Computer Methods in Applied Mechanics and Engineering 447 (2025) 118382

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

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

Linear model reduction using spectral proper orthogonal decomposition

Peter Frame [a,*], Cong Lin [b], Oliver T. Schmidt [b], Aaron Towne [a]

[a] Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI, USA
[b] Department of Mechanical and Aerospace Engineering, University of California, San Diego, CA, USA

ARTICLE INFO

Keywords:
Space-time model reduction
Spectral POD
Linear dynamical systems

ABSTRACT

Most model reduction methods reduce the state dimension and then temporally evolve a set of coefficients that encode the state in the reduced representation. In this paper, we instead employ an efficient representation of the entire trajectory of the state over some time interval of interest and then solve for the static coefficients that encode the trajectory on the interval. We use spectral proper orthogonal decomposition (SPOD) modes, which are provably optimal for representing long trajectories and substantially outperform any representation of the trajectory in a purely spatial basis (e.g., POD). We develop a method to solve for the SPOD coefficients that encode the trajectories for forced linear dynamical systems given the forcing and initial condition, thereby obtaining the accurate prediction of the dynamics afforded by the SPOD representation of the trajectory. The method, which we refer to as spectral solution operator projection (SSOP), is derived by projecting the general time-domain solution for a linear time-invariant system onto the SPOD modes. We demonstrate the new method using two examples: a linearized Ginzburg-Landau equation and an advection-diffusion problem. In both cases, the error of the proposed method is orders of magnitude lower than that of POD-Galerkin projection and balanced truncation. The method is also fast, with CPU time comparable to or lower than both benchmarks in our examples. Finally, we describe a data-free space-time method that is a derivative of the proposed method and show that it is also more accurate than balanced truncation in most cases.

1. Introduction

The expense of many modern computational models can prohibit their use in applications where speed is required. In a design optimization problem, for example, many simulations with different boundary conditions or parameters must be performed. In control applications, simulations may need to be conducted in real time to inform actuation. Model reduction techniques strive to deliver the orders-of-magnitude speedup necessary to enable adequately fast simulation for these and other problems with only a mild sacrifice in accuracy.

The great majority of model reduction methods employ the following two-step strategy: (i) find an accurate compression of the state of the system at a particular time and (ii) find equations that evolve the coefficients that represent the state in this representation. The POD-Galerkin method [1–3], perhaps the most widely used starting point for model reduction, is representative of this approach. The proper orthogonal decomposition (POD) modes are an efficient means of representing the state in that with relatively few POD coefficients, the state can often be represented to high accuracy. In a POD-Galerkin reduced-order model (ROM), these coefficients

* Corresponding author.
E-mail address: pframe@umich.edu (P. Frame).

https://doi.org/10.1016/j.cma.2025.118382
Received 11 April 2025; Received in revised form 2 September 2025; Accepted 3 September 2025
Available online 15 September 2025

are then evolved by projecting the governing equations into the space of POD modes, yielding a much smaller dynamical system to evolve. Many alternative choices have been explored for both steps. Examples of the compression step include using balanced truncation modes [4] and autoencoders [5,6], and examples of deriving the equations in the reduced space include using Petrov-Galerkin projections [7–9] and learning the equations from data [10,11]. All of these approaches to model reduction, however, fall within the two-step strategy outlined above.

We have investigated a different approach in this work: instead of representing the state (at a particular time) in a reduced manner, we instead employ a reduced representation for the entire trajectory, i.e., the state's evolution for some time interval. Whereas POD modes are the most efficient (linear) representation of the state, they are far from the most efficient representation of trajectories. This is true intuitively – to represent a trajectory with POD modes, one has to specify the POD coefficients for each time step along the trajectory, but from one time step to the next, the POD coefficients are highly correlated. The analog of POD for entire trajectories is space-time POD [12–14]. Space-time POD modes are themselves time- and space-dependent, so to represent a trajectory, they are weighted by static coefficients. These modes are the most efficient linear representation of trajectories in the sense that to represent a trajectory to some desired accuracy, fewer degrees of freedom are needed if the trajectory is represented with space-time POD modes than any other linear encoding scheme. Unfortunately, space-time POD modes have a number of characteristics that make them undesirable for model reduction; computing them requires much training data, storing them is memory intensive, and computing space-time inner products, which would be necessary in a space-time ROM method, is expensive.

Fortunately, an efficient space-time basis that does not share the undesirable properties of space-time POD modes exists. Spectral POD (SPOD) modes are most naturally formulated as a POD in the frequency domain. More precisely, at every temporal frequency, there exists a set of spatial modes that optimally represent the spatial structure at that frequency. These modes are the SPOD modes, and they may be thought of as space-time modes where each spatial mode $\boldsymbol{\psi}_{k,j}$ at frequency ω_k has the time dependence $e^{i\omega_k t}$. Each mode is associated with an energy, and these energies may be compared across frequencies; for example, the second mode at one frequency may have more energy than the first mode at another frequency. The fact that motivates this work is that the most energetic SPOD modes are also an excellent basis for representing trajectories. In fact, SPOD modes converge to space-time POD modes as the time interval becomes long, so for long intervals, the representation of a trajectory with SPOD modes is nearly as accurate (on average) as the space-time POD representation, which is optimal among all linear representations [14].

With this motivation, the goal of this work is to develop an algorithm to solve quickly for the SPOD coefficients that represent a trajectory for forced linear dynamical systems given the initial condition and forcing, as shown in Fig. 1. If these coefficients can be obtained accurately, then the resulting error will be substantially lower than that of a POD-Galerkin model with the same number of modes. The method works as follows. The SPOD coefficients at a given frequency are related to the (temporal) Fourier transform of

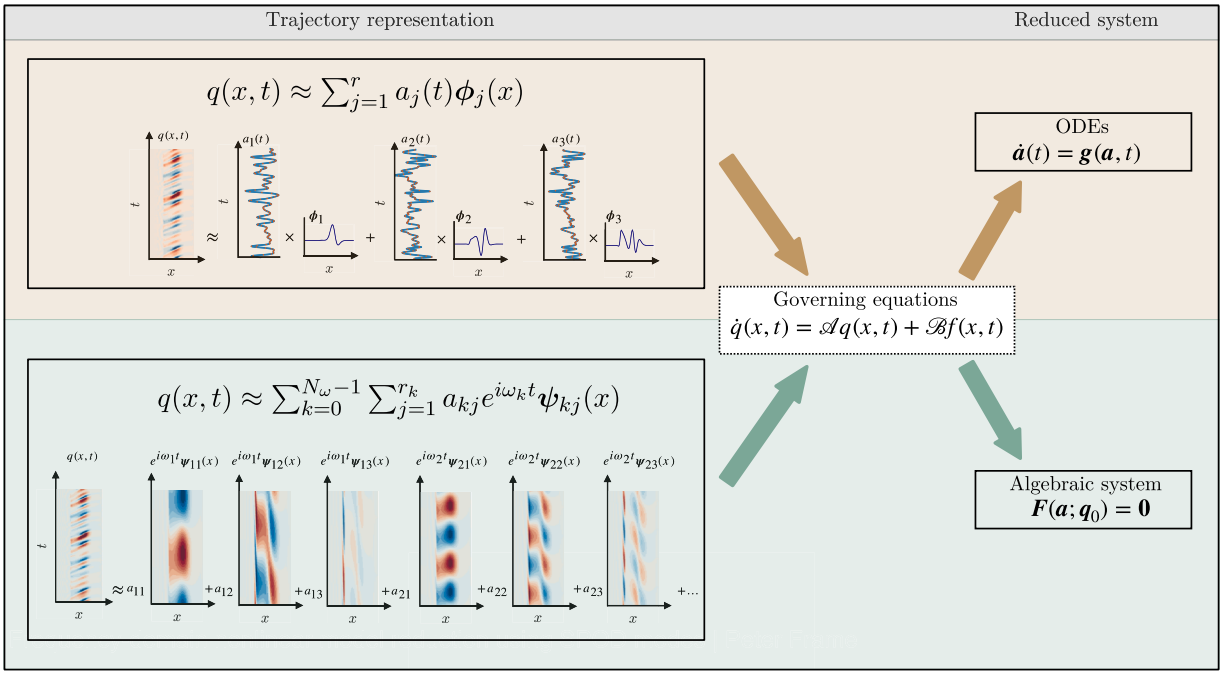

Fig. 1. The proposed model reduction approach (bottom panel) compared against a standard space-only linear model reduction (top panel). To represent a trajectory, the space-only basis vectors are multiplied by time-dependent coefficients; the SPOD modes with their oscillatory time dependence are multiplied by static coefficients. In the space-only case, the coefficients are obtained by integrating a set of ODEs forward in time, whereas in the SPOD case, the coefficients are obtained by solving a linear algebraic system.

the state at the same frequency, which in turn is related to the forcing and initial condition. We derive these relations by starting from the fundamental solution to linear time-invariant (LTI) systems, then performing a discrete Fourier transform analytically, and finally projecting the result onto the SPOD modes at each frequency. Using the LTI solution as a starting point makes the method applicable to general linear systems, avoiding the requirement that the system be periodic, which might otherwise arise with a temporal Fourier basis. The operators involved are all linear and are precomputed, leaving only small matrix-vector multiplications to be done online. The method amounts to a space-time projection of a space-time solution operator, and we refer to it as *Spectral Solution Operator Projection* (SSOP).

We demonstrate the method on two problems: a linearized Ginzburg-Landau problem with a spatial dimension of $N_x = 220$ and an advection-diffusion problem with $N_x = 9604$. We show that, indeed, we can solve for the SPOD coefficients accurately, resulting in two-orders-of-magnitude lower error than even the projection of the solution onto the same number of POD modes, which is itself a lower bound for the error in any time-domain Petrov-Galerkin method, such as balanced truncation (BT) [4]. We show that this accuracy improvement does not come with an increase in CPU time; the computational cost of our method is similar to that of POD-Galerkin projection and balanced truncation, consistent with scaling estimates we derive.

The SPOD modes that form the basis for the method are obtained from data. A data-free version of the method is made possible by a connection between SPOD modes and the left singular vectors of the resolvent operator. The resolvent operator comes directly from the matrices that define the LTI system, so its singular value decomposition does not rely on data. As established by Ref. [15], the left singular vectors of the resolvent operator at a given frequency are equivalent to the SPOD modes at the same frequency if the forcing in the system is spatially white. Many studies [16–20], however, have shown that remarkable similarity between the two types of modes persists even when the forcing is far from white. Given that the SPOD modes and resolvent modes are similar, the latter serves as an excellent space-time basis for trajectories. We find that the data-free method that results from substituting the resolvent modes for the SPOD modes in the proposed method yields lower error than balanced truncation, the state-of-the-art data-free method.

Though the space-time approach is uncommon, we are not the first to attempt it [21–25]. Previous methods have formed space-time basis vectors by assigning time dependence to POD modes, i.e., where each space-time basis vector is formed as the Kronocker product of a POD mode and a time dependent function. The time dependent functions are obtained from time series of the corresponding POD coefficients in the training data. We believe that the representational advantage of SPOD modes relative to previous choices of space-time basis, as well as their analytic time dependence, make them a more compelling choice for model reduction.

Using SPOD modes for linear model reduction has been explored before by Refs. [26] and [27]. Both methods can be viewed as reduced-order models for the frequency-domain equation $(i\omega\mathbf{I} - \mathbf{A})\hat{\boldsymbol{q}}(\omega) = \mathbf{B}\hat{\boldsymbol{f}}(\omega)$, where $\hat{\boldsymbol{q}}$ and $\hat{\boldsymbol{f}}$ are the Fourier transformed state and forcing, respectively, and $\mathbf{A}$ and $\mathbf{B}$ are the standard state-space matrices. However, this equation cannot capture transient behavior; it describes the steady state when the system is subjected to a periodic forcing, and this restriction precludes these ROMs from being applicable to many systems. In this paper, we derive a different frequency-domain equation that captures both transient and steady-state behavior and use it as the starting point for the proposed reduced-order model. By projecting a space-time solution operator onto SPOD modes, we prevent unretained modes from affecting the accuracy of retained coefficients. This was achieved by Ref. [27] (in the periodic case) by employing a Petrov-Galerkin projection, but not by Ref. [26] wherein a Galerkin projection was used. Another advancement of this work relative to Refs. [26,27] is the use of different numbers of SPOD modes at different frequencies. This allows for more computational resources to be devoted to the more energetic frequencies. We also note that SPOD modes have been used to construct time-domain ROMs [28,29]. However, these models are only superficially related to the space-time ROM proposed in this paper.

Our approach may also be related to the harmonic balance method [30,31]. In this technique, the governing equations for a temporally periodic system are Fourier-transformed in time, resulting in a set of nonlinear equations to be solved for the transformed fields. This has been applied to the periodic flows arising in turbomachinery [32], resulting in significant computational speedup due to the relatively small number of relevant harmonics. Harmonic balance does not employ a spatial reduction, and our method may be viewed as a spatially-reduced harmonic balance method for linear problems. Another important difference is that our method is applicable to non-periodic systems as well as periodic ones.

One requirement for the application of the proposed method bears mentioning here: the first step in the method is to take the temporal Fourier transform of the forcing. Therefore, the forcing on the entire time interval of interest must be available before starting computation in the reduced-order model. Any space-time or frequency-domain method that incorporates forcing, such as the ones mentioned above [21–24,26,27,33], is restricted in the same way. The proposed method is designed for applications where the forcing is known beforehand, such as, e.g., adjoint-based optimization and open loop control.

The remainder of this paper is organized as follows. In Section 2, we discuss the properties of POD, space-time POD, and SPOD that are relevant to the method. We present the method in Section 3, and demonstrate it applied to a linearized Ginzburg-Landau problem and a scalar transport problem in Section 4. We conclude the paper in Section 5.

2. Space-only, space-time, and spectral POD

We briefly review the space-only, space-time, and spectral forms of POD here; see Refs. [14] and [15] for additional details. The most significant point for the purposes of this paper is the fact that spectral POD modes approach space-time POD modes as the time interval becomes long, and thus are very efficient in representing trajectories.

2.1. Space-only POD

Space-only POD aims to reconstruct snapshots of the state by adding together prominent modes weighted by expansion coefficients. In the continuous setting, the state $\boldsymbol{q} : \Omega \to \mathbb{C}^{N_v}$ is a function that maps elements of the spatial domain $\mathbf{x} \in \Omega$ to the N_v state variables. The first space-only POD mode $\boldsymbol{\phi}_1 : \Omega \to \mathbb{C}^{N_v}$ is defined to maximize the functional $\lambda[\boldsymbol{\phi}(\mathbf{x})]$, which quantifies the expected value of the energy captured by its argument,

$$\lambda[\boldsymbol{\phi}(\mathbf{x})] = \frac{\mathbb{E}\left[|\langle \boldsymbol{q}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x})\rangle_x|^2\right]}{\|\boldsymbol{\phi}(\mathbf{x})\|_x^2}, \tag{2.1a}$$

$$\boldsymbol{\phi}_1(\mathbf{x}) = \arg\max \lambda[\boldsymbol{\phi}(\mathbf{x})]. \tag{2.1b}$$

The subsequent modes are defined to maximize the energy captured under the constraint that they are orthogonal to all previous ones,

$$\boldsymbol{\phi}_j(\mathbf{x}) = \arg\max_{\langle \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}_{k<j}(\mathbf{x})\rangle_x = 0} \lambda[\boldsymbol{\phi}(\mathbf{x})]. \tag{2.2}$$

The space-only inner product $\langle \cdot, \cdot \rangle_x$, which defines the energy captured, is defined as an integral over the spatial domain Ω,

$$\langle \boldsymbol{q}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x})\rangle_x = \int_\Omega \boldsymbol{\phi}^*(\mathbf{x}) \mathsf{W}(\mathbf{x}) \boldsymbol{q}(\mathbf{x}) d\mathbf{x}, \tag{2.3}$$

where $\mathsf{W}(\mathbf{x})$ is a weight matrix used to account for inter-variable importance or possibly to preference certain regions of the domain. The space-only norm $\|\cdot\|_x$ is induced by the space-only inner product. One can show [12,15] that the solution to the optimization problem Eqs. (2.1b) and (2.2) is modes that are eigenfunctions of the space-only correlation tensor,

$$\int_\Omega \mathsf{C}(\mathbf{x}_1, \mathbf{x}_2) \mathsf{W}(\mathbf{x}_2) \boldsymbol{\phi}_j(\mathbf{x}_2) d\mathbf{x}_2 = \lambda_j \boldsymbol{\phi}_j(\mathbf{x}_1), \tag{2.4}$$

where the eigenvalue is equal to the energy of the mode, i.e., $\lambda_j = \lambda[\boldsymbol{\phi}_j]$, and the correlation tensor is

$$\mathsf{C}(\mathbf{x}_1, \mathbf{x}_2) = \mathbb{E}[\boldsymbol{q}(\mathbf{x}_1) \boldsymbol{q}^*(\mathbf{x}_2)]. \tag{2.5}$$

2.2. Space-time POD

Whereas space-only POD modes optimally represent snapshots, space-time POD modes optimally represent trajectories over the time window $[0, T]$. The formulation is much the same as in space-only POD; the modes optimize the expected energy Eqs. (2.1b) and (2.2), but in space-time POD, the inner product involved in defining the energy functional includes time as well as space,

$$\langle \boldsymbol{q}(\mathbf{x}, t), \boldsymbol{\phi}(\mathbf{x}, t)\rangle_{x,t} = \int_0^T \int_\Omega \boldsymbol{\phi}^*(\mathbf{x}, t) \mathsf{W}(\mathbf{x}) \boldsymbol{q}(\mathbf{x}, t) d\mathbf{x}\, dt. \tag{2.6}$$

The space-time POD modes that solve this optimization are eigenfunctions of the space-time correlation $\mathsf{C}(\mathbf{x}_1, t_1, \mathbf{x}_2, t_2) = \mathbb{E}[\boldsymbol{q}(\mathbf{x}_1, t_1) \boldsymbol{q}^*(\mathbf{x}_2, t_2)]$, and the eigenvalues represent the energy of each mode. The most important property of space-time POD modes for the purpose of this paper is that the space-time POD reconstruction of a trajectory achieves lower error, on average, than the reconstruction with the same number of modes in any other space-time basis. More concretely, using the first r space-time POD modes to reconstruct the trajectory,

$$\tilde{\boldsymbol{q}}(\mathbf{x}, t) = \sum_{j=1}^{r} \boldsymbol{\phi}_j(\mathbf{x}, t) \langle \boldsymbol{q}(\mathbf{x}, t), \boldsymbol{\phi}_j(\mathbf{x}, t)\rangle_{x,t}, \tag{2.7}$$

yields lower expected error

$$\mathbb{E}[\|\tilde{\boldsymbol{q}}(\mathbf{x}, t) - \boldsymbol{q}(\mathbf{x}, t)\|_{x,t}^2] \tag{2.8}$$

than would any other space-time basis. The expected error Eq. (2.8) is measured over space and time using the norm $\|\cdot\|_{x,t}$ induced by the inner product Eq. (2.6).

