

Space-Time Model Reduction Using Spectral Proper Orthogonal Decomposition

by

Peter K. Frame

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in The University of Michigan
2026

Doctoral Committee:

Professor Aaron Towne, Chair
Professor Jesse Capecelatro
Professor Karthik Duraisamy
Professor Krzysztof Fidkowski

Peter K. Frame
pframe@umich.edu
ORCID iD: 0000-0002-9919-1796
© Peter K. Frame 2026

DEDICATION

To my parents — you've given me so much.

ACKNOWLEDGEMENTS

My time in Ann Arbor, much of which has been spent developing the work in this thesis, will always feel significant to me. I want to thank those who have contributed to my decision to choose this path, and those who have made my time here so memorable.

My first thanks go to my advisor, Aaron Towne. Aaron, our meetings are often the highlight of my workweek, and I look forward to them as a time for rigorous yet energetic discussion of results and ideas. I'm particularly grateful for the freedom and encouragement you've given me over the years to be creative and to dive into the topics I find interesting, even when this has come at the cost of near-term research progress.

I'm grateful to the other members of our group, past and present, including Ali, Junoh, Rutvij, Min, Yuhao, Kai, James, Saarthak, Henry, Marildo, and Ravee. Our daily long lunches are usually a welcome break from work, though I suppose we can't complain if they sometimes lead to a useful idea for one of our projects. Ali and Junoh, we overlapped for almost five years, and the two of you graduating felt like the end of an era.

To my friends here and elsewhere, thanks for providing balance while I pursued this degree. Ray, who'd have guessed that we'd go from teammates in middle and high school, to roommates in college, to (soon-to-be) co-authors in graduate school. I'm grateful to my other lifelong friends Solomon and Jack, and it'd be great if Zoom chess and Zoom guitar return permanently to 'in-person' at some point. Eli and Codie, thanks for always being up for something active, either around town or across the county. Thomas, it's too bad I didn't convince you to join our group, but that didn't stop us from sharing an office and lots of good times outside of it. Thanks to Tobias and James for sharing my enthusiasm for physics — I'm glad we've kept up our nerdy chats since we all left UCSD, and I've probably wandered a hundred aimless miles through Ann Arbor while on the phone with you. I am indebted to my friends from LLNL, Torin, Meredith, Helen, Thomas, and Jessie — oddly enough, that happy summer driving from the lab to Yosemite every weekend was what made me decide to go to graduate school. Thanks to all the people I met living at Escher for the many late-night conversations, and for adding a distinct quirkiness to my experience here. Lisa and Claire, you welcomed me to Ann Arbor from the beginning, and turned into my parents-away-from-home while I've been here.

I owe the most to my partner, Maya, and to my parents, Kari and Jim. Maya, you've

added so much joy and direction to my life. Mom and Dad, you never put pressure on me to go along any particular path, and you've supported me in everything I've done.

The work in this thesis was supported by the National Science Foundation grant no. 2237537.

TABLE OF CONTENTS

DEDICATION	ii
ACKNOWLEDGEMENTS	iii
LIST OF FIGURES	viii
LIST OF ACRONYMS	xiii
ABSTRACT	xiv
1 Introduction	1
1.1 The need for model reduction	1
1.2 The standard framework: space-only model reduction	2
1.2.1 Encoding methods	3
1.2.2 Producing the reduced dynamical system	5
1.2.3 Time-delay embedding methods	9
1.2.4 Outlook on model reduction	9
1.3 Space-time model reduction	10
1.3.1 The space-time framework	10
1.3.2 Existing space-time ROMs	11
1.3.3 Proposed approach	13
1.4 Thesis overview	14
2 Space-Time Proper Orthogonal Decomposition and the Hankel Matrix	16
2.1 Space-only POD	19
2.1.1 Formulation	19
2.1.2 With discrete data	20
2.2 Spectral POD	22
2.2.1 Formulation	22
2.2.2 With discrete data	23
2.3 Space-time POD	24
2.3.1 Formulation	24
2.3.2 Space-time POD on long times becomes spectral POD	26
2.3.3 Space-time POD on short times becomes space-only POD	26
2.3.4 Space-time POD with discrete data	27
2.4 The Hankel matrix	28
2.5 Hankel singular vectors are space-time POD modes	29

2.5.1	Height of the Hankel matrix corresponds to T	30
2.5.2	Width of the Hankel matrix informs convergence	33
2.5.3	Summary of the connection	35
2.6	Fully exploiting ergodicity: more accurate modes	37
2.7	Numerical experiments: Lid-driven cavity flow	39
2.7.1	Simulation description	40
2.7.2	Spaced columns yield better modes	40
2.7.3	Faster computation by sampling the Hankel matrix	41
2.7.4	Convergence to space-only POD for short times	43
2.7.5	Convergence to spectral POD for long times	45
2.7.6	Fully Exploiting Ergodicity	46
2.8	Chapter summary	51
3	Linear Spectral Solution Operator Projection	52
3.1	Introduction	52
3.2	SPOD modes for encoding trajectories	53
3.2.1	Trajectory representation	53
3.2.2	Encoding using SPOD modes	54
3.3	Resolvent analysis	58
3.4	Spectral solution operator projection method	59
3.4.1	Solution operator	60
3.4.2	Evaluating the inner product	60
3.4.3	Relation to previous SPOD-based methods	62
3.4.4	Operator approximations	63
3.4.5	Computational complexity	65
3.4.6	Data-free method	66
3.5	Examples	66
3.5.1	Linearized Ginzburg-Landau problem	67
3.5.2	Scalar transport problem	83
3.6	Chapter summary	84
4	Generalizing SSOP to Nonlinear Systems	87
4.1	Nonlinear spectral solution operator projection	88
4.1.1	Formulation	88
4.1.2	Constructing the nonlinearity	90
4.1.3	Affine parametric dependence	94
4.1.4	Solving the nonlinear system	96
4.1.5	Analysis of the solution procedure	97
4.1.6	Algorithms and computational complexity	100
4.2	Results	105
4.2.1	Standard Ginzburg-Landau system	106
4.2.2	Quadratic Ginzburg-Landau system	114
4.3	Chapter summary	116
5	Preliminary Work on Application to the Navier-Stokes Equations	118

5.1	Formulation of the incompressible Navier-Stokes equations	119
5.1.1	Incompatibility with the standard form	119
5.1.2	Leray projection form	119
5.2	Time-stepping approximation of ROM operators	120
5.2.1	Test case	121
5.2.2	Transient operator	121
5.2.3	Resolvent operator	123
5.3	Steps remaining for a complete model	126
6	Conclusion	127
6.1	Summary	127
6.2	Future work	130
	BIBLIOGRAPHY	132

LIST OF FIGURES

FIGURE

1.1	Space-only and space-time reduced-order modeling frameworks (top and bottom panels, respectively). The space-time encoding leverages the spatiotemporal correlations present in the trajectory $q(x, t)$ to achieve greater efficiency.	12
2.1	The height of the Hankel matrix d , along with the time step Δt between snapshots, determines the time window T over which the space-time correlation C of the system is accounted for in the Hankel singular modes. Shorter Hankel matrices, therefore, severely truncate the correlations, whereas taller Hankel matrices retain more of these correlations.	31
2.2	Uncorrelated columns (left) and Hankel (right) approaches to approximating the correlation matrix from short (b,c) and long (d,e) time series. (a) A time series used to sample temporal realizations with different samplings. (b) Correlations from uncorrelated columns using a short time series. (c) Correlations from the Hankel matrix using the short times series. (d) Correlations from uncorrelated columns using a long time series. (e) Correlations from a Hankel matrix using the long time series. Though both (d) and (e) are accurate, the correlations in (d) come at significantly lower computational cost.	36
2.3	The dimensions of the Hankel matrix can be interpreted in terms of the integral space-time eigenvalue problem (2.3.6). The height of the Hankel matrix corresponds to T , the length on which the space-time POD modes optimally represent the flow. The width, though there is subtlety about independence, informs how accurately the correlation matrix is approximated, and hence the accuracy of the modes.	37
2.4	Schematic of the lid-driven cavity flow. The lid moves to the right and drives the flow. Contours show a snapshot of the turbulent kinetic energy.	41
2.5	Convergence of the modes with the width m of the matrix: (blue) Hankel matrix; (red) uncorrelated data matrix. The modes from the uncorrelated data matrix converge faster than those from the Hankel matrix. This improved convergence is observed because the columns of the uncorrelated data matrix represent data from more of the attractor than the columns of the Hankel matrix of the same size.	42

2.6	Similarity between modes from a downsampled Hankel matrix, retaining m/m_H columns, and the Hankel modes. Both the mean (solid) and the median (dashed) are reported for the 200 trials. The first mode (blue) and the second mode (red) are more similar than the third mode (green). For the parameters used, retaining 1/10 and 1/5 of the columns yields modes that capture over 97% and 99.9% of the energy of the Hankel modes, respectively, for both the first and second modes, on average (mean).	44
2.7	Convergence of space-time POD modes ϕ_T to space-only POD modes ϕ for small T . The spatial dependence of the first space-time mode at each T is compared to the first space-only mode using the space-only inner product, averaged over the time evolution of the space-time mode.	45
2.8	Convergence of space-time POD to spectral POD, visualized in terms of the power spectral density of the space-time POD modes for various values of T : (blue) $T = 50$; (red) $T = 100$; (green) $T = 200$; (purple) $T = 400$. As T is increased, more and more of the energy of the modes is contained to an increasingly narrow band of frequencies, indicating convergence toward a discrete frequency, consistent with the time dependence of SPOD modes.	46
2.9	PDFs of energy captured by the first modes of the Hankel (blue) and Toeplitz (red) methods for various choices of m , d , and N_x . Both methods use the same time series of length $m+d-1$ to approximate the modes. The PDFs are calculated with 1000 samples and the energies are divided by the maximum possible energy of one mode. These parameter values are, (a): $m = 30$, $d = 30$, $N_x = 1$. (b): $m = 100$, $d = 30$, $N_x = 1$. (c): $m = 300$, $d = 30$, $N_x = 1$. (d): $m = 300$, $d = 30$, $N_x = 3$. (e) $m = 600$, $d = 30$, $N_x = 3$. (f) $m = 10$, $d = 10$, $N_x = 1$	48
2.10	Mean (circles) and median (squares) accuracy of Hankel modes (blue) vs. Toeplitz modes (red) calculated from 1000 time series of the most energetic N_x points of the lid-driven cavity flow. Accuracy is measured as the square inner product against a ‘true mode.’ When the data is severely limited (m is relatively small), the difference is more pronounced.	49
2.11	PDFs of the energy captured by the first four modes of the Toeplitz and Hankel methods, normalized by the maximum possible energy by four modes. These are calculated for $m = 30$, $d = 30$, $N_x = 1$. The difference is more severe for the latter modes, which we observed for the majority of the parameter choices that we tested.	50
3.1	Number of degrees of freedom (DOFs) required to achieve 2% representation error as a function of the length of the time interval of the trajectory $[0, T]$. The convergence of SPOD to space-time POD proven in Chapter 2 can be seen. The data are from the Ginzburg-Landau system introduced in Section 3.5.	57
3.2	The relative error for the SPOD and POD trajectory encoders as a function of r , the number of modes used.	58

3.3	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 $x = 0$	69
3.4	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 (3.5.4)) as a function of r , the average number of modes per frequency.	70
3.5	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.	71
3.6	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.	74
3.7	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 (3.5.5).	75
3.8	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.	75
3.9	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 $\mu_0 = 0.397$	77
3.10	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.	79
3.11	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.	80

3.12	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.	81
3.13	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.	81
3.14	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.	82
3.15	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.	85
3.16	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.	85
4.1	Spectral Solution Operator Projection: A system of nonlinear algebraic equations is obtained by performing a space-time projection of a linear solution operator onto the set of SPOD modes used to encode trajectories. The linear solution operator returns the solution to the linear system given the initial condition and forcing. Nonlinearities are handled in this framework as a state-dependent forcing. The system is solved via a fixed-point iteration, which may be viewed as a perturbation series around the linear solution.	89
4.2	A trajectory of the (standard) Ginzburg-Landau system with $\mu_0 = 0.229$. The green lines demarcate the location of the forcing. The spatiotemporal structure enables space-time modes, like SPOD modes, to represent the trajectory far more efficiently than space-only modes.	107
4.3	SPOD energies. (a) The energy not captured by the first rN_ω modes, calculated as the ratio of the sum of the energies of the excluded modes to the sum of the energies of all the modes. The dashed line corresponds to the r value used to determine the cutoff in (b). (b) The energy of the included (red) and excluded (blue) modes as a function of ω for $r = 5$. Inclusion of a given mode ψ_{km} is determined by whether the associated energy is among the rN_ω largest energies over all frequencies i.e., whether $\lambda_{km} \geq \tilde{\lambda}_{(rN_\omega)}$ (see (3.2.10)). The cutoff energy $\tilde{\lambda}_{(rN_\omega)}$ is shown in orange. The number of modes at each frequency $r(\omega)$ is shown in green on the right axis; the mean is $r = 5$	108

4.4	Error as a function of time for the proposed method and for POD-G, both with $r = 5$. The curves represent the mean error over the 30 test trajectories, and the shaded regions indicate the standard deviation of these errors. The dashed curves show the respective projection errors, which serve as lower bounds for the two methods.	110
4.5	Error, averaged over time, for SSOP and for POD-G as a function of r , the number of modes used in the ROMs. The dashed curves again represent the projection error onto the POD and SPOD modes.	111
4.6	CPU time for SSOP in comparison to that of POD-G as a function of the number of modes used in the ROMs.	111
4.7	The error as a function of time for a variety of non-stochastic forcings for both SSOP and POD-G with $r = 5$ modes, along with the respective projection errors (dashed). The SSOP performance is robust for these out-of-sample forcings. . .	112
4.8	The methods tested on a range of Ginzburg-Landau systems parameterized by μ_0 with $r = 5$. In (a), the model is trained using data from the same GL system. In (b), the models are trained using data from $\mu_0^* = 0.229$. The dashed curves correspond to the projection error for the modes from the training system. . . .	113
4.9	The impact of ω_l on ω_k as defined by (4.2.7) via triadic interaction with $\omega_i = \omega_k - \omega_l$. (a) shows T_{kl} , a proxy for the strength of all interactions between the SPOD modes at ω_l with those at ω_i . (b) shows how many of these interactions are needed to account for 98% of T_{kl}	115
4.10	(a) The total number of interactions between SPOD modes at ω_l and $\omega_k - \omega_l$ with $r = 5$. (b) The number of these interactions that are retained when $\epsilon = 10^{-1.8}$. Only 1.7% of the interactions in (a) are retained with this threshold.	116
4.11	The relative error (a) and CPU time (b) for the method with $r = 5$ using the sparse-triadic-interaction approximation of the nonlinearity as a function of the number of triadic interactions retained. The x -axis shows the percentage of the total triadic interactions used. The red ticks indicate results using the DEIM approximation of the nonlinearity.	117
5.1	Comparison of a nonlinear trajectory and the linear SSOP prediction using the transient operator. The trajectory is projected onto the first and third POD modes. Results for the second mode are qualitatively similar to those for the first.	122
5.2	Residual for the time-stepping approximation of the resolvent adjoint applied to the first SPOD mode, as a function of frequency. The spike at the vortex-shedding frequency goes to a value of roughly 80.	125

LIST OF ACRONYMS

BT	Balanced Truncation
CPU	Central Processing Unit
DEIM	Discrete Empirical Interpolation Method
DFT	Discrete Fourier Transform
DMD	Dynamic Mode Decomposition
ERA	Eigensystem Realization Algorithm
FFT	Fast Fourier Transform
FOM	Full-Order Model
HAVOK	Hankel Alternative View of Koopman
IFFT	Inverse Fast Fourier Transform
INS	Incompressible Navier-Stokes equations
LTI	Linear Time-Invariant
PDF	Probability Density Function
POD	Proper Orthogonal Decomposition
POD-G	POD-Galerkin projection
PSD	Power Spectral Density
ROM	Reduced-Order Model
SPOD	Spectral Proper Orthogonal Decomposition
SSA	Singular Spectrum Analysis
SSOP	Spectral Solution Operator Projection
SVD	Singular Value Decomposition
TKE	Turbulent Kinetic Energy

ABSTRACT

Simulations of physical processes are ubiquitous in science and engineering, but they can be prohibitively expensive, particularly when many cases must be run. The field of model reduction comprises a collection of methods with an ambitious goal: reduce the computational cost of simulations by orders of magnitude while maintaining tolerable accuracy. The standard framework for producing a reduced-order model (ROM) is to *(i)* find a low-dimensional encoding of the state of the dynamical system, then *(ii)* derive a dynamical system governing the evolution of this low-dimensional encoding. While model reduction has had many successes, new techniques are needed for the widespread adoption of ROMs. This thesis explores space-time model reduction, an alternative framework wherein *(i)* the entire trajectory of the state of the system over a time interval $[0, T]$ is encoded, then *(ii)* the degrees of freedom in the encoding are solved for simultaneously.

First, we investigate space-time encodings. We establish a connection between space-time proper orthogonal decomposition (POD) and the singular value decomposition of a particular Hankel matrix used in popular time-delay embedding methods. This connection provides a clear interpretation of the Hankel modes grounded in classical POD theory, and we gain insights into these modes by analyzing the equivalent discrete space-time POD problem. Based on these insights, we present several methods for improving the Hankel modes.

Second, the main contribution of this thesis is the development of a space-time ROM, complete with a space-time encoding scheme and a method to solve quickly and accurately for the encoding online. The method encodes trajectories using spectral POD (SPOD) modes, which provide a spatial basis at each temporal frequency. These modes have several properties that make them an ideal choice for space-time model reduction, including near-optimality for encoding long trajectories. The method solves for the degrees of freedom in the SPOD encoding using a novel technique, which we call spectral solution operator projection (SSOP): we project the analytical solution to the linearized governing equations onto the SPOD modes. This yields precomputable transfer functions that map the initial condition and forcing to the SPOD coefficients that result from the linearized dynamics. Nonlinearity is accounted for in this framework as an additional state-dependent forcing on the system, resulting in a nonlinear system of algebraic equations for the SPOD expansion coefficients.

We solve this system using a fixed-point iteration.

We demonstrate the capabilities of SSOP by testing it on linear and nonlinear Ginzburg-Landau systems, and an advection-diffusion system. These tests include cases where the testing conditions are qualitatively different from those used for training, such as using different system parameters and using non-statistically-stationary forcings. In all cases, we find that SSOP delivers orders-of-magnitude lower error than benchmark ROMs at similar cost. Indeed, the method achieves substantially lower error than the lower bound for space-only ROMs based on linear encodings. We then describe ongoing work applying SSOP to the incompressible Navier-Stokes equations. These equations pose additional challenges related to approximating the transfer functions involved in the method, and we discuss techniques for overcoming these challenges.

We conclude that SSOP is a promising avenue for further work on model reduction. While it requires more care in the precomputation steps than most existing methods, we view the reduction in error, particularly the near-exact recovery of the space-time encoding of the solution, as a significant result.

CHAPTER 1

Introduction

1.1 The need for model reduction

Computational models for physical phenomena are surprisingly expensive. Consider, for example, a two-dimensional simulation of the flow past a cylinder. Integrating the solution over many oscillations can require hours of CPU time, even at modest Reynolds number. Increasing the Reynolds number increases the range of scales, requiring a finer grid and shorter time step, and the cost grows accordingly. Making the simulation three-dimensional may further increase the cost by orders of magnitude. And this flow is among the simplest of practical interest.

The scientific and industrial applications for computational models often involve much more complex flows. The turbulent convection of the molten iron in the outer core of the earth, for example, contains scales ranging from meters to thousands of kilometers [43]. Many engineering applications also require simulations to be run quickly, either because the simulation results are needed promptly, or because many simulations must be conducted in succession. In model-predictive control, a simulation time of milliseconds can be required to inform real-time actuation input [55, 53, 80]. Aircraft manufacturers often run a simulation at each step in an optimization loop over possible airfoil designs [59, 79]. Techniques for uncertainty quantification often use Monte-Carlo methods which involve many simulations [10, 140], requiring each one to be cheap. While these examples are drawn from fluid mechanics, which is the focus of this thesis, the need for fast simulations extends across science and engineering.

Developing techniques to accelerate simulations is a broad goal, and there are many approaches to doing so including better hardware, more computing (parallelization), better numerical algorithms, and simplified physical models. The approach that is the focus of this thesis is reduced-order modeling.

The field of model reduction comprises methods that build a reduced, i.e., computationally

cheap, model using data from and/or the equations governing a model that is regarded as the ground-truth. The ground-truth model, hereafter called the full-order model (FOM), may be a high-fidelity simulation or an experiment, and while accurate it may be expensive to run. The goal of the reduced-order model (ROM) is to produce accurate predictions of the behavior of the FOM in new conditions and at low cost, thus alleviating the need to run the full-order model. Reduced-order models often require an up-front computational cost to construct, which usually involves generating and processing data from the FOM, and in many cases, doing additional calculations using the equations governing the FOM. While this thesis focuses on fluid mechanics, reduced-order models, including the ones developed herein, are formulated for general dynamical systems.

The goal of model reduction is an ambitious one, and the availability of general-purpose techniques to accelerate simulations would be of enormous value in the scientific and engineering fields that rely on expensive simulations. However, ROMs are not yet sufficiently robust for widespread use in science and engineering. The majority of ROMs operate within a particular approach, which we refer to as the space-only framework. While this framework has many successes, ROMs remain far from a general-purpose tool for accelerating simulations, and we argue that this motivates the investigation of alternative approaches. In this thesis we explore space-time model reduction, an interesting, but far less common approach that addresses some limitations of the space-only approach. In the remainder of this chapter, we describe the space-only framework in detail, introducing concepts and notation that are used throughout the thesis, and give an outlook on its use and limitations. We then describe the space-time framework, and situate the contributions of this thesis within the existing space-time literature.

1.2 The standard framework: space-only model reduction

We focus in this thesis on time dependent problems, though some ROMs are designed for time-independent problems. While some ROMs are constructed without explicit governing equations, many of the methods discussed below, and the methods developed in this thesis, make use of them. We consider full-order evolution equations of the form

$$\dot{\mathbf{q}} = \mathbf{h}(\mathbf{q}, t; \boldsymbol{\mu}), \quad (1.2.1)$$

The function $\mathbf{h}$ returns the time derivative of the state $\mathbf{q} \in \mathbb{C}^{N_x}$. It is a function of the state, time and any parameters $\boldsymbol{\mu} \in \mathbb{C}^{N_\mu}$ that define the system. In later chapters we assume a less general dependence of $\mathbf{h}$ on its arguments.

The great majority of model reduction methods use the following two-step approach: *(i)* find an efficient encoding of the state of the dynamical system, replacing the high-dimensional state space with a low-dimensional one, then *(ii)* derive a dynamical system for the encoding [8, 112]. The motivation for this approach is the following: the instantaneous state of the system of interest is often made up by a few prominent coherent structures, and moreover, the dynamics are dominated by the interactions of these structures. This allows the state to be represented accurately by a relatively small number of degrees of freedom, and ideally, for these degrees of freedom to be evolved accurately by a reduced system that ignores the impact of unretained features on the large-scale structures.

