank Christian Cierpka, Florian Heyder, Patrick Mäder, and Karl Worthmann for fruitful discussions.

[1] G. Hinton, L. Deng, D. Yu *et al.*, Deep neural networks for acoustic modeling in speech recognition, IEEE Signal Process. Mag. **29**, 82 (2012).
[2] M. I. Jordan and T. M. Mitchell, Machine learning: Trends, perspectives, and prospects, Science **349**, 255 (2015).
[3] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature (London) **521**, 436 (2015).
[4] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Netw. **61**, 85 (2015).
[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, 2016).
[6] J. Ling, A. Kurzawski, and J. Templeton, Reynolds averaged turbulence modeling using deep neural networks with embedded invariance, J. Fluid Mech. **807**, 155 (2016).
[7] J. N. Kutz, Deep learning in fluid dynamics, J. Fluid Mech. **814**, 1 (2017).
[8] K. Duraisamy, G. Iaccarino, and H. Xiao, Turbulence modeling in the age of data, Annu. Rev. Fluid Mech. **51**, 357 (2019).
[9] R. Fang, D. Sondak, P. Protopapas, and S. Succi, Neural network models for the anisotropic Reynolds stress tensor in turbulent channel flow, J. Turbul. **21**, 525 (2020).
[10] A. Mohan, D. Tretiak, M. Chertkov, and D. Livescu, Spatio-temporal deep learning models of 3D turbulence with physics informed diagnostics, J. Turbul. **21**, 484 (2020).
[11] S. Brunton, B. R. Noack, and P. Koumoutsakos, Machine learning for fluid mechanics, Annu. Rev. Fluid Mech. **52**, 477 (2020).
[12] O. Ronneberger, P. Fischer, and T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015: Proceedings of 18th International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, October 5–9, 2015, Part III*, edited by N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Lecture Notes in Computer Science Vol. 9351 (Springer, Cham, 2015), p. 234.
[13] E. Fonda, A. Pandey, J. Schumacher, and K. R. Sreenivasan, Deep learning in turbulent convection networks, Proc. Natl. Acad. Sci. USA **116**, 8667 (2019).
[14] M. S. Emran and J. Schumacher, Large-scale mean patterns in turbulent convection, J. Fluid Mech. **776**, 96 (2015).
[15] A. Pandey, J. D. Scheel, and J. Schumacher, Turbulent superstructures in Rayleigh-Bénard convection, Nat. Commun. **9**, 2118 (2018).
[16] M. Raissi, P. Perdikaris, and G. E. Karniadakis, Physics-informed neural networks: A deep learning

framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. **378**, 686 (2019).
[17] N. Geneva and N. Zabaras, Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks, J. Comput. Phys. **403**, 109056 (2020).
[18] M. Raissi, A. Yazdani, and G. E. Karniadakis, Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations, Science **367**, 1026 (2020).
[19] R. King, O. Hennigh, A. T. Mohan, and M. Chertkov, From deep to physics-informed learning of turbulence: Diagnostics, arXiv:1810.07785.
[20] A. T. Mohan, D. Daniel, M. Chertkov, and D. Livescu, Compressed convolutional LSTM: An efficient deep learning framework to model high fidelity 3D turbulence, arXiv:1903.00033.
[21] S. Hochreiter and J. Schmidhuber, Long short-term memory, Neural Comput. **9**, 1735 (1997).
[22] P. A. Srinivasan, L. Guastoni, H. Azizpour, P. Schlatter, and R. Vinuesa, Predictions of turbulent shear flows using deep neural networks, Phys. Rev. Fluids **4**, 054603 (2019).
[23] J. Moehlis, H. Faisst, and B. Eckhardt, A low-dimensional model for turbulent shear flows, New J. Phys. **6**, 56 (2004).
[24] H. Jaeger and H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, Science **304**, 78 (2004).
[25] M. Lukoševičius and H. Jaeger, Reservoir computing approaches to recurrent neural network training, Comput. Sci. Rev. **3**, 127 (2009).
[26] S. Pandey, J. Schumacher, and K. R. Sreenivasan, A perspective on machine learning in turbulent flows, J. Turbul. **21**, 567 (2020).
[27] P. R. Vlachas, J. Pathak, B. R. Hunt, T. P. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos, Forecasting of spatio-temporal chaotic dynamics with recurrent neural networks: A comparative study of reservoir computing and backpropagation algorithms, Neural Netw. **126**, 191 (2020).
[28] Z. Lu, J. Pathak, B. R. Hunt, M. Girvan, R. Brockett, and E. Ott, Reservoir observers: Model-free inference of unmeasured variables in chaotic systems, Chaos **27**, 041102 (2017).
[29] J. Pathak, B. R. 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).
[30] F. Chillà and J. Schumacher, New perspectives in turbulent Rayleigh-Bénard convection, Eur. Phys. J. E **35**, 58 (2012).
[31] J. D. Scheel, M. S. Emran, and J. Schumacher, Resolving the fine-scale structure in turbulent Rayleigh-Bénard convection, New J. Phys. **15**, 113063 (2013).
[32] L. Sirovich and H. Park, Turbulent thermal convection in a finite domain: Part I. Theory, Phys. Fluids A **2**, 1649 (1990).
[33] H. Park and L. Sirovich, Turbulent thermal convection in a finite domain: Part II. Numerical results, Phys. Fluids A **2**, 1659 (1990).
[34] J. Bailon-Cuba, M. S. Emran, and J. Schumacher, Aspect ratio dependence of heat transfer and large-scale flow in turbulent convection, J. Fluid Mech. **655**, 152 (2010).
[35] J. Bailon-Cuba and J. Schumacher, Low-dimensional model of turbulent Rayleigh-Bénard convection in a Cartesian cell with square domain, Phys. Fluids **23**, 077101 (2011).
[36] K. Krischer, R. Rico-Martínez, I. G. Kevrekidis, H. H. Rotermund, G. Ertl, and J. L. Hudson, Model identification of a catalytic spatiotemporally varying reaction, AIChE J. **39**, 89 (1993).
[37] Z. Y. Wan, P. Vlachas, P. Koumoutsakos, and T. P. Sapsis, Data-assisted reduced-order modeling of extreme events in complex dynamical systems, PLoS One **13**, e0197704 (2018).
[38] P. R. Vlachas, W. Byeon, Z. Y. Wan, T. P. Sapsis, and P. Koumoutsakos, Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks, Proc. R. Soc. A **474**, 20170844 (2018).
[39] Z. Deng, Y. Chen, Y. Liu, and K. C. Kim, Time-resolved turbulent velocity field reconstruction using a long short-term memory (LSTM)-based artificial intelligence framework, Phys. Fluids **31**, 075108 (2019).
[40] A. T. Mohan and D. V. Gaitonde, A deep learning based approach to reduced order modeling for turbulent flow control using LSTM neural networks, arXiv:1804.09269.