2.3. Spectral POD

Spectral POD is most easily understood as the frequency domain variant of space-only POD for statistically stationary systems. In other words, SPOD modes at a particular frequency optimally reconstruct (in the same sense as above) the state at that frequency, on average. The property that makes them attractive for model reduction, however, is that SPOD modes are also the long-time limit of space-time POD modes for statistically stationary systems. These ideas are made precise below, but for a more complete discussion, see Ref. [15].

Spectral POD modes at frequency k maximize

$$\lambda_k[\boldsymbol{\psi}(\mathbf{x})] = \frac{\mathbb{E}\left[|\langle \hat{\boldsymbol{q}}_k(\mathbf{x}), \boldsymbol{\psi}(\mathbf{x})\rangle_x|^2\right]}{\|\boldsymbol{\psi}(\mathbf{x})\|_x^2}, \tag{2.9}$$

again subject to the constraint that each mode $\boldsymbol{\psi}_{k,j}(\mathbf{x})$ is orthogonal to the previous ones at that frequency $\boldsymbol{\psi}_{k,i<j}(\mathbf{x})$. The Fourier-transformed state $\hat{\boldsymbol{q}}_k : \Omega \to \mathbb{C}^{N_v}$ is defined as

$$\hat{\boldsymbol{q}}_k(\mathbf{x}) = \int_{-\infty}^{\infty} e^{-i\omega_k t} \boldsymbol{q}(\mathbf{x}, t) dt. \tag{2.10}$$

The solution to the optimization Eq. (2.9) is that the modes are eigenvectors of the cross-spectral density $\mathsf{S}_k(\mathbf{x}_1, \mathbf{x}_2) = \mathbb{E}[\hat{\boldsymbol{q}}_k(\mathbf{x}_1)\hat{\boldsymbol{q}}_k^*(\mathbf{x}_2)]$,

$$\int_{\Omega} \mathsf{S}_k(\mathbf{x}_1, \boldsymbol{x}_2)\mathsf{W}(\mathbf{x}_2)\boldsymbol{\psi}_{k,j}(\mathbf{x}_2) d\mathbf{x}_2 = \lambda_{k,j}\boldsymbol{\psi}_{k,j}(\mathbf{x}_1). \tag{2.11}$$

The eigenvalue is again equal to the energy of the mode, i.e., $\lambda_{k,j} = \lambda_k[\boldsymbol{\psi}_{k,j}]$. Modes at frequency k have an implicit time dependence of $e^{i\omega_k t}$.

SPOD modes and their energies become identical to space-time POD modes as the time interval on which the latter are defined becomes long [12,14,15]. Thus, the SPOD modes with the largest energies among all frequencies are the dominant space-time POD modes (for long times) and are most efficient for reconstructing long-time trajectories. We denote by $\tilde{\lambda}_j$ the j-th largest SPOD eigenvalue among all frequencies, which may be compared to the space-time POD eigenvalues. The convergence in the energy of the trajectory they capture is relatively fast, so for time intervals beyond a few correlation times, the SPOD modes capture nearly as much energy as the space-time modes [14]. If the simulation time of a reduced-order model is long enough for this convergence to be met, the ability of the SPOD modes to capture structures is not diminished relative to that of space-time POD modes.

SPOD modes also have two properties that make them more suitable for model reduction than space-time modes: they have analytic time dependence, and they are separable in space and time. The former makes some analytic progress possible in writing the equations that govern the modes and enables Fourier theory to be applied. The latter means that storing the modes requires N_t times less memory, where N_t is the number of times steps in the simulation.

Fig. 2 shows the convergence in the representational ability of space-time POD and SPOD as the time interval becomes long. Specifically, to represent trajectories with some level of accuracy, 98 %, say, one needs the same number of SPOD coefficients as space-time POD coefficients if the interval is long compared to the correlation time in the system. This convergence in representation ability occurs because the SPOD modes themselves converge to space-time POD modes in the limit of a long time interval. With space-only POD, one must specify the coefficients for every time step, which leads to a far less efficient encoding of the data because the coefficients are highly correlated from one time step to the next. That SPOD modes are near-optimal in representing trajectories, and that they are substantially more efficient than space-only POD modes, motivate this work. If one can efficiently solve for some number of the SPOD coefficients of a trajectory, then these coefficients will lead to substantially lower error than solving for the same number of space-only POD coefficients.

2.4. Discretization and truncation

Upon spatial discretization, the state and modes become vectors in $\mathbb{C}^{N_x}$, where N_x denotes the number of gridpoints multiplied by N_v. Frequency is also discretized, and a finite number N_ω of evenly spaced frequencies is retained. The lowest one, ω_1, induces a time $T = 2\pi/\omega_1$, which determines the interval $[0, T]$ on which the modes are periodic. The trajectories themselves are, of course, not periodic on this interval, so T is the longest we may use SPOD modes for prediction, though, if a longer prediction is needed, the

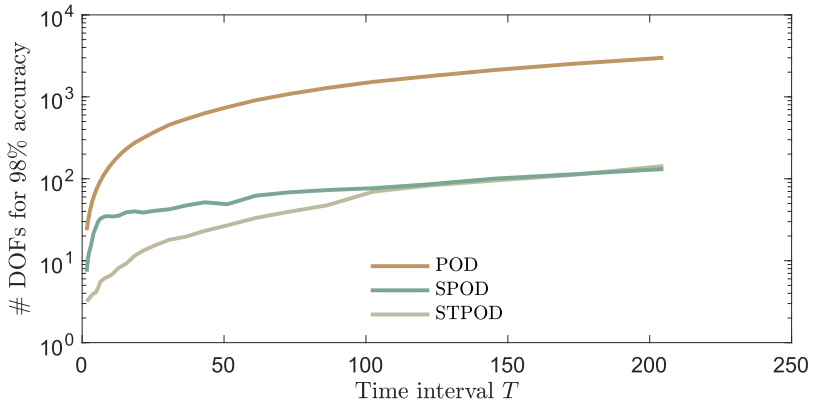

Fig. 2. Number of degrees of freedom (DOFs) required to achieve 98 % representation accuracy of trajectories as a function of the length of the time interval of the trajectory $[0, T]$. For POD, one must specify all the mode coefficients at every time step, whereas spectral and space-time POD modes are themselves time-dependent. Thus, by leveraging spatiotemporal correlations, fewer DOFs are needed to represent a trajectory to a given accuracy by specifying the SPOD or space-time POD coefficients. As the time interval becomes long, the SPOD and space-time POD modes become equally efficient at representing trajectories. The data come from the Ginzburg-Landau system introduced in Section 4, and the details on the comparison of POD and SPOD with the same number of degrees of freedom can be found there.

method may be repeated. The fastest frequency corresponds to a time step Δt. The discrete Fourier transform of a trajectory $\boldsymbol{q}(t)$ at frequency $\omega_k = 2\pi k/T$ is defined as

$$\hat{\boldsymbol{q}}_k = \sum_{j=0}^{N_\omega - 1} \boldsymbol{q}(j\Delta t) e^{-i\omega_k j \Delta t}, \tag{2.12}$$

where $\hat{\boldsymbol{q}}_k$ and $\boldsymbol{q}(j\Delta t)$ are both in $\mathbb{C}^{N_x}$.

With the spatial discretization, integration over space becomes matrix-vector multiplication. For example, the full set of discrete SPOD modes at frequency ω_k are defined as the eigenvectors of the (weighted) cross-spectral density matrix

$$\mathbf{S}_k \mathbf{W} \boldsymbol{\Psi}_{k,\text{full}} = \boldsymbol{\Psi}_{k,\text{full}} \boldsymbol{\Lambda}_{k,\text{full}}. \tag{2.13}$$

Here, $\mathbf{S}_k \in \mathbb{C}^{N_x \times N_x}$ is the cross-spectral density, and $\mathbf{W} \in \mathbb{C}^{N_x \times N_x}$ is a weight matrix. $\boldsymbol{\Psi}_{k,\text{full}} \in \mathbb{C}^{N_x \times N_x}$ is the matrix with the full set discrete SPOD modes as its columns, and $\boldsymbol{\Lambda}_{k,\text{full}}$ is the diagonal matrix of corresponding eigenvalues, which are the SPOD mode energies. However, in practice, the matrix $\mathbf{S}_k$ is not formed, and too little data will be available to create a full-rank set of SPOD modes. Instead, the SPOD modes are calculated by the method of snapshots [15,34] or by using the singular value decomposition. In particular, given r_d trajectories from which to obtain SPOD modes, each N_ω time steps in length, the discrete Fourier transform (DFT) of each trajectory is taken. This yields r_d realizations of the k-th frequency, for every frequency k. These can be formed into a data matrix

$$\mathbf{Q}_k = \left[\hat{\boldsymbol{q}}_k^1, \hat{\boldsymbol{q}}_k^2, \ldots, \hat{\boldsymbol{q}}_k^{r_d}\right], \tag{2.14}$$

where $\hat{\boldsymbol{q}}_k^i \in \mathbb{C}^{N_x}$ is the k-th frequency of the DFT of the i-th trajectory. The SPOD modes at frequency ω_k and the associated energies may then be obtained by first taking the singular value decomposition $\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^* = 1/\sqrt{r_d}\mathbf{W}^{1/2}\mathbf{Q}_k$. The r_d available SPOD modes $\boldsymbol{\Psi}_k^{r_d} \in \mathbb{C}^{N_x \times r_d}$ are then given by $\mathbf{W}^{-1/2}\mathbf{U}$ and the energies $\boldsymbol{\Lambda}_k^{r_d} \in \mathbb{R}^{r_d \times r_d}$ by $\boldsymbol{\Sigma}^2$ [15]. In practice, the r_d trajectories are usually obtained as (possibly overlapping) sub-trajectories of one long trajectory, as is done in Welch's method [35] for computing power spectra.

The set of available SPOD modes at the k-th frequency $\boldsymbol{\Psi}_k^{r_d}$ must be truncated in forming a reduced-order model, however, the order of this truncation should not be the same for all frequencies. Intuitively, allocating more SPOD modes to the energetic frequencies leads to more accuracy than keeping the number of modes constant across frequency. We denote the mean number of modes retained at each frequency as r, thus $N_\omega r$ modes are retained in total. We determine the number of SPOD modes retained at frequency ω_k as the number of modes at this frequency that are among the $N_\omega r$ most energetic overall. That is,

$$r_k = |\{l : \lambda_{k,l} \geq \tilde{\lambda}_{N_\omega r}\}|, \tag{2.15}$$

where $\tilde{\lambda}_i$ denotes the i-th largest eigenvalue over all frequencies. With this notation established, we denote the retained SPOD modes at frequency ω_k as $\boldsymbol{\Psi}_k \in \mathbb{C}^{N_x \times r_k}$, and the corresponding energies as $\boldsymbol{\Lambda}_k \in \mathbb{R}^{r_k \times r_k}$

The trajectory $\boldsymbol{q}(t)$ on the interval $[0, T]$ is then approximated using the retained SPOD modes as

$$\boldsymbol{q}(t) \approx \frac{1}{N_\omega} \sum_{k=0}^{N_\omega - 1} \boldsymbol{\Psi}_k \boldsymbol{a}_k e^{i\omega_k t}. \tag{2.16}$$

The vector of expansion coefficients at frequency ω_k is given by

$$\boldsymbol{a}_k = \boldsymbol{\Psi}_k^* \mathbf{W} \hat{\boldsymbol{q}}_k \in \mathbb{C}^{r_k}. \tag{2.17}$$

3. Spectral solution operator projection method

Our goal is to derive an SPOD-based method to solve the linear ordinary differential equation

$$\dot{\boldsymbol{q}}(t) = \mathbf{A}\boldsymbol{q}(t) + \mathbf{B}\boldsymbol{f}(t), \tag{3.1a}$$

$$\boldsymbol{y}(t) = \mathbf{C}\boldsymbol{q}(t) \tag{3.1b}$$

on the interval $t \in [0, T]$, where $\boldsymbol{q}(t) \in \mathbb{C}^{N_x}$ is the state, $\mathbf{A} \in \mathbb{C}^{N_x \times N_x}$ is the system matrix, $\boldsymbol{f}(t) \in \mathbb{C}^{N_f}$ is some known forcing that is mapped onto the system by the matrix $\mathbf{B} \in \mathbb{C}^{N_x \times N_f}$, and $\boldsymbol{y} \in \mathbb{C}^{N_y}$ is an observable extracted from the state by the matrix $C \in \mathbb{C}^{N_y \times N_x}$. Given the forcing and initial condition $\boldsymbol{q}(0) = \boldsymbol{q}_0$, our goal is to find the retained SPOD coefficients for the trajectory $\boldsymbol{q}(t)$, thereby obtaining the near-optimal rank-$N_\omega r$ space-time representation of the trajectory. With these coefficients, $\boldsymbol{y}(t)$ can be easily obtained taking the inverse DFT of $\hat{\boldsymbol{y}}_k = \mathbf{C}\boldsymbol{\Psi}_k \boldsymbol{a}_k$.

The starting point for finding the retained coefficients is Eq. (2.17), which gives $\boldsymbol{a}_k$ in terms of $\hat{\boldsymbol{q}}_k$. The Fourier-transformed state $\hat{\boldsymbol{q}}_k$ must be obtained from the known forcing and initial condition, and we derive this relation in the following subsection.

3.1. Frequency-domain equation

We begin by inserting the analytic solution to Eq. (3.1a) into the definition of the DFT Eq. (2.12),

$$\hat{\boldsymbol{q}}_k = \sum_{j=0}^{N_\omega - 1} e^{-i\omega_k j \Delta t} \left(e^{\mathbf{A} j \Delta t} \boldsymbol{q}_0 + \int_0^{j\Delta t} e^{\mathbf{A}(j\Delta t - t')} \mathbf{B}\boldsymbol{f}(t')\, dt' \right). \tag{3.2}$$

The term in parentheses is the well-known solution to the linear ODE Eq. (3.1a) [36]. Analytic progress can be made with the assumptions that the forcing can be written as $\boldsymbol{f}(t) = \sum_{k=0}^{N_\omega - 1} \hat{\boldsymbol{f}}_k e^{i\omega_k t}$, and that the matrices $i\omega_k \mathbf{I} - \mathbf{A}$ are invertible for all k. To aid in the ensuing discussion, we split Eq. (3.2) into two parts: the response to the initial condition and the response to the forcing. The first of these is

$$\hat{\boldsymbol{q}}_{k,ic} = \sum_{j=0}^{N_\omega - 1} e^{(\mathbf{A} - i\omega_k \mathbf{I})j\Delta t} \boldsymbol{q}_0. \tag{3.3}$$

This term may be evaluated by noticing that it is a matrix geometric sum, so, it can be written as

$$\hat{\boldsymbol{q}}_{k,ic} = (\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t})^{-1} (\mathbf{I} - e^{(\mathbf{A} - i\omega_k) N_\omega \Delta t}) \boldsymbol{q}_0. \tag{3.4}$$

Because $e^{i\omega_k N_\omega \Delta t} = 1$, this simplifies to

$$\hat{\boldsymbol{q}}_{k,ic} = (\mathbf{I} - e^{(\mathbf{A} - i\omega_k)\Delta t})^{-1} (\mathbf{I} - e^{\mathbf{A}T}) \boldsymbol{q}_0. \tag{3.5}$$

Note that the assumption that $i\omega_k \mathbf{I} - \mathbf{A}$ is invertible implies that $\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t}$ is also invertible.