1.2.1 Encoding methods

Mathematically, the encoding scheme of the state is given by a pair of functions $\mathbf{e}^x : \mathbb{C}^{N_x} \rightarrow \mathbb{C}^r$ and $\mathbf{d}^x : \mathbb{C}^r \rightarrow \mathbb{C}^{N_x}$. The encoder $\mathbf{e}^x$ maps the state $\mathbf{q}$ to an r dimensional encoding $\mathbf{a} \in \mathbb{C}^r$, and the decoder $\mathbf{d}^x$ maps the encoding back to the state space. The encoding scheme is effective if the encoding is low-dimensional, i.e., $r \ll N_x$, and if the decoding of the encoded state is near the original state, i.e., $\mathbf{d}^x(\mathbf{e}^x(\mathbf{q})) \approx \mathbf{q}$ for states $\mathbf{q}$ that are realized by the system. The encoding scheme is usually derived from data from the FOM, though in some cases [86, 47, 58], it is derived from the governing equations.

In practice, it is standard to decompose the state in terms of a mean and fluctuation $\mathbf{q} = \bar{\mathbf{q}} + \mathbf{q}'$, and to formulate the encoding scheme for the fluctuation, i.e., $\mathbf{a} = \mathbf{e}^x(\mathbf{q}')$ and $\mathbf{q} \approx \bar{\mathbf{q}} + \mathbf{d}^x(\mathbf{a})$. We assume throughout that this decomposition has already been performed, so hereafter $\mathbf{q}$ denotes the fluctuation to the mean.

1.2.1.1 Linear encodings

Many common encodings are linear in that the encoding and decoding functions are both linear in their inputs. In this case, the encoder and decoder are matrices (if the state is a finite-dimensional vector), and the encoding/decoding process amounts to a projection onto the subspace spanned by the columns of the decoding matrix. Let the decoder be given by $\Phi \in \mathbb{C}^{N_x \times r}$, meaning the decoded state $\tilde{\mathbf{q}} \in \mathbb{C}^{N_x}$ is

$$\tilde{\mathbf{q}} = \Phi \mathbf{a}. \quad (1.2.2)$$

With this decoder, the best encoder is the one that minimizes the error

$$\|\mathbf{q} - \Phi \mathbf{a}\|_x^2. \quad (1.2.3)$$

It is common to use a weighted L^2 norm, given by $\|\mathbf{q}\|_x^2 = \mathbf{q}^* \mathbf{W} \mathbf{q}$, where $\mathbf{W}$ is a positive-definite matrix, and the subscript, used throughout the thesis, is used to distinguish this space-only norm from a space-time one introduced later. With this norm, the optimizing encoded state is given by $\mathbf{a} = (\Phi^* \mathbf{W} \Phi)^{-1} \Phi^* \mathbf{W} \mathbf{q}$, meaning the optimal encoder is given by the (weighted) pseudo-inverse of the decoding matrix. The encoding/decoding process then amounts to an orthogonal projection of the state onto the span of Φ , i.e., $\tilde{\mathbf{q}} = \mathbf{P} \mathbf{q}$ where $\mathbf{P} = \Phi(\Phi^* \mathbf{W} \Phi)^{-1} \Phi^* \mathbf{W}$. The error in the representation only depends on this span, so it is convenient to work with orthonormal decoders, i.e., ones where $\Phi^* \mathbf{W} \Phi = \mathbf{I}$. In this case, the encoding is

$$\mathbf{a} = \Phi^* \mathbf{W} \mathbf{q} \quad (1.2.4)$$

and the projection matrix is $\mathbf{P} = \Phi \Phi^* \mathbf{W}$. Another convenient property of these encoding schemes is that the error is given by $\|\mathbf{q} - \Phi \mathbf{a}\|_x^2 = \|\mathbf{q}\|_x^2 - \|\mathbf{a}\|_2^2$, which may be interpreted as a generalized form of Parseval's theorem. Familiar examples of such orthonormal linear encodings include representation by a truncated Fourier series, and representation by a truncated set of normal eigenvectors of a linearized system.

Proper orthogonal decomposition (POD) modes [96, 52, 76, 77] provide the most common basis for linear encodings in space-only model reduction. POD is used across many fields. Its use in model reduction derives from that fact that it is the optimal linear encoding; it minimizes the average error between the state and the decoder approximation thereof. POD will be discussed in detail in Chapter 2.

An advantage of linear encodings is that the encoded state $\mathbf{a}$ is interpretable as the coefficients of the state $\mathbf{q}$ in the basis given by the columns of the decoding matrix. The simplicity of linear encodings can also be helpful when deriving the reduced dynamical system. However, linear encodings can be inefficient for representing the state in certain problems [71, 6]. Consider, for example, the one-dimensional advection equation $\dot{u}(x, t) = -\frac{\partial}{\partial x} u(x, t)$, with a step-function initial condition $u(x, 0) = 1$ for $x < 0$ and $u(x, 0) = 0$ for $x \geq 0$. The step function simply translates in the positive direction at constant speed, but despite this simple behavior, linear encodings of the solution require many degrees of freedom for even modest accuracy. It has been shown [90] that with any linear encoding scheme, the L^2 norm of the error in the representation of the solution is at least $\frac{C}{\sqrt{r}}$ for some time. Therefore, the accuracy in the representation only converges at a rate of $\frac{1}{\sqrt{r}}$ with the number of modes r . While such precise error bounds are only available for a few problems, this slow decay in error with r is observed empirically for advection-dominated problems with sharp gradients [98].

1.2.1.2 Nonlinear encodings

The limitations of linear encodings has motivated the development of nonlinear encodings. The additional flexibility afforded by nonlinearity can enable more efficient representations of the state. In the one-dimensional advection problem described above, for example, an encoding that simply returns the location of the discontinuity can achieve an exact representation of the solution using only one degree of freedom. Encoding schemes where the encoder is linear, but the decoder is a quadratic function of its inputs were first developed for model reduction in the structural dynamics community [58, 114], and these schemes have been shown to improve the convergence for advection-dominated problems [38]. Encoding schemes that generalize projection onto eigenmodes to nonlinear systems have also been developed [47]. These schemes also involve a linear encoder and a polynomial decoder. Perhaps the most common nonlinear encoding schemes – both in model reduction and other fields — are autoencoders. These parametrize both $\mathbf{e}^x$ and $\mathbf{d}^x$ as neural networks, and train them to minimize the error in the representation of the state using data from the problem of interest. Autoencoders were first developed in the machine learning community [113, 49], but have been employed in model reduction [66, 48, 71]. While nonlinear encoding schemes are usually more efficient than linear ones in that they achieve lower error with the same number of degrees of freedom, nonlinearity can make the process of learning the dynamics more difficult.

1.2.2 Producing the reduced dynamical system

With an encoding defined, the second step in the standard framework is to produce a dynamical system governing the encoded state of the form

$$\dot{\mathbf{a}} = \mathbf{g}(\mathbf{a}, t; \boldsymbol{\mu}), \quad (1.2.5)$$

where $\mathbf{g} : \mathbb{C}^r \rightarrow \mathbb{C}^r$.

1.2.2.1 Projection with linear encodings

With a linear encoding the most common approach is Galerkin projection. Substituting $\mathbf{q} \approx \Phi \mathbf{a}$ into (1.2.1) gives

$$\Phi \dot{\mathbf{a}} = \mathbf{h}(\Phi \mathbf{a}, t; \boldsymbol{\mu}), \quad (1.2.6)$$

which can be used to obtain an equation giving $\dot{\mathbf{a}}$ in terms of $\mathbf{a}$ and t , as desired. However, (1.2.6) is overdetermined, as there are N_x equations and only r unknowns. The Galerkin

approach is to left-multiply both sides by $\Phi^* \mathbf{W}$, which gives

$$\dot{\mathbf{a}} = \Phi^* \mathbf{W} \mathbf{h}(\Phi \mathbf{a}, t; \boldsymbol{\mu}). \quad (1.2.7)$$

This is equivalent to minimizing the norm of the residual $\|\Phi \dot{\mathbf{a}} - \mathbf{h}(\Phi \mathbf{a}, t; \boldsymbol{\mu})\|_x^2$ over $\dot{\mathbf{a}}$, and it amounts to an orthogonal projection of the full-order right-hand side $\mathbf{h}$ onto the span of the columns of Φ .

The overdetermined system (1.2.6) can also be closed by performing an oblique projection, or, equivalently, by left-multiplying both sides of (1.2.6) by a different matrix $\Psi^* \mathbf{W}$, where $\Psi \in \mathbb{C}^{N_x \times r}$ is a different set of basis vectors. This is called Petrov-Galerkin projection, and it results in the following reduced system:

$$\dot{\mathbf{a}} = (\Psi^* \mathbf{W} \Phi)^{-1} \Psi^* \mathbf{W} \mathbf{h}(\Phi \mathbf{a}, t; \boldsymbol{\mu}). \quad (1.2.8)$$

Galerkin projection is recovered as the case where $\Psi = \Phi$. This choice of Ψ can have a surprising impact on the quality of the resulting ROM, with good choices leading to substantially more accuracy [16, 17].

1.2.2.2 Projection with nonlinear encodings

A similar approach can be taken when the encoding is nonlinear. Substituting $\mathbf{q} \approx \mathbf{d}^x(\mathbf{a})$ into (1.2.1) and applying the chain rule gives

$$\mathbf{J}(\mathbf{a}) \dot{\mathbf{a}} = \mathbf{h}(\mathbf{d}^x(\mathbf{a}), t; \boldsymbol{\mu}), \quad (1.2.9)$$

where $\mathbf{J}(\mathbf{a}) = \frac{\partial}{\partial \mathbf{a}} \mathbf{d}^x(\mathbf{a}) \in \mathbb{C}^{N_x \times r}$ is the Jacobian of the decoder about the current encoded state. This is again an overdetermined system. The residual $\|\mathbf{J}(\mathbf{a}) \dot{\mathbf{a}} - \mathbf{h}(\mathbf{d}^x(\mathbf{a}), t; \boldsymbol{\mu})\|_x^2$ is minimized by left-multiplying by the pseudo-inverse of $\mathbf{J}(\mathbf{a})$. This can be interpreted as Galerkin projection generalized to the nonlinear encoding: it is an orthogonal projection of the full-order right-hand side $\mathbf{h}$ onto the tangent space of the manifold given by the decoder at the current encoded state. Petrov-Galerkin projection generalizes to the nonlinear case as well, and it entails an oblique projection onto the same tangent space [71].

An important difference relative to the linear case is that the projection changes as the system evolves, since the Jacobian depends on $\mathbf{a}$, and this can complicate the precomputation process.

1.2.2.3 Precomputations and hyper-reduction

The purpose of deriving the reduced system $\dot{\mathbf{a}} = \mathbf{g}(\mathbf{a}, t; \boldsymbol{\mu})$ is to arrive at a system that is cheap to integrate forward in time. However, as written, the systems above are actually similarly expensive to the FOM. Evaluating the function $\mathbf{g}$ involves decoding the state, evaluating the full-order right-hand side $\mathbf{h}$, and then performing the projection. Each of these operations scales (at least) with N_x , the large-scale state dimension that ROMs are designed to avoid, and thus there is no speedup as written. With the right precomputations — offline calculations involved in building the ROM before using it online — these operations can usually be avoided, although doing so can require additional approximations in some cases.

Consider the linear time-invariant (LTI) system

$$\dot{\mathbf{q}} = \mathbf{A}\mathbf{q} + \mathbf{B}\mathbf{f}(t), \quad (1.2.10)$$

with $\mathbf{A} \in \mathbb{C}^{N_x \times N_x}$ and $\mathbf{B} \in \mathbb{C}^{N_x \times N_f}$. With a linear encoding scheme given by the decoder Φ , Galerkin projection gives the following reduced system:

$$\dot{\mathbf{a}} = \mathbf{A}^r \mathbf{a} + \mathbf{B}^r \mathbf{f}(t). \quad (1.2.11)$$

The matrices $\mathbf{A}^r = \Phi^* \mathbf{W} \mathbf{A} \Phi \in \mathbb{C}^{r \times r}$ and $\mathbf{B}^r = \Phi^* \mathbf{W} \mathbf{B} \in \mathbb{C}^{r \times N_f}$ can be precomputed, and the resulting system can then be integrated forward in time at a cost that scales with r (and N_f) rather than N_x .

A similar technique can be used for a quadratically nonlinear system of the form $\dot{\mathbf{q}} = \mathbf{A}\mathbf{q} + \mathbf{b}(\mathbf{q}, \mathbf{q}) + \mathbf{B}\mathbf{f}(t)$, where $\mathbf{b} : \mathbb{C}^{N_x} \times \mathbb{C}^{N_x} \rightarrow \mathbb{C}^{N_x}$ is a bilinear function. By precomputing both the matrices from the linear case and the tensor $\mathbf{N}_{ijk} = \phi_i^* \mathbf{W} \mathbf{b}(\phi_j, \phi_k)$, the reduced system can be written as

$$\dot{\mathbf{a}} = \mathbf{A}^r \mathbf{a} + \sum_{j,k=1}^r \mathbf{N}_{ijk} a_j a_k + \mathbf{B}^r \mathbf{f}(t). \quad (1.2.12)$$

Evaluating the right-hand side of this system now scales with r^3 rather than N_x , and the additional precomputation scales with $N_x r^2$.

In principle, one could approximate a general nonlinearity as a sum of polynomial terms, carrying out the necessary precomputations for each one. However, this quickly becomes too expensive — the online phase scales with r^{k+1} for a k -th order polynomial nonlinearity. In practice, for third-order and higher nonlinearities, so-called hyper-reduction techniques are used. The most common among these approximate the nonlinear term with a precomputed

reduced basis. The nonlinear term is then sampled at p points online at each time step, using the result to obtain the basis coefficients. With such methods, the evaluation of the nonlinear term scales with pr rather than N_x . Most often, the basis used for the nonlinear terms is POD modes obtained from data for the nonlinear term, and the sample points are obtained by a greedy selection process called the discrete empirical interpolation method (DEIM) [21].

In the examples above, $\mathbf{h}$ does not depend on parameters, and parameter dependence can complicate the process of precomputing the necessary matrices for fast online model evaluation. The objective is to precompute quantities such that the ROM operators can be formed quickly given a new parameter vector, i.e., without any computations that scale with N_x . While this is often not achievable [8], one special, but common, case which can be dealt with elegantly is affine parameter dependence. Consider the LTI system $\dot{\mathbf{q}} = \mathbf{A}(\mu)\mathbf{q} + \mathbf{B}\mathbf{f}(t)$, where $\mathbf{A}$ has the following dependence on the scalar parameter μ ,

$$\mathbf{A}(\mu) = \mathbf{A}_0 + \mu\mathbf{A}_1. \quad (1.2.13)$$

By precomputing $\mathbf{A}_0^r = \Phi^* \mathbf{W} \mathbf{A}_0 \Phi$ and $\mathbf{A}_1^r = \Phi^* \mathbf{W} \mathbf{A}_1 \Phi$, the reduced linear operator $\mathbf{A}^r(\mu)$ involved in a Galerkin ROM for this system can be formed as $\mathbf{A}^r(\mu) = \mathbf{A}_0^r + \mu\mathbf{A}_1^r$ without any computations that scale with N_x . Affine dependence is the more general case when $\mathbf{A}(\mu)$ can be written as a linear combination of matrices $\mathbf{A}_i$, each weighted by a scalar function $\zeta_i(\mu)$ of the parameters (see (4.1.22)). Full-order model operators often depend affinely on physical parameters like Reynolds number, conductivity, or wave speed, which can be exploited to obtain efficient ROMs. On the other hand, geometric parameters usually lead to non-affine dependence, requiring further approximations [8].

1.2.2.4 Learning the dynamics from data

An alternative to projection-based model reduction is to obtain the operators that define the ROM directly from data. These methods obtain data for the time derivative of the encoded state $\dot{\mathbf{a}}$ via finite differences, then solve a regression problem to find the operators in the reduced system. Operator inference [100, 69, 37] exploits known structure of the governing equations to obtain a reduced system: if the equations are known to be quadratically non-linear, for example, then operator inference entails solving a least-squares problem to find the operators in (1.2.12). Operator inference is used with linear encodings, usually POD, to ensure that the functional form of the reduced system is the same as that of the full-order system. Learning methods which do not exploit any known structure of the governing equations also exist, and these are often used in conjunction with autoencoders [84, 40, 138].

These methods can take advantage of the additional reduction enabled by the nonlinear encoding, but they sacrifice the benefit of known structure in the governing equations.

Methods that build a ROM without use of governing equations, i.e., that treat the source of data as a “black box”, are called non-intrusive. These are particularly useful when the governing equations are either unavailable, as in an experiment, or too cumbersome to work with, as can be the case in complicated simulations. However, it is generally expected that these non-intrusive ROMs have trouble generalizing to test conditions that differ substantially from the data used to train them.

1.2.3 Time-delay embedding methods

The methods described above all encode the instantaneous state of the system. An alternative is to incorporate temporal information into the encoding. Time-delay embedding methods do this by encoding the state of the system using data from some number d previous time steps $\mathbf{q}_i, \mathbf{q}_{i-1}, \mathbf{q}_{i-2}, \dots, \mathbf{q}_{i-(d-1)}$. This idea goes back to Takens [129], who showed that even a scalar time series of sufficient length can encode the state of the system with no approximation. Reduced-order models have been developed based on this idea, and many [12, 61, 13, 20, 27, 67] obtain reduced coordinates by taking the left singular vectors of a Hankel matrix, which is a data matrix with time delay embeddings as columns. While the encoding of the state in these methods incorporates both space and time, the dynamics are still Markovian, and the reduced system is still in the form of an evolution equation for the encoded state. The connection between these time-delay-based encodings and space-time proper orthogonal decomposition is the focus of Chapter 2.

1.2.4 Outlook on model reduction

POD-Galerkin projection is perhaps the most widely used approach, and it serves as an important benchmark against which to compare new approaches. It has been successfully applied [4, 109, 88, 111] to a range of problems, particularly those in which the dynamics are dominated by a small number of energetic coherent structures. A particular Petrov-Galerkin approach, called least-squares Petrov-Galerkin [16, 15], has been an important advance, improving the accuracy and applicability of projection-based ROMs. This approach has led to application of projection-based ROMs to problems such as flow past the Ahmed body [17] and airplane geometries [42], culminating in the development of production-level software such as **Pressio** [107]. Though non-intrusive methods like operator inference generally do not improve on the accuracy of projection-based methods, and often result in poorer generalizability outside the training data, they do lower the barrier to applying ROMs,

which is quite valuable. Despite these successes, adoption of ROMs is somewhat limited, and a number of issues remain. Projection-based ROMs often lack stability [104, 9]. They can also require a large number of modes to achieve tolerable accuracy, and the resulting ROM can thus be expensive to integrate.

A large literature has developed to address these issues within the space-only framework. For example, stabilization schemes [115, 64] and closure models [57, 105] have been investigated to address the issue of stability. In addition to the nonlinear encodings described above, methods requiring fewer modes have been developed by using different encodings or hyperreduction schemes tailored to different regions of parameter space [103, 99]. Similarly, methods that update the encoding basis in the online phase [97, 54] have been applied to complex problems, albeit at greater cost than typical ROMs. Given the fact, however, that the majority of existing ROMs share the same basic space-only architecture, we contend that there is value in exploring fundamentally different formulations. The space-time approach described in the next section is a promising, if nascent, alternative.

1.3 Space-time model reduction

In this section, we describe the space-time framework for model reduction, which is the focus of this thesis.

1.3.1 The space-time framework

Space-only ROMs are motivated by the observation that the state of the system at a given time is often made up of a few prominent coherent structures. A broader observation is that the evolution of the state over a time interval is often made up of a few spatially and temporally coherent motions, or spatiotemporal coherent structures. Whereas the former observation suggests that an efficient encoding of the instantaneous state is possible, the latter suggests that an efficient encoding of the trajectory of the state over a time interval $[0, T]$ is possible. An encoder/decoder framework for trajectory representation, analogous to the one described above for state representation, is given in Chapter 3. In short, the encoder becomes a function that maps the trajectory on $[0, T]$, discretized in both space and time, to a low-dimensional vector, and the decoder maps this vector back to the vector space of trajectories. Since both the spatial and temporal dimensions are reduced in tandem, this is called a space-time reduction. The space-time analog of POD is called space-time POD [76, 33], which minimizes a certain mean L^2 error over the interval.

Space-time encodings can be used for model reduction by modifying the two-step process

used in space-only model reduction as follows:

- (i) Find an efficient encoding for the entire trajectory of the state over some time interval of interest. With the entire evolution of the state represented in the encoding, the degrees of freedom that represent the solution no longer evolve in time.
- (ii) Solve for these static degrees of freedom either by solving a system of simultaneous equations or an optimization.

As we show in Chapter 2, many time-delay embedding methods make use of an encoding that is an approximation of space-time POD. However, these methods use these embeddings to encode the state, rather than the trajectory, and the resulting ROMs retain the Markovian structure of space-only ROMs.

As can be seen in Figure 1.1, the potential advantage of the space-time approach over the space-only one is that solutions to problems of interest exhibit spatiotemporal coherent structures, which offer an extremely efficient means of describing the dynamics. In a space-only ROM, one is nonetheless solving for an encoding of the trajectory – the space-only encoding of the state at all time steps may be viewed as such – but this trajectory encoding is far from optimal. Indeed, the space-only POD coefficients, for example, are highly correlated from one time step to the next, so they provide a redundant (i.e., suboptimal) encoding of the entire trajectory. Thus, with the same total degrees of freedom, the space-time framework offers the potential for substantial additional accuracy by leveraging spatiotemporal correlations.

The key questions for a space-time ROM are the following. (1) To what extent is lower representation error achieved by the space-time encoding? (2) Can one solve for the space-time encoding coefficients accurately? POD-Galerkin projection (POD-G), e.g., does not exactly recover the projection of the solution onto the POD modes due to the influence of unclosed terms. To what extent does this persist for a space-time approach? (3) Can the system of equations in the space-time ROM be solved in similar time to the time integration required in the space-only ROM? In other words, does the space-time approach take the same (or less) CPU time as the space-only approach for the same number of degrees of freedom? If the answers to these are all ‘yes’, then the space-time ROM will be much more accurate than a space-only ROM at the same CPU time.