[41] S. Pawar, S. M. Rahman, H. Vaddireddy, O. San, A. Rasheed, and P. Vedula, A deep learning enabler for nonintrusive reduced order modeling of fluid flows, Phys. Fluids **31**, 085101 (2019).
[42] P. F. Fischer, An overlapping Schwarz method for spectral element solution of the incompressible Navier-Stokes equations, J. Comput. Phys. **133**, 84 (1997).
[43] B. Podvin and A. Sergent, Proper orthogonal decomposition investigation of turbulent Rayleigh-Bénard convection in a rectangular cavity, Phys. Fluids **24**, 105106 (2012).
[44] B. Podvin and A. Sergent, A large-scale investigation of wind reversal in a square Rayleigh-Bénard cell, J. Fluid Mech. **766**, 172 (2015).
[45] L. Sirovich, Turbulence and the dynamics of coherent structures. Part I: Coherent structures, Q. Appl. Math. **XLV**, 561 (1987).
[46] J. Weiss, A tutorial on the Proper Orthogonal Decomposition, in *AIAA Aviation 2019 Forum* (AIAA, Reston, VA, 2019) p. 3333.
[47] P. Holmes, J. L. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry* (Cambridge University Press, Cambridge, 1996).
[48] T. Strauss, W. Wustlich, and R. Labahn, Design strategies for weight matrices of echo state networks, Neural Comput. **24**, 3246 (2012).
[49] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, Algorithms for hyper-parameter optimization, in *NIPS'11: Proceedings of the 24th International Conference on Neural Information Processing Systems*, edited by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger (ACM, New York, 2011), pp. 2546–2554.
[50] J. Snoek, H. Larochelle, and R. P. Adams, Practical Bayesian optimization of machine learning algorithms, in *NIPS'12: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (ACM, New York, 2012), pp. 2951–2959.
[51] H. Jaeger, *Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the Echo State Network Approach* (GMD-Forschungszentrum Informationstechnik, Bonn, 2002), Vol. 5.
[52] J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, R. Brockett, and E. Ott, Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data, Chaos **27**, 121102 (2017).
[53] A. K. De, V. Eswaran, and P. K. Mishra, Dynamics of plumes in turbulent Rayleigh-Bénard convection, Eur. J. Mech. B: Fluids **72**, 164 (2018).
[54] M. Mattheakis, P. Protopapas, D. Sondak, M. Di Giovanni, and E. Kaxiras, Physical symmetries embedded in neural networks, arXiv:1904.08991.
[55] C. Gallicchio and A. Micheli, Richness of deep echo state networks, in *Advances in Computational Intelligence: 15th International Work-Conference on Artificial Neural Networks, IWANN 2019, Gran Canaria, Spain, June 12–14, 2019, Part I*, edited by I. Rojas, G. Joya, and A. Catala, Lecture Notes in Computer Science Vol. 11506 (Springer, Cham, 2019), p. 480.
[56] O. Pauluis and J. Schumacher, Self-aggregation of clouds in conditionally unstable moist convection, Proc. Natl. Acad. Sci. USA **108**, 12623 (2011).
[57] P. Gentine, M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis, Could machine learning break the convection parametrization deadlock? Geophys. Res. Lett. **45**, 5742 (2018).