Second, the response to the forcing is

$$\sum_{j=0}^{N_\omega - 1} e^{-i\omega_k j\Delta t} \int_0^{j\Delta t} e^{\mathbf{A}(j\Delta t - t')} \mathbf{B} \boldsymbol{f}(t')\, dt'. \tag{3.6}$$

To evaluate this term, we first insert the Fourier expansion of the forcing and take the constant matrix exponential out of the integral, giving

$$\hat{\boldsymbol{q}}_{k,force} = \frac{1}{N_\omega} \sum_{j=0}^{N_\omega - 1} e^{-i\omega_k j\Delta t} e^{\mathbf{A} j\Delta t} \int_0^{j\Delta t} \sum_{l=0}^{N_\omega - 1} e^{(i\omega_l \mathbf{I} - \mathbf{A})t'} \mathbf{B} \hat{\boldsymbol{f}}_l \, dt'. \tag{3.7}$$

Integration inverts the matrix in the exponential, and the expression evaluates to

$$\hat{\boldsymbol{q}}_{k,force} = \frac{1}{N_\omega} \sum_{j=0}^{N_\omega - 1} e^{-i\omega_k j\Delta t} \sum_{l=0}^{N_\omega - 1} \mathbf{R}_l \left(e^{i\omega_l j\Delta t} - e^{\mathbf{A} j\Delta t} \right) \mathbf{B} \hat{\boldsymbol{f}}_l. \tag{3.8}$$

We refer to $\mathbf{R}_k = (i\omega_k \mathbf{I} - \mathbf{A})^{-1}$ as the resolvent operator. Eq. (3.8) can then be written as

$$\hat{\boldsymbol{q}}_{k,force} = \frac{1}{N_\omega} \sum_{j=0}^{N_\omega - 1} \sum_{l=0}^{N_\omega - 1} \mathbf{R}_l \left(e^{i(\omega_l - \omega_k)j\Delta t} - e^{(\mathbf{A} - i\omega_k \mathbf{I})j\Delta t} \right) \mathbf{B} \hat{\boldsymbol{f}}_l. \tag{3.9}$$

The frequency difference term in Eq. (3.9) evaluates to zero for $\omega_l \neq \omega_k$, and the other term may be evaluated by the same geometric sum argument as before. The entire forcing term in Eq. (3.2) becomes

$$\hat{\boldsymbol{q}}_{k,force} = \mathbf{R}_k \mathbf{B} \hat{\boldsymbol{f}}_k - \frac{1}{N_\omega} (\mathbf{I} - e^{(\mathbf{A} - i\omega_k)\Delta t})^{-1} (\mathbf{I} - e^{\mathbf{A}T}) \sum_{l=0}^{N_\omega - 1} \mathbf{R}_l \mathbf{B} \hat{\boldsymbol{f}}_l. \tag{3.10}$$

Recombining our simplified expressions for $\hat{\boldsymbol{q}}_{k,ic}$ and $\hat{\boldsymbol{q}}_{k,force}$, we obtain the following equation for the k-th component of the DFT of the state:

$$\hat{\boldsymbol{q}}_k = \mathbf{R}_k \mathbf{B} \hat{\boldsymbol{f}}_k + \left(\mathbf{I} - e^{(\mathbf{A} - i\omega_k)\Delta t} \right)^{-1} \left(\mathbf{I} - e^{\mathbf{A}T} \right) \left(\boldsymbol{q}_0 - \frac{1}{N_\omega} \sum_{l=0}^{N_\omega - 1} \mathbf{R}_l \mathbf{B} \hat{\boldsymbol{f}}_l \right). \tag{3.11}$$

The first term in Eq. (3.11) $\mathbf{R}_k \mathbf{B} \hat{\boldsymbol{f}}_k$, is familiar: it is the component at frequency ω_k of the steady-state response to a periodic forcing. The second term in Eq. (3.11) represents the transient. This transient may persist for the entirety of the interval $[0, T]$, so including it is crucial.

Refs. [26,27] took an expression equivalent to the first term as their starting point for SPOD-based model reduction. Specifically, they began with the equation

$$\mathbf{L}_k \hat{\boldsymbol{q}}_k = \mathbf{B} \hat{\boldsymbol{f}}_k, \tag{3.12}$$

where $\mathbf{L}_k = (i\omega_k \mathbf{I} - \mathbf{A})$ is the inverse of the resolvent operator. We stress that unless the solution $\boldsymbol{q}(t)$ is T-periodic, this relation is incorrect, and this issue was not fully appreciated in either Ref. [26] or [27]. We note that [37] has discussed this issue in another context.

Ref. [26] proceeded to reduce Eq. (3.12) by applying a Galerkin projection with $\boldsymbol{\Psi}_k$ as the trial basis, resulting in the equation $\boldsymbol{a}_k = (\boldsymbol{\Psi}_k^* \mathbf{W} \mathbf{L}_k \boldsymbol{\Psi}_k)^{-1} \mathbf{B} \hat{\boldsymbol{f}}_k$. Even in T-periodic systems, this reduction does not recover the exact SPOD coefficients — the unretained modes influence the retained coefficients. An innovation made Ref. [27] was to use a Petrov-Galerkin projection of Eq. (3.12), resulting in the equation $\boldsymbol{a}_k = (\boldsymbol{\Phi}_k^{F*} \mathbf{W} \mathbf{L}_k \boldsymbol{\Psi}_k)^{-1} \boldsymbol{\Phi}_k^{F*} \mathbf{W} \mathbf{B} \hat{\boldsymbol{f}}$. Ref. [27] showed that the test basis $\boldsymbol{\Phi}_k^F$ can be chosen in such a way that the retained SPOD coefficients are exact (for T-periodic systems). The expression for the test basis $\boldsymbol{\Phi}_k^F$ required to achieve this involved the statistics of the forcing and was written in terms of the singular value decomposition of the product of the resolvent operator and a matrix that contained these forcing statistics.

The T-periodic Petrov-Galerkin method from Ref. [27] may be alternatively derived by left-multiplying the full-order equation $\hat{\boldsymbol{q}}_k = \mathbf{R}_k\mathbf{B}\hat{\boldsymbol{f}}_k$ by $\boldsymbol{\Psi}_k^*\mathbf{W}$, resulting in the following relation for the SPOD coefficients $\boldsymbol{a}_k = \boldsymbol{\Psi}_k^*\mathbf{W}\mathbf{R}_k\mathbf{B}\hat{\boldsymbol{f}}_k$. Since our goal is to arrive at a ROM without the constraint of periodicity, we instead apply the left-multiplication to Eq. (3.11).

While the frequency domain Eq. (3.11) remains valid regardless of the stability of the system as long as the assumptions listed above are met, we do not recommend the method in the case of an unstable system unless $T\lambda_{\max}$ is relatively close to unity. When there is significant exponential growth along the unstable eigenvectors, i.e., $e^{T\lambda_{\max}}$ is large, capturing these directions, and the projection of the forcing onto them is crucial, so eigensystem methods are superior.

3.2. Reduction and operator approximations

As described above, we proceed by left-multiplying Eq. (3.11) according to Eq. (2.17),

$$\boldsymbol{a}_k = \boldsymbol{\Psi}_k^*\mathbf{W}\mathbf{R}_k\mathbf{B}\hat{\boldsymbol{f}}_k + \boldsymbol{\Psi}_k^*\mathbf{W}\left(\mathbf{I} - e^{(\mathbf{A}-i\omega_k\mathbf{I})\Delta t}\right)^{-1}\left(\mathbf{I} - e^{\mathbf{A}T}\right)\left(\boldsymbol{q}_0 - \frac{1}{N_\omega}\sum_{l=0}^{N_\omega-1}\mathbf{R}_l\mathbf{B}\hat{\boldsymbol{f}}_l\right). \tag{3.13}$$

If the system is small, all operators may be computed analytically. However, for large systems, direct computation of, e.g., the inverse that defines the resolvent operator or the matrix exponentials, is not tractable. For these systems, these operators must be approximated.

3.2.1. Steady-state operator

We begin with the operator $\mathbf{M}_k = \boldsymbol{\Psi}_k^*\mathbf{W}\mathbf{R}_k\mathbf{B}$. An accurate and simple-to-implement approximation of $\mathbf{M}_k$ can be obtained by leveraging the availability of data as follows. Defining

$$\boldsymbol{g}_k^i = \mathbf{L}_k\hat{\boldsymbol{q}}_k^i, \tag{3.14}$$

where, again, $\mathbf{L} = (i\omega_k\mathbf{I} - \mathbf{A})$ is the inverse of the resolvent and $\hat{\boldsymbol{q}}_k^i$ is the i-th realization of $\hat{\boldsymbol{q}}_k$ in the training data, we have

$$\hat{\boldsymbol{q}}_k^i = \mathbf{R}_k\boldsymbol{g}_k^i. \tag{3.15}$$

Forming each $\boldsymbol{g}_k^i$ is not computationally expensive so long as $\mathbf{A}$ is sparse. Using the many realizations of the training data (the same ones used to generate the SPOD modes), we have

$$\mathbf{Q}_k = \mathbf{R}_k\mathbf{G}_k, \tag{3.16}$$

where $\mathbf{Q}_k = [\hat{\boldsymbol{q}}_k^1, \hat{\boldsymbol{q}}_k^2, \dots, \hat{\boldsymbol{q}}_k^{r_d}]$ and $\mathbf{G}_k = [\boldsymbol{g}_k^1, \boldsymbol{g}_k^2, \dots, \boldsymbol{g}_k^{r_d}]$. We approximate the resolvent operator $\mathbf{R}_k$ as

$$\mathbf{R}_k \approx \mathbf{Q}_k\mathbf{G}^+, \tag{3.17}$$

where the pseudoinverse is defined $\mathbf{G}^+ = (\mathbf{G}^*\mathbf{W}\mathbf{G})^{-1}\mathbf{G}^*\mathbf{W}$. It may be shown that the action of this approximate resolvent is equivalent to an orthogonal projection into the column space of $\mathbf{G}_k$, followed by a multiplication by the resolvent. In other words, $\mathbf{Q}_k\mathbf{G}^+ = \mathbf{R}_k\mathbf{P}_g$, where $\mathbf{P}_g$ is an orthogonal projection matrix (in the $\mathbf{W}$-based norm). This means that $\mathbf{Q}_k\mathbf{G}^+$ is an effective approximation of $\mathbf{R}_k$ to the extent that the vectors to which it is applied are near the column space of $\mathbf{G}_k$. By inserting this approximation of the resolvent operator into the expression $\mathbf{M}_k = \boldsymbol{\Psi}_k^*\mathbf{W}\mathbf{R}_k\mathbf{B}$, we define the matrix $\mathbf{E}_k$ as

$$\mathbf{M}_k \approx \mathbf{E}_k = \boldsymbol{\Psi}_k^*\mathbf{W}\mathbf{Q}_k\mathbf{G}_k^+\mathbf{B}. \tag{3.18}$$

The total cost of these operations scales linearly with N_x and quadratically with r_d, thus the approximation avoids the superlinear N_x scaling required by most methods for computing the (non-approximate) action of the resolvent operator. We note that if more accuracy is required, time-stepping approaches, such as those developed by [38], may be used.

3.2.2. Transient operator

Next, we approximate the operator $\boldsymbol{\Psi}_k^*\mathbf{W}\left(\mathbf{I} - e^{(\mathbf{A}-i\omega_k\mathbf{I})\Delta t}\right)^{-1}\left(\mathbf{I} - e^{\mathbf{A}T}\right)$. This is accomplished using the r_d available SPOD modes at each frequency. We first multiply by the identity, expressed as $\boldsymbol{\Psi}_{k,\text{full}}\boldsymbol{\Psi}_{k,\text{full}}^*\mathbf{W}$, in various places,

$$\begin{aligned}
&\boldsymbol{\Psi}_k^*\mathbf{W}\left(\mathbf{I} - e^{(\mathbf{A}-i\omega_k\mathbf{I})\Delta t}\right)^{-1}\left(\mathbf{I} - e^{\mathbf{A}T}\right)\\
&\quad= \boldsymbol{\Psi}_k^*\mathbf{W}\left(\mathbf{I} - e^{(\mathbf{A}-i\omega_k\mathbf{I})\Delta t}\right)^{-1}\boldsymbol{\Psi}_{k,\text{full}}\boldsymbol{\Psi}_{k,\text{full}}^*\mathbf{W}\left(\mathbf{I} - e^{\mathbf{A}T}\right)\boldsymbol{\Psi}_{k,\text{full}}\boldsymbol{\Psi}_{k,\text{full}}^*\mathbf{W}. &\text{(3.19a)}\\
&\quad= \mathbf{P}_k\left(\mathbf{I} - e^{(\boldsymbol{\Psi}_{k,\text{full}}^*\mathbf{W}\mathbf{A}\boldsymbol{\Psi}_{k,\text{full}} - i\omega_k\mathbf{I})\Delta t}\right)^{-1}\left(\mathbf{I} - e^{\boldsymbol{\Psi}_{k,\text{full}}^*\mathbf{W}\mathbf{A}\boldsymbol{\Psi}_{k,\text{full}}T}\right)\boldsymbol{\Psi}_{k,\text{full}}^*\mathbf{W}. &\text{(3.19b)}
\end{aligned}$$

The full-rank set of SPOD modes are brought into the matrix exponentials in Eq. (3.19b), and it is straightforward to show that this does not introduce any approximation. In Eq. (3.19b), the matrix $\mathbf{P}_k = \begin{bmatrix}\mathbf{I}_{r_k} & \mathbf{0}\end{bmatrix} \in \mathbb{R}^{r_k\times r_d}$ selects the first r_k rows of the matrix it multiplies. Finally, truncating the operators in Eq. (3.19b), i.e., $\boldsymbol{\Psi}_{k,\text{full}} \to \boldsymbol{\Psi}_k^{r_d}$, and denoting $\tilde{\mathbf{A}} = \boldsymbol{\Psi}_k^{r_d *}\mathbf{W}\mathbf{A}\boldsymbol{\Psi}_k^{r_d}$, the approximated term is

$$\boldsymbol{\Psi}_k^*\mathbf{W}\left(\mathbf{I} - e^{(\mathbf{A}-i\omega_k\mathbf{I})\Delta t}\right)^{-1}\left(\mathbf{I} - e^{\mathbf{A}T}\right) \approx \mathbf{P}_k\left(\mathbf{I} - e^{(\tilde{\mathbf{A}}_k - i\omega_k\mathbf{I})\Delta t}\right)^{-1}\left(\mathbf{I} - e^{\tilde{\mathbf{A}}_k T}\right)\boldsymbol{\Psi}_k^{r_d *}\mathbf{W}. \tag{3.20}$$

As desired, all matrix exponentials and inverses are of size $r_d \times r_d$, which makes them tractable.

The forcing sum in Eq. (3.13), which is difficult to compute directly because it involves resolvents, must also be approximated. In the unreduced equations, this term is the same at each frequency, so the sum over frequencies only needs to be computed once. Any approximation of this term should be the same for each frequency to avoid quadratic scaling in N_ω. The natural choice is to approximate each term by

$$\mathbf{R}_l\mathbf{B}\hat{\boldsymbol{f}}_l \approx \boldsymbol{\Psi}_l\boldsymbol{\Psi}_l^*\mathbf{W}\mathbf{R}_l\mathbf{B}\hat{\boldsymbol{f}}_l. \tag{3.21}$$

This approximation is accurate because the SPOD modes at the l-th frequency are the best basis for $\hat{\boldsymbol{q}}_l$ and are thus a very good basis for $\mathbf{R}_l\mathbf{B}\hat{\boldsymbol{f}}_l$, which is the steady-state component of $\hat{\boldsymbol{q}}_l$. The operator $\boldsymbol{\Psi}_l^*\mathbf{W}\mathbf{R}_l\mathbf{B}$ may again be approximated with $\mathbf{E}_l$.