1.3.2 Existing space-time ROMs

Space-time model reduction is far less developed than the space-only variant. The space-time encodings used in many existing space-time ROMs are formed by attaching a time

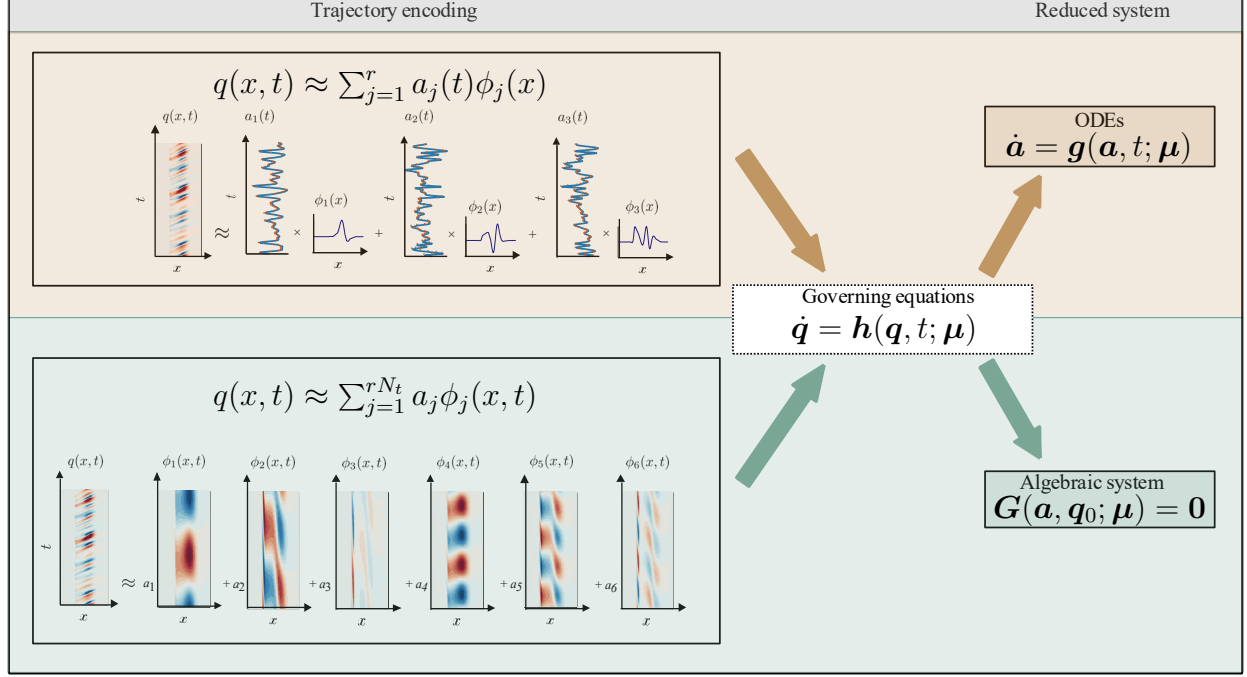


Figure 1.1: Space-only and space-time reduced-order modeling frameworks (top and bottom panels, respectively). The space-time encoding leverages the spatiotemporal correlations present in the trajectory $q(x, t)$ to achieve greater efficiency.

dependence to space-only POD modes [25, 24, 68, 50]. In most cases, the time dependence for a given POD mode is tailored to that mode, which is accomplished by taking the singular value decomposition (SVD) of a data matrix containing as its columns different temporal evolutions of the associated POD coefficient. In other cases [25], the same temporal basis is used for each POD mode. To find the coefficients, Refs. [25, 24, 68] minimize the square norm of the residual between the time derivative of the decoded trajectory $\tilde{q}(t)$ and the decoding inserted into the right-hand side, integrated over the time interval of interest. That is, they solve the following optimization

$$\min_{\mathbf{a}} \int_0^T \left\| \frac{d}{dt} \mathbf{d}^{x,t}(\mathbf{a}, t) - \mathbf{h}(\mathbf{d}^{x,t}(\mathbf{a}, t), t; \boldsymbol{\mu}) \right\|_x^2 dt, \quad (1.3.1)$$

where $\mathbf{d}^{x,t}$ is the space-time decoder and $\mathbf{a}$ is now the encoding of the entire trajectory (the notation will be made more precise in Chapter 3). Alternatively, Ref. [50] trains a machine learning model that outputs the encoding given inputs to the system.

In addition to the work in this thesis and the articles it is based on [32, 35], several methods have been proposed that use spectral proper orthogonal decomposition (SPOD) modes for space-time model reduction. The following are methods closely related to the proposed

approach. Refs. [74, 130] developed Galerkin and Petrov-Galerkin SPOD-based ROMs, respectively, for linear systems. These models are only capable of handling systems in which the solution is known to be periodic. Recently, Ref. [73] proposed a nonlinear SPOD-based model, somewhat similar to the Galerkin models in Refs. [74, 130], but which was generalized to handle nonlinearities. Ref. [73] then applied this model to find periodic solutions in a canonical incompressible viscous flow. This latter model can be seen as a spatially reduced version of the harmonic balance method [46], a method which derives a relation between the various frequencies of a temporally periodic solution to the Navier-Stokes equations. The harmonic balance method has been applied in turbomachinery problems [46, 45], where the solution is assumed to be periodic with the fundamental frequency given by the blade-passing frequency, and has also been used to study boundary layer transition [106]. In both, just a few harmonics are retained. All the methods above require a periodic solution, something that might be natural to expect of methods based on the Fourier transform. However, SPOD modes are effective at representing both periodic and aperiodic trajectories, and since the latter is far more common, there is a greater need for reduced-order models for that case. The approach proposed in Chapters 3 and 4 can handle both periodic and aperiodic problems.

1.3.3 Proposed approach

In Chapters 3 and 4, we present a novel space-time model reduction method called spectral solution operator projection (SSOP). The method makes the following choices for the two steps that define a space-time ROM:

- (i) The method uses spectral POD (SPOD) modes to encode the trajectory. SPOD provides a spatial basis at each temporal frequency that is tailored to the structures that appear in the solution at that frequency. As we show analytically in Chapter 2, this basis converges to space-time POD in a certain limit, implying that it is near-optimal for encoding trajectories. Moreover, we show that it has a number of other properties that together make it an ideal choice for space-time encodings for model reduction. SPOD has been used before for model reduction [74, 130], but these works have not formulated effective methods for solving online for the coefficients, leading to large errors.
- (ii) To solve online for the space-time encoding, we apply a space-time projection to the fundamental solution operator for linear systems. In the case of linear governing equations, this results in transfer functions that map the initial condition and forcing to the SPOD encoding; in the nonlinear case, it results in a nonlinear system to solve for the coefficients. The benefit of this global-in-time approach, as opposed to minimizing a

local-in-time residual as in (1.3.1), is that the coefficients obtained are nearly exactly, with errors arising from approximating the transfer functions. The approach is novel, and it is among the most significant contributions of this thesis.

Returning to the three key questions for space-time ROMs listed above, we find the method is based on an efficient encoding, solves accurately for the encoding coefficients, and does so quickly.

1.4 Thesis overview

The key contributions of this thesis are the following.

- Chapter 2: Established a connection between space-time POD and the SVD of a Hankel matrix, the latter of which is used in many time-delay embedding methods. Since space-time POD is well-understood, this connection is useful for understanding, and in some cases improving, the properties of model reduction methods based on delay embedding.
- Chapter 3: Developed a novel approach to space-time model reduction for linear systems: Rather than minimizing a local-in-time residual over the coefficients, we project the fundamental solution to linear systems onto the space-time modes used to encode the trajectory. The result is a transfer function from the initial condition and forcing inputs to the space-time coefficients that encode the trajectory. The benefit of this approach is that the coefficients obtained are nearly exact, with errors arising from approximating the transfer functions. This is in contrast to the errors in previous approaches to both space-only and space-time model reduction, where the accuracy of retained coefficients is impacted by unretained ones.
- Chapter 3: Derived expressions for the transfer functions in the case of spectral POD modes, coining the resulting method Spectral Solution Operator Projection. Leveraging the fact that SPOD modes have analytic (complex-exponential) time dependence, the temporal portion of the space-time projection can be done analytically. Doing so, however, requires careful formulation, and two previous SPOD-based ROMs resulted in $\mathcal{O}(1)$ errors due to subtleties with the periodicity of SPOD modes. Our formulation avoids these issues.
- Chapter 4: Extended the SSOP framework to nonlinear systems by treating the nonlinearity as an additional forcing on the system. Crucially, this additional forcing depends

on the solution itself, i.e., the nonlinearity at a given time is a function of the solution at that time. The nonlinear coupling is handled by adding over a sparsified set of triadic interactions between SPOD modes in the case of a quadratic nonlinearity, and by using an adaptation of a standard hyper-reduction method otherwise.

- Chapter 4: Developed a fixed-point iteration approach to solve the nonlinear system of equations for the SPOD coefficients. This approach can be interpreted as an iterative correction to the linear solution.
- Chapters 3 and 4: Tested SSOP on both linear and nonlinear Ginzburg-Landau problems, as well as a two-dimensional advection-diffusion system, all for unseen data. In all cases, the method is shown to produce orders-of-magnitude lower error than both benchmark space-only ROMs, and the lower bound for space-only ROMs based on linear encodings. With this reduction in error, SSOP is comparable in computational cost to the benchmarks.
- Chapters 3 and 4: Tested the predictive capacity of SSOP using test data qualitatively different from the training data. This includes testing on trajectories generated with non-statistically-stationary forcings (and training on statistically stationary ones), and testing on different system parameter values than those used for training. In all cases, the method achieved substantially lower error than the benchmarks at similar cost.
- Chapter 5: Presents preliminary work on applying SSOP to the incompressible Navier-Stokes equations, with the goal of an open-source implementation in the spectral element code Nek5000 for community use.

The material in Chapters 2, 3, and 4 is based on Refs. [33, 32, 35], respectively. The material in Chapter 5 is based on ongoing work that is not yet submitted for publication. In addition to the work presented in this thesis, the author’s doctoral research has included work on the phenomenon of transient growth in hydrodynamic stability theory. This work is not included in this thesis, but is published in Ref. [34], and developed further in Ref. [44].

CHAPTER 2

Space-Time Proper Orthogonal Decomposition and the Hankel Matrix

The aim of this chapter is twofold. First, we describe three variants of proper orthogonal decomposition, space-only, spectral, and space-time. These methods are integral to the remainder of this dissertation, with spectral POD being the basis for the model reduction approach developed in the chapters that follow, and the other variants underlying important points of comparison. Second, we establish a connection between space-time POD and the singular value decomposition of a particular Hankel matrix, which is used in various model reduction methods. This connection provides insight into the properties of these Hankel-based methods, and in some cases, guidance about how to improve the modes that underlie them.

Originally introduced to the fluid dynamics community by Lumley [76] in 1967, proper orthogonal decomposition is known by a variety of names in other areas including principal component analysis, Karhunen-Loève decomposition, and empirical orthogonal functions. In POD, the flow data is analyzed statistically, and the objective is to search for the modes that most efficiently represent the data. Specifically, POD modes are defined to minimize the reconstruction error, as measured by the average square inner product, compared to any other basis of the same dimension. As introduced by Lumley, the most general version of POD seeks to describe the time evolution of the flow for a prespecified window of time, so the basis functions are functions both of space and of time. This general version is called space-time POD, and the reconstruction of a flow over the time window consists of these basis functions multiplied by constant coefficients. Space-time POD has been used infrequently in the literature. Notable exceptions include application of space-time POD to optimally describe transients [41], generalize dimension reduction methods [108], and study acoustic intermittency in the form of bursts in jets via conditioning [120, 119].

Following the work of Sirovich [127] and Aubry [3], today the most popular form of POD involves modes that are functions of space only. To represent a time-dependent flow, these

modes are multiplied by time-varying coefficients. We refer to this form of POD as space-only POD [131], but note that it is often referred to simply as POD in the literature. Space-only POD has been used extensively to form Galerkin-based reduced-order models [4, 111, 112], reduce physically meaningful structures from flow data [51, 85], and reduce the data needed to store flow data [102]. We will show that space-only POD can be understood within the more general space-time POD framework as the limit as the time interval on which the optimization problem is defined goes to zero.

An increasingly popular variant of POD is spectral POD (SPOD). While also introduced in the original work of Lumley [76, 77], this form of POD was rarely employed until recently [131, 18, 128, 122]. Here, the objective is to optimally represent statistically stationary flows in the frequency domain. At each frequency, SPOD provides a set of spatial modes that capture the portion of the flow at that frequency more accurately, on average, than any other basis of the same order. SPOD can be formulated as the limit of space-time POD as the time interval on which the modes are defined goes to infinity.

Many other techniques exist for analyzing time-series data. At the heart of several of them, especially ones popular in the dynamical systems community, is analysis of a Hankel matrix. A Hankel matrix has constant skew diagonals, i.e., the (i, j) entry of the matrix only depends on $i + j$. Common applications include system identification and minimal system realization [30], and the use of the Hankel matrix in this context goes back to the 1960s [124]. The Hankel matrix plays a key role in singular spectral analysis (SSA) [134] and in the eigensystem realization algorithm (ERA) [61]. Recently, the Hankel matrix has been used in the context of fluid dynamics. It is central to balanced truncation [86], which was made scalable to fluid dynamics problems with balanced POD [137, 110]. The Hankel matrix is also used in a variant of dynamic mode decomposition (DMD) [117] called Hankel DMD [2], where the goal is to extract the spectrum and modes of the Koopman operator for some dynamical system by performing DMD on a Hankel matrix of observables.

The entries of the Hankel matrix are taken from time series data, and moving down or right along the columns or rows of the matrix corresponds to moving forward in time. Therefore, the columns of the Hankel matrix are delay embeddings of the dynamical system that produced the time series. Delay embedding, which goes back to work from the 1980s [92, 129], is a method of encoding the state of a dynamical system by recording the time history of one (or a few) of its observables. Intuitively, the state of an n -dimensional system should be determined by n independent observables, which don't need to be the original degrees of freedom of the system [92]. This idea was later made rigorous by Takens [129], and, under weak conditions, $2n + 1$ entries of a time series are needed to determine the state of the system.

A recurring theme in applications of the Hankel matrix in dynamical systems is the singular value decomposition (SVD). Both SSA and ERA obtain their bases from the SVD of the Hankel matrix. In balanced truncation and balanced POD, the Hankel singular values are used. In the Hankel alternative view of Koopman (HAVOK) framework [13], the SVD of a Hankel matrix of data is used in order to form a low-rank linear model of the dynamics on some chaotic attractor. The modes in their linear model are the left singular vectors of the Hankel matrix formed from a time series of the dynamical system. Convolutional coordinates, used to represent the state of a dynamical system at a particular time in terms of its representation in some predefined temporal basis, can be defined in terms of the continuous SVD (Schmidt decomposition) of a continuous Hankel matrix [65]. The left singular vectors of the Hankel matrix have recently been called principal component trajectories and used for control [27].

The connection between space-time POD and the Hankel singular modes is established by observing that the Hankel matrix multiplied by its Hermitian transpose provides an approximation of the space-time correlation matrix, and the eigendecomposition of this matrix defines discrete space-time POD modes. Analyzing this space-time correlation matrix leads to insights into the modes and, in some cases, guidance about how to improve them. The Hankel modes are not the only approximation one can form of the space-time POD modes using the available data, and in many cases, they are not the most practical approximation.

With a surplus of available data, a sufficiently accurate space-time correlation matrix can be formed by throwing out many of the columns before taking the SVD, which can drastically reduce the cost and memory requirements of obtaining the modes. In fact, unless data is quite scarce, we show that it is inadvisable to form the entire Hankel matrix to obtain the modes for large systems. Discarding columns works because the Hankel matrix is a data matrix of trajectories, but each trajectory is very similar to the adjacent ones in the Hankel matrix, so a given trajectory provides little additional information. In the opposite limit, i.e., where data is severely limited, we derive a more accurate approximation of the space-time POD modes than that provided by the Hankel matrix SVD. The approximation works by exploiting the ergodicity of the system (ergodicity is assumed in forming the Hankel matrix) to form a more accurate space-time correlation matrix. We show that these modes are substantially more accurate for low dimensional systems than Hankel modes when data is severely limited as compared to converged Hankel modes.

The connection to space-time POD also makes clear the assumed inner product, which defines the sense in which the Hankel modes are optimal and clarifies the impact of the time step between successive snapshots on the approximation. Additionally, we show that the height and width of the Hankel matrix determine the extent to which temporal correlation

is accounted for in defining the modes, and the convergence of the correlation, respectively. Finally, we show that in the limits of short and long delays, space-only and spectral POD are recovered, respectively.

2.1 Space-only POD

In this section on space-only POD, and in the ones that follow on spectral and space-time POD, we first introduce the continuous formulation of the POD method. In the continuous setting, we describe the modes in terms of expected values over ensembles. We then describe the discrete formulation of the methods, and in this setting, the modes are obtained from data and the expected values become sample averages over the realizations within the data. We view the latter as a discrete and finite-data approximation of the former.

2.1.1 Formulation

Space-only POD modes are vectors that form an efficient basis for snapshots of the flow $\mathbf{q}(\mathbf{x})$. These snapshots are thought of as vectors in a Hilbert space with an inner product $\langle \cdot, \cdot \rangle_x$, and the first POD mode is the direction in this space with along which the projection of $\mathbf{q}(\mathbf{x})$ has the most variance. That is, the first space-only POD mode $\phi_1(\mathbf{x})$ is the solution to the following optimization problem,

$$\lambda[\phi(\mathbf{x})] = \frac{1}{\|\phi(\mathbf{x})\|_x^2} \mathbb{E} [|\langle \mathbf{q}(\mathbf{x}), \phi(\mathbf{x}) \rangle_x|^2], \quad (2.1.1a)$$

$$\phi_1(\mathbf{x}) = \arg \max \lambda[\phi(\mathbf{x})]. \quad (2.1.1b)$$

Here, the expectation operator $\mathbb{E}[\cdot]$ acts over time, and λ is a functional which takes any function as input (in the appropriate function space) and returns the variation in the flow along it. The first POD mode ϕ_1 maximizes this functional. Typically, the modes are normalized to unity, but for clarity we will keep the magnitude in the formulae. The norm $\|\cdot\|_x$ is defined in the usual way by the inner product, and the inner product takes the form

$$\langle \mathbf{q}_1(\mathbf{x}), \mathbf{q}_2(\mathbf{x}) \rangle_x = \int_{\Omega} \mathbf{q}_2^*(\mathbf{x}) \mathbf{W}(\mathbf{x}) \mathbf{q}_1(\mathbf{x}) d\mathbf{x}, \quad (2.1.2)$$

where $\mathbf{W}$ is a weight matrix and Ω is the spatial domain of interest. The weight matrix is often used to make the norm correspond to some physical definition of energy, e.g., turbulent kinetic energy, or to give preference to certain flow variables or regions of the flow. In solving (2.1.1b), it is helpful to rewrite (2.1.1a) as an inner product with the correlation tensor

[76, 109],

$$\lambda[\phi(\mathbf{x})] = \frac{1}{\|\phi(\mathbf{x})\|_x^2} \left\langle \left\langle \phi(\mathbf{x}'), \mathbf{C}(\mathbf{x}, \mathbf{x}') \right\rangle_x, \phi(\mathbf{x}) \right\rangle_x, \quad (2.1.3)$$

where the correlation is defined as

$$\mathbf{C}(\mathbf{x}_1, \mathbf{x}_2) = \mathbb{E}[\mathbf{q}(\mathbf{x}_1)\mathbf{q}(\mathbf{x}_2)^*]. \quad (2.1.4)$$

The correlation tensor is symmetric (under interchange of its two arguments), so it has orthogonal eigenfunctions $\{\boldsymbol{\nu}_1(\mathbf{x}), \boldsymbol{\nu}_2(\mathbf{x}), \dots\}$, which can be ordered so that the associated eigenvalues $\{\lambda_1 \geq \lambda_2 \geq \dots \geq 0\}$ are non-increasing. These are eigenfunctions in the sense that they satisfy

$$\langle \boldsymbol{\nu}_k(\mathbf{x}_2), \mathbf{C}(\mathbf{x}_1, \mathbf{x}_2) \rangle_x = \lambda_k \boldsymbol{\nu}_k(\mathbf{x}_1). \quad (2.1.5)$$

Writing ϕ in the basis of these eigenfunctions provides insight into the maximization problem (2.1.1), and the energy of ϕ (2.1.3) can be written in terms of the expansion coefficients used to express it [76],

$$\lambda[\phi(\mathbf{x})] = \sum_{k=1}^{\infty} |c_k|^2 \lambda_k \quad \text{for normalized } \phi(\mathbf{x}), \quad (2.1.6)$$

where the expansion coefficient $c_k = \langle \phi(\mathbf{x}), \boldsymbol{\nu}_k(\mathbf{x}) \rangle_x$ is the projection of $\phi(\mathbf{x})$ on the k^{th} eigenfunction. The solution that maximizes (2.1.6) is $c_1 = 1, c_{\neq 1} = 0$, because $\lambda_1 > \lambda_{\neq 1}$. This tells us that the first POD mode is the eigenvector of the correlation tensor with the greatest eigenvalue. To define the latter modes, we maximize $\lambda[\phi(\mathbf{x})]$ over all $\phi(\mathbf{x})$ orthogonal to previous modes. A simple inductive argument shows that if the first k POD modes are the first k eigenfunctions of the correlation tensor, then the $k+1^{\text{st}}$ mode must be the $k+1^{\text{st}}$ eigenfunction because this orthogonality condition tells us that $c_{<k+1} = 0$ for the $k+1^{\text{st}}$ mode. The POD modes are therefore the eigenfunctions of the correlation tensor, $\phi_k = \boldsymbol{\nu}_k$, and their energies are the eigenvalues, $\lambda[\phi_k(\mathbf{x})] = \lambda_k$. In terms of the inner product, the POD modes satisfy

$$\int_{\Omega} \mathbf{C}(\mathbf{x}_1, \mathbf{x}_2) \mathbf{W}(\mathbf{x}_2) \phi_k(\mathbf{x}_2) d\mathbf{x}_2 = \lambda_k \phi_k(\mathbf{x}_1). \quad (2.1.7)$$

2.1.2 With discrete data

In practice, space-only POD modes are approximated using data from a simulation or experiment defined on a discrete set of points and sampled in time. To approximate the true continuous space-only POD modes, the data is used to approximate a correlation matrix whose eigendecomposition gives the modes. Denoting one discrete snapshot as $\mathbf{q}_j \in \mathbb{C}^{N_x}$, we need a correlation matrix of the form $\mathbf{C}^{\mathbf{x}} \in \mathbb{C}^{N_x \times N_x}$, where each element represents the

correlation between two components of the flow at different points. We label this spatial correlation matrix with an $\mathbf{x}$ superscript to distinguish it from other correlations that arise later in the paper. It can be approximated as

$$\mathbf{C}^{\mathbf{x}} = \frac{1}{m} \mathbf{Q} \mathbf{Q}^* \quad \text{with} \quad \mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_m] \in \mathbb{C}^{N_x \times m}. \quad (2.1.8)$$

The data matrix $\mathbf{Q}$ contains an ensemble of snapshots, and $\frac{1}{m} \mathbf{Q} \mathbf{Q}^*$ is an approximation of $\mathbf{C}$ because it gives each component of the correlation as a sample average over the realizations,

$$(\mathbf{Q} \mathbf{Q}^*)_{ij} = \sum_{k=1}^m (\mathbf{q}_k)_i (\mathbf{q}_k^*)_j. \quad (2.1.9)$$

Discrete space-only POD modes are given by the eigenvectors of the discrete correlation matrix multiplied by the weight,

$$\mathbf{C}^{\mathbf{x}} \mathbf{W} \Phi = \Phi \Lambda, \quad (2.1.10)$$

where $\mathbf{W}$ is the discrete weight matrix. This weight is used both as the discrete version of the continuous weight and to account for numerical quadrature of the integral in (2.1.7). The columns of the matrix $\Phi = [\phi_1, \phi_2, \dots]$ are the discrete space-only POD modes, and the diagonal matrix Λ contains the corresponding eigenvalues.

The discrete space-only POD modes are also related to the left singular vectors of the data matrix $\mathbf{Q}$ and the weight matrix, and the details of this relation will prove important to understanding the connection between the SVD of the Hankel matrix and POD. If we take the singular value decomposition to obtain

$$\frac{1}{\sqrt{m}} \mathbf{W}^{\frac{1}{2}} \mathbf{Q} = \mathbf{U} \Sigma \mathbf{V}^*, \quad (2.1.11)$$

then we can write the correlation tensor multiplied by the weight as $\mathbf{C}^{\mathbf{x}} \mathbf{W} = \mathbf{W}^{-\frac{1}{2}} \mathbf{U} \Sigma \mathbf{V}^* \mathbf{V} \Sigma \mathbf{U}^* \mathbf{W}^{\frac{1}{2}}$. Because both $\mathbf{U}$ and $\mathbf{V}$ are orthonormal, multiplying by $\mathbf{W}^{-\frac{1}{2}} \mathbf{U}$ we have

$$\mathbf{C}^{\mathbf{x}} \mathbf{W} \mathbf{W}^{-\frac{1}{2}} \mathbf{U} = \mathbf{W}^{-\frac{1}{2}} \mathbf{U} \Sigma^2. \quad (2.1.12)$$

This constitutes an eigendecomposition of $\mathbf{C}^{\mathbf{x}} \mathbf{W}$, and therefore the POD modes are related to the left singular vectors of the data matrix and the energies to the singular values,

$$\Phi = \mathbf{W}^{-\frac{1}{2}} \mathbf{U} \quad \text{and} \quad \Lambda = \Sigma^2. \quad (2.1.13)$$

It is important to remember that the correlation matrix is approximate because it is obtained from finite data. If the data are highly correlated, e.g., if the snapshots are taken

from a time series whose length is on the same order as the characteristic timescale of the flow or shorter, then the correlation tensor, and hence the POD modes, will be inaccurate. Both will increase in accuracy with the number of realizations and the independence of the realizations.