The equations can, at this point, be written as

$$\boldsymbol{a}_k = \mathbf{E}_k\hat{\boldsymbol{f}}_k + \mathbf{F}_k\left(\boldsymbol{q}_0 - \frac{1}{N_\omega}\sum_l \boldsymbol{\Psi}_l\mathbf{E}_l\hat{\boldsymbol{f}}_l\right), \tag{3.22}$$

where $\mathbf{F}_k = \left(\mathbf{I} - e^{(\tilde{\mathbf{A}}_k - i\omega_k\mathbf{I})\Delta t}\right)^{-1}\left(\mathbf{I} - e^{\tilde{\mathbf{A}}_k T}\right)\boldsymbol{\Psi}_k^*\mathbf{W} \in \mathbb{C}^{r_k\times N_x}$. The operators have been approximated, so far, to avoid $\mathcal{O}(N_x^3)$ scaling in the offline phase of the algorithm. To avoid $\mathcal{O}(N_x)$ scaling in the online phase, one final approximation must be made. The term in parentheses in Eq. (3.22) is multiplied on the left by $\mathbf{F}_k$ for every frequency ω_k, leading to $N_xN_\omega r$ scaling. This can be avoided by storing the term in parentheses in Eq. (3.22) in a rank-p reduced basis $\boldsymbol{\Phi} \in \mathbb{C}^{N_x\times p}$ and precomputing the product of this basis with each $\mathbf{F}_k$. This basis should represent the initial condition and forcing sum terms accurately, and in practice, we choose POD modes of the state. With this approximation, the equations become

$$\boldsymbol{a}_k = \mathbf{E}_k\hat{\boldsymbol{f}}_k + \mathbf{H}_k\left(\boldsymbol{\Phi}^*\mathbf{W}\boldsymbol{q}_0 - \frac{1}{N_\omega}\sum_l \mathbf{T}_l\mathbf{E}_l\hat{\boldsymbol{f}}_l\right), \tag{3.23}$$

where $\mathbf{H}_k = \mathbf{F}_k\boldsymbol{\Phi} \in \mathbb{C}^{r_k\times p}$ and $\mathbf{T}_l = \boldsymbol{\Phi}^*\mathbf{W}\boldsymbol{\Psi}_l \in \mathbb{C}^{p\times r_l}$. Given an initial condition $\boldsymbol{q}_0$ and forcing $\boldsymbol{f}(t)$, the online stage of the method consists of taking the Fourier transform of the forcing, inserting this and the initial condition into Eq. (3.23) in order to get the SPOD coefficients, then transforming back to the time domain. The details are given in the following subsection.

3.3. Formal statement of the algorithm and scaling

The offline and online phases of the SSOP method are shown in Algorithms 1 and 2, respectively.

Algorithm 1 SSOP (offline).

Inputs: $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{W}$, $\{\boldsymbol{\Psi}_i^{r_d}\}$, $\{\boldsymbol{\Lambda}_i^{r_d}\}$,$\{\mathbf{G}_i\}$, $\{r_i\}$, $\boldsymbol{\Phi}$
for $k \in \{1, 2, \dots, N_\omega\}$ **do**
  $\boldsymbol{\Psi}_k \leftarrow [\boldsymbol{\psi}_{k,1}, \dots, \boldsymbol{\psi}_{k,r_k}]$ ▷ Retained SPOD modes
  $\mathbf{L}_k \leftarrow i\omega_k\mathbf{I} - \mathbf{A}$ ▷ Inverse of resolvent
  $\mathbf{E}_k \leftarrow \boldsymbol{\Psi}_k^*\mathbf{W}\mathbf{Q}_k\mathbf{G}_k^+\mathbf{B}$ ▷ Precomputation of first operator
  $\tilde{\mathbf{A}}_k \leftarrow \boldsymbol{\Psi}_k^{r_d *}\mathbf{W}\mathbf{A}\boldsymbol{\Psi}_k^{r_d}$ ▷ Reduced $\mathbf{A}$ to compute matrix exponentials
  $\mathbf{P}_k \leftarrow \begin{bmatrix}\mathbf{I}_{r_k} & \mathbf{0}\end{bmatrix}$ ▷ Row selector matrix
  $\mathbf{H}_k \leftarrow \mathbf{P}_k(\mathbf{I} - e^{(\tilde{\mathbf{A}}_k - i\omega_k\mathbf{I})\Delta t})^{-1}(\mathbf{I} - e^{\tilde{\mathbf{A}}_k T})(\boldsymbol{\Psi}_k^{r_d *}\mathbf{W}\boldsymbol{\Phi})$ ▷ Precomputation of second operator
  $\mathbf{T}_k \leftarrow \boldsymbol{\Phi}^*\mathbf{W}\boldsymbol{\Psi}_k$ ▷ Precomputation of third operator
  $\mathbf{C}_k^{\boldsymbol{\Psi}} \leftarrow \mathbf{C}\boldsymbol{\Psi}_k$ ▷ $\mathbf{C}$ in SPOD basis
end for

Inputs: $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, the system matrices; $\mathbf{W}$, the weight matrix; $\{\boldsymbol{\Psi}_i^{r_d}\}$, the r_d SPOD modes at each frequency; $\{\boldsymbol{\Lambda}_i^{r_d}\}$, the r_d SPOD energies at each frequency; $\{\mathbf{G}_i\}$, the matrix with columns defined in Eq. (3.14); $\{r_i\}$, the number of modes to be kept at each frequency; $\boldsymbol{\Phi}$, the basis for reducing the initial condition and forcing terms.
Outputs: $\{\mathbf{E}_i\}$, $\{\mathbf{H}_i\}$, $\{\mathbf{T}_i\}$, the operators in Eq. (3.23) for each frequency; $\{\mathbf{C}_i^{\boldsymbol{\Psi}}\}$, the operators that map the SPOD mode coefficients to $\boldsymbol{y}$ for each frequency.

So long as the offline time is feasible, which we show next, the online scaling is the salient cost. In calculating the online cost, we count the operations necessary to go from the time domain forcing and initial condition to the SPOD coefficients. In practice, $\boldsymbol{y}$ is likely small, thus multiplying the SPOD coefficients by $\mathbf{C}\boldsymbol{\Psi}_k$ and taking the inverse DFT contributes insignificantly to the scaling. The complexity of the online algorithm is

$$\mathcal{O}((rp + rN_f + N_f\log N_\omega)N_\omega). \tag{3.24}$$

This time should be compared with the complexity of a POD-Galerkin model, which is $\mathcal{O}((rN_f + r^2)N_t)$. The two are similar, and the differences are due to how p compares with r, how N_t compares with N_ω, and the constants involved. In our numerical experiments, we find that the SPOD method is slightly faster for equal rank (but much more accurate). In Appendix B we detail a further approximation that removes the N_f scaling from Eq. (3.24) by employing the discrete empirical interpolation method (DEIM) [39] to approximate the forcing, and other N_x-tall vectors, via sparse samplings. This can lead to significant speed-up in cases where N_f is large.

Algorithm 2 SSOP (online).

Input parameters: $\boldsymbol{q}_0$, $\boldsymbol{f}$, $\mathbf{W}$, $\{\boldsymbol{\Psi}_i\}$, $\boldsymbol{\Phi}$, $\{\mathbf{E}_i\}$, $\{\mathbf{H}_i\}$, $\{\mathbf{T}_i\}$, $\{\mathbf{C}_i^{\boldsymbol{\Psi}}\}$
$\hat{\boldsymbol{f}} \leftarrow \mathtt{FFT}(\boldsymbol{f})$ ▷ FFT of forcing
$\boldsymbol{a}_0^{\boldsymbol{\Phi}} = \boldsymbol{\Phi}^* \mathbf{W} \boldsymbol{q}_0$ ▷ Reduced initial condition
$\tilde{\boldsymbol{a}}_0^{\boldsymbol{\Phi}} \leftarrow \mathbf{0}_{p\times 1}$ ▷ Initializing forcing sum term
for $k \in \{1, 2, \dots, N_\omega\}$ **do**
  $\boldsymbol{b}_k \leftarrow \mathbf{E}_k \hat{\boldsymbol{f}}_k$
  $\tilde{\boldsymbol{a}}_0^{\boldsymbol{\Phi}} \leftarrow \tilde{\boldsymbol{a}}_0^{\boldsymbol{\Phi}} + \frac{1}{N_\omega} \mathbf{T}_k \boldsymbol{b}_k$ ▷ Forcing sum
end for
for $k \in \{1, 2, \dots, N_\omega\}$ **do**
  $\boldsymbol{a}_k \leftarrow \boldsymbol{b}_k + \mathbf{H}_k(\boldsymbol{a}_0^{\boldsymbol{\Phi}} - \tilde{\boldsymbol{a}}_0^{\boldsymbol{\Phi}})$ ▷ Assigning SPOD coefficients
  $\hat{\boldsymbol{y}}_k \leftarrow \mathbf{C}_k^{\boldsymbol{\Psi}} \boldsymbol{a}_k$ ▷ Constructing observable in frequency domain
end for
$\boldsymbol{y} \leftarrow \mathtt{IFFT}(\hat{\boldsymbol{y}})$ ▷ Observable in time domain

Inputs: $\boldsymbol{q}_0$, the initial condition; $\boldsymbol{f}$, the forcing as a function of time; $\mathbf{W}$, the weight matrix; $\{\boldsymbol{\Psi}_i\}$, the retained SPOD modes for each frequency; $\boldsymbol{\Phi}$, the basis for reducing the initial condition and forcing terms; $\{\mathbf{E}_i\}$, $\{\mathbf{H}_i\}$, $\{\mathbf{T}_i\}$, the operators in Eq. (3.23) for each frequency; $\{\mathbf{C}_i^{\boldsymbol{\Psi}}\}$, the operators that map the SPOD mode coefficients to $\boldsymbol{y}$ for each frequency.
Output: $\boldsymbol{y}$, the observable in the time domain.

In Algorithm 1, we have assumed that the SPOD modes for the $\hat{\boldsymbol{q}}$ and $\hat{\boldsymbol{g}}$ have already been obtained. These modes can be obtained using the techniques described in, e.g., Refs. [15,40], and the cost for this step is $\mathcal{O}(r_d^2 N_x N_\omega)$, where again r_d is the number of SPOD modes obtained from the data, which is the same as the number of temporal blocks formed in Welch's algorithm. If evaluation of $\mathbf{A}\boldsymbol{q}$ scales linearly with N_x (i.e., $\mathbf{A}$ is sparse), then the offline scaling is $\mathcal{O}(r_d^2 N_x N_\omega)$. If the evaluation of $\mathbf{A}\boldsymbol{q}$ scales quadratically with N_x, then the offline scaling is $\mathcal{O}(r_d^2 N_x N_\omega + r_d N_x^2 N_\omega)$.

3.4. Data-free method

Here, we briefly outline a data-free version of the method. Ref. [15] established a connection between SPOD modes, which come from data, and resolvent modes, which come directly from the system matrices. We first define the singular value decomposition

$$\mathbf{X}_q \mathbf{R}_k \mathbf{B} \mathbf{X}_f^{-1} = \tilde{\mathbf{U}}_k \boldsymbol{\Sigma}_k \tilde{\mathbf{V}}_k^*. \tag{3.25}$$

Here, $\mathbf{X}_q$ is the Cholesky factor of the weight matrix that defines the energy of the state, and $\mathbf{X}_f$ is the Cholesky factor of the weight matrix that defines the norm of the forcing, i.e., $\mathbf{W} = \mathbf{X}_q^* \mathbf{X}_q$, $\mathbf{W}_f = \mathbf{L}_f^* \mathbf{L}_f$ and $\|\boldsymbol{f}\|^2 = \boldsymbol{f}^* \mathbf{W}_f \boldsymbol{f}$. The resolvent response modes at frequency ω_k are then defined as $\mathbf{U}_k = \mathbf{X}_q^{-1} \tilde{\mathbf{U}}_k$. The relation between these modes and the SPOD modes is the following: if the forcing at frequency ω_k is white in space with respect to the forcing norm, i.e., $\mathbb{E}[\hat{\mathbf{f}}_k \hat{\mathbf{f}}_k^*] = \alpha \mathbf{W}_f^{-1}$ for some constant α, then the SPOD modes are equivalent to the resolvent modes,

$$\boldsymbol{\Psi}_k = \mathbf{U}_k, \tag{3.26}$$

and the SPOD energies are proportional to the square singular values,

$$\boldsymbol{\Lambda}_k = \alpha^2 \boldsymbol{\Sigma}_k^2. \tag{3.27}$$

Formally, this equivalence holds only if the forcing is white; however it has been demonstrated extensively in the fluid mechanics literature that the two sets of modes are often remarkably similar even when the forcing is far from white [16–20]. This can be leveraged to formulate a data-free ROM by substituting the resolvent modes for the SPOD modes at each frequency. We refer to this version of the method as resolvent SSOP.

The scaling for a straightforward computation of the inverse that defines the resolvent, and the matrix exponential and inverse in Eq. (3.13) is cubic in N_x. However, data-free methods [38] exist that can be used to drastically reduce this scaling, as is required for large systems.

The natural point of comparison for the resolvent SSOP method is balanced truncation, which is also data-free. We make this comparison in Section 4.1, finding that at most parameter values, resolvent SSOP is more accurate. Balanced truncation shares the worst-case offline scaling, and approximations that reduce the scaling exist as well [41,42].

4. Examples

Here, we demonstrate the proposed method on two examples: a linearized Ginzburg-Landau problem and a scalar transport problem. The former is a dense system of dimension $N_x = 220$, and the latter is a sparse system of dimension $N_x = 9604$. For the Ginzburg-Landau system, we compare the accuracy and cost of the proposed method to those of POD-Galerkin projection and a statistics-enhanced version of balanced truncation. The error is roughly two orders of magnitude lower for the proposed method than

the other two (depending on the case), and the CPU time is similar for all methods. For the scalar transport case, the offline time for balanced truncation makes the method (in its unapproximated form) infeasible, so we compare only to a POD-Galerkin model. The proposed method is again orders of magnitude more accurate at similar CPU cost.

The SSOP method with an average of r modes at each frequency is compared to the two time-domain methods with r modes. In both time-domain methods, we use an integrator with an adaptive time step, but calculate the error at the temporal grid corresponding to the fastest frequency used in the SSOP method (which was longer than the time steps taken by the integrators in both examples). This means that the time-domain methods, as well as the proposed SSOP method, use rN_ω degrees of freedom to represent the solution over the interval. We will show that the SSOP and the two time-domain methods at the same r are comparable in terms of CPU time, so the comparison that ultimately is the most meaningful is the accuracy of the methods at the same r.

4.1. Linearized Ginzburg-Landau problem

In continuous space, the complex linearized Ginzburg-Landau equation is

$$\dot{q}(x,t) = \mathcal{A}q(x,t) + f(x,t), \tag{4.1}$$

where

$$\mathcal{A} = -\nu\frac{\partial}{\partial x} + \gamma\frac{\partial^2}{\partial x^2} + \mu(x), \tag{4.2}$$

and $f(x,t)$ is a forcing. Following Ref. [43], we set $\nu = 2 + 0.4i$ and $\gamma = 1 - i$. The parameter $\mu(x)$ takes the form

$$\mu(x) = (\mu_0 - c_\mu^2) + \frac{\mu_2}{2}x^2, \tag{4.3}$$

with $c_\mu = 0.2$, $\mu_2 = -0.01$ [43]. The parameter μ_0 is a bifurcation parameter; the linearized system transitions from global stability to global instability when μ_0 exceeds 0.397. We set $\mu_0 = 0.229$ [15] for the majority of our numerical experiments and later explore the effectiveness of the ROMs as μ_0 varies from 0.079 to 0.379. The system can be interpreted as an advection-diffusion equation with a local exponential growth term. The equation supports traveling wave behavior in the positive x-direction and is stable in the sense that all the eigenvalues of the linear operator (discretized or continuous) are negative, so all solutions to the unforced equations decay asymptotically. Whether the exponential term promotes local growth or local decay depends on the sign of $\mu(x)$. With the parameters used, $\mu(x)$ is positive when $x \in [-6.15, 6.15]$ and negative elsewhere, so as waves move through this region, they grow substantially before decaying once again after passing through it.

When the equation is discretized in space, it takes the general form in (3.1). Following Refs. [43,44], we use a pseudo-spectral Hermite discretization with $N_x = 220$ collocation points [15], and solve the discretized equations using MATLAB's `ode45`, an explicit

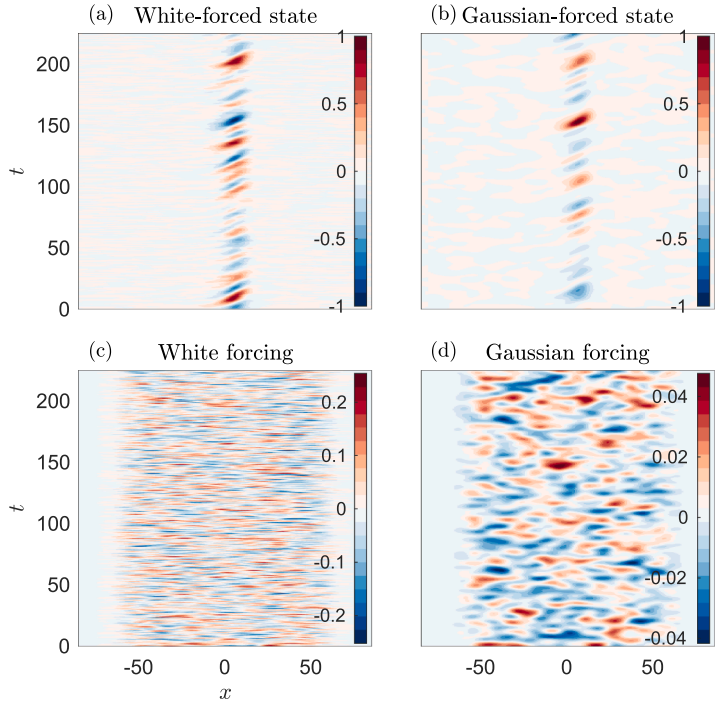

Fig. 3. Ginzburg-Landau state and forcing trajectories: (a) the state resulting from the white forcing in (c); (b) the state resulting from the Gaussian forcing in (d). Both forcings have the same spatial correlation, but the short temporal correlation in the white forcing leads to more jagged structures in the corresponding state. Both trajectories consist of waves traveling in the positive x-direction that are amplified in a region near $x = 0$.