2.2 Spectral POD

Spectral POD produces an optimal frequency domain representation of statistically stationary flows. At each frequency, SPOD modes reconstruct the Fourier transform of a flow more accurately, on average, than any other reconstruction of the same order.

2.2.1 Formulation

Spectral POD can also be cast as an optimization problem, analogous to that of space-only POD, as

$$\lambda_\omega[\phi(\mathbf{x})] = \frac{1}{\|\phi(\mathbf{x})\|_x^2} \mathbb{E} \left[|\langle \hat{\mathbf{q}}_\omega(\mathbf{x}), \phi(\mathbf{x}) \rangle_x|^2 \right], \quad (2.2.1a)$$

$$\psi_{\omega,1}(\mathbf{x}) = \arg \max \lambda_\omega[\phi(\mathbf{x})]. \quad (2.2.1b)$$

Here, the expectation operator acts over the Fourier transform of segments of the flow. These segments can be subsections of a longer time series or correspond to separate realizations of the flow. Similar to the space-only case, λ_ω is a functional that returns the energy in the flow at frequency ω captured by the argument. The Fourier transformed flow is defined as

$$\hat{\mathbf{q}}_\omega(\mathbf{x}) = \int_{-\infty}^{\infty} \mathbf{q}(\mathbf{x}, t) e^{-i\omega t} dt, \quad (2.2.2)$$

and the inner product is the same as before, given in (2.1.2). Analogous to the space-only case, the optimization problem (2.2.1a) can be rewritten in terms of correlations,

$$\lambda_\omega[\phi(\mathbf{x})] = \frac{1}{\|\phi(\mathbf{x})\|_x^2} \left\langle \left\langle \phi(\mathbf{x}'), \mathbf{S}_\omega(\mathbf{x}, \mathbf{x}') \right\rangle_x, \phi(\mathbf{x}) \right\rangle_x, \quad (2.2.3)$$

where $\mathbf{S}_\omega$, called the cross-spectral density tensor, is the Fourier transform pair of the statistically stationary space-time correlation tensor,

$$\mathbf{S}_\omega(\mathbf{x}_1, \mathbf{x}_2) = \int_{-\infty}^{\infty} \mathbf{C}(\mathbf{x}_1, \mathbf{x}_2, \tau) e^{-i\omega\tau} d\tau \quad (2.2.4)$$

with

$$\mathbf{C}(\mathbf{x}_1, \mathbf{x}_2, \tau) = \mathbb{E}[\mathbf{q}(\mathbf{x}_1, t + \tau)\mathbf{q}^*(\mathbf{x}_2, t)]. \quad (2.2.5)$$

Note that in statistically stationary flow, the correlation tensor $\mathbf{C}$ only depends on one time variable, τ , which represents the difference in the two times in a more general $\mathbf{C}$ that arises in space-time POD.

The mathematical structure of (2.2.3) is the same as in space-only POD, so to maximize (2.2.3) we solve the integral eigenvalue problem

$$\int_{\Omega} \mathbf{S}_{\omega}(\mathbf{x}_1, \mathbf{x}_2) \mathbf{W}(\mathbf{x}_2) \boldsymbol{\psi}_{\omega,k}(\mathbf{x}_2) d\mathbf{x}_2 = \lambda_k \boldsymbol{\psi}_{\omega,k}(\mathbf{x}_1), \quad (2.2.6)$$

where λ_k is the k^{th} largest eigenvalue of $\mathbf{S}_{\omega}$, and $\boldsymbol{\psi}_{\omega,k}$ is the k^{th} SPOD mode at frequency ω .

2.2.2 With discrete data

As we saw in 2.1.2, data can be used to compute approximate space-only POD modes by forming a discrete spatial correlation matrix $\mathbf{C}$. Analogously, approximate spectral POD modes can be computed by using data to form a discrete cross-spectral density matrix. To do this, instead of taking the Fourier transform of the space-time correlation matrix, the Wiener-Khinchin theorem is invoked, which states that the cross-spectral density tensor is equivalent to the correlation between points in the flow in Fourier space,

$$\mathbf{S}_{\omega}(\mathbf{x}_1, \mathbf{x}_2) = \mathbb{E}[\hat{\mathbf{q}}_{\omega}(\mathbf{x}_1)\hat{\mathbf{q}}_{\omega}^*(\mathbf{x}_2)]. \quad (2.2.7)$$

To find $\mathbf{S}_{\omega}$ in this way, many realizations of the flow are needed in the frequency domain, at a particular frequency: $\hat{\mathbf{Q}}_{\omega} = [\hat{\mathbf{q}}_{\omega,1}, \hat{\mathbf{q}}_{\omega,2}, \dots, \hat{\mathbf{q}}_{\omega,m}]$, where $\hat{\mathbf{q}}_{\omega,i} \in \mathbb{C}^{N_x}$ is the spatially discretized representation of the Fourier transform of the i^{th} flow realization, at frequency ω . These frequency domain realizations come from applying the discrete Fourier transform to time series data, and to get more than one realization at each frequency from a single time series, the time series is broken up into m (possibly overlapping) blocks, and the discrete Fourier transform is taken of each block [131]. With this data matrix of Fourier realizations, the cross-spectral density matrix can be approximated as

$$\mathbf{S}_{\omega} = \frac{1}{m} \hat{\mathbf{Q}}_{\omega} \hat{\mathbf{Q}}_{\omega}^*. \quad (2.2.8)$$

The product of the data matrix with its transpose approximates the discrete cross-spectral density tensor because it averages products of different components of $\hat{\mathbf{q}}_{\omega}$ over the realiza-

tions. The SPOD modes at frequency ω are the eigenvectors of this approximated cross-spectral density tensor multiplied by the weight matrix,

$$\mathbf{S}_\omega \mathbf{W} \boldsymbol{\Psi}_\omega = \boldsymbol{\Psi}_\omega \boldsymbol{\Lambda}_\omega, \quad (2.2.9)$$

where $\boldsymbol{\Psi}_\omega = [\psi_{\omega,1}, \psi_{\omega,2} \dots]$, and $\boldsymbol{\Lambda}_\omega$ is the diagonal matrix of eigenvalues at frequency ω . As before, the modes can be obtained using the SVD of the (weighted) data matrix of realizations in Fourier space,

$$\frac{1}{\sqrt{m}} \mathbf{W}^{\frac{1}{2}} \hat{\mathbf{Q}}_\omega = \mathbf{U}_\omega \boldsymbol{\Sigma}_\omega \mathbf{V}_\omega^*. \quad (2.2.10)$$

The modes and energies are then given by

$$\boldsymbol{\Psi}_\omega = \mathbf{W}^{-\frac{1}{2}} \mathbf{U}_\omega \quad \text{and} \quad \boldsymbol{\Lambda}_\omega = \boldsymbol{\Sigma}_\omega^2. \quad (2.2.11)$$

2.3 Space-time POD

In this section, we describe a generalization of space-only POD and spectral POD called space-time POD, in which the goal is to find modes that optimally represent the flow evolution on a finite-time window $[0, T]$.

2.3.1 Formulation

The goal is formalized in the same way as the previous two cases, by maximizing an expression involving an inner product,

$$\lambda_{x,t}[\boldsymbol{\phi}(\mathbf{x}, t)] = \frac{1}{\|\boldsymbol{\phi}(\mathbf{x}, t)\|_{x,t}^2} \mathbb{E} \left[|\langle \mathbf{q}(\mathbf{x}, t), \boldsymbol{\phi}(\mathbf{x}, t) \rangle_{x,t}|^2 \right], \quad (2.3.1a)$$

$$\boldsymbol{\phi}_1(\mathbf{x}, t) = \arg \max \lambda_{x,t}[\boldsymbol{\phi}(\mathbf{x}, t)]. \quad (2.3.1b)$$

For the space-time case, however, the inner product involves integration over space and time,

$$\langle \mathbf{q}_1(\mathbf{x}, t), \mathbf{q}_2(\mathbf{x}, t) \rangle_{x,t} = \int_0^T \int_{\Omega} \mathbf{q}_2^*(\mathbf{x}, t) \mathbf{W}(\mathbf{x}, t) \mathbf{q}_1(\mathbf{x}, t) d\mathbf{x} dt, \quad (2.3.2)$$

and the space-time norm $\|\cdot\|_{x,t}$ is induced in the usual way from this space-time inner product. The expectation operator acts over finite time segments of the flow. These segments can be separate realizations of the flow or can be extracted from a single long time series if the flow is ergodic. The space-time optimization problem (2.3.1) has a different physical meaning

than the space-only optimization and the spectral optimization problems (2.1.1) and (2.2.1), respectively. The desired mode here will be a function of both space and time, and its accuracy is measured by an inner product over both space and time as well, whereas, in the previous two cases, the modes were functions of only space. Mathematically, however, the problems are quite similar; instead of the spatial domain, Ω , we now have the spatiotemporal domain $\mathcal{D} \equiv \Omega \times [0, T]$, and the modes are now a functions of both space and time, $\mathbf{z} \equiv [\mathbf{x}, t]^T$. With these definitions, the space-time optimization problem can be written

$$\lambda_{x,t}[\phi(\mathbf{z})] = \frac{1}{\|\phi(\mathbf{z})\|_{x,t}^2} \mathbb{E}[\langle \mathbf{q}(\mathbf{z}), \phi(\mathbf{z}) \rangle_{x,t}^2], \quad (2.3.3a)$$

$$\phi_1(\mathbf{z}) = \arg \max \lambda_{x,t}[\phi(\mathbf{z})], \quad (2.3.3b)$$

where the inner product is

$$\langle \mathbf{q}_1(\mathbf{z}), \mathbf{q}_2(\mathbf{z}) \rangle_{x,t} = \int_{\mathcal{D}} \mathbf{q}_2^*(\mathbf{z}) \mathbf{W}(\mathbf{z}) \mathbf{q}_1(\mathbf{z}) d\mathbf{z}. \quad (2.3.4)$$

The maximization is the same mathematically as (2.1.1), so the solution is that the optimal space-time modes are the eigenfunctions of the correlation between two points in the spatiotemporal domain,

$$\int_{\mathcal{D}} \mathbf{C}(\mathbf{z}_1, \mathbf{z}_2) \mathbf{W}(\mathbf{z}_2) \phi_k(\mathbf{z}_2) d\mathbf{z}_2 = \lambda_k \phi_k(\mathbf{z}_1). \quad (2.3.5)$$

Rewriting this again in terms of space and time, the space-time POD modes satisfy

$$\int_0^T \int_{\Omega} \mathbf{C}(\mathbf{x}_1, t_1, \mathbf{x}_2, t_2) \mathbf{W}(\mathbf{x}_2, t_2) \phi_k(\mathbf{x}_2, t_2) d\mathbf{x}_2 dt_2 = \lambda_k \phi_k(\mathbf{x}_1, t_1), \quad (2.3.6)$$

where the space-time correlation tensor is defined as

$$\mathbf{C}(\mathbf{x}_1, t_1, \mathbf{x}_2, t_2) = \mathbb{E}[\mathbf{q}(\mathbf{x}_1, t_1) \mathbf{q}(\mathbf{x}_2, t_2)^*]. \quad (2.3.7)$$

In section 2.5, we will show that the Hankel SVD modes constitute a discrete approximation of space-time POD modes. Before doing so, we first show that spectral and space-only POD can be recovered from space-time POD in the limits of long and short time windows, respectively. The latter result is novel to the best of our knowledge.

2.3.2 Space-time POD on long times becomes spectral POD

Here, we show that in the limit that $T \rightarrow \infty$, the space-time modes become spectral POD modes. This limit has been shown in the literature, e.g., in [77, 131], but here we give a different proof. We restrict our attention to ergodic flows, though all that is needed for the results to be valid is that the flow is wide-sense statistically stationary, i.e., the first and second moments of the flow don't change with time. The integral eigenvalue problem that defines the modes when $T \rightarrow \infty$ is

$$\int_{-\infty}^{\infty} \int_{\Omega} \mathbf{C}(\mathbf{x}_1, t_1, \mathbf{x}_2, t_2) \mathbf{W}(\mathbf{x}_2) \phi(\mathbf{x}_2, t_2) d\mathbf{x}_2 dt_2 = \lambda \phi(\mathbf{x}_1, t_1). \quad (2.3.8)$$

For convenience, the bounds on time have been shifted to cover the entire real line. Because the flow is stationary, the correlation must only depend on the difference of the two times, not on the times themselves, so

$$\mathbf{C}(\mathbf{x}_1, t_1, \mathbf{x}_2, t_2) \rightarrow \mathbf{C}(\mathbf{x}_1, \mathbf{x}_2, t_1 - t_2). \quad (2.3.9)$$

Switching the order of integration, we have

$$\int_{\Omega} \int_{-\infty}^{\infty} \mathbf{C}(\mathbf{x}_1, \mathbf{x}_2, t_1 - t_2) \mathbf{W}(\mathbf{x}_2) \phi(\mathbf{x}_2, t_2) dt_2 d\mathbf{x}_2 = \lambda \phi(\mathbf{x}_1, t_1), \quad (2.3.10)$$

which is a convolution of $\mathbf{C}$ with ϕ_k . Taking the Fourier transform of both sides and using the convolution theorem, we have

$$\int_{\Omega} \mathbf{S}_{\omega}(\mathbf{x}_1, \mathbf{x}_2) \mathbf{W}(\mathbf{x}_2) \hat{\phi}_{\omega}(\mathbf{x}_2) d\mathbf{x}_2 = \lambda \hat{\phi}_{\omega}(\mathbf{x}_1). \quad (2.3.11)$$

The modes must satisfy (2.3.11) for every ω . SPOD modes, $\psi_{\omega'}$ at some ω' , are solutions as they are defined by this equation for $\omega = \omega'$, and are zero for $\omega \neq \omega'$. The SPOD mode with the k^{th} greatest energy λ (over the modes at all frequencies) is the k^{th} space-time POD mode.

2.3.3 Space-time POD on short times becomes space-only POD

Here, we show an analogous result for space-only POD. Specifically, we investigate what happens to the space-time POD modes in the limit that the time T on which they evolve is short compared to the time scales of the flow. In this limit, the space-time correlation

between two points $(\mathbf{x}_1, t_1)$ and $(\mathbf{x}_2, t_2)$ is only a function of their locations in space because t_1 and t_2 are negligibly different. Intuitively, we can now think of this integral over space and time as an integral over space multiplied by time because there is no time dependence in the integrand, so long as the weight doesn't depend on time (which would be an unusual choice). Rewriting condition (2.3.6) with a constant weight and correlation, we have

$$\int_0^T \int_{\Omega} \mathbf{C}(\mathbf{x}_1, \mathbf{x}_2) \mathbf{W}(\mathbf{x}_2) \phi_k(\mathbf{x}_2, t_2) d\mathbf{x}_2 dt_2 = \lambda_k \phi_k(\mathbf{x}_1, t_1). \quad (2.3.12)$$

Because the left-hand side is not a function of t_1 , the modes must indeed be constant in time, and the condition on the modes becomes an integral over space multiplied by time,

$$T \int_{\Omega} \mathbf{C}(\mathbf{x}_1, \mathbf{x}_2) \mathbf{W}(\mathbf{x}_2) \phi_k(\mathbf{x}_2) d\mathbf{x}_2 = \lambda_k \phi_k(\mathbf{x}_1). \quad (2.3.13)$$

This condition is exactly that of space-only POD, as shown in (2.1.7). As a result, the space-time POD modes converge to the space-only POD modes and their energies are proportional to the space-only POD energies (with proportionality constant T).

2.3.4 Space-time POD with discrete data

Given time series data on a spatial grid from an experiment or simulation, how do we compute the space-time POD modes? Above, we have seen that the solution to the space-time POD problem is given by eigendecomposition of the space-time correlation, which becomes a matrix when space and time are discrete. Building this matrix requires any ensemble of finite-time realizations of the flow. If the time T that the space-time POD modes are to evolve on is represented by d time steps with our temporal discretization, then one realization of the flow can be written as a vector with d snapshots stacked on top of one another,

$$\mathbf{y} = \begin{bmatrix} \mathbf{q}_1 \\ \mathbf{q}_2 \\ \vdots \\ \mathbf{q}_d \end{bmatrix} \in \mathbb{C}^{N_x d}. \quad (2.3.14)$$

With many of these realizations, we can approximate the correlation matrix $\mathbf{C} \in \mathbb{C}^{N_x d \times N_x d}$

in the same manner as before,

$$\mathbf{C} = \frac{1}{m} \mathbf{Y} \mathbf{Y}^* \quad \text{with} \quad \mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_m] \in \mathbb{C}^{(N_x d) \times m}. \quad (2.3.15)$$

Analogous to the space-only POD and SPOD cases, this approximation works because the correlation between two points in the spatio-temporal domain is approximated as the product of the flow at those points averaged over the realizations. We will see later in section 2.6, however, that if the flow is assumed to be ergodic and the realizations are all to be formed from one long time series, this construction of the correlation matrix does not fully exploit the ergodicity of the system, and it is possible to create a more accurate approximation of the correlation. The discretized POD modes are obtained as

$$\mathbf{C} \mathbf{W} \Phi = \Phi \Lambda, \quad (2.3.16)$$

where $\mathbf{W} \in \mathbb{C}^{N_x d \times N_x d}$ is the discretized weight matrix.

Similar to the space-only POD and SPOD cases, the modes on this domain can also be obtained from the SVD of the weighted data matrix,

$$\frac{1}{\sqrt{m}} \mathbf{W}^{\frac{1}{2}} \mathbf{Y} = \mathbf{U} \Sigma \mathbf{V}^*. \quad (2.3.17)$$

The space-time POD modes and energies are then

$$\Phi = \mathbf{W}^{-\frac{1}{2}} \mathbf{U} \quad \text{and} \quad \Lambda = \Sigma^2. \quad (2.3.18)$$

2.4 The Hankel matrix

A Hankel matrix is a matrix whose skew-diagonals are constant, i.e., the (i, j) entry only depends on $i + j$,

$$H = \begin{bmatrix} g_1 & g_2 & \cdots & g_m \\ g_2 & g_3 & \cdots & g_{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ g_d & g_{d+1} & \cdots & g_{m+d-1} \end{bmatrix}. \quad (2.4.1)$$

In dynamical systems theory, these matrices are formed from a time series of observables $g_1, g_2, g_3, \dots, g_{m+d-1}$. This means that the columns of H are time-delay embeddings of the dynamical system. In this paper, we focus our attention on the more general block Hankel

matrix

$$\mathbf{H} = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_m \\ \mathbf{q}_2 & \mathbf{q}_3 & \cdots & \mathbf{q}_{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{q}_d & \mathbf{q}_{d+1} & \cdots & \mathbf{q}_{m+d-1} \end{bmatrix}, \quad (2.4.2)$$

formed from a time series of vector-valued observables $\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_d, \dots, \mathbf{q}_{m+d-1}$.

Applications of the Hankel matrix include the singular spectrum analysis (SSA) [134], the eigensystem realization algorithm (ERA) [61] and the Hankel alternative view of Koopman (HAVOK) [13]. All of these methods depend on the SVD of the Hankel matrix,

$$\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*, \quad (2.4.3)$$

and in particular use the left singular vectors of the Hankel matrix as a basis, which have been called principal component trajectories [27]. The connections to time-delay embedding, Koopman theory, and dynamic mode decomposition have garnered increasing interest in analysis of the Hankel matrix.

2.5 Hankel singular vectors are space-time POD modes

Here, we demonstrate the connection between space-time POD modes and Hankel modes by showing that one way of computing the unweighted space-time POD modes is to take the SVD of a Hankel matrix. As we will show, this connection provides insight into the sense in which the Hankel SVD modes are optimal and the impacts of the time step, rows, and columns on the properties of the modes.

For ergodic systems, the flow realizations needed to construct the approximate correlation matrix can be extracted from a single long time series. Given a time series of snapshots, $\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_d, \dots, \mathbf{q}_{m+d-1}$, we may extract m realizations of length d by creating the first realization $\mathbf{y}_1 = [\mathbf{q}_1^T \mathbf{q}_2^T, \dots, \mathbf{q}_d^T]^T$, then advancing one time step over to create the second realization, $\mathbf{y}_2 = [\mathbf{q}_2^T \mathbf{q}_3^T, \dots, \mathbf{q}_{d+1}^T]^T$, and so on. Stacking these as columns in the data matrix, the result is a block Hankel matrix,

$$\mathbf{H} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_m] = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_m \\ \mathbf{q}_2 & \mathbf{q}_3 & \cdots & \mathbf{q}_{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{q}_d & \mathbf{q}_{d+1} & \cdots & \mathbf{q}_{m+d-1} \end{bmatrix}. \quad (2.5.1)$$

Comparing (2.5.1) and (2.3.15) we see that the Hankel matrix $\mathbf{H}$ provides one way of gener-

ating the data matrix $\mathbf{Y}$. Therefore, using (2.3.17) and (2.3.18) and taking the SVD of the (weighted) Hankel matrix,

$$\frac{1}{\sqrt{m}} \mathbf{W}^{\frac{1}{2}} \mathbf{H} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^*, \quad (2.5.2)$$

we obtain the space-time POD modes as

$$\mathbf{\Phi} = \mathbf{W}^{-\frac{1}{2}} \mathbf{U} \quad \text{and} \quad \mathbf{\Lambda} = \mathbf{\Sigma}^2. \quad (2.5.3)$$

If the weight $\mathbf{W}$ is uniform, the modes come directly from the Hankel SVD,

$$\mathbf{\Phi} = \mathbf{U} = \text{lsv}(\mathbf{H}). \quad (2.5.4)$$

That is, the left singular vectors of the Hankel matrix give a discrete approximation of the space-time POD modes *in the case of a uniform weight*.

This connection to space-time POD implies the sense in which the Hankel modes are optimal: they minimize the expected value of the square norm of the projection error, where the norm is the ℓ_2 norm, i.e., the weight is uniform. Uniformity is a reasonable choice for the weight matrix if the spatial grid is uniform. However, if the grid is non-uniform, a uniform weight matrix corresponds to a non-uniform continuous weight. For example, if the flow data is defined on a cylindrical grid, a uniform weight matrix will bias accuracy toward the center of the domain over the outside. If the desired weight is non-uniform, as, e.g., is likely to be the case if the grid is non-uniform in space, then the discrete space-time POD modes can be obtained using equations (2.5.3) and (2.5.2).