Runge-Kutta (4,5) integrator. Following Ref. [15], training data to compute the SPOD modes is generated from a single long run of the forced system comprising a short transient followed by 12000 time steps of data with $\Delta t = 0.2$. The transient is discarded so that the modes are independent of the initial condition used in the run, and the 12000 time steps are segmented into $r_d = 142$ overlapping blocks, each of length $N_\omega = 1024$ time steps.

Fig. 3(a,b) shows two space-time trajectories of the state q, each 1024 time steps in length. The diagonally oriented structures demonstrate the traveling wave behavior of the system, and it is clear that the waves are amplified and then attenuated as they pass through $x = 0$. These space-time trajectories are to space-time POD (and thus SPOD) as snapshots are to POD: the more structure there is in the trajectories, the fewer space-time modes are needed to accurately represent them. For example, in Fig. 3(a) the state is forced with band-limited temporally white noise and a Gaussian spatial correlation while the state in Fig. 3(b) is forced with a temporally, as well as spatially, Gaussian noise. The resulting state from the white forcing has more detailed structures – a good proxy for higher rank behavior – so trajectories with this forcing require more space-time modes to be accurately represented. The temporally white and temporally Gaussian forcings are shown in Fig. 3(c,d). Note that the forcing occupies the entire domain in this example, i.e., $N_f = N_x$. We thus choose $p = N_x$ vectors in the intermediary basis used to reduce the transient operator, because this will not have a large impact on the CPU time in this case. The SPOD method, therefore, scales like N_x, as do POD-Galerkin projection and balanced truncation, and there is no scaling benefit gained by using an intermediary basis, so we set it to the identity in this example. Despite this spatially extensive forcing, all three ROMs maintain a significant advantage over the FOM due to the latter being dense, resulting from the pseudo-spectral discretization.

Fig. 4 illustrates various features of the mode energies. The top panel shows the SPOD mode energies λ as a function of ω. Each curve is a particular mode number as a function of frequency. The decision to retain $N_\omega r$ total modes in the ROM selects an energy threshold below which modes are not retained (indicated by the dashed line in the figure). After ordering the energies of all mode numbers at all frequencies, the threshold is given by the $N_\omega r$-th energy $\tilde{\lambda}_{N_\omega r}$. At frequencies where a given mode number is above (red) the energy threshold (dashed), it is retained. Where it is below (blue) the energy threshold, it is truncated. The green curve shows the number of modes that meet or exceed this threshold as a function of ω. For example, at the dominant frequency in the white noise case, 22 modes are retained, whereas at the highest and lowest frequencies, only 3 are retained. The bottom panel shows

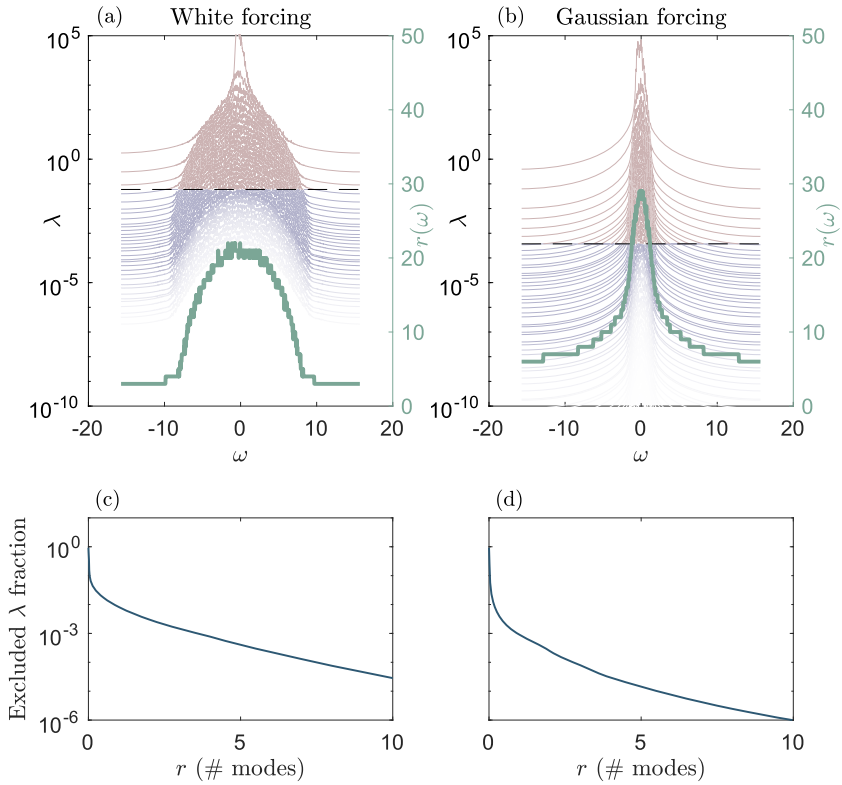

Fig. 4. SPOD mode energies: (a-b, left axis) energy λ of the retained and unretained modes. The top red curve is the energy of the first SPOD mode as a function of frequency ω. The lower red and blue curves are the energies of the lower mode numbers, as functions of frequency. The retained modes (red) are the overall highest-energy modes, and the threshold (dashed) is determined as the energy of the $N_\omega r = 10240$-th most energetic mode; (a-b, right axis) number of modes that clear the threshold as a function of frequency. (c-d): the fraction of excluded energy (see Eq. (4.4)) as a function of r, the average number of modes per frequency. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the fraction of energy that is excluded depending on the number of modes retained, as given by the formula

$$\frac{\sum_{k=0}^{N_\omega-1}\sum_{j=r_k+1}^{N_x}\lambda_{kj}}{\sum_{k=0}^{N_\omega-1}\sum_{j=1}^{N_x}\lambda_{kj}}. \tag{4.4}$$

This quantity represents the fraction of the energy in the training data that is not representable by the retained modes. It depends on r, the average number of modes retained per frequency, and as r is increased the excluded energy decreases as the SPOD basis at each frequency is enriched. For the Gaussian forcing case, the excluded energy fraction drops more quickly initially, indicating that the state is more accurately represented with a given number of SPOD modes in the Gaussian case relative to the white case.

The test data comprise 173 trajectories, again with $\Delta t = 0.2$ and with each trajectory of length 1024 time steps. These test trajectories are generated as sub-trajectories from a long run, and the initial conditions and forcings used are not present in the training data. For each ROM, we calculate the error at the 1024 points for each trajectory as the square norm of the difference with the FOM solution. Throughout this section, we compare the performance of the proposed SSOP method to that of POD-Galerkin projection and an enhanced version of balanced truncation. The reader is referred to Ref. [45] for a description of POD-Galerkin projection. We use the MATLAB function `balreal` to generate the reduced system matrices and basis for balanced truncation, which is based on Ref. [4,46]. In its usual form, balanced truncation does not make use of (i.e., does not require) data or knowledge of the statistics of the problem it is applied to. It may be improved with this information by 'whitening' the forcing, i.e., transforming the system to one where the forcing is spatially white before performing the usual balanced truncation algorithm. In this application, where the forcing is far from spatially white, we observe that this variant of balanced truncation substantially outperforms the standard version, so we use it as a benchmark along with POD-Galerkin projection. We solve the reduced equations with the MATLAB function `ode45` for both methods, and run both the FOM and all ROMs using six cores of an Intel Xenon 6128 processor. The cost of building the ROM (with $r = 10$) was 0.7 % of the cost generating the FOM data.

Fig. 5 shows the three ROM approximations of a trajectory, along with the errors in these approximations. All ROMs here use $r = 2$ modes, and the error field shown here is the absolute value of the difference between the FOM and ROM trajectories, i.e., $|\tilde{q}(x,t) - q(x,t)|$ where $\tilde{q}(x,t)$ is the ROM result. The enhanced balanced truncation produces a better result than POD-Galerkin projection, but the error of the SSOP method is much lower than both benchmarks.

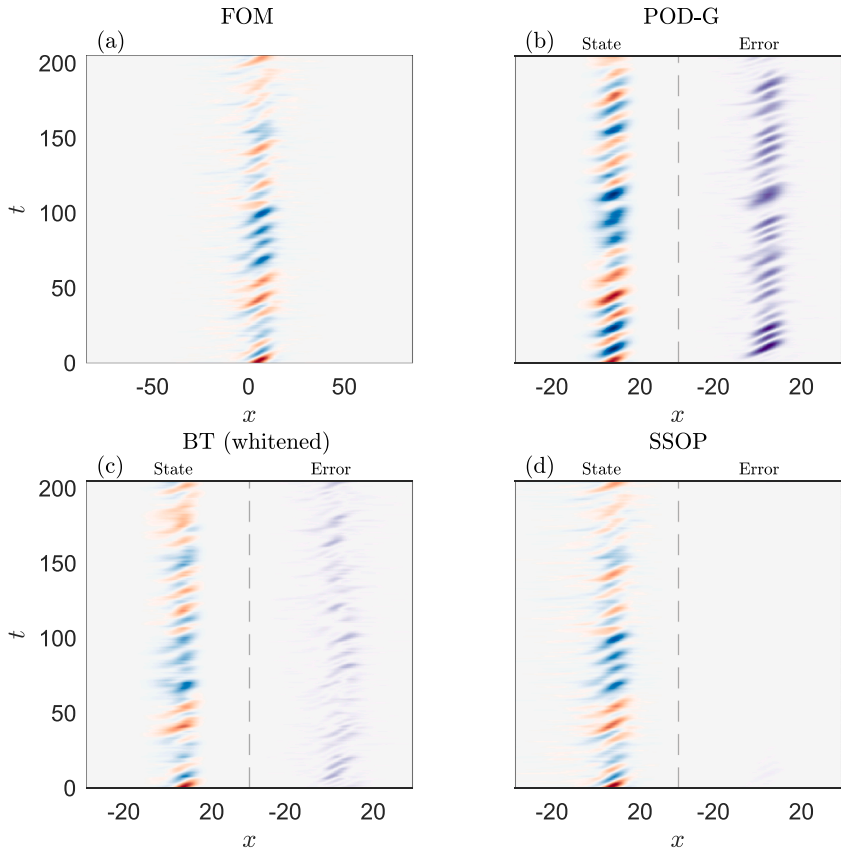

Fig. 5. FOM trajectory (a) and ROM predictions thereof (b-d) along with the errors for the Ginzburg-Landau system. The error fields shown are the absolute value of the difference of the FOM and ROM trajectories. The peak error value (and the upper limit on the error color scale) is 87 % of the peak absolute value of the state.

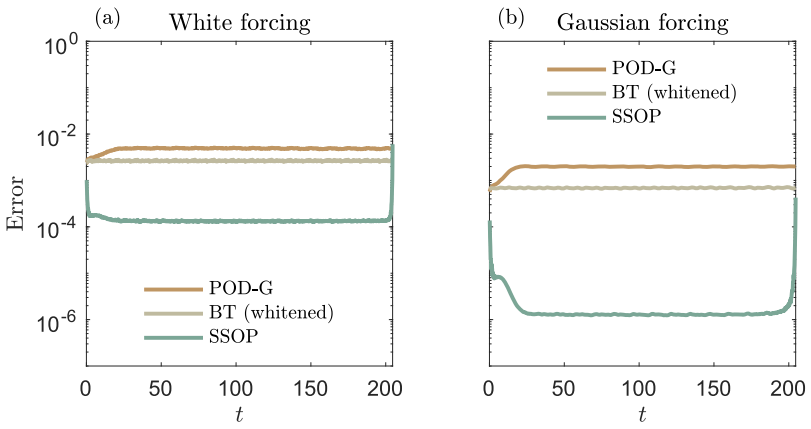

Fig. 6. Error with $r = 10$ modes relative to the FOM for the Ginzburg-Landau system, averaged over 173 trajectories. The large difference in accuracy is due, in large part, to the ability of the SPOD modes to represent trajectories more accurately than space-only modes. This difference is larger in the Gaussian forcing case.

Hereafter, we compute the error as a function of time as the square norm of the difference of the ROM and FOM solutions, averaged over the test trajectories and normalized by the mean square norm of the FOM solution, i.e.,

$$e(t) = \frac{\sum_{i=1}^{N_{\text{test}}} \|\tilde{\boldsymbol{q}}^i(t) - \boldsymbol{q}^i(t)\|_x^2}{\frac{1}{T}\sum_{i=1}^{N_{\text{test}}} \int_0^T \|\boldsymbol{q}^i(t')\|_x^2 \, \mathrm{d}t'}. \tag{4.5}$$

We also report the mean of this quantity over time.

Fig. 6 shows the error, defined in Eq. (4.5), for the various ROMs. For the POD-Galerkin and enhanced balanced truncation models, 10 spatial modes are used for each of the 1024 time steps. For the SSOP method, the $10 \times 1024 = 10240$ most energetic SPOD modes are used, and are distributed over the frequencies as shown in Fig. 4. Most notably, the error is nearly two orders of magnitude smaller for the SSOP method than it is for the other two methods. Given the analysis in the previous sections, this is not surprising: the SPOD modes are (nearly) optimal in that the representation error with some number of SPOD modes is smaller than (nearly) every other space-time basis. Again, this representation is recovered by the SSOP method up to the errors introduced from approximating the operators and the non-periodicity of the forcing on the temporal interval. The error from all methods is larger in the white-noise forcing case. This is to be expected because the resulting behavior of the state is higher rank in this case relative to the Gaussian forcing. The error of the SSOP method decreases by more in the Gaussian forcing case because it takes explicit advantage of the additional spatiotemporal coherence relative to the white forcing case.

Next, we investigate the dependence of the error on the number of modes retained. Fig. 7 shows, as one might expect, that the error in all methods decreases with the number of modes retained. The gulf between the SSOP method and the two space-only ROMs is roughly maintained over the range of modes shown for both the white and Gaussian forcing cases. The dashed lines are the projection of the full-order solution onto each respective set of modes, which we refer to as the representation error. For example,

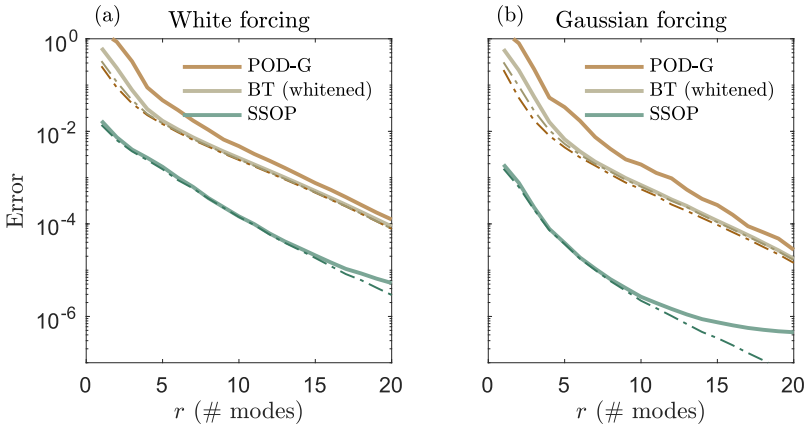

Fig. 7. Error as a function of the number of modes for the Ginzburg-Landau system. The values reported here are the time averages of the error defined in Eq. (4.5).

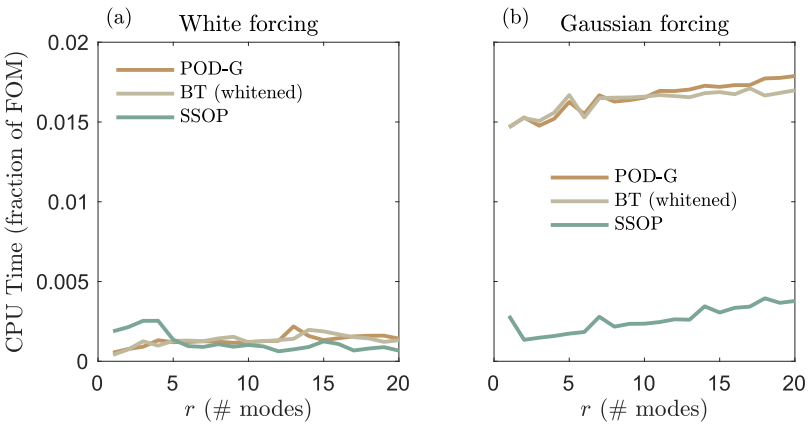

Fig. 8. Average CPU time as a function of the number modes for the Ginzburg-Landau system. The values are normalized by the CPU time of the FOM.

the dashed green line is the SPOD mode representation error, i.e., the error of the FOM solution projected into the span of the SPOD modes. We emphasize that the motivation for this work is the fact that the SPOD representation error is substantially below the POD representation error. The SSOP solution error and representation error are nearly identical before the accuracy of the former is limited by the full-order frequency domain error at around 16 modes and 10^{-6} error. Until this point, the SSOP method indeed achieves the lowest error possible using SPOD modes, which is nearly the lowest error with any set of space-time modes. The SSOP solution error is not only lower than the POD-Galerkin and enhanced balanced truncation solution errors, but also the respective representation errors of these bases. The POD representation error is a lower bound for the error for any time-domain Petrov-Galerkin method, such as balanced truncation or least-squares Petrov-Galerkin [7], because this is precisely the quantity that POD modes minimize. Indeed, the balanced truncation error is within this bound. We view the fact that the SSOP solution error is significantly below the POD representation error as one of the major achievements of this work.

In Fig. 8, we show the CPU time as a function of the number of modes retained. All values are reported as a fraction of the FOM time of 3543 seconds in total for the 173 runs. The SSOP CPU time scales linearly with the number of modes retained, but here, there are too few modes to see this scaling. Nonetheless, the SSOP method is substantially faster than the two benchmarks and runs in roughly two thousandths of the FOM time. This time includes the time to take the Fourier transform of the forcing and the inverse Fourier transform of the response. The times for the white forcing cases and Gaussian forcings are comparable for the SSOP method, but the FOM takes substantially longer with the white forcing.

Thus far, we have tested the method on a system with the same statistics as the system on which the model was trained. In many applications, this assumption is too generous; a model trained on one system may be used to predict the behavior of a different system. We test the robustness of the method by training it on the Ginzburg-Landau system where the bifurcation parameter is $\mu_0 = 0.229$, then testing it on a range of Ginzburg-Landau systems with $\mu_0 \in [0.079, 0.379]$ with increments of 0.03. Physically, this range corresponds to shifting from modal behavior at the lower end to strongly non-modal behavior at the upper end [43]. The training system ($\mu_0 = 0.229$) is the center of this range, but its behavior is more similar to systems at the lower end than the higher end; one metric for this is the optimal transient growth [47,48], which is just above 1 for $\mu_0 = 0.079$, approximately 5 for $\mu_0 = 0.229$, and nearly 200 for $\mu_0 = 0.379$. We compare the performance of the SSOP model on the out-of-sample data to that of the POD-Galerkin model. For both, using training data from one system consists of simulating that system and obtaining the SPOD or POD modes from the data. To build the model on the test system, the ROM operators are computed with the modes from the training system and the $\mathbf{A}$ and $\mathbf{B}$ operators from the test system. We also compare results to the enhanced balanced truncation, which does not use data but does use the statistics of the forcing.

Before considering out-of-sample conditions, in Fig. 9(a), we first show the performance where the ROMs are trained and tested on the same system, i.e., μ_0 is the same. The error is computed with $r = 10$ for all models and is shown for the range of μ_0. The dashed lines once again denote the projection of the FOM solution onto the respective modes; they are a lower bound for the respective methods. SSOP maintains its substantial accuracy superiority relative to the two baseline methods and to the POD projection over the range of μ_0. Both the balanced truncation and SSOP models take advantage of the lower rank behavior at the high μ_0 values, while the POD-Galerkin model is slightly less accurate due to the strong non-normality of $\mathbf{A}$ at high μ_0.

Fig. 9(b) shows the results for the same range of test systems, but where the POD-Galerkin and SSOP models use modes from $\mu_0 = 0.229$. The enhanced balanced truncation is the same between (a) and (b) because it only uses the statistics of the forcing, not the state, which are the same between the systems. Most notably, the SSOP model delivers low error over the range of test systems when it is trained on $\mu_0 = 0.229$. Even at high μ_0, where the system is dominated by non-modal mechanisms that are far less prevalent in the training system [43], the SSOP method is remarkably accurate. In fact, the SSOP method trained on the 'wrong' data ((b), solid green) is more accurate than the projection of the POD modes from the 'right' data ((a), dashed brown). That the system at, e.g., $\mu_0 = 0.379$ is substantially different from the one at $\mu_0 = 0.229$ may be seen by comparing the projection errors (dashed) for the POD and SPOD modes in (a) and (b) at $\mu_0 = 0.379$. For example, the error in the projection of the $\mu_0 = 0.379$ system onto the $\mu_0 = 0.379$ POD modes ((a), dashed brown) is an order of magnitude lower than the error in the projection of the $\mu_0 = 0.379$ system onto the

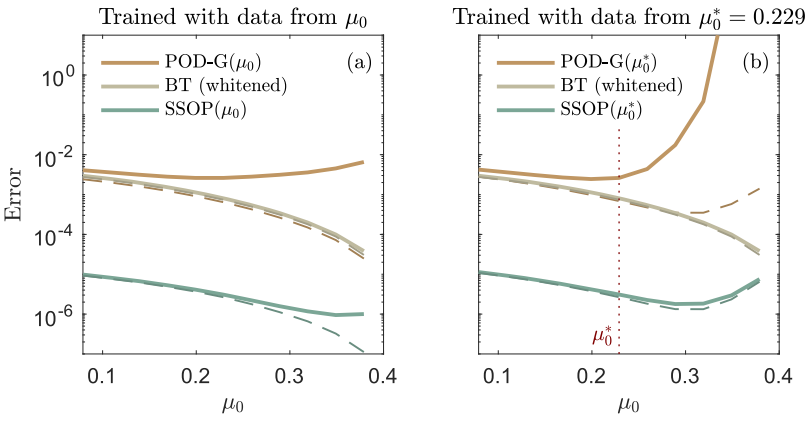

Fig. 9. Data-driven ROM performance across a range of Ginzburg-Landau systems parameterized by μ_0: (a) Models are trained and tested on the same system for a range of μ_0; (b) Models are trained on a system with $\mu_0 = 0.229$ and tested on a range of μ_0. All models use $r = 10$. The enhanced version of balanced truncation in this figure does not use data to obtain its test and trial bases, so the BT results are the same in (a) and (b). It is included among the data-driven results because it uses the forcing statistics to whiten the system. All values of μ_0 shown are smaller than the critical value of $\mu_0 = 0.397$.

$\mu_0 = 0.229$ POD modes ((b), dashed brown). The same can be said of the SPOD projection error between the two. The difference in the systems may also be seen in the fact that the POD-Galerkin model built from the $\mu_0 = 0.229$ POD modes is unstable at $\mu_0 = 0.379$ ((b), solid brown).

Next, we investigate the robustness of the model to out-of-sample forcings using the following two tests. First, we train the model with $r = 5$ on the data generated by the white forcing and test it on the Gaussian forcing. We compare the statistics of the model output over 142 blocks, namely the variance as a function of x and the power spectral density as a function of ω. Second, we use the same white-forcing-trained model to predict trajectories for four different non-stationary forcings. Unlike the white and Gaussian forcings, which are active in the entire domain, these non-stationary forcings are only nonzero in the interval $x \in [-12, 12]$. Within this region, the forcings are given by $f(x,t) = T(t)X(x)$ where $X(x)$ is a Gaussian bump centered at $x = -10$. We test four choices of $T(t)$.

Fig. 10 shows the results of the first test. The FOM variance, i.e., the turbulent kinetic energy, is shown in panel (a) and the errors in the ROM predictions thereof are shown in panel (c). Likewise, the FOM power spectral density is shown in panel (b) and the errors in the ROM PSDs are shown in panel (d). Both the mean energy and the PSD are normalized such that their peak is 1, and both errors are computed as the absolute value of the difference between the FOM and ROM statistics. For both quantities, the proposed method produces more accurate statistics for the out-of sample forcings than does either POD-Galerkin projection or balanced truncation. We also note that the error in the SSOP prediction of the state for this test is 84 and 20 times lower than the POD-Galerkin and balanced truncation predictions, respectively. Fig. 10 shows the results of the first test. The FOM variance, i.e., the turbulent kinetic energy, is shown in panel (a) and the errors in the ROM predictions thereof are shown in panel (c). Likewise, the FOM power spectral density is

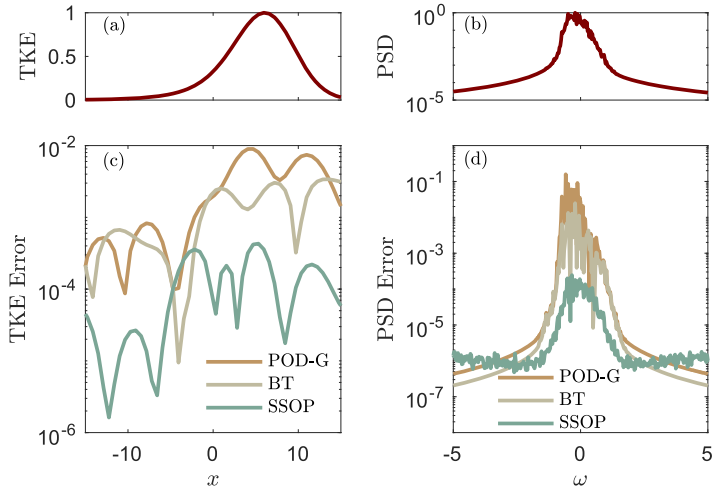

Fig. 10. Error in the prediction of statistics for the Ginzburg-Landau system: (a) the FOM variance (TKE) as a function of x; (b) the power spectral density (PSD); (c) the ROM errors in predictions of the TKE; (d) the ROM errors in predictions of the PSD. The ROMs are built using data from the white-noise forcing.

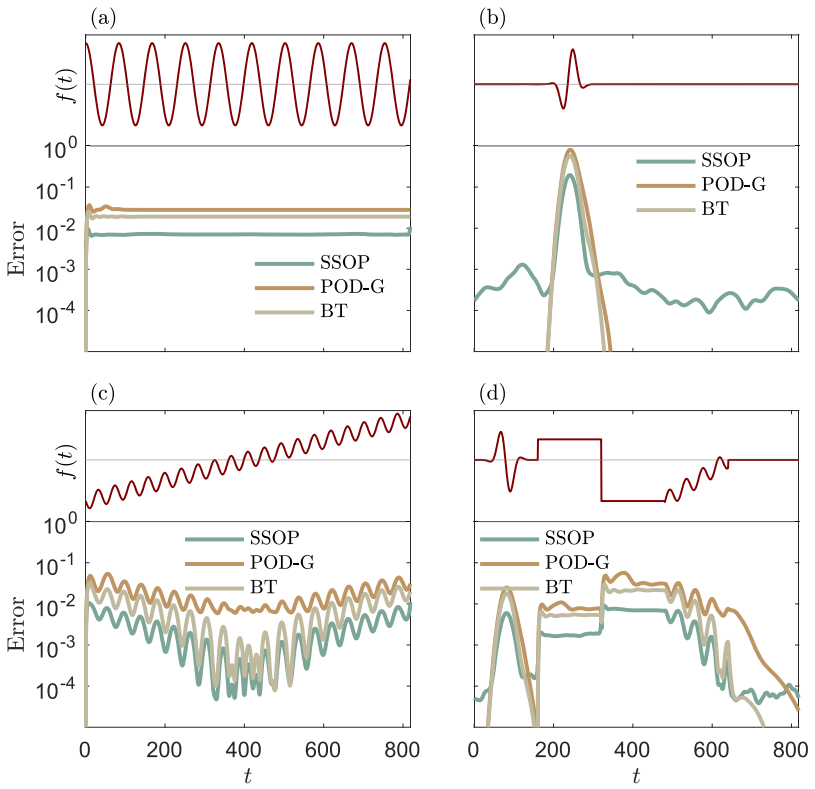

Fig. 11. Results of the models trained with the white forcing on four out-of-sample non-stationary forcings to the Ginzburg-Landau system. The time-dependence of these forcings is shown in the top panel, and the resulting error for the three ROMs in the bottom panel. Respectively, the mean errors for the SSOP, POD-Galerkin, and balanced truncation models are (a) .007, .03, .02; (b) .008, .03, .02; (c) .002, .02, .007; (d) .002, .01, .007.

shown in panel (b) and the errors in the ROM PSDs are shown in panel (d). Both the mean energy and the PSD are normalized such that their peak is 1, and both errors are computed as the absolute value of the difference between the FOM and ROM statistics. For both quantities, the proposed method produces more accurate statistics for the out-of sample forcings than does either POD-Galerkin projection or balanced truncation. We also note that the error in the SSOP prediction of the state for this test is 84 and 20 times lower than the POD-Galerkin and balanced truncation predictions, respectively.

Fig. 11 shows the four non-stationary forcings, along with the error in the ROM predictions of the state for each. In each case, the errors are normalized by the mean square energy of the solution over the interval. The SSOP error is lower than that of the other two methods for all tests, but the difference is smaller than in previous tests. This likely may be attributed to the fact that the support of the forcing is quite different than in the training data for the model. Nevertheless, the method gives relatively accurate predictions for this out-of-sample forcing.

Fig. 12 compares the accuracy of the resolvent SSOP model, described in Section 3.4, to that of non-whitened balanced truncation, both with $r = 10$. The test is conducted for the same range of μ_0 values as used in Fig. 9. The non-whitened version of balanced truncation does not transform the system to one where the forcing is spatially white, and, as such, is data-free. Notably, the resolvent SSOP method, which is also data-free, outperforms balanced truncation, the current gold-standard for linear model reduction, over most of the range. The online algorithm for resolvent SSOP is identical to SSOP, so its cost is also comparable to balanced truncation. We also note resolvent SSOP nearly recovers the projection of the FOM solution onto the resolvent modes, as can be seen by comparing the solid and dashed green lines in Fig. 12.

4.2. Scalar transport problem

Next, we demonstrate the proposed algorithm on an advection-diffusion system modeling the transport of a scalar quantity in a steady fluid flow. The flow profile is the mean of a lid-driven cavity flow simulation at $Re = 30,000$. This problem differs from the Ginzburg-Landau example in three important ways: it is substantially larger ($N_x = 9604$ as opposed to 220 in the Ginzburg-Landau case), the matrix $\mathbf{A}$ is sparse, and the forcing occupies only a subset of the domain. The former means that the model is too large to compute the matrix operations without the approximations we described earlier. With this large N_x, computing the Gramians in balanced truncation is too costly as well, so we do not compare to it here, though we do note that there are effective data-driven approximations of it as well [41,42]. A balanced truncation model must have greater error than the POD representation error, which

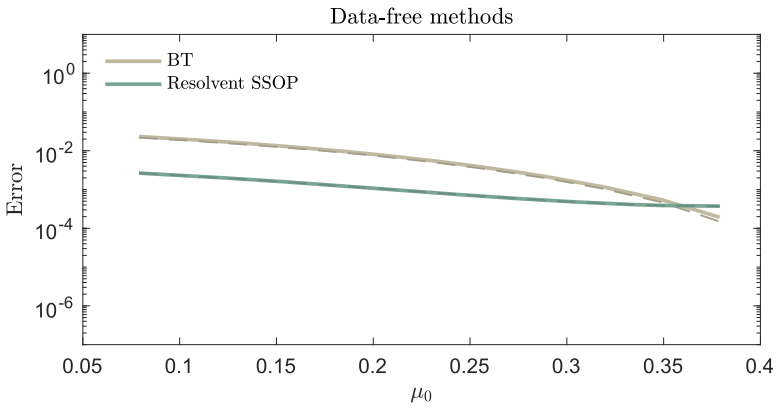

Fig. 12. The accuracy of the resolvent SSOP model and unwhitened balanced truncation for Ginzburg-Landau systems with a range of μ_0. Both methods are data-free; the resolvent modes come directly from the system matrices in the governing equation, and 'unwhitened' indicates that we did not transform the system to one where the forcing is white using forcing data before performing balanced truncation. The dashed lines are the projections of the FOM solution onto the respective modes.

we do report, and can be expected to share the CPU time of a POD-Galerkin model. That the forcing does not occupy the entire domain means that there is a CPU time savings in using the intermediary basis described in Section 3.2.2.