2.5.1 Height of the Hankel matrix corresponds to T

The connection between space-time POD and singular modes of the Hankel matrix reveals two interpretations of the number of rows d in the Hankel matrix. First, it determines, for a fixed time step Δt , the time window T on which the space-time POD modes optimally represent the flow. The window also depends on the time step, specifically, $T = (d - 1)\Delta t$. Second, the significance of the choice of d can also be understood in terms of the space-time correlation matrix implied by the Hankel matrix — the Hankel singular modes will account for a section of the space-time correlation of the system of width T , as shown in Figure 2.1. The properties of the correlation outside this window are discarded; they play no role in the definition of the modes and cannot be represented by the modes.

Two interesting limits of this truncation occur when $(d - 1) * \Delta t \ll \tau_C$ and when $(d - 1)\Delta t \gg \tau_C$, where τ_C is the maximum correlation time of the flow between any two

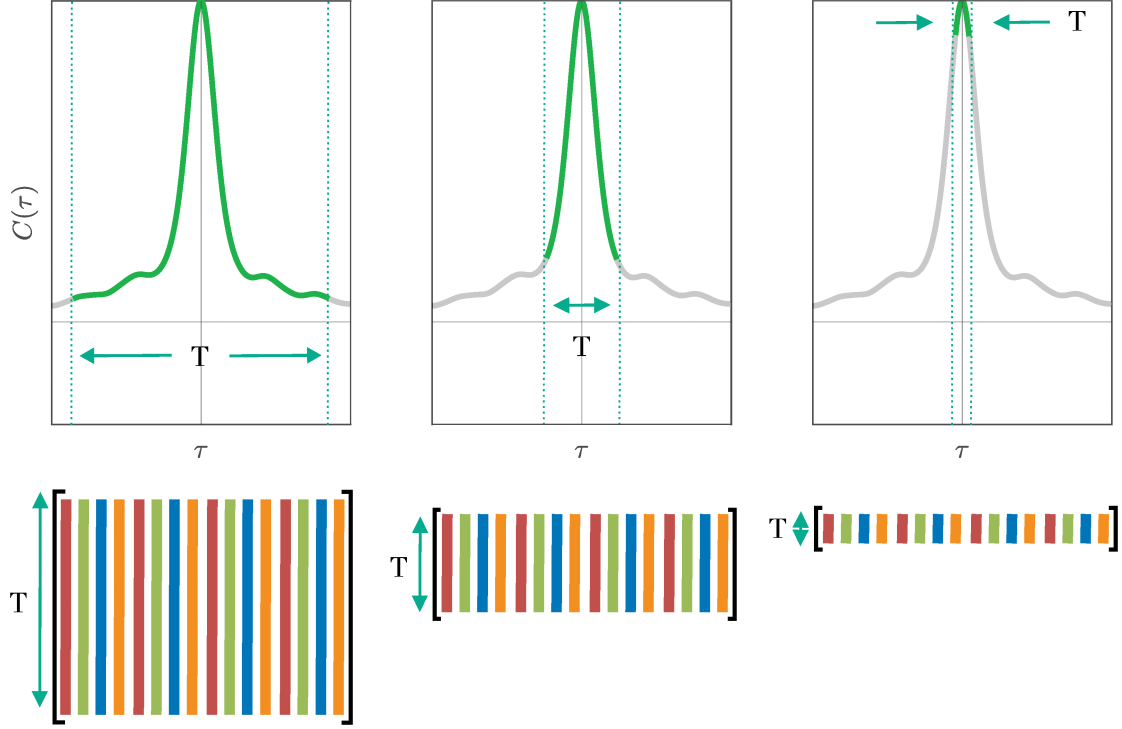


Figure 2.1: The height of the Hankel matrix d , along with the time step Δt between snapshots, determines the time window T over which the space-time correlation C of the system is accounted for in the Hankel singular modes. Shorter Hankel matrices, therefore, severely truncate the correlations, whereas taller Hankel matrices retain more of these correlations.

locations in space. When the height of the Hankel matrix corresponds to a time much smaller than the correlation time, we should expect to get spatial modes identical to those from discrete space-only POD for the same reason discussed in Section 2.3.3: the discrete space-time POD modes are eigenvectors of the correlation matrix, and if $(d - 1) * \Delta t = T \ll \tau_C$, then this correlation matrix is constant in time. In this limit, the correlation between two points in space is independent of the time difference, so

$$\mathbf{C}_{ij} = \mathbf{C}_{i+aN_x, j+bN_x}, \quad (2.5.5)$$

where N_x is the number of spatial gridpoints and a and b are any integers, so long as the indices are valid. Incrementing one of the indices by N_x corresponds to moving one time step forward but not shifting in space, but we have assumed that the correlation matrix is

constant in time. It can therefore be written in block form as

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}^{\mathbf{x}} & \mathbf{C}^{\mathbf{x}} & \dots & \mathbf{C}^{\mathbf{x}} \\ \mathbf{C}^{\mathbf{x}} & \ddots & & \vdots \\ \vdots & & & \\ \mathbf{C}^{\mathbf{x}} & \dots & & \mathbf{C}^{\mathbf{x}} \end{bmatrix}, \quad (2.5.6)$$

where $\mathbf{C}^{\mathbf{x}} \in \mathbb{C}^{N_x \times N_x}$ is the discrete correlation matrix at no time lag, as defined in 2.1.8. $\mathbf{C}$ has only N_x non-zero eigenvalues, and these eigenvalues are $d\Lambda^{\mathbf{x}}$, the space-only eigenvalues scaled by the number of time steps. The associated eigenvectors of $\mathbf{C}$ are

$$\phi_k = \begin{bmatrix} \phi_k^{\mathbf{x}} \\ \phi_k^{\mathbf{x}} \\ \vdots \\ \phi_k^{\mathbf{x}} \end{bmatrix}, \quad (2.5.7)$$

where $\phi_k^{\mathbf{x}}$ is the k^{th} eigenvector of the space-only correlation $\mathbf{C}^{\mathbf{x}}$. Since each $N_x \times 1$ block of ϕ_k is the same, the space-time POD modes are constant in time and have the spatial form of space-only POD. Thus, we've recovered space-only POD in the short-time limit, as we might expect from the discussion of the continuous case in section 2.3.3. The SVD of a very short Hankel matrix will therefore give modes with little time dependence, which could have been obtained with less computational effort using space-only POD instead.

Above, as well as in section 2.3.3, we assumed no temporal variation of the correlation function. Gibson et al. [39] showed that for the special case of a scalar Hankel matrix, if small variations in time to the correlation matrix are allowed, then the resulting POD basis is the Legendre polynomials. In intuitive terms, the Legendre polynomials appear because, over short times, the flow will be well approximated by its Taylor series truncated at a few terms, and each term is much more important than the next. Therefore, with k modes, the basis should span the set of degree $k - 1$ polynomials and also be orthogonal with respect to the inner product. If the inner product is uniform, then the basis satisfying these conditions is the Legendre polynomials. On a vector time series, we observe a full set of modes with no time dependence, consistent with the order zero Legendre polynomial, followed by full sets of modes with time dependence corresponding to each subsequent Legendre polynomial and negligible energy. The constant modes are the space-only POD modes, and their energies are the space-only POD energies multiplied by d . This confirms that our analysis assuming no temporal variation, and the resulting conclusions that the space-time modes converge to space-only modes, hold.

The other limit, $(d - 1)\Delta t = T \gg \tau_C$, occurs when the height of the Hankel matrix corresponds to a time much longer than any correlation time in the flow. From section 2.3.2, we would expect that the modes oscillate with a pure frequency as SPOD modes do. Indeed, taking the SVD of a Hankel matrix becomes a discrete Fourier transform as the height of the Hankel matrix grows to infinity [12, 11]. This limit has been used in the context of time-delay embedding [63].

This connection to SPOD means that Hankel SVD is also related to two other popular methods, resolvent analysis and dynamic mode decomposition (DMD). Resolvent analysis is an input-output formulation of the Navier-Stokes equations in the frequency domain [82, 133] wherein the non-linear terms (about some base/mean flow) are interpreted as forcing inputs to a linearized system. Towne et al. [131] showed that if these forcings do not preference any spatial structure, i.e., they are white in space, then resolvent modes are equivalent to SPOD modes. Hence, due to the connection we have established, resolvent modes are SVD modes of an asymptotically tall Hankel matrix if the non-linear terms are white. DMD is a data-driven method that seeks the linear operator which most closely maps snapshots at one time to their value one time-step later, in the least-squares sense. SPOD modes are optimally averaged DMD modes [131], so too then are Hankel SVD modes in the case that the Hankel matrix is asymptotically tall.

Finally, we note that the time step between rows (along columns) signifies the temporal discretization of the continuous eigenvalue problem (2.3.6) which defines the modes. Next, we leverage the distinct meaning of the time step between rows and columns to approximate the modes at reduced cost.

2.5.2 Width of the Hankel matrix informs convergence

The connection with space-time POD also reveals the impact of the width of the Hankel matrix on its singular modes. The convergence of the discrete space-time POD modes is determined by the accuracy of the approximation of the correlation matrix formed from the data. Each element of the correlation is proportional to the sample average of products of one entry in the column with another, averaged over the columns of the Hankel matrix,

$$\mathbf{C}_{ij} = (\mathbf{H}\mathbf{H}^*)_{ij} = \sum_{k=1}^m \mathbf{H}_{ik} \mathbf{H}_{jk}. \quad (2.5.8)$$

Thus, the width of the Hankel matrix m determines the number of realizations that contribute to approximating the correlation.

However, because two adjacent columns in the Hankel matrix are only one time step apart,

their contributions to each element of the correlation matrix are far from independent, so the accuracy of the correlation is not just a function of the number of columns. Indeed, one could imagine a Hankel matrix that has many columns but only represents data over a short time during which the underlying system does not fully explore its phase space. There are two criteria necessary for the modes to be accurate: there must be enough columns in the Hankel matrix so that their sample average converges, and the data must be representative of the underlying attractor. The latter may require the data to be taken over a long time, leading to a very wide Hankel matrix, thus a costly SVD. We point out here that the time step between columns signifies the time between two successive realizations, in contrast to the time step between rows.

Alternatively, we may throw out many of the columns of the Hankel matrix, so that the columns are less correlated, then take the SVD of this matrix,

$$\mathbf{Q}_{UC} = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_{s+1} & \mathbf{q}_{2s+1} & \cdots & \mathbf{q}_m \\ \mathbf{q}_2 & \mathbf{q}_{s+2} & \mathbf{q}_{2s+2} & \cdots & \mathbf{q}_{m+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{q}_d & \mathbf{q}_{s+d} & \mathbf{q}_{2s+d} & \cdots & \mathbf{q}_{m+d-1} \end{bmatrix}. \quad (2.5.9)$$

Here, s represents the spacing in time between columns, and the Hankel matrix is recovered if $s = 1$. $\mathbf{Q}_{UC}\mathbf{Q}_{UC}^*$ still forms an approximation of the correlation matrix, so its singular vectors will approximate the space-time POD modes. If $\mathbf{H}$ is formed from data representative of the attractor, then $\mathbf{Q}_{UC}$ will be as well, and if we retain enough columns, the correlation matrix will be accurate. An appropriate choice of s , one which negligibly impacts the accuracy of the modes, depends on the amount of data available, the system dimension, the embedding dimension, and the time step. The Hankel matrix is the data matrix with the most columns possible from the data, and in practice it is likely overkill, and overly costly, if the time series is long. If the data is lacking, however, the Hankel matrix will produce a more accurate correlation and hence more accurate modes than any other spacing.

This strategy of removing most of the columns provides a substantial cost reduction. The time complexity of the SVD scales quadratically with the smaller dimension of the matrix and linearly with the larger one. Therefore, this strategy reduces the computation time by a factor of s^2 if the Hankel matrix is taller than it is wide, which is usually the case for fluids and other high-dimensional problems, and by a factor of s otherwise. Critically, this reduction is achieved without changing the window T or the time step Δt along the columns, such that the meaning of the modes discussed in section 2.5.1 and the temporal discretization of the integral in (2.3.6) remain unchanged.

Obtaining the space-time POD modes from a long time series of data is, in some ways,

analogous to finding the power spectrum of a signal from a single long time series. It is well known [136] in the latter case that one should break the long time series into many shorter, possibly overlapping blocks, and then average the spectra obtained from each block. The ideal overlap is determined by the analyst’s trade-off preferences between accuracy and cost and the particularities of the problem, but little is gained by going beyond, say, 90% overlap. Certainly, it would be ill-advised to use the maximum overlap, i.e., shifting each block by only one time step. But, in the case of obtaining space-time POD modes, using the full Hankel matrix is imposing such an excessive overlap. This maximum overlap should only be used if data is scarce and forming the Hankel matrix and taking its SVD do not pose time or memory problems.

A cartoon of the correlations produced from uncorrelated columns vs. Hankel columns with different amounts of data is shown in Figure 2.2. The top two correlation graphics show how the uncorrelated columns (left) and Hankel (right) approaches might approximate the correlation with limited data. In this case, the Hankel matrix is likely to produce more accurate correlations than the matrix with uncorrelated columns because the latter may have too few to converge averages. However, the limited data is not representative of the underlying attractor, so although sampling this data more and more (Hankel approach) and computing the correlations may converge, it will not converge to the true correlations. When the time series is long enough to be representative of the attractor (bottom), both approaches will produce accurate correlations so long as enough columns are included in the uncorrelated matrix to converge averages. We also note that in the case of a short time series, we derive a method in section 2.6 which produces a more accurate approximation of the correlations than the Hankel approach with the same data.

2.5.3 Summary of the connection

To summarize, we have shown that Hankel singular modes constitute an approximation of space-time POD modes. The block Hankel matrix is formed as a data matrix whose columns are discretized temporal flow realizations, and when multiplied by its conjugate transpose, gives the correlation matrix from space-time POD. The eigenvectors of this matrix, which are the space-time POD modes, are the left singular vectors of the Hankel matrix due to the well-known equivalence of the SVD and eigendecomposition. This connection sheds light on the sense in which the Hankel modes are optimal: they are optimal in the case of a uniform weight, which is often an undesirable norm. Understanding Hankel singular modes as approximate space-time POD modes also reveals that the time step between rows and columns of the Hankel matrix need not be the same. Indeed, in many practical cases,

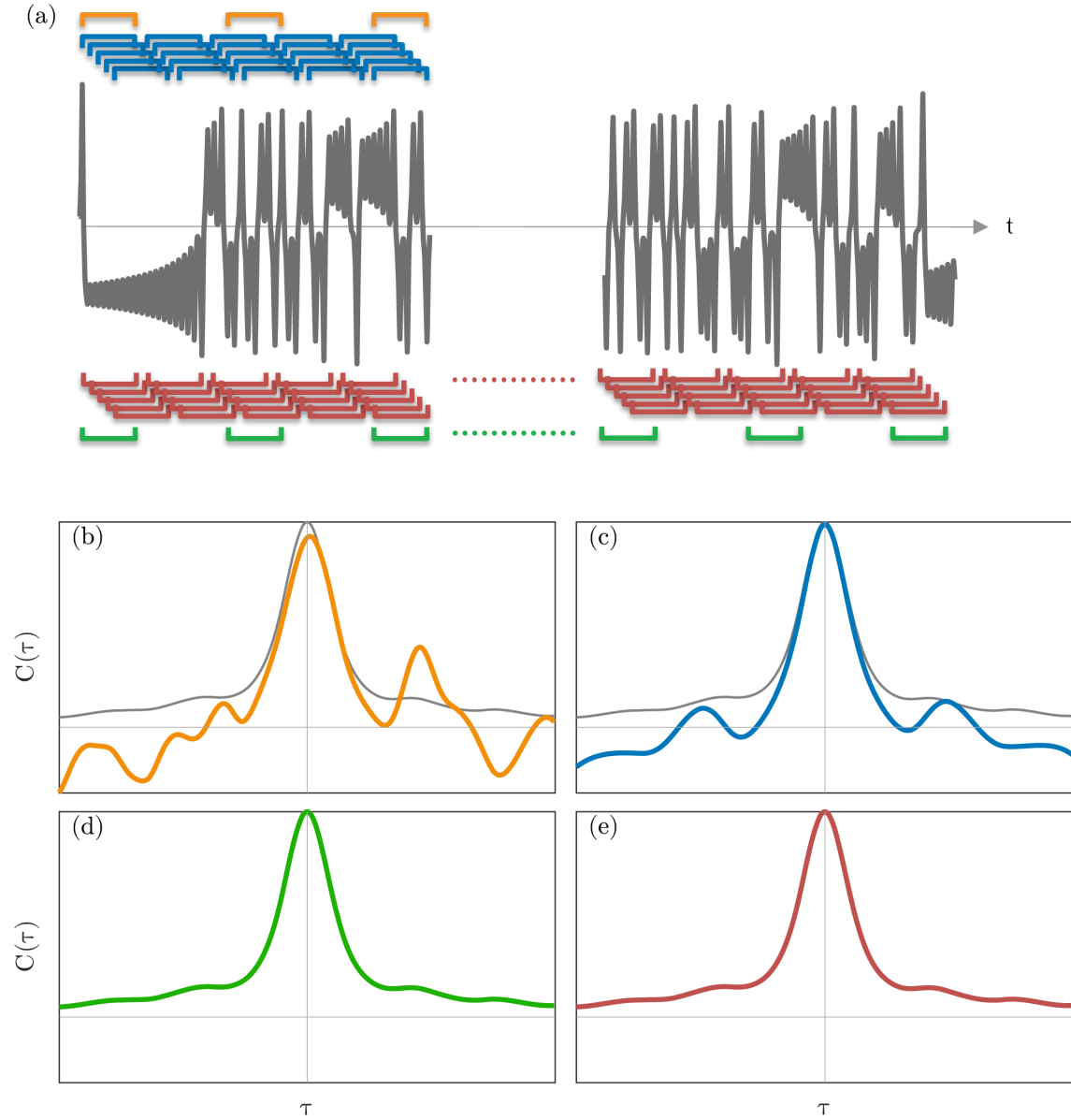


Figure 2.2: Uncorrelated columns (left) and Hankel (right) approaches to approximating the correlation matrix from short (b,c) and long (d,e) time series. (a) A time series used to sample temporal realizations with different samplings. (b) Correlations from uncorrelated columns using a short time series. (c) Correlations from the Hankel matrix using the short times series. (d) Correlations from uncorrelated columns using a long time series. (e) Correlations from a Hankel matrix using the long time series. Though both (d) and (e) are accurate, the correlations in (d) come at significantly lower computational cost.

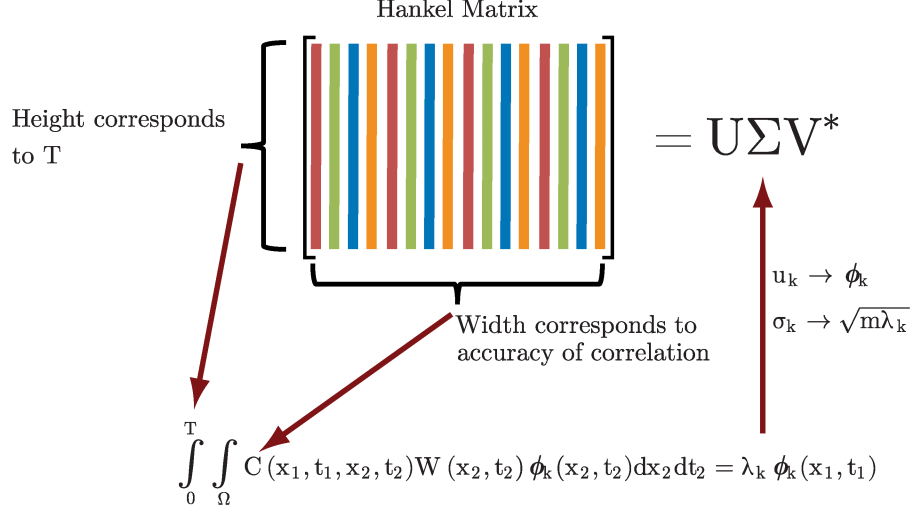


Figure 2.3: The dimensions of the Hankel matrix can be interpreted in terms of the integral space-time eigenvalue problem (2.3.6). The height of the Hankel matrix corresponds to T , the length on which the space-time POD modes optimally represent the flow. The width, though there is subtlety about independence, informs how accurately the correlation matrix is approximated, and hence the accuracy of the modes.

discarding many of the columns of the Hankel matrix negligibly impacts the accuracy of the modes but significantly reduces the time for the SVD. The time window corresponding to the height of the Hankel matrix $T = (d - 1)\Delta t$ is the time over which the Hankel modes optimally represent the flow as well as the window of space-time correlations accounted for in calculating the modes. Finally, in the limits of short and tall Hankel matrices, corresponding to evolution on short and long time windows, the Hankel modes become discrete space-only POD modes and discrete SPOD modes, respectively. Much of the discussion in this section is summarized in Figure 2.3.

2.6 Fully exploiting ergodicity: more accurate modes

As discussed in section 2.5, the Hankel matrix implies a particular approximation of the space-time correlation, which in turn determines the approximation of space-time POD modes provided by singular modes of the Hankel matrix. In this section, we show how a more accurate approximation of the correlation matrix can be constructed by fully exploiting the ergodicity of the underlying system, which produces more accurate modes if there is a shortage of data, as we will demonstrate later in section 2.7.6. This approach has been used in the context of SSA [134], but to our knowledge, has never been employed for vector-valued data.

Ergodicity is assumed in constructing the Hankel matrix: instead of taking columns from different sample paths of the flow, columns are formed from different sections in one long sample path, so there is an assumption that these two are equivalent. However, in constructing the Hankel matrix (or a down-sampling thereof) and then taking its SVD, the assumed ergodicity is not fully exploited. To see this, look at the correlation $\mathbf{C} = \mathbf{H}\mathbf{H}^* \in \mathbb{C}^{N_x d \times N_x d}$. Broken up into its spatial block structure, this correlation is written

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} & \mathbf{C}_{13} & \dots & \mathbf{C}_{1d} \\ \mathbf{C}_{12}^T & \mathbf{C}_{22} & \mathbf{C}_{23} & \dots & \mathbf{C}_{2d} \\ \mathbf{C}_{13}^T & \mathbf{C}_{23}^T & \mathbf{C}_{33} & \dots & \mathbf{C}_{3d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_{1d}^T & \mathbf{C}_{2d}^T & \mathbf{C}_{3d}^T & \dots & \mathbf{C}_{dd} \end{bmatrix}, \quad (2.6.1)$$

where the matrix

$$\mathbf{C}_{ij} \in \mathbb{C}^{N_x \times N_x} \quad (2.6.2)$$

represents the correlations between all points in the spatial domain between times i and j . Because the flow is ergodic, these correlations should only depend on the difference in times, e.g., $\mathbf{C}_{12}$ should be equal to $\mathbf{C}_{56}$, but by calculating this correlation with the Hankel matrix, this will not normally be the case. Indeed, writing each of these correlation blocks in terms of elements of the time series,

$$\mathbf{C}_{ij} = \sum_{k=1}^m \mathbf{q}_{i+k-1} \mathbf{q}_{j+k-1}^*, \quad (2.6.3)$$

we see, for example, that the $(1,1)$ block of $\mathbf{C}$ does not depend on the last element of the time series while the (d,d) block does, so they will not be equivalent as we know they should be from ergodicity. Instead, the fully converged correlation should have the symmetric block Toeplitz structure,

$$\tilde{\mathbf{C}} = \begin{bmatrix} \mathbf{C}_0 & \mathbf{C}_1 & \mathbf{C}_2 & \dots & \mathbf{C}_{d-1} \\ \mathbf{C}_1^T & \mathbf{C}_0 & \mathbf{C}_1 & \dots & \mathbf{C}_{d-2} \\ \mathbf{C}_2^T & \mathbf{C}_1^T & \mathbf{C}_0 & \dots & \mathbf{C}_{d-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_{d-1}^T & \mathbf{C}_{d-2}^T & \mathbf{C}_{d-3}^T & \dots & \mathbf{C}_0 \end{bmatrix}. \quad (2.6.4)$$

That the (block) diagonals of $\mathbf{C} = \mathbf{H}\mathbf{H}^*$ are not constant indicates that ergodicity is not fully exploited in constructing the correlation.