The continuous governing equations for the scalar transport case may be written in the same form as Eq. (4.1), where $\mathcal{A}$ is now defined as

$$\mathcal{A} = -\boldsymbol{u}(\mathbf{x}) \cdot \nabla + \eta \nabla^2, \tag{4.6}$$

and where $\boldsymbol{u}(\mathbf{x})$ is the mean flow in the lid-driven cavity. We take $\eta = 0.001$ and the velocity of the lid to be Gaussian in x (as opposed to a constant) to avoid discontinuities at the upper corners. The problem is nondimensionalized such that the maximum speed, occurring in the center of the lid, is 1. We prescribe a forcing that is stochastic with Gaussian spatial and temporal autocorrelation in a region of Gaussian support centered at $\overline{\boldsymbol{x}} = [0.75, 0.25]^T$; its statistics are given by

$$C_{ff}(\boldsymbol{x}_1 \boldsymbol{x}_2, t_1, t_2) = \exp\left[-\left(\frac{|\boldsymbol{x}_1 - \overline{\boldsymbol{x}}|^2 + |\boldsymbol{x}_2 - \overline{\boldsymbol{x}}|^2}{l^2} + \frac{|\boldsymbol{x}_2 - \boldsymbol{x}_1|^2}{\xi^2} + \frac{(t_2 - t_1)^2}{\tau^2} \right) \right], \tag{4.7}$$

where $|\cdot|$ is Euclidean distance. Here, $l = 0.1$ is the spatial width of the support of the forcing, $\xi = 0.07$ is the spatial correlation length, and $\tau = 1$ is the temporal correlation length. We use a second-order finite difference discretization with 98 points in both directions and Dirichlet boundary conditions so as to mimic a heat bath. The FOM is solved using MATLAB's `ode45`. The vorticity of the underlying velocity field and a snapshot of the transported scalar are shown in Fig. 13. The red dashed circle in the latter indicates the region in which the equations are substantially forced – it is the radius at which the autocorrelation of the forcing defined by Eq. (4.7) drops by a factor of e from its peak value.

For this example, we gather 50,000 time steps of FOM data spaced $\Delta t = 0.5$ apart from which to calculate SPOD modes and approximate operators. Time is nondimensionalized such that it takes one time unit for the lid to cross the cavity. In this example, the time step used in the FOM integration was much smaller – more than an order of magnitude for most times – due to the stiffness of the system. The data was then segmented into 648 overlapping blocks, each 256 time steps in length, so $N_\omega = 256$, and $T = 128$. The test

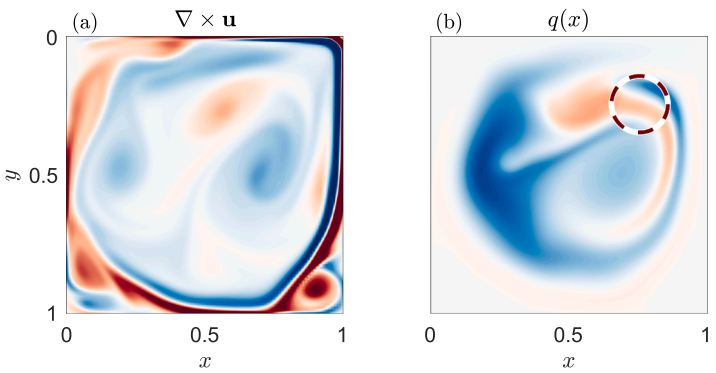

Fig. 13. Visualization of the scalar transport problem: (a) the vorticity of the steady velocity field that transports the scalar; (b) a snapshot of the scalar field with the forcing region highlighted.

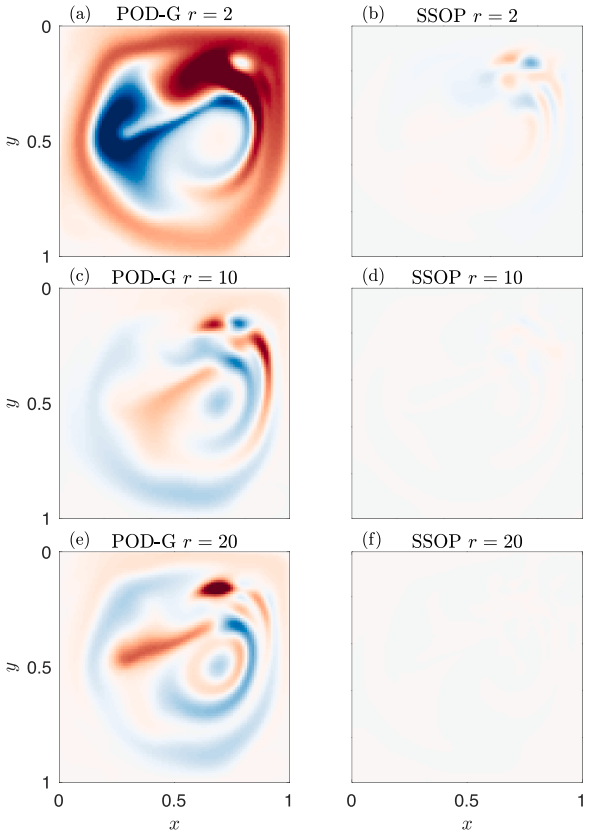

Fig. 14. Error snapshots for the scalar transport problem: (a, c, e) POD-Galerkin ROM error for 2, 10, and 20 modes; (b, d, f) SSOP ROM error for the same numbers of modes. The extremes of the color scale are 31 % of the maximum absolute value of the FOM solution, and the maximum in the 2-mode POD-Galerkin error field is 62 % of the maximum absolute value of the FOM solution.

data is 128 trajectories of the system, and error is defined in the same way as before – as the square norm of the difference of the ROM and FOM solutions averaged over all test trajectories and normalized by the mean square norm of the solution itself. Both the ROMs and the FOM use 56 cores on a pair of Intel Xeon 6242R processors. The cost of building the 10-mode ROM, including 168 seconds to compute the SPOD modes, was 30 % of the cost of generating the FOM data. The additional offline time for the POD-Galerkin ROM was trivial in comparison to the cost of running the FOM to generate data from which to obtain the POD modes. Finally, we use $p = 50$ for the intermediary basis.

Fig. 14 shows error field snapshots for the two ROMs for 2, 10, and 20 modes. The error field for the POD-Galerkin model with 20 modes is larger than that for the proposed model with 2 modes.

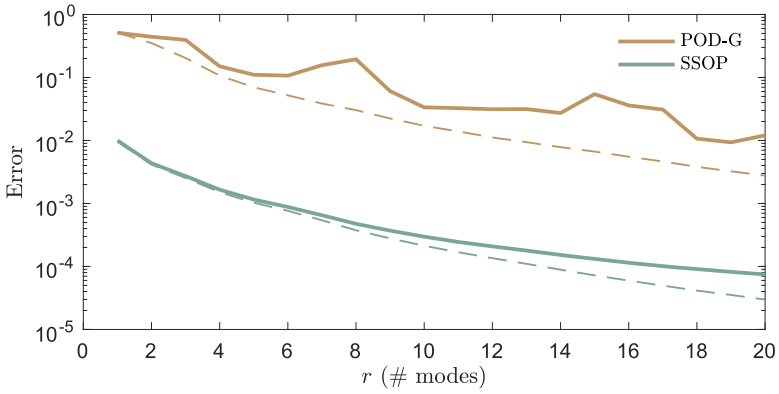

Fig. 15. Accuracy of the proposed method compared to that of a POD-Galerkin model applied to the scalar transport problem. The dashed lines are the error of the full-order solution projected on the respective bases and are lower bounds for the error of the methods.

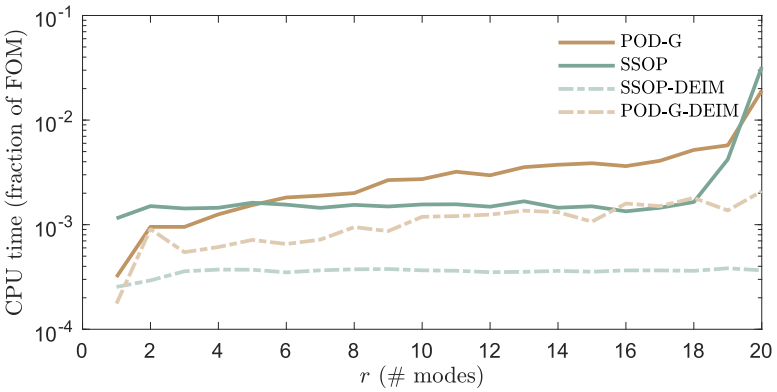

Fig. 16. Average CPU time to solve the scalar transport problem as a function of the number of space-time modes used as a fraction of the FOM CPU time. The timing results from using DEIM to remove the N_f scaling from the complexity of the model (and from the POD-Galerkin model) are also shown, and the details for this method are given in Appendix B.

Fig. 15 shows the accuracy over a range of mode numbers. Once again, the proposed method produces error two orders of magnitude lower than that of the POD-Galerkin model. The dashed lines are again the representation error. The error of the POD projection is a lower bound for the POD-Galerkin error, as well as any spatial Petrov-Galerkin method, such as balanced truncation. We see that in this problem, like in the Ginzburg-Landau problem, the SSOP model far outperforms even this bound and is indeed fairly close to its own error bound.

Fig. 16 shows the CPU time required to solve the scalar transport problem. The SSOP model is slightly faster in this case than the POD-Galerkin one. Too few modes are used to see the asymptotic linear scaling with the number of modes. Timing results from using DEIM to remove the N_f scaling are also shown for both methods, and the DEIM-augmented version of the method is described in Appendix B. There, we show that the error is not meaningfully affected by the DEIM approximation, but we also discuss some drawbacks of the augmented version of the method.

5. Conclusions

Space-time bases allow for a more accurate representation of a trajectory than do space-only bases with the same number of coefficients. In particular, the SPOD encoding of a trajectory with some number of coefficients may be orders of magnitude more accurate than the POD encoding of the same trajectory with the same number of coefficients. The obvious objectives are, therefore, to solve for these coefficients quickly and accurately, and we have pursued these for linear time-invariant systems.

The method works as follows. The SPOD coefficients $\boldsymbol{a}_k$ at frequency ω_k are given by the Fourier transform of the trajectory $\hat{\boldsymbol{q}}_k$ at that frequency left-multiplied by the (weighted) transpose of the SPOD modes. In a linear system, $\hat{\boldsymbol{q}}_k$ obeys a linear relation involving the forcing (at all frequencies) and the initial condition. We derive this relation and left-multiply it by the transpose of the SPOD modes, which amounts to a projection of a solution operator onto SPOD modes. The resulting matrices are small and may be precomputed, and the online phase of the method involves small matrix-vector products to obtain the SPOD coefficients given the forcing and initial condition.

We show, via two examples, that the SSOP method can indeed accomplish both objectives outlined above: it takes comparable CPU time to benchmark time-domain methods like POD-Galerkin projection and balanced truncation, and the solution from SSOP is roughly two orders of magnitude more accurate than both benchmarks. In fact, the SSOP solution is nearly two orders of magnitude more accurate than the projection of the FOM solution onto the POD modes, which is the lower bound on error for any time-domain Petrov-Galerkin method. We also demonstrated the robustness of the method by training and testing it on different systems.

A few negative aspects of the method are worth mentioning. The most limiting is that the method requires the entire forcing over the time interval of interest to be known before beginning the computation; the first step of the method is to take a FFT of the forcing, which cannot be done without the entire forcing in time. For some applications, this prevents the method from being applicable, while for others, it is not a problem. Second, the method works on a prepresribed interval $[0, T]$. If one wishes to obtain the solution longer than this interval, one can repeat the method with the value at the end of the interval as the initial condition. This is more cumbersome than extending the solution in a time-domain method. Finally, SPOD modes require more training data than POD modes, which limits the applicability of the proposed method in cases where training data is scarce. Where these disadvantages are not obstacles, however, we have shown the proposed method to be substantially more accurate at the same CPU cost compared to standard methods for linear model reduction. We hope that this will aid in applications of linear model reduction and increase interest in space-time methods.

Finally, we also described a resolvent-based version of the SSOP method that is data-free. Leveraging the ability of resolvent modes obtained directly from the linear system to approximate SPOD modes [15], the resolvent SSOP method uses resolvent modes in place of SPOD modes within the SSOP method, eliminating the need for training data. We showed this method to be more accurate than balanced truncation, the state of the art for data-free model reduction. We see this result as quite promising, calling for further work

comparing the method to balanced truncation. We also believe that this result underscores the promise of space-time techniques for model reduction.

One clear area for further inquiry is whether the method can be generalized to the case of nonlinear governing equations. The method uses the fundamental solution to the LTI system, and, of course, no analogous solution exists for general nonlinear systems. One possible remedy is to treat the nonlinearity that would result from the present SPOD coefficients as an additional forcing on the system, and then again use the fundamental LTI solution. Nonlinearity as a forcing to a linear system is the perspective taken by resolvent analysis [16], and the successes of that field in predicting turbulent structures are cause for optimism that this perspective would be useful for model reduction. With such an approach, Eq. (3.23) would become a coupled system of nonlinear algebraic equations, which would be solved online for the SPOD coefficients given the initial condition and forcing.

CRediT authorship contribution statement

Peter Frame: Writing – original draft, Validation, Software, Methodology, Investigation, Conceptualization; **Cong Lin:** Writing – review & editing, Software, Methodology; **Oliver T. Schmidt:** Writing – review & editing, Methodology; **Aaron Towne:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Aaron Towne reports financial support was provided by National Science Foundation. Oliver Schmidt reports financial support was provided by National Science Foundation. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

P.F. and A.T. gratefully acknowledge funding from the National Science Foundation grant no. 2237537. C.L. and O.S. gratefully acknowledge funding from the National Science Foundation grant no. 2046311.

Appendix A. Fourier representation of the time derivative

Here, we illustrate why substituting $i\omega_k\hat{\boldsymbol{q}}_k$ for $\dot{\boldsymbol{q}}$ when going to the frequency domain is only correct if $\boldsymbol{q}$ starts and ends at the same value on the interval. It is easier to do this using the continuous definition of the Fourier coefficients, rather than the DFT, so we compute the integral

$$\hat{\boldsymbol{q}}_k = \frac{1}{T}\int_0^T \dot{\boldsymbol{q}} e^{-i\omega_k t}\, dt. \tag{A.1}$$

This integral may be solved by parts,

$$\hat{\boldsymbol{q}}_k = \frac{1}{T}\boldsymbol{q}\Big|_0^T - \int_0^T -i\omega_k \boldsymbol{q}(t) e^{-i\omega_k t} dt. \tag{A.2}$$

The boundary term gives $\frac{\Delta \boldsymbol{q}}{T}$ [49], where $\Delta\boldsymbol{q} = \boldsymbol{q}(T) - \boldsymbol{q}(0)$, and the integral is the naive term $i\omega_k\hat{\boldsymbol{q}}_k$. The Fourier representation of the derivative is thus

$$\hat{\dot{\boldsymbol{q}}}_k = i\omega_k\hat{\boldsymbol{q}}_k + \frac{\Delta\boldsymbol{q}}{T}. \tag{A.3}$$

Appendix B. DEIM-augmented algorithm

Here, we present a means of improving the cost scaling in cases where the dimension of the forcing N_f is large but the forcing is spatially structured. The idea is to use a sparse sampling of the forcing vectors, which are size N_f, using the discrete empirical interpolation method (DEIM) [39]. We also sparse sample the two vectors that are size N_x, the initial condition and the forcing sum terms in Eq. (3.22), though these terms can also be handled using an intermediary basis, as before. The rank-p DEIM approximation of a vector $\boldsymbol{v} \in \mathbb{C}^{N_x}$ is $\boldsymbol{v} \approx \mathbf{U}_{\boldsymbol{v}}(\mathbf{P}_{\boldsymbol{v}}^T\mathbf{U}_{\boldsymbol{v}})^{-1}\mathbf{P}_{\boldsymbol{v}}^T\boldsymbol{v}$, where the columns of $\mathbf{U}_{\boldsymbol{v}} \in \mathbb{C}^{N_x\times p}$ are the POD modes for the ensemble from which $\boldsymbol{v}$ is a sample, and $\mathbf{P}_{\boldsymbol{v}}^T \in \{0,1\}^{p\times N_x}$ samples p elements from $\boldsymbol{v}$ and is formed via the DEIM algorithm.