To calculate each correlation $\mathbf{C}_i$ from a time series, $\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_{m+d-1}$, we simply take the sample average of the product of all points in the time series i time steps apart,

$$\mathbf{C}_i = \frac{1}{m+d-i-1} \sum_{k=1}^{m+d-i-1} \mathbf{q}_k \mathbf{q}_{k+i}^*. \quad (2.6.5)$$

After building $\tilde{\mathbf{C}}$ as in (2.6.4), the modes are obtained as $\tilde{\mathbf{C}}\mathbf{W}\Phi = \Phi\Lambda$. The difference in accuracy of the modes can be substantial when data is in short supply and when m/d is not large, as we demonstrate in section 2.7.6.

The entries of the Toeplitz matrix (2.6.5) are formed using more terms than are used in forming the Hankel-based correlations (2.6.3). The ratio depends on the entry calculated but is proportional to $\frac{m}{m+d}$; therefore as m/d increases, the advantage in the Toeplitz approach diminishes.

The relative cost of computing modes using the Hankel and Toeplitz approaches depends on the relative size of $N_x d$ and m . The time scaling for the Hankel approach comes from the SVD in both cases, which is $\mathcal{O}(N_x^2 d^2 m)$ for $N_x d < m$ and $\mathcal{O}(N_x d m^2)$ otherwise. The time scaling for the Toeplitz approach comes from passing through the time series to calculate all blocks of the correlation and from taking the eigendecomposition of the correlation. The former scales as $\mathcal{O}((m+d)dN_x^2)$ and the latter as $\mathcal{O}(N_x^3 d^3)$. The ratio of the scalings of the Toeplitz and Hankel algorithms is thus

$$\begin{cases} \max(\frac{1}{d}, \frac{N_x d}{m}) & \text{if } N_x d < m, \\ \left(\frac{N_x d}{m}\right)^2 & \text{if } N_x d > m. \end{cases} \quad (2.6.6)$$

In fluids applications, $N_x d$ is usually much greater than m , so this approach will scale much worse than the Hankel approach. However, in many other dynamics applications, including many classic applications of the Hankel matrix, N_x is small, often 1, and d substantially smaller than m [13, 27], so this approach both leads to more accurate modes and better scaling. In short, the Toeplitz method scales more favorably than building the Hankel matrix and taking the SVD when $N_x d < m$, but when $N_x d > m$ the algorithm is slower.

2.7 Numerical experiments: Lid-driven cavity flow

In this section, we demonstrate our theoretical results with data from a 2D lid-driven cavity flow at Reynolds number $Re = 22,000$ [19]. First, we compare the convergence of modes with a true Hankel matrix against a data matrix whose columns are uncorrelated, showing

that with the same number of columns (flow realizations) the uncorrelated matrix generates better modes. We then show a related, but more practical, example. Given an m_H -column Hankel matrix, one can form an uncorrelated data matrix by sampling m of its columns. We show that for m significantly smaller than m_H the modes are nearly identical, which can be used to reduce computational cost. Next, we demonstrate that space-time POD converges to space-only POD on very short time intervals, and that the modes become Fourier in time on very long time intervals, indicating convergence to spectral POD. Finally, we show that with limited data, exploiting the ergodicity of the system produces more accurate modes than the Hankel approach.

2.7.1 Simulation description

We generate data for square lid-driven cavity flow at $Re = \frac{Uh}{\nu} = 22,000$, where U is the speed of the lid, h is the height of the square cavity, and ν is the viscosity of the fluid. The incompressible Navier-Stokes equations are solved using a Crank-Nicolson method for the viscous term and an Adams-Bashforth method for the nonlinear term [83]. The domain is discretized with $N_x = N_y = 120$ grid-points. Data is generated by starting the simulation with zero velocity everywhere except the top, running until the initial transients have vanished and the statistics become stationary, and then collecting the time series data. The time is nondimensionalized so that in one unit of time the lid slides the cavity width, and the simulation time step is 5×10^{-4} . We sample this data with $\Delta t = 0.1$. Figure 2.4 shows a snapshot of the flow to illustrate the setup and spatial scale of variation of the flow.

2.7.2 Spaced columns yield better modes

In section 2.5.2, we argued that an m -column data matrix with uncorrelated columns will produce a more accurate correlation matrix, and hence more accurate modes, than an m -column Hankel matrix, when measured against converged modes. Here, this is demonstrated with the lid-driven cavity flow: we form two data matrices — one Hankel, and the other uncorrelated, with the structure in (2.5.9). For the sake of memory, which scales like $N_x m d$, we choose a downsampled version of u , the x component of velocity, as our observable of the state $\mathbf{q}$, with 24 points in each direction. We choose $T = 2$, so that the modes evolve on the convective time scale of the flow, and $d = \frac{T}{\Delta t} + 1 = 21$. Each column of the matrices is an element of $\mathbb{C}^{N_x d}$, where $N_x d = 24 \times 24 \times 21 = 12,096$. The space-time POD modes are obtained as the left singular vectors of the two matrices, and their convergence is evaluated by taking the square of their inner product with a fully converged mode, which is obtained from a data matrix with 40,000 columns, all far apart in time. The variation in different

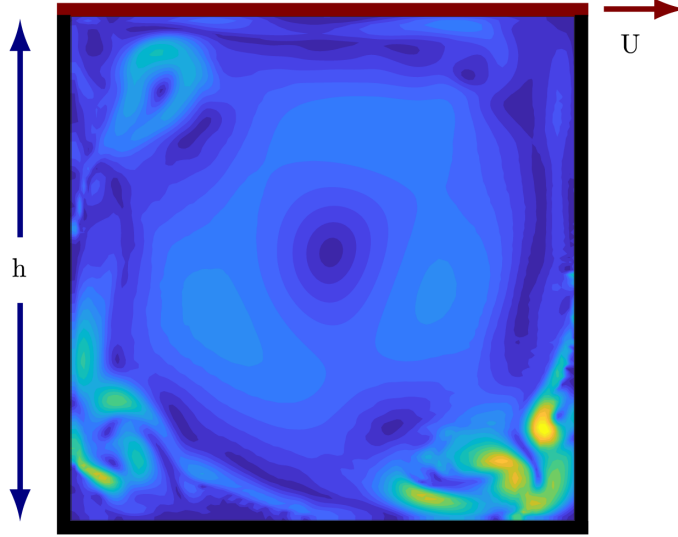


Figure 2.4: Schematic of the lid-driven cavity flow. The lid moves to the right and drives the flow. Contours show a snapshot of the turbulent kinetic energy.

realizations of the chaotic system leads to variation in the accuracy of the modes calculated from them. To account for this, we repeat the process of finding the modes 400 times for each m and take the mean square inner product with the fully converged mode.

Figure 2.5 shows the results. As expected, the modes from the uncorrelated data matrix outperform those from the Hankel matrix. The x -axis is logarithmic, and the two curves are roughly a constant horizontal distance from one another. This means that to achieve the same accuracy as an uncorrelated data matrix with m columns, a Hankel matrix must have cm columns, where c is some constant independent of m ; here it is roughly 8, though this number depends sensitively on the parameters used.

2.7.3 Faster computation by sampling the Hankel matrix

Choosing between taking the SVD of a Hankel matrix or an uncorrelated data matrix of the same size may be unrealistic — simulation or experimental data can be in short supply, and it is not trivial to simply generate the extra data needed to construct the uncorrelated data matrix. A more relevant question might be: given a time series of snapshots, should one form the entire Hankel matrix or can many of the columns be omitted? The latter has the advantage of reducing computational cost, in terms of both CPU time and memory requirements, while the former may be more accurate if data is scarce. The accuracy loss depends on how many columns are removed, but we demonstrate here that the loss of accuracy can be negligible for orders of magnitude computational speedup.

Again, we choose $T = 2$, so $d = 21$. Starting with a Hankel matrix with $m_H = 5000$

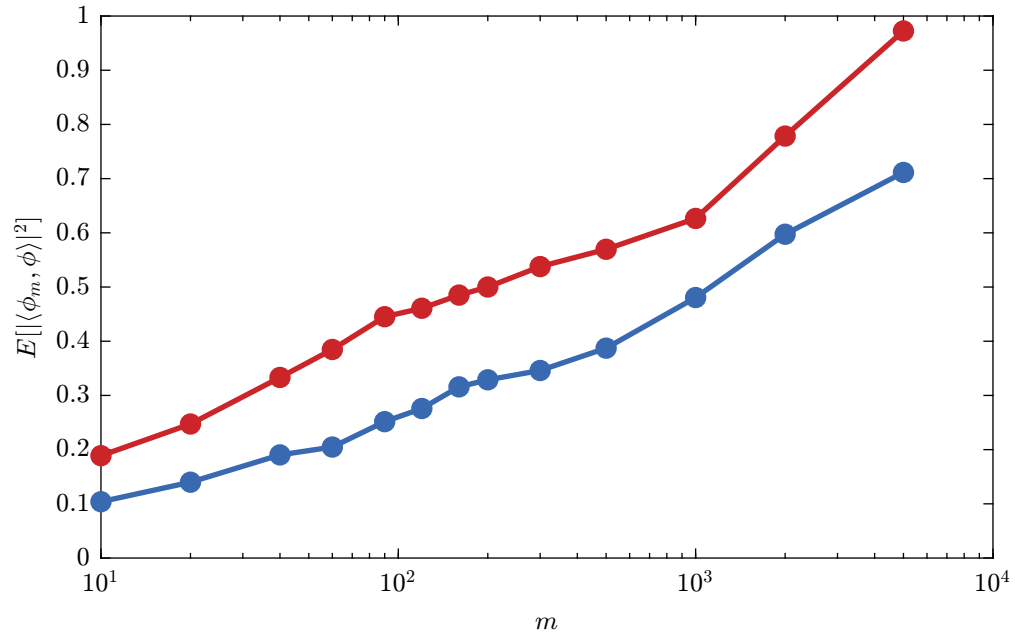


Figure 2.5: Convergence of the modes with the width m of the matrix: (blue) Hankel matrix; (red) uncorrelated data matrix. The modes from the uncorrelated data matrix converge faster than those from the Hankel matrix. This improved convergence is observed because the columns of the uncorrelated data matrix represent data from more of the attractor than the columns of the Hankel matrix of the same size.

columns, we form a reduced data matrix by retaining only m columns with the maximum possible spacing. For example, if $m = 51$, we use every 100th column of the Hankel matrix to form the data matrix. Removing columns changes the time step along rows but not along columns, and our analysis in section 2.5.2 indicates that this only changes the rate of convergence of the modes, not what the modes converge to. We take the SVD of both the Hankel and data matrices and compare each mode from the data matrix to its respective Hankel matrix mode using the square of the inner product, i.e., $|\langle \phi_k^H, \phi_k^D \rangle_{x,t}|^2$ where ϕ_k^H is the k -th mode from the Hankel matrix and ϕ_k^D is the k -th mode from the data matrix; values of 1 and 0 indicate identical modes and orthogonal modes, respectively. We repeat this process 200 times to see how well, on average, the modes from the much smaller data matrix approximate the modes from the full Hankel matrix.

Figure 2.6 shows this accuracy metric as a function of m for the first three modes from each matrix, reporting both the mean (solid) and the median (dashed) of the 200 trials. Both the first and second modes from the $m = 500$ data matrix capture over 97% of the energy of the corresponding Hankel modes (mean), and over 99% (median). For $m = 1000$, the first and second modes both capture over 99.9% of the corresponding Hankel modes. Even these modest reductions in m offer significant computational savings, scaling here like the square of the reduction in m .

The ratio of the number of columns in the sampled Hankel matrix to that of the full Hankel matrix needed to achieve a given accuracy is not constant. For example, if the time step Δt is smaller, a sparser sampling of the Hankel matrix can be used to achieve the same accuracy because k columns of the Hankel matrix represent less of the underlying flow. The speedup gained by sampling the Hankel matrix can be significant; for problems with a complicated spatial domain, the height of the data matrices, $N_x d$, will likely be much larger than the number of columns, m . Therefore the time complexity of the SVD is m^2 , and reductions in m save considerable time. We reiterate that this cost savings is enabled by the insight, provided by the connection between Hankel singular modes and space-time POD, that the time step along rows and columns have different meanings and need not be equal.

2.7.4 Convergence to space-only POD for short times

Here, we demonstrate the convergence of the space-time POD modes to space-only POD modes as the time interval they are defined on approaches zero, $T \rightarrow 0$. Space-only POD modes are defined on Ω and space-time POD modes are defined on $\Omega \times [0, T]$, so in order to compare the two, we take the square of the space-only inner product, averaged over time. In Figure 2.7, we see that the spatial part of the space-time modes indeed converges to space-

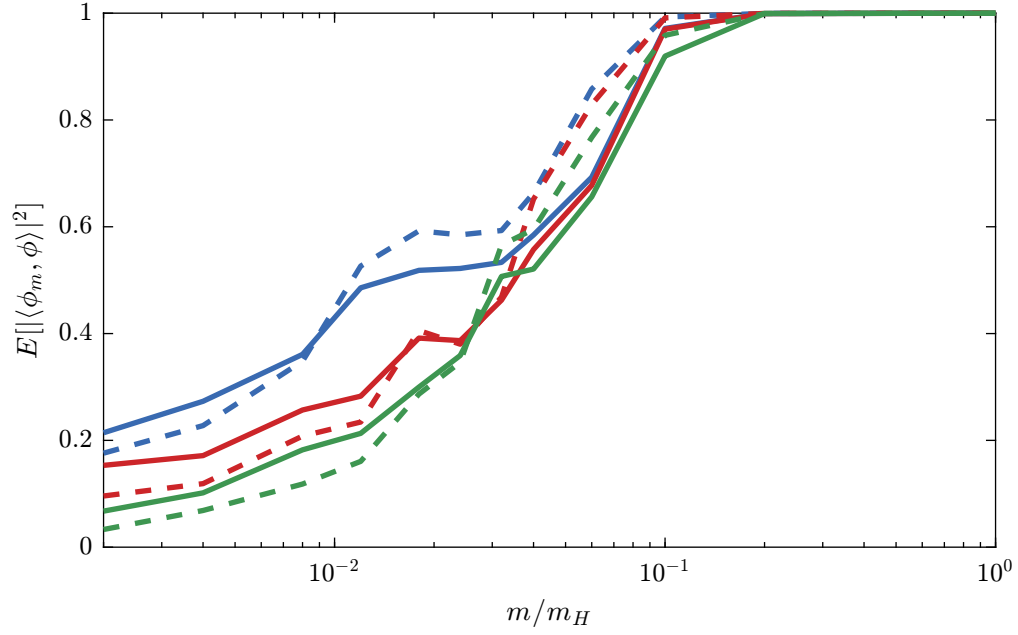


Figure 2.6: Similarity between modes from a downsampled Hankel matrix, retaining m/m_H columns, and the Hankel modes. Both the mean (solid) and the median (dashed) are reported for the 200 trials. The first mode (blue) and the second mode (red) are more similar than the third mode (green). For the parameters used, retaining 1/10 and 1/5 of the columns yields modes that capture over 97% and 99.9% of the energy of the Hankel modes, respectively, for both the first and second modes, on average (mean).

only modes as $T \rightarrow 0$. In fact, the square inner product of the two remains near unity even when the space-time modes are defined over a time on the order of the convective time-scale; when $T = 1$ it is 0.937, and it only drops below 0.9 when $T > 1.4$.

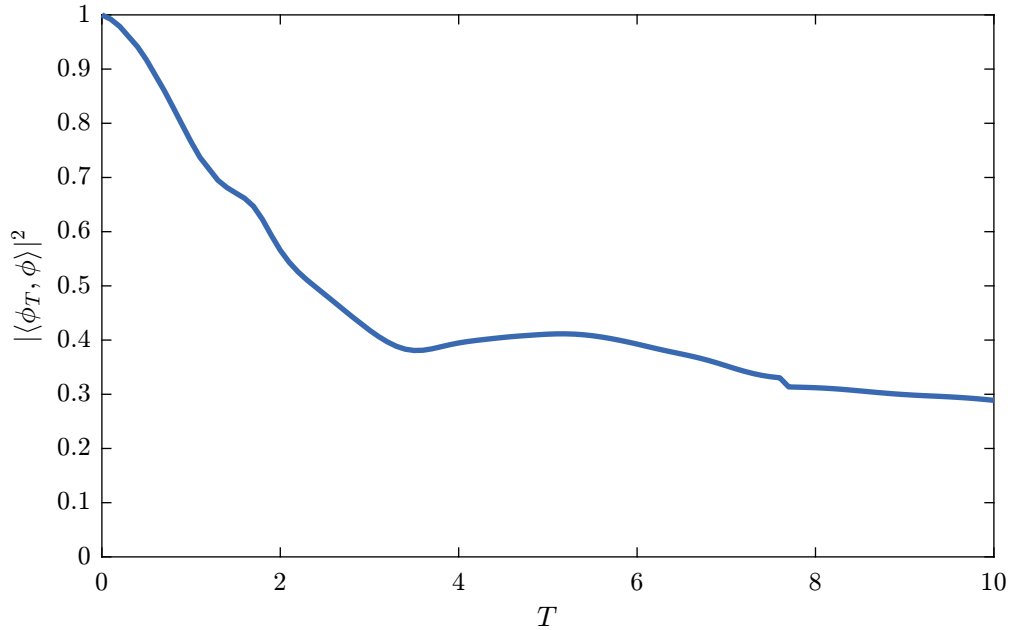


Figure 2.7: Convergence of space-time POD modes ϕ_T to space-only POD modes ϕ for small T . The spatial dependence of the first space-time mode at each T is compared to the first space-only mode using the space-only inner product, averaged over the time evolution of the space-time mode.

2.7.5 Convergence to spectral POD for long times

Here, we demonstrate that as the time interval becomes long, the time dependence of the modes becomes Fourier. Modes are computed via the SVD of a data matrix with large T and 800 well-spaced columns. We do this for different values of T and examine the frequency content of the modes. The expectation is that the modes will become delta functions in the frequency domain as T increases since we know that SPOD modes, which have purely Fourier time dependence, are the limits of space-time POD for long time windows. The spectral content of the space-time modes may vary over the spatial extent of the mode, so we use the norm of their Fourier component,

$$\|\hat{\phi}(\omega)\|_x^2 = \langle \hat{\phi}(\omega, \mathbf{x}), \hat{\phi}(\omega, \mathbf{x}) \rangle_x, \quad (2.7.1)$$

which is equivalent to the power spectral density of the mode integrated over the domain, as a measure of their overall spectral content. Figure 2.8 demonstrates the convergence with increasing T of the spectra of the leading mode to a delta function in frequency.

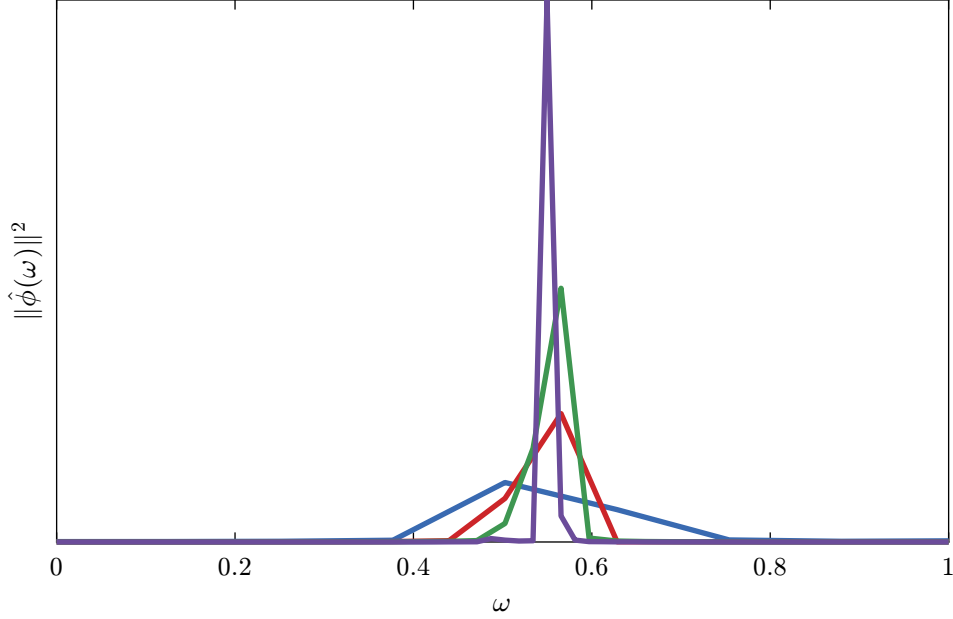


Figure 2.8: Convergence of space-time POD to spectral POD, visualized in terms of the power spectral density of the space-time POD modes for various values of T : (blue) $T = 50$; (red) $T = 100$; (green) $T = 200$; (purple) $T = 400$. As T is increased, more and more of the energy of the modes is contained to an increasingly narrow band of frequencies, indicating convergence toward a discrete frequency, consistent with the time dependence of SPOD modes.

2.7.6 Fully Exploiting Ergodicity

Finally, we demonstrate that the modes from the Toeplitz correlation matrix offer a significant improvement over those from the Hankel data matrix. We extract the modes by taking the SVD and eigendecomposition of the Hankel and Toeplitz matrices, respectively, and calculate the energy of the flow captured by the first few modes of both methods. We repeat this many times to generate PDFs of the energy captured, observing that the Toeplitz modes capture more energy. We also compare the modes from both methods to fully converged Hankel modes and tabulate the results for different parameters, again finding that the Toeplitz modes are more accurate.

Energy of the space-time POD modes, (2.3.1), is an appropriate metric for evaluating the performance of the two methods for obtaining modes because it is the quantity optimized

in the definition of the POD problem. Analogous to (2.1.6), we may rewrite the energy of a mode in terms of its projection coefficients onto the exact modes,

$$\lambda[\phi] = \sum_{k=1}^{\min(m, N_x d)} c_k^2 \lambda_k \quad \text{with} \quad c_k = \langle \phi, \phi_k \rangle_{x,t} / \|\phi\|_{x,t}, \quad (2.7.2)$$

where ϕ_k is the k^{th} space-time POD mode. We calculate the energy of the modes produced by the Hankel and Toeplitz methods using the above formula, where the ‘true’ modes are calculated using a Hankel matrix built from a 40,000-long time series with the same N_x and d .

Given m and d , and an $m+d-1$ -long time series, we compute the modes using the $N_x d \times m$ Hankel matrix and the $N_x d \times N_x d$ Toeplitz matrix and calculate their energies. Forming the Toeplitz matrix and taking its eigendecomposition is infeasible for large systems, so here, we take the N_x points in the simulation with the highest variance for small values of N_x . Of course, the accuracy of the modes from either method depends on the time series used, so we repeat this process 1000 times, generating PDFs for different choices of m , d , and N_x . A PDF with significant support only near unity indicates a high probability of producing an accurate mode.

Figure 2.9 shows the PDF for the energy captured by the first mode from each method as a fraction of the energy of the true optimal mode (λ_1). Beginning with (a) in Figure 2.9, we see that the mean of the modes from the Toeplitz method is higher than that of the modes from the Hankel method. Both methods give a bimodal distribution, and these peaks represent instances where the method mistakes the true second mode for the first mode. There is a deep trough between the peaks of the Toeplitz method, indicating that it produces a mode close to the first or to the second true mode rather than some combination thereof. The Hankel method displays some of this behavior, but it is significantly less pronounced, indicating that it produces something between the two true modes more often than the Toeplitz method. This behavior is also present in (b), (c), and (d).

As expected, for a fixed value of d and N_x , increasing m yields a higher probability of more accurate modes, and the modes from both methods in (c) are more accurate than those in (b), which are more accurate than those in (a). We also see that as $\frac{m}{d}$ increases, the difference in the accuracy of the two methods decreases, and the PDFs of the two methods become closer going from (a) to (b) and from (b) to (c). We see this behavior because for $\frac{m}{d} \gg 1$, all of the elements on the same diagonal of $\mathbf{C} = \mathbf{H}\mathbf{H}^*$ are sums of mostly the same products of terms from the time series, so $\mathbf{C}$ is nearly Toeplitz, whereas for $\frac{m}{d} \approx 1$ this is not true. In (e), $\frac{m}{d} = 50$, and the difference between the methods is hard to see by looking at the PDFs, though there is still some difference in the means and medians, especially for

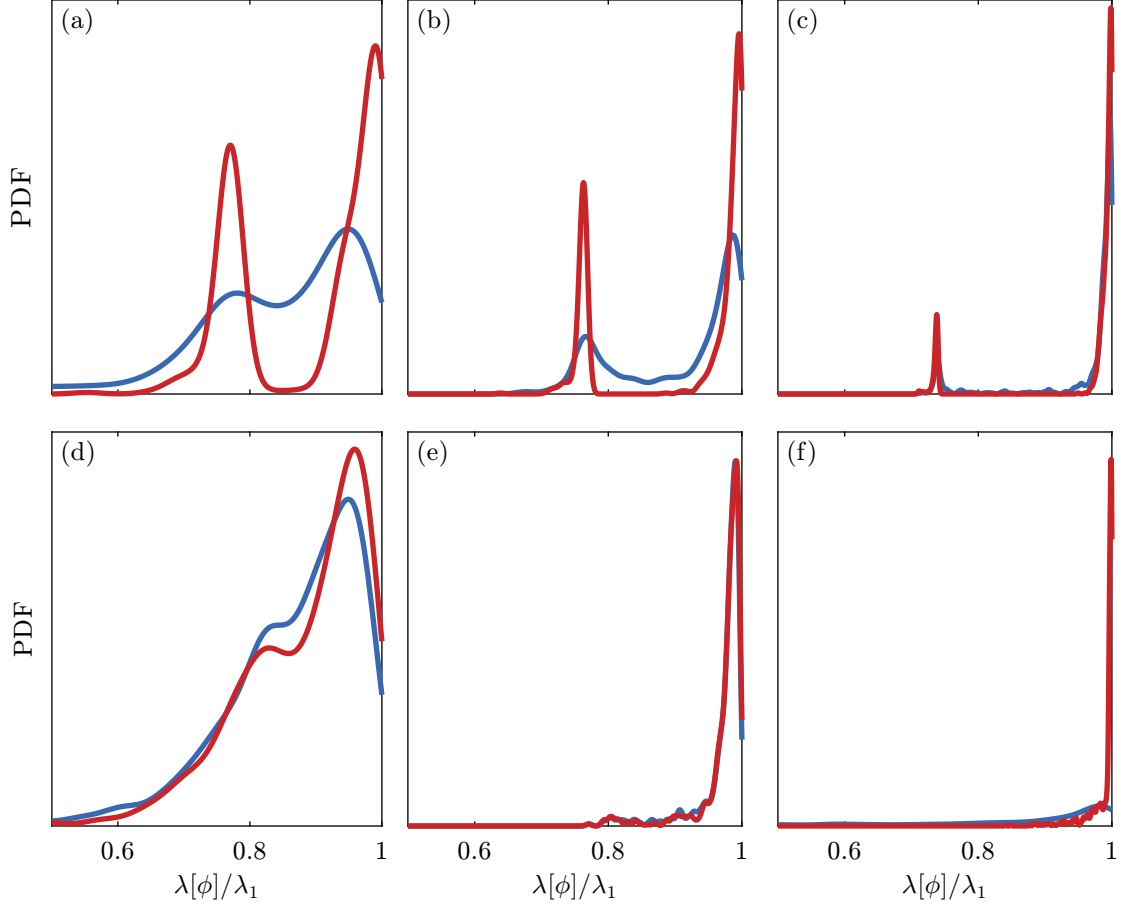


Figure 2.9: PDFs of energy captured by the first modes of the Hankel (blue) and Toeplitz (red) methods for various choices of m , d , and N_x . Both methods use the same time series of length $m + d - 1$ to approximate the modes. The PDFs are calculated with 1000 samples and the energies are divided by the maximum possible energy of one mode. These parameter values are, (a): $m = 30$, $d = 30$, $N_x = 1$. (b): $m = 100$, $d = 30$, $N_x = 1$. (c): $m = 300$, $d = 30$, $N_x = 1$. (d): $m = 300$, $d = 30$, $N_x = 3$. (e) $m = 600$, $d = 30$, $N_x = 3$. (f) $m = 10$, $d = 10$, $N_x = 1$.

the second mode, as reported in Figure 2.10.

For fixed values of d and m , increasing N_x worsens the accuracy of the modes, and the modes in (d) are less accurate on average than those in (b). Similarly, increasing d worsens the accuracy of the modes, as can be seen by comparing (a) and (f). In fact, the modes in (f) are more accurate from both methods than those in (a) despite being calculated with lower m . Increasing N_x or d decreases the quality of the modes because it causes them to be in a higher dimensional space, which makes them more difficult to approximate. If m is large enough, however, (signifying a long time series from which to calculate the modes), both methods will converge for any N_x and d .

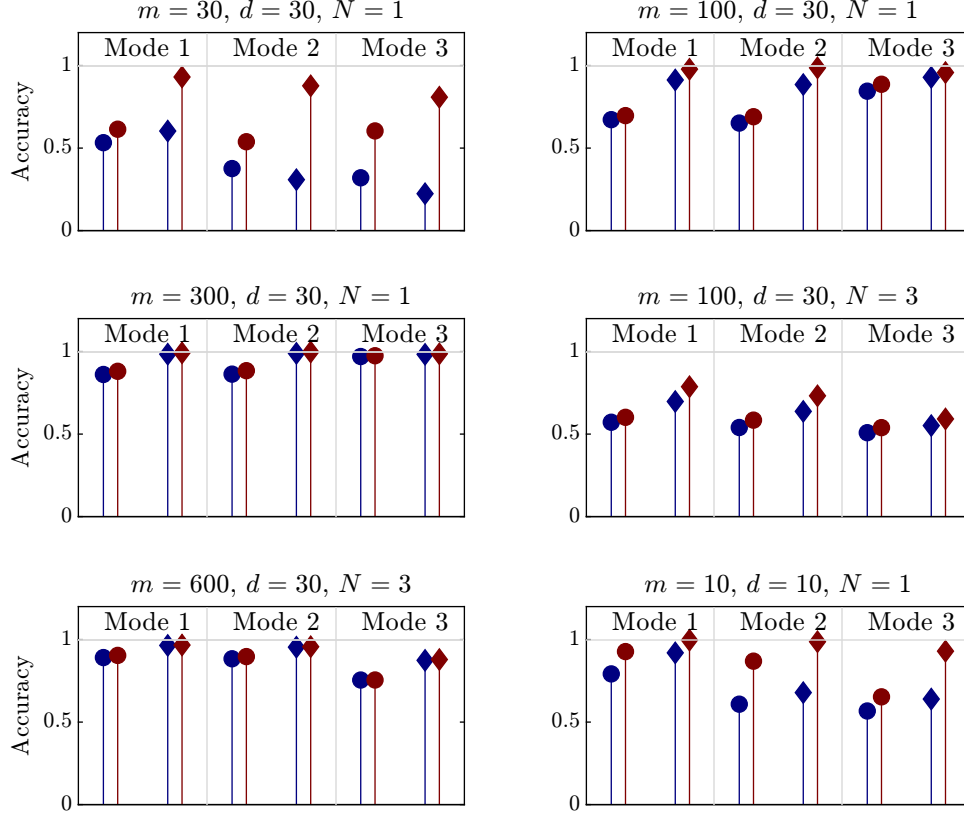


Figure 2.10: Mean (circles) and median (squares) accuracy of Hankel modes (blue) vs. Toeplitz modes (red) calculated from 1000 time series of the most energetic N_x points of the lid-driven cavity flow. Accuracy is measured as the square inner product against a ‘true mode.’ When the data is severely limited (m is relatively small), the difference is more pronounced.

To evaluate the accuracy of the latter modes, say the k^{th} , we add the energies of the modes up to the k^{th} . We do this instead of simply calculating $\lambda[\phi_k]$ because this would erroneously reward switching the first and the k^{th} mode. We denote the sum of the energies as

$$\lambda[\phi_{1,2,\dots,k}] = \sum_{i=1}^k \lambda[\phi_i], \quad (2.7.3)$$

and this quantity is interpreted as the energy captured by the first k modes. Again, we generate PDFs of these energies using 1000 time series of length $m + d - 1$, computing modes using both methods, and calculating the energy they capture. Figure 2.11 shows the results for the first four modes using $m = 30$, $d = 30$, $N_x = 1$. The energy captured by the first two Toeplitz modes is significantly greater than that of the first two Hankel modes. The same can be said about the first three and first four Toeplitz modes. These observations hold for the majority of parameter combinations.

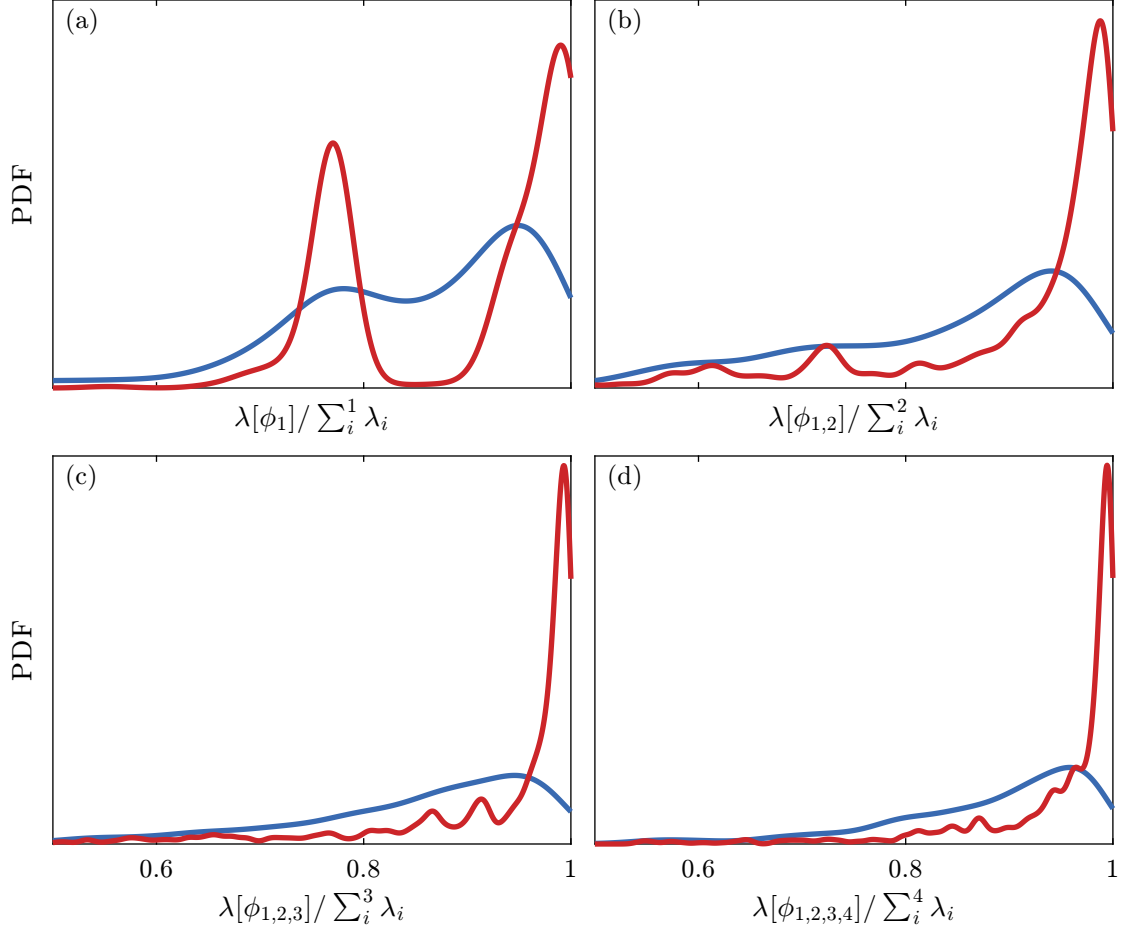


Figure 2.11: PDFs of the energy captured by the first four modes of the Toeplitz and Hankel methods, normalized by the maximum possible energy by four modes. These are calculated for $m = 30$, $d = 30$, $N_x = 1$. The difference is more severe for the latter modes, which we observed for the majority of the parameter choices that we tested.

A second metric for evaluating the accuracy of the modes is to compare them directly to fully converged modes, via the square inner product. Again a value near unity indicates an accurate mode, though a value near zero for, e.g., the third mode does not indicate that that mode captures no energy, just that it captures energy orthogonal to that of the converged third mode. We calculate these converged modes from the 40,000-long time series. For each choice of m , d , and N_x , we calculate the modes from the Hankel and Toeplitz matrices, take the inner product with the converged modes, and repeat 1000 times. We record the mean and median of the square inner products for the first three modes in Figure 2.10.

As d increases, the number of realizations m needed to get accurate modes also increases for both methods. Similarly, as N_x increases, m may need to be increased to retain accuracy. Intuitively, this tells us that we need more data to find a mode in a higher dimensional space.

The improvement of the Toeplitz method decays with m/d , because if this quantity is large, the difference between $\mathbf{H}\mathbf{H}^*$ and the Toeplitz $\tilde{\mathbf{C}}$ becomes small, as discussed in section 2.6. Again, we note that the Toeplitz modes are significantly more accurate when m is not significantly greater than d . Also, there is not a big difference when m is very large, because here the modes are converged. Because a large N_x necessitates a large m for accuracy, m/d will be large and thus the improvement gained by the Toeplitz method is marginal for large N_x . However, these are the cases for which the time complexity of the Toeplitz method is already prohibitive.

2.8 Chapter summary

This chapter begins with a rigorous description of space-only, spectral, and space-time POD, and demonstrates how the former two variants can be recovered as limits of the latter variant. Following this, we show that the SVD of the Hankel matrix used in a variety of ROMs gives an approximation of space-time POD modes. This connection provides insight into the qualities of the Hankel modes, and we give interpretations of them in terms of the integral that defines the space-time eigenvalue problem. We also give examples of how the Hankel modes can be improved by leveraging insights from the connection to space-time POD.

CHAPTER 3

Linear Spectral Solution Operator Projection

3.1 Introduction

Motivated by the trajectory representation properties of SPOD modes, the goal of this chapter 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. 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 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) [86]. 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. [131], 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 [82, 7, 122, 89, 101], 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.

Before introducing SSOP, we describe the details of using SPOD modes for trajectory representation in Section 3.2, and introduce the resolvent operator in Section 3.3, which is relevant for this chapter and the two that follow.

3.2 SPOD modes for encoding trajectories

Here, we formalize the problem of encoding trajectories, and describe how SPOD can be used to achieve efficient trajectory encodings. We also define the discrete spatial and space-time norms and inner products that are used throughout Chapters 3 and 4.

3.2.1 Trajectory representation

Here, we formalize the notion of a space-time encoding scheme in analogy to the space-only encoding framework introduced in Section 1.2.1. We take the trajectory on the interval to be a vector of its values at times $0, \Delta t, \dots, (N_\omega - 1)\Delta t$, where $N_\omega \Delta t = T$. That is, the trajectory is a vector $\mathbf{q}_{\mathcal{J}} = [\mathbf{q}_0^T, \mathbf{q}_1^T, \dots, \mathbf{q}_{N_\omega-1}^T]^T$ where $\mathbf{q}_j$ is the state at time $t_j = j\Delta t$. Throughout this chapter and the following one, a subscript $\mathcal{J}$ is used to indicate the set or vector that is formed by taking the subscript to be each element of the vector $\mathcal{J} = [0, 1, \dots, N_\omega - 1]^T$. For example $t_{\mathcal{J}}$ denotes the vector $[0, \Delta t, \dots, (N_\omega - 1)\Delta t]^T$. We denote the number of time steps N_ω for reasons related to the discrete Fourier transform, which will become clear later.

An encoding scheme for trajectories comprises an encoder $\mathbf{e}^{x,t} : \mathbb{C}^{N_x N_\omega} \rightarrow \mathbb{C}^{r N_\omega}$ and a decoder $\mathbf{d}_{\mathcal{J}}^{x,t} : \mathbb{C}^{r N_\omega} \rightarrow \mathbb{C}^{N_x N_\omega}$. The encoding scheme is effective if $\mathbf{q}_{\mathcal{J}} \approx \mathbf{d}_{\mathcal{J}}^{x,t}(\mathbf{e}^{x,t}(\mathbf{q}_{\mathcal{J}}))$ for trajectories of the system. More precisely, the expected error averaged over the trajectory,

$\mathbb{E}[\|\mathbf{q}_{\mathcal{J}} - \mathbf{d}_{\mathcal{J}}^{x,t}(\mathbf{e}^{x,t}(\mathbf{q}_{\mathcal{J}}))\|_{x,t}^2]$, should be small. The space-time norm is induced in the usual way from a space-time inner product $\|\mathbf{q}_{\mathcal{J}}\|_{x,t} = \sqrt{\langle \mathbf{q}_{\mathcal{J}}, \mathbf{q}_{\mathcal{J}} \rangle_{x,t}}$, and this inner product is given by

$$\langle \mathbf{q}_{\mathcal{J}}^1, \mathbf{q}_{\mathcal{J}}^2 \rangle_{x,t} = \sum_{j=0}^{N_{\omega}-1} \langle \mathbf{q}_j^1, \mathbf{q}_j^2 \rangle_x, \quad (3.2.1)$$

The space-only inner product is $\langle \mathbf{q}^1, \mathbf{q}^2 \rangle_x = (\mathbf{q}^1)^* \mathbf{W} \mathbf{q}^2$ for a positive definite weight matrix $\mathbf{W}$, and the space-only norm $\|\cdot\|_x$ is induced from this inner product in the usual way.

The analog of POD for the case of trajectories is space-time POD [76, 33] — it is the optimal linear encoding/decoding scheme for trajectories in the sense that it minimizes $\mathbb{E}[\|\mathbf{q}_{\mathcal{J}} - \mathbf{d}_{\mathcal{J}}^{x,t}(\mathbf{e}^{x,t}(\mathbf{q}_{\mathcal{J}}))\|_{x,t}^2]$. As discussed in the following subsection, SPOD modes can be used to represent trajectories, and a primary motivation for the SSOP method is that the SPOD representation approaches the accuracy of the space-time POD representation for statistically stationary systems as $T \rightarrow \infty$. Another important point of comparison is the POD representation of trajectories, wherein each time step of the trajectory is encoded using r POD modes $\Phi \in \mathbb{C}^{N_x \times r}$, i.e., where $\mathbf{d}_{\mathcal{J}}^{x,t}(\mathbf{e}^{x,t}(\mathbf{q}_{\mathcal{J}})) = \Phi \Phi^* \mathbf{W} \mathbf{q}_j$.

3.2.2 Encoding using SPOD modes

SPOD modes can be used to represent a trajectory $\mathbf{q}_{\mathcal{J}}$ by representing the trajectory at each frequency $\hat{\mathbf{q}}_k$ using the SPOD modes at the same frequency. The trajectory at the k -th frequency is given by the discrete Fourier transform (DFT) as

$$\hat{\mathbf{q}}_k = \text{DFT}_k[\mathbf{q}_{\mathcal{J}}] = \sum_{j=0}^{N_{\omega}-1} \mathbf{q}_j e^{-i\omega_k t_j}, \quad (3.2.2)$$

where $\omega_k = 2\pi(k - N_{\omega}\Theta(k - N_{\omega}/2))/T$, where Θ is 0 if its argument is negative, and is 1 otherwise. This frequency of the trajectory may be represented by a linear combination of the first r_k SPOD modes as that frequency as

$$\hat{\mathbf{q}}_k \approx \Psi_k \mathbf{a}_k. \quad (3.2.3)$$

In practice, the SPOD modes are obtained using the method described in Section 2.2.2. With N_d temporal blocks of data, N_d SPOD modes are available at each frequency. We denote the full set of modes at ω_k as $\Psi_k^{N_d} = [\psi_{k1}, \psi_{k2}, \dots, \psi_{kN_d}] \in \mathbb{C}^{N_x \times N_d}$. The truncated set of modes, used in (3.2.3), is $\Psi_k \in \mathbb{C}^{N_x \times r_k}$, which includes the first r_k columns of $\Psi_k^{N_d}$. The vector $\mathbf{a}_k \in \mathbb{C}^{r_k}$ in (3.2.3) contains the expansion coefficients. It will become clear below why it is advantageous to retain different numbers of modes at different frequencies. The

expected value of the error in the representation at this frequency can be expressed in terms of the truncated SPOD energies as

$$\mathbb{E}[\|\hat{\mathbf{q}}_k - \mathbf{\Psi}_k \mathbf{a}_k\|_x^2] = \sum_{m=r_k+1}^{N_x} \lambda_{km}, \quad (3.2.4)$$

If $\hat{\mathbf{q}}_k$ is known, the SPOD coefficients may be found by taking inner products between $\hat{\mathbf{q}}_k$ and the modes as

$$\mathbf{a}_k = \mathbf{\Psi}_k^* \mathbf{W} \hat{\mathbf{q}}_k. \quad (3.2.5)$$

This may be written in terms of a space-time encoder $\mathbf{a}_k = \mathbf{e}_k^{x,t}(\mathbf{q}_{\mathcal{J}})$, where the k -th frequency of the encoder is given by

$$\mathbf{e}_k^{x,t}(\mathbf{q}_{\mathcal{J}}) = \mathbf{\Psi}_k^* \mathbf{W} \text{DFT}_k[\mathbf{q}_{\mathcal{J}}]. \quad (3.2.6)$$

The corresponding decoder is given by taking using the SPOD coefficients at each frequency to reconstruct the trajectory in the frequency domain, then applying the inverse DFT to the result. Specifically, the j -th time step of the decoded trajectory is

$$\tilde{\mathbf{q}}_j = \mathbf{d}_j^{x,t}(\mathbf{a}_{\mathcal{K}}) = \frac{1}{N_{\omega}} \sum_{k=0}^{N_{\omega}-1} \mathbf{\Psi}_k \mathbf{a}_k e^{i\omega_k t_j}. \quad (3.2.7)$$

Hereafter, a subscript $\mathcal{K}$ is used to denote all frequencies, analogous to our use of the subscript $\mathcal{J}$ for all times. The relevant metric for accuracy is the spatial error averaged over the times in the interval, which is the square of the space-time norm of the difference. Due to Parseval's theorem, this is proportional to the error summed over frequencies,

$$\|\tilde{\mathbf{q}}_{\mathcal{J}} - \mathbf{q}_{\mathcal{J}}\|_{x,t}^2 = \frac{1}{N_{\omega}} \sum_{k=0}^{N_{\omega}-1} \|\mathbf{\Psi}_k \mathbf{a}_k - \hat{\mathbf{q}}_k\|_x^2, \quad (3.2.8)$$

where $\tilde{\mathbf{q}}_{\mathcal{J}}$ is the decoded trajectory. Taking the expected value of both sides and using (3.2.4), we have

$$\mathbb{E}[\|\tilde{\mathbf{q}}_{\mathcal{J}} - \mathbf{q}_{\mathcal{J}}\|_{x,t}^2] = \frac{1}{N_{\omega}} \sum_{k=0}^{N_{\omega}-1} \sum_{m=r_k+1}^{N_x} \lambda_{km}. \quad (3.2.9)$$

Thus, the expected accuracy of the trajectory representation is determined by the energies of the truncated SPOD modes. This informs the best allocation of modes across frequencies: for the best accuracy given a total number of modes, one should retain the modes with the highest energies over all frequencies. We denote the average number of modes retained by r , and denote the j -th largest SPOD energy at any frequency by $\tilde{\lambda}_j$. The number of modes retained at the k -th frequency r_k is then the number of modes at this frequency with energy

greater than or equal to $\tilde{\lambda}_{(rN_\omega)}$,

$$r_k = |\{m : \lambda_{km} \geq \tilde{\lambda}_{(rN_\omega)}\}|, \quad (3.2.10)$$

where $|\cdot|$ denotes the cardinality of the set, and the parentheses in the subscript are a reminder that r and N_ω are multiplied and are not separate indices.