The DEIM algorithm is run for the forcing in the time domain, giving a set of sample points $\mathbf{P}_{\boldsymbol{f}}^T \in \mathbb{C}^{p\times N_f}$ from which the forcing can be reconstructed accurately. The structures in the forcing at different frequencies will, in general, be different, so it is best to use

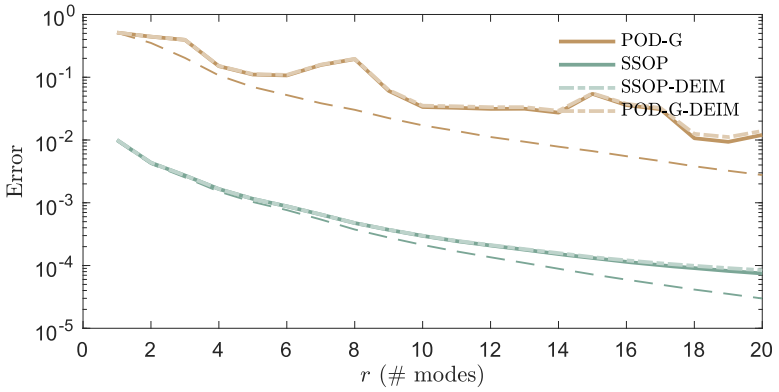

Fig. B.17. Accuracy of the DEIM-augmented version of the method compared to POD-Galerkin projection (and the DEIM version thereof) applied to the scalar transport problem. As long as enough sample points are used, DEIM does not introduce additional error. Again, the dashed lines are the error of the full-order solution projected on the respective bases.

a different spatial basis at each frequency to complete the DEIM approximation. We label the spatial basis for the forcing at the k-th frequency $\mathbf{U}_{\hat{f}_k}$. The approximation for the k-th forcing is $\hat{\boldsymbol{f}}_k \approx \mathbf{U}_{\hat{f}_k}(\mathbf{P}_f^T\mathbf{U}_{\hat{f}_k})^{-1}\mathbf{P}_f^T\hat{\boldsymbol{f}}_k$, so the first term in Eq. (3.22) is now

$$\mathbf{E}_k\hat{\boldsymbol{f}}_k \approx \mathbf{K}_{\hat{f}_k}\mathbf{P}_f^T\hat{\boldsymbol{f}}_k, \tag{B.1}$$

where the matrix $\mathbf{K}_{\hat{f}_k} = \mathbf{E}_k\mathbf{U}_{\hat{f}_k}(\mathbf{P}_f^T\mathbf{U}_{\hat{f}_k})^{-1} \in \mathbb{C}^{r_k\times p}$ is precomputed for each frequency. Note that the same sampling is used for each frequency; if the sampling were different for each frequency, one would need to take the DFT of the entire forcing (or the union of all the samplings), which would negate the scaling benefit of sparse sampling. This approach for the forcing term can be used in conjunction with the intermediary basis approach for the initial condition and forcing sum terms in Eq. (3.22). These latter terms can also be handled with DEIM, which we show now.

The initial condition and forcing sum terms can be approximated with DEIM matrices $\mathbf{U}_{q_0}$ and $\mathbf{P}_{q_0}^T$ for the initial condition, and $\mathbf{U}_{fs}$ and $\mathbf{P}_{fs}^T$ for the forcing sum. For the former, these matrices are formed by gathering all initial conditions within the training data and running the DEIM algorithm to obtain $\mathbf{U}_{q_0}$ and $\mathbf{P}_{q_0}^T$. From these matrices, the initial condition multiplied by $\mathbf{F}_k$ is approximated as

$$\mathbf{F}_k\boldsymbol{q}_0 \approx \mathbf{K}_{q_0}\mathbf{P}_{q_0}^T\boldsymbol{q}_0, \tag{B.2}$$

where $\mathbf{K}_{q_0,k} = \mathbf{F}_k\mathbf{U}_{q_0}(\mathbf{P}_{q_0}^T\mathbf{U}_{q_0})^{-1} \in \mathbb{C}^{r_k\times p}$ is precomputed. Similarly, $\mathbf{U}_{fs}$ and $\mathbf{P}_{fs}^T$ are obtained by running DEIM on the set of forcing sums, which must be calculated from each trajectory in the training data. With these matrices, the forcing sum multiplied by $\mathbf{F}_k$ is approximated as

$$\mathbf{F}_k\frac{1}{N_\omega}\sum_l \boldsymbol{\Psi}_l\mathbf{E}_l\hat{\boldsymbol{f}}_l \approx \frac{1}{N_t}\mathbf{K}_{fs,l}\sum_l \mathbf{T}_{l,fs}\mathbf{E}_l\hat{\boldsymbol{f}}_l, \tag{B.3}$$

where $\mathbf{K}_{fs,l} = \mathbf{F}_k\mathbf{U}_{fs}(\mathbf{P}_{fs}^T\mathbf{U}_{fs})^{-1} \in \mathbb{C}^{r_k\times p}$ and $\mathbf{T}_{l,fs} = \mathbf{P}_{fs}^T\boldsymbol{\Psi}_l \in \mathbb{C}^{p\times r_l}$ are both precomputed. These operators are kept separate (as opposed to multiplied as a precomputation step) to avoid N_ω^2 scaling. With these approximations, the DEIM-augmented equation is

$$\boldsymbol{a}_k = \mathbf{K}_{\hat{f}_k}\mathbf{P}_f^T\hat{\boldsymbol{f}}_k + \mathbf{K}_{q_0}\mathbf{P}_{q_0}^T\boldsymbol{q}_0 - \frac{1}{N_t}\mathbf{K}_{fs,l}\sum_l \mathbf{T}_{l,fs}\mathbf{E}_l\hat{\boldsymbol{f}}_l. \tag{B.4}$$

This method removes the N_f scaling and replaces it with p, thus, the DEIM-augmented version of the algorithm scales like $\mathcal{O}((rp + p\log N_\omega)N_\omega)$.

To demonstrate the augmented version of the method, we apply it to the scalar transport problem described in the main text. The forcing in this problem meets the conditions for a large improvement: $N_f = 2050$ is large, but the forcing is spatially structured, coming from Eq. (4.7), so it can be approximated via sparse sampling effectively. Fig. B.17 shows the error of the DEIM-augmented version with $p = 200$ along with that of the non-augmented version. There is almost no error sacrifice relative to the non-approximated method with this number of sample points. The timing for the method applied to the scalar transport problem is shown in Fig. 16, along with the timing for a DEIM-augmented version of POD-Galerkin projection. Indeed, DEIM offers substantial speedup, both for the proposed method and for POD-Galerkin projection.

Two drawbacks of the DEIM-augmented version lead us to favor the non-augmented method in most cases. First, the DEIM-augmented version of the method relies on the initial condition and forcing being accurately approximated by sparse samplings, and these sparse samplings will only be accurate if the initial condition and forcing are similar in character to those in the training data. Second, the DEIM-augmented version is cumbersome to implement, requiring more precomputation of modes and matrices, relative to the non-augmented method.

References

[1] N. Aubry, P. Holmes, J.L. Lumley, E. Stone, The dynamics of coherent structures in the wall region of a turbulent boundary layer, J. Fluid Mech. 192 (1988) 115–173. https://doi.org/10.1017/s0022112088001818

[2] B.R. Noack, K. Afanasiev, M. Morzyński, G. Tadmor, F. Thiele, A hierarchy of low-dimensional models for the transient and post-transient cylinder wake, J. Fluid Mech. 497 (2003) 335–363. https://doi.org/10.1017/s0022112003006694

[3] C.W. Rowley, T. Colonius, R.M. Murray, Model reduction for compressible flows using POD and Galerkin projection, Physica D 189 (1–2) (2004) 115–129. https://doi.org/10.1016/j.physd.2003.03.001

[4] B. Moore, Principal component analysis in linear systems: controllability, observability, and model reduction, IEEE Trans. Automat. Contr. 26 (1) (1981) 17–32. https://doi.org/10.1109/tac.1981.1102568

[5] K. Lee, K.T. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, J. Comput. Phys. 404 (2020) 108973. https://doi.org/10.1016/j.jcp.2019.108973

[6] F.J. Gonzalez, M. Balajewicz, Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems, arXiv preprint arXiv:1808.01346. (2018).

[7] K. Carlberg, C. Bou-Mosleh, C. Farhat, Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations, Int. J. Numer. Methods Eng. 86 (2) (2010) 155–181. https://doi.org/10.1002/nme.3050

[8] K. Carlberg, C. Farhat, J. Cortial, D. Amsallem, The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows, J. Comput. Phys. 242 (2013) 623–647. https://doi.org/10.1016/j.jcp.2013.02.028

[9] S.E. Otto, A. Padovan, C.W. Rowley, Optimizing oblique projections for nonlinear systems using trajectories, SIAM J. Sci. Comput. 44 (3) (2022) A1681–A1702. https://doi.org/10.1137/21m1425815

[10] B. Peherstorfer, K. Willcox, Data-driven operator inference for nonintrusive projection-based model reduction, Comput. Methods Appl. Mech. Eng. 306 (2016) 196–215. https://doi.org/10.1016/j.cma.2016.03.025

[11] A. Padovan, B. Vollmer, D.J. Bodony, Data-driven model reduction via non-intrusive optimization of projection operators and reduced-order dynamics, 2024, https://doi.org/10.48550/ARXIV.2401.01290

[12] J.L. Lumley, The structure of inhomogeneous turbulent flows, Atmospheric Turbulence and Radio Wave Propagation (1967).

[13] O.T. Schmidt, P.J. Schmid, A conditional space–time POD formalism for intermittent and rare events: example of acoustic bursts in turbulent jets, J. Fluid Mech. 867 (2019). https://doi.org/10.1017/jfm.2019.200

[14] P. Frame, A. Towne, Space-time POD and the Hankel matrix, PLoS ONE 18 (8) (2023) e0289637. https://doi.org/10.1371/journal.pone.0289637

[15] A. Towne, O.T. Schmidt, T. Colonius, Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis, J. Fluid Mech. 847 (2018) 821–867.

[16] B.J. McKeon, A.S. Sharma, A critical-layer framework for turbulent pipe flow, J. Fluid Mech. 658 (2010) 336–382. https://doi.org/10.1017/s002211201000176x

[17] S. Beneddine, D. Sipp, A. Arnault, J. Dandois, L. Lesshafft, Conditions for validity of mean flow stability analysis, J. Fluid Mech. 798 (2016) 485–504. https://doi.org/10.1017/jfm.2016.331

[18] O.T. Schmidt, A. Towne, G. Rigas, T. Colonius, G.A. Brès, Spectral analysis of jet turbulence, J. Fluid Mech. 855 (2018) 953–982. https://doi.org/10.1017/jfm.2018.675

[19] P.A.S. Nogueira, A.V.G. Cavalieri, P. Jordan, V. Jaunet, Large-scale streaky structures in turbulent jets, J. Fluid Mech. 873 (2019) 211–237. https://doi.org/10.1017/jfm.2019.365

[20] E. Pickering, G. Rigas, P.A.S. Nogueira, A.V.G. Cavalieri, O.T. Schmidt, T. Colonius, Lift-up, Kelvin–Helmholtz and Orr mechanisms in turbulent jets, J. Fluid Mech. 896 (2020). https://doi.org/10.1017/jfm.2020.301

[21] M. Yano, A.T. Patera, K. Urban, A space-time hp-interpolation-based certified reduced basis method for Burgers' equation, Math. Models Methods Appl. Sci. 24 (09) (2014) 1903–1935. https://doi.org/10.1142/s0218202514500110

[22] E.J. Parish, K.T. Carlberg, Windowed least-squares model reduction for dynamical systems, J. Comput. Phys. 426 (2021) 109939. https://doi.org/10.1016/j.jcp.2020.109939

[23] Y. Choi, P. Brown, W. Arrighi, R. Anderson, K. Huynh, Space–time reduced order model for large-scale linear dynamical systems with application to Boltzmann transport problems, J. Comput. Phys. 424 (2021) 109845. https://doi.org/10.1016/j.jcp.2020.109845

[24] Y. Kim, K. Wang, Y. Choi, et al., Efficient space–time reduced order model for linear dynamical systems in python using less than 120 lines of code, Mathematics 9 (14) (2021). https://doi.org/10.3390/math9141690

[25] C. Hoang, K. Chowdhary, K. Lee, J. Ray, Projection-based model reduction of dynamical systems using space–time subspace and machine learning, Comput. Methods Appl. Mech. Eng. 389 (2022) 114341. https://doi.org/10.1016/j.cma.2021.114341

[26] C. Lin, Model Order Reduction in the frequency Domain via Spectral Proper Orthogonal Decomposition, Master's thesis, University of Illinois at Urbana-Champaign (2019). https://doi.org/10.13140/RG.2.2.10653.56802

[27] A. Towne, Space-time Galerkin projection via spectral proper orthogonal decomposition and resolvent modes, in: AIAA Scitech 2021 Forum, American Institute of Aeronautics and Astronautics, 2021. https://doi.org/10.2514/6.2021-1676

[28] T. Chu, O.T. Schmidt, A stochastic SPOD-Galerkin model for broadband turbulent flows, Theor. Comput. Fluid Dyn. 35 (6) (2021) 759–782. https://doi.org/10.1007/s00162-021-00588-6

[29] C. Ji, D. Xie, S. Zhang, A. Maqsood, Spectral proper orthogonal decomposition reduced-order model for analysis of aerothermoelasticity, AIAA J. 61 (2) (2023) 793–807. https://doi.org/10.2514/1.j062304

[30] K.C. Hall, J.P. Thomas, W.S. Clark, Computation of unsteady nonlinear flows in cascades using a harmonic balance technique, AIAA J. 40 (5) (2002) 879–886. https://doi.org/10.2514/2.1754

[31] K.C. Hall, K. Ekici, J.P. Thomas, E.H. Dowell, Harmonic balance methods applied to computational fluid dynamics problems, Int. J. Comut. Fluid Dyn. 27 (2) (2013) 52–67. https://doi.org/10.1080/10618562.2012.742512

[32] K. Ekici, K.C. Hall, Nonlinear analysis of unsteady flows in multistage turbomachines using harmonic balance, AIAA J. 45 (5) (2007) 1047–1057. https://doi.org/10.2514/1.22888

[33] Y. Choi, K. Carlberg, Space–time least-squares petrov–Galerkin projection for nonlinear model reduction, SIAM J. Sci. Comput. 41 (1) (2019) A26–A58. https://doi.org/10.1137/17M1120531

[34] L. Sirovich, Turbulence and the dynamics of coherent structures. II. symmetries and transformations, Q. Appl. Math. 45 (3) (1987) 573–582. https://doi.org/10.1090/qam/910463

[35] P. Welch, The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms, IEEE Trans. Audio Electroacoust. 15 (2) (1967) 70–73. https://doi.org/10.1109/TAU.1967.1161901

[36] M.D. Grigoriu, Linear Dynamical Systems, Springer International Publishing, 2021. https://doi.org/10.1007/978-3-030-64552-6

[37] E. Martini, A.V.G. Cavalieri, P. Jordan, L. Lesshafft, Accurate Frequency Domain Identification of ODEs with Arbitrary Signals, 2019. https://doi.org/10.48550/ARXIV.1907.04787

[38] A. Farghadan, E. Martini, A. Towne, Scalable resolvent analysis for three-dimensional flows, J. Comput. Phys. 524 (2025) 113695. https://doi.org/10.1016/j.jcp.2024.113695

[39] S. Chaturantabut, D.C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. Sci. Comput. 32 (5) (2010) 2737–2764. https://doi.org/10.1137/090766498

[40] O.T. Schmidt, A. Towne, An efficient streaming algorithm for spectral proper orthogonal decomposition, Comput. Phys. Commun. 237 (2019) 98–109. https://doi.org/10.1016/j.cpc.2018.11.009

[41] K. Willcox, J. Peraire, Balanced model reduction via the proper orthogonal decomposition, AIAA J. 40 (11) (2002) 2323–2330. https://doi.org/10.2514/2.1570

[42] C.W. Rowley, Model reduction for fluids, using balanced proper orthogonal decomposition, Int. J. Bifurcation Chaos 15 (03) (2005) 997–1013. https://doi.org/10.1142/s0218127405012429
[43] S. Bagheri, D.S. Henningson, J. Hœpffner, P.J. Schmid, Input-output analysis and control design applied to a linear model of spatially developing flows, Appl. Mech. Rev. 62 (2) (2009). https://doi.org/10.1115/1.3077635
[44] K.K. Chen, C.W. Rowley, Optimal actuator and sensor placement in the linearised complex Ginzburg–Landau system, J. Fluid Mech. 681 (2011) 241–260. https://doi.org/10.1017/jfm.2011.195
[45] C.W. Rowley, S.T.M. Dawson, Model reduction for flow analysis and control, Annu. Rev. Fluid Mech. 49 (1) (2017) 387–417. https://doi.org/10.1146/annurev-fluid-010816-060042
[46] A. Laub, M. Heath, C. Paige, R. Ward, Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms, IEEE Trans. Automat. Contr. 32 (2) (1987) 115–122. https://doi.org/10.1109/tac.1987.1104549
[47] L.N. Trefethen, Spectra and Pseudospectra: The Behaviour of Non-Normal Matrices and Operators, Springer Berlin Heidelberg, 1999, p. 217–250. https://doi.org/10.1007/978-3-662-03972-4_6
[48] P.J. Schmid, Nonmodal stability theory, Annu. Rev. Fluid Mech. 39 (1) (2007) 129–162. https://doi.org/10.1146/annurev.fluid.38.050304.092139
[49] E. Martini, D. Rodríguez, A. Towne, A.V.G. Cavalieri, Efficient computation of global resolvent modes, J. Fluid Mech. 919 (2021). https://doi.org/10.1017/jfm.2021.364