The SPOD encoding/decoding scheme may be written in terms of the forward and inverse DFTs as

$$\tilde{\mathbf{q}}_j = \frac{1}{N_\omega} \sum_{k=0}^{N_\omega-1} e^{i\omega_k j \Delta t} \hat{\mathbf{P}}_k \sum_{l=0}^{N_\omega-1} e^{-i\omega_k l \Delta t} \mathbf{q}_l, \quad (3.2.11)$$

where $\hat{\mathbf{P}}_k$ is $\mathbf{\Psi}_k \mathbf{\Psi}_k^* \mathbf{W}$. The l -th time in the trajectory thus has the following impact on the j -th time in the representation

$$\left(\frac{1}{N_\omega} \sum_{k=0}^{N_\omega-1} e^{i\omega_k (j-l) \Delta t} \hat{\mathbf{P}}_k \right) \mathbf{q}_l, \quad (3.2.12)$$

and j -th time in the projected trajectory may be written as the following discrete convolution

$$\tilde{\mathbf{q}}_j = \sum_{l=0}^{N_\omega-1} \mathbf{P}_{[j-l]} \mathbf{q}_l, \quad (3.2.13)$$

where

$$\mathbf{P}_s = \text{IDFT}_s[\hat{\mathbf{P}}_\kappa] = \frac{1}{N_\omega} \sum_{k=0}^{N_\omega-1} e^{i\omega_k s \Delta t} \hat{\mathbf{P}}_k, \quad (3.2.14)$$

and where $[j-l]$ indicates $j-l \bmod N_\omega$. The representation in terms of the convolution is useful for interpreting the behavior of the method in Section 3.5.

The SPOD trajectory representation (3.2.7) may be viewed as a rank- rN_ω space-time representation, i.e., it is a sum of spatiotemporal modes with constant coefficients of the form $\sum_{i=1}^{rN_\omega} \boldsymbol{\xi}_i a_i$ with $\boldsymbol{\xi}_i \in \mathbb{C}^{N_\omega N_x}$. In the SPOD representation, each spatiotemporal mode is an SPOD mode at some frequency ω_k multiplied by the associated oscillatory time dependence $e^{i\omega_k t}$. The most accurate space-time representation is space-time POD modes — these lead to lower trajectory representation error than any other space-time basis. The fact that motivates our use of SPOD modes is that as the time interval $T \rightarrow \infty$, the SPOD modes approach space-time POD modes [76, 77, 33].

Figure 3.1 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 error, 2%, 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. The error

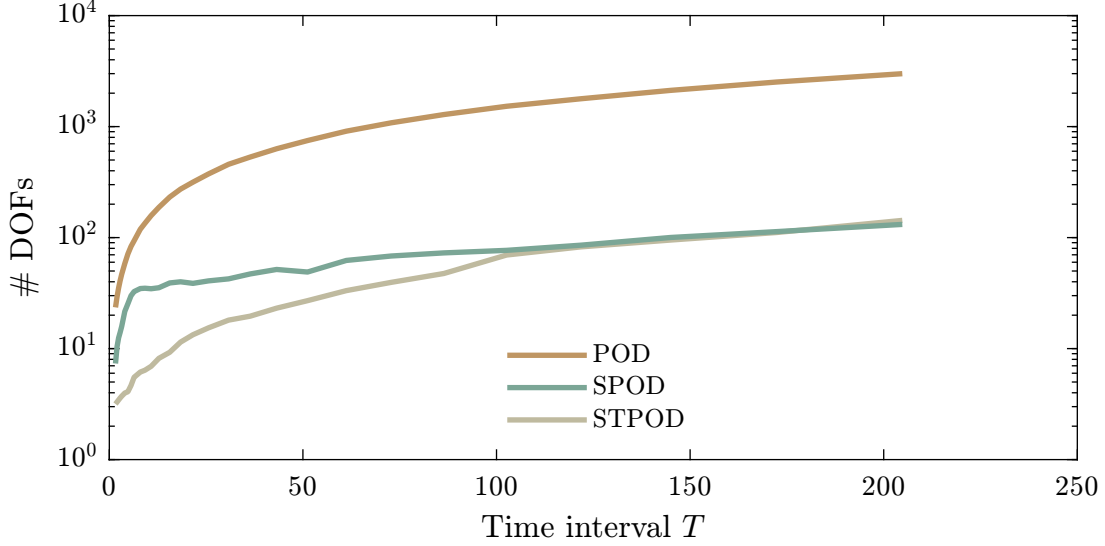


Figure 3.1: Number of degrees of freedom (DOFs) required to achieve 2% representation error as a function of the length of the time interval of the trajectory $[0, T]$. The convergence of SPOD to space-time POD proven in Chapter 2 can be seen. The data are from the Ginzburg-Landau system introduced in Section 3.5.

is defined in 3.2.15 below. The data are from the Ginzburg-Landau system described in Section 3.5.

The efficacy of the SPOD modes can also be compared to that of POD modes by calculating the relative error produced by both encoding schemes as a function of the number of degrees of freedom. The relative error is

$$\frac{\|\tilde{\mathbf{q}}_{\mathcal{J}} - \mathbf{q}_{\mathcal{J}}\|_{x,t}^2}{\|\mathbf{q}_{\mathcal{J}}\|_{x,t}^2}, \quad (3.2.15)$$

where $\tilde{\mathbf{q}}_{\mathcal{J}}$ is given by either the POD or SPOD encoding/decoding. In Figure 3.2, we show these errors as a function of r , the number of modes, averaged over 173 trajectories of the linear Ginzburg-Landau system described later in the paper. Figure 3.2 demonstrates the superior representational power of SPOD modes over POD modes; with the same number of total coefficients, the SPOD representation results in more than two orders of magnitude lower error. Space-time POD is not included in this plot as computing the requisite $N_{\omega}r$ modes requires a prohibitively large amount of data.

The goal of this chapter and the next is to develop a reduced-order model for nonlinear systems that solves for the SPOD coefficients quickly and accurately, so as to recover the accuracy shown in Figure 3.2 at low cost.

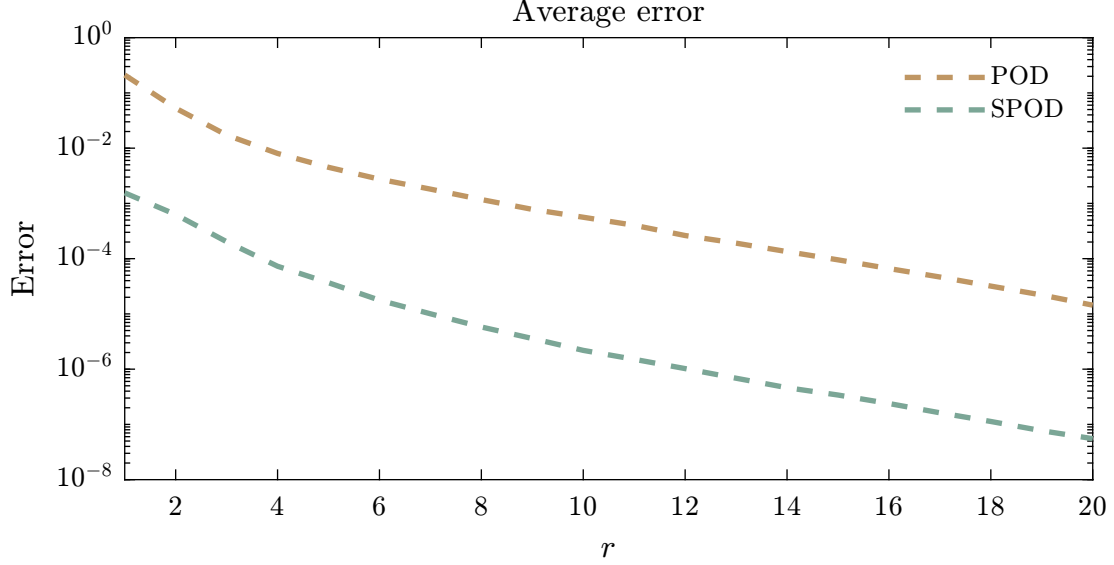


Figure 3.2: The relative error for the SPOD and POD trajectory encoders as a function of r , the number of modes used.

3.3 Resolvent analysis

Here, we introduce the resolvent operator, which is a fundamental object in linear systems theory, and is relevant to the method developed in this chapter and in the two that follow.

Consider the LTI system

$$\dot{\mathbf{q}} = \mathbf{A}\mathbf{q} + \mathbf{B}\mathbf{f}, \quad (3.3.1)$$

where $\mathbf{q}(t) \in \mathbb{C}^{N_x}$ is the state, $\mathbf{A} \in \mathbb{C}^{N_x \times N_x}$ is the system matrix, and $\mathbf{f}(t) \in \mathbb{C}^{N_f}$ is some known forcing that is mapped onto the system by the matrix $\mathbf{B} \in \mathbb{C}^{N_x \times N_f}$. The resolvent operator $\mathbf{R}(\omega) \in \mathbb{C}^{N_x \times N_x}$ is the transfer function that relates the Fourier transformed-state $\hat{\mathbf{q}}(\omega)$ to the Fourier-transformed forcing $\hat{\mathbf{f}}(\omega)$,

$$\hat{\mathbf{q}}(\omega) = \overbrace{(i\omega\mathbf{I} - \mathbf{A})^{-1} \mathbf{B}}^{\mathbf{R}(\omega)} \hat{\mathbf{f}}(\omega). \quad (3.3.2)$$

Note that here, the Fourier transforms that define $\hat{\mathbf{q}}(\omega)$ and $\hat{\mathbf{f}}(\omega)$ are continuous and defined on an infinite time interval, unlike the DFT that defined $\hat{\mathbf{q}}_k$ in Section 3.2.

Much insight can be gained into the behavior of the system by studying the properties of $\mathbf{R}(\omega)$, and one particularly important tool for doing so is the singular value decomposition [133, 29, 82, 81]. The squares of the singular values of the resolvent operator are interpreted as gains that relate the energy that is input to the linear system by the forcing in the direction of the corresponding right singular vector to the energy in the state in the direction of the

corresponding left singular vector. This analysis can be predictive of structures that appear in the system, especially in cases of high gain separation, i.e., when the first singular value is much larger than the second. In this case, the portion of the forcing that projects onto the first right singular vector is amplified much more than any other portion of the forcing, and the resulting structure will thus resemble the first left singular vector.

This intuitive analysis was made rigorous by Ref. [131] who showed that 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. We discuss this equivalence in more detail in Section 3.4.6, where we use it to formulate a data-free version of the method developed in this chapter by substituting the resolvent modes for the SPOD modes in the method.

While (3.3.1) is linear, the resolvent operator can be used to gain insight into nonlinear systems by considering the nonlinearity as a forcing on the linear system. This insight, first made by Ref. [82], has led to a significant literature on using the resolvent operator to understand and model even turbulent flows [26, 87, 101]. A particularly remarkable result is that even when the nonlinearity is far from white, the resolvent modes often are predictive of the structures that appear in the flow [82, 123, 75, 7], and direct comparisons between SPOD modes and resolvent modes have been made [122, 89, 101].

3.4 Spectral solution operator projection method

The method aims to find, as accurately as possible, the SPOD coefficients of the solution to the LTI system

$$\dot{\mathbf{q}} = \mathbf{A}(\boldsymbol{\mu})\mathbf{q} + \mathbf{B}\mathbf{f}, \quad (3.4.1)$$

Specifically, we seek the SPOD coefficients of $\mathbf{q}_{\mathcal{T}}$, the temporally discretized solution to this system on the interval $[0, T]$. In (3.4.1), $\mathbf{f} \in \mathbb{C}^{N_f}$ is a known time-dependent forcing, and the matrices $\mathbf{A}(\boldsymbol{\mu}) \in \mathbb{C}^{N_x \times N_x}$, and $\mathbf{B} \in \mathbb{C}^{N_x \times N_f}$ are the usual system matrices, the former of which depends affinely on the parameter vector $\boldsymbol{\mu} \in \mathbb{R}^{N_\mu}$. We suppress this parameter dependence for now; after deriving the nonlinear reduced-order model in Chapter 4, we reintroduce it and show how the reduced matrices involved may be computed quickly given $\boldsymbol{\mu}$. We also assume that the system has been transformed such that the mean of the state is zero.

3.4.1 Solution operator

The SPOD coefficients for the solution can be written in terms of a space-time inner product. The m -th SPOD coefficient at the k -th frequency, for example, is

$$a_{km} = \langle \mathbf{q}_{\mathcal{J}}, \boldsymbol{\psi}_{km} e^{i\omega_k t_{\mathcal{J}}} \rangle_{x,t}, \quad (3.4.2)$$

where $a_{km} = (\mathbf{a}_k)_m$ and where the space-time inner product is given by (3.2.1). We write the first argument of the inner product in terms of a solution operator

$$a_{km} = \langle \mathbf{L}[\mathbf{q}_0, \mathbf{B}\mathbf{f}(t); t_{\mathcal{J}}], \boldsymbol{\psi}_{km} e^{i\omega_k t_{\mathcal{J}}} \rangle_{x,t}. \quad (3.4.3)$$

This solution operator $\mathbf{L}[\mathbf{q}_0, \mathbf{g}(t); t_j]$ maps the initial condition, forcing, and a time t_j to $\mathbf{q}(t_j)$ as

$$\mathbf{L}[\mathbf{q}_0, \mathbf{g}(t); t_j] = e^{\mathbf{A}t_j} \mathbf{q}_0 + \int_0^{t_j} e^{\mathbf{A}(t_j-t')} \mathbf{g}(t') dt'. \quad (3.4.4)$$

The name of the method is due to the fact that (3.4.3) can be thought of as a space-time projection of the solution operator onto the SPOD modes.

3.4.2 Evaluating the inner product

Analytic progress can be made on evaluating the inner product (3.4.3) by breaking it into a DFT of the solution operator and a spatial inner product as

$$\hat{\mathbf{q}}_k = \sum_{j=0}^{N_{\omega}-1} \left(e^{\mathbf{A}j\Delta t} \mathbf{q}_0 + \int_0^{j\Delta t} e^{\mathbf{A}(j\Delta t-t')} \mathbf{B}\mathbf{f}(t') dt' \right) e^{-i\omega_k j\Delta t}. \quad (3.4.5a)$$

$$\mathbf{a}_k = \boldsymbol{\Psi}_k^* \mathbf{W} \hat{\mathbf{q}}_k \quad (3.4.5b)$$

where the left multiplication by $\boldsymbol{\Psi}_k^* \mathbf{W}$ effects the spatial inner product for all modes at the k -th frequency. The sum in (3.4.5a) can be computed analytically assuming that (i) the forcing is of the form $\mathbf{f}(t) = \sum_{l=0}^{N_{\omega}-1} \hat{\mathbf{f}}_l e^{i\omega_l t}$, and (ii) that $i\omega_l \mathbf{I} - \mathbf{A}$ is invertible for all included l (this latter assumption can be relaxed if needed).

To aid in the ensuing discussion, we split (3.4.5a) into two parts: the response to the initial condition and the response to the forcing. The first of these is

$$\hat{\mathbf{q}}_{k,ic} = \sum_{j=0}^{N_{\omega}-1} e^{(\mathbf{A}-i\omega_k \mathbf{I})j\Delta t} \mathbf{q}_0. \quad (3.4.6)$$

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

written as

$$\hat{\mathbf{q}}_{k,ic} = (\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t})^{-1} (\mathbf{I} - e^{(\mathbf{A} - i\omega_k)N_\omega \Delta t}) \mathbf{q}_0. \quad (3.4.7)$$

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

$$\hat{\mathbf{q}}_{k,ic} = (\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t})^{-1} (\mathbf{I} - e^{\mathbf{A}T}) \mathbf{q}_0. \quad (3.4.8)$$

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

$$\hat{\mathbf{q}}_{k,force} = \sum_{j=0}^{N_\omega-1} e^{-i\omega_k j \Delta t} \int_0^{j\Delta t} e^{\mathbf{A}(j\Delta t-t')} \mathbf{B} \mathbf{f}(t') dt'. \quad (3.4.9)$$

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

$$\hat{\mathbf{q}}_{k,force} = \frac{1}{N_\omega} \sum_{j=0}^{N_\omega-1} e^{-i\omega_k j \Delta t} e^{\mathbf{A}j\Delta t} \int_0^{j\Delta t} \sum_{l=0}^{N_\omega-1} e^{(i\omega_l \mathbf{I} - \mathbf{A})t'} \mathbf{B} \hat{\mathbf{f}}_l dt'. \quad (3.4.10)$$

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

$$\hat{\mathbf{q}}_{k,force} = \frac{1}{N_\omega} \sum_{j=0}^{N_\omega-1} e^{-i\omega_k j \Delta t} \sum_{l=0}^{N_\omega-1} \mathbf{R}_l (e^{i\omega_l j \Delta t} - e^{\mathbf{A}j\Delta t}) \mathbf{B} \hat{\mathbf{f}}_l, \quad (3.4.11)$$

where $\mathbf{R}_k = (i\omega_k \mathbf{I} - \mathbf{A})^{-1}$ is the resolvent operator at the k -th frequency. Equation (3.4.11) can then be written as

$$\hat{\mathbf{q}}_{k,force} = \frac{1}{N_\omega} \sum_{j=0}^{N_\omega-1} \sum_{l=0}^{N_\omega-1} \mathbf{R}_l (e^{i(\omega_l - \omega_k)j\Delta t} - e^{(\mathbf{A} - i\omega_k \mathbf{I})j\Delta t}) \mathbf{B} \hat{\mathbf{f}}_l. \quad (3.4.12)$$

The frequency difference term in (3.4.12) 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 (3.4.5a) becomes

$$\hat{\mathbf{q}}_{k,force} = \mathbf{R}_k \mathbf{B} \hat{\mathbf{f}}_k - \frac{1}{N_\omega} (\mathbf{I} - e^{(\mathbf{A} - i\omega_k) \Delta t})^{-1} (\mathbf{I} - e^{\mathbf{A}T}) \sum_{l=0}^{N_\omega-1} \mathbf{R}_l \mathbf{B} \hat{\mathbf{f}}_l. \quad (3.4.13)$$

Recombining our simplified expressions for $\hat{\mathbf{q}}_{k,ic}$ and $\hat{\mathbf{q}}_{k,force}$, we obtain the following equation

for the k -th component of the DFT of the trajectory:

$$\hat{\mathbf{q}}_k = \mathbf{R}_k \mathbf{B} \hat{\mathbf{f}}_k + \left(\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t} \right)^{-1} \left(\mathbf{I} - e^{\mathbf{A}T} \right) \left(\mathbf{q}_0 - \frac{1}{N_\omega} \sum_{l=0}^{N_\omega-1} \mathbf{R}_l \mathbf{B} \hat{\mathbf{f}}_l \right). \quad (3.4.14)$$

The first term in (3.4.14) $\mathbf{R}_k \mathbf{B} \hat{\mathbf{f}}_k$, is familiar: it is the component at frequency ω_k of the steady-state response to a periodic forcing. The second term in (3.4.14) represents the transient. This transient may persist for the entirety of the interval $[0, T]$, so including it is crucial.

Finally, we left-multiply (3.4.14) by the SPOD modes according to (3.4.5b) to obtain the following expression for the SPOD coefficients,

$$\mathbf{a}_k = \Psi_k^* \mathbf{W} \mathbf{R}_k \mathbf{B} \hat{\mathbf{f}}_k + \Psi_k^* \mathbf{W} \left(\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t} \right)^{-1} \left(\mathbf{I} - e^{\mathbf{A}T} \right) \left(\mathbf{q}_0 - \frac{1}{N_\omega} \sum_{l=0}^{N_\omega-1} \mathbf{R}_l \mathbf{B} \hat{\mathbf{f}}_l \right). \quad (3.4.15)$$

3.4.3 Relation to previous SPOD-based methods

Refs. [74, 130] took an expression equivalent to the first term in (3.4.14) as their starting point for SPOD-based model reduction. Specifically, they began with the equation

$$\mathbf{L}_k \hat{\mathbf{q}}_k = \mathbf{B} \hat{\mathbf{f}}_k, \quad (3.4.16)$$

where $\mathbf{L}_k = (i\omega_k \mathbf{I} - \mathbf{A})$ is the inverse of the resolvent operator. We stress that unless the solution $\mathbf{q}(t)$ is T -periodic, this relation is incorrect, and this issue was not fully appreciated in either Ref. [74] or Ref. [130]. We note that [78] has discussed this issue in another context.

Ref. [74] proceeded to reduce 3.4.16 by applying a Galerkin projection with Ψ_k as the trial basis, resulting in the equation $\mathbf{a}_k = (\Psi_k^* \mathbf{W} \mathbf{L}_k \Psi_k)^{-1} \mathbf{B} \hat{\mathbf{f}}_k$. Even in T -periodic systems, this reduction does not recover the exact SPOD coefficients — the unretained modes influence the retained coefficients. An innovation made by Ref. [130] was to use a Petrov-Galerkin projection of (3.4.16), resulting in the equation $\mathbf{a}_k = (\Phi_k^{F*} \mathbf{W} \mathbf{L}_k \Psi_k)^{-1} \Phi_k^{F*} \mathbf{W} \mathbf{B} \hat{\mathbf{f}}_k$. Ref. [130] showed that the test basis Φ_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 Φ_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. [130] may be alternatively derived by left-multiplying the full-order equation $\hat{\mathbf{q}}_k = \mathbf{R}_k \mathbf{B} \hat{\mathbf{f}}_k$ by $\Psi_k^* \mathbf{W}$, resulting in the following relation for the SPOD coefficients $\mathbf{a}_k = \Psi_k^* \mathbf{W} \mathbf{R}_k \mathbf{B} \hat{\mathbf{f}}_k$. Since our goal is to arrive at a ROM

without the constraint of periodicity, we instead apply the left-multiplication to (3.4.14).

While the frequency domain equation (3.4.14) 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 eigen-system methods are superior.

3.4.4 Operator approximations

If the system is small, all operators in (3.4.15) 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, and we describe an approximation method for both below. We return to the question of approximating these operators in Chapter 5.

3.4.4.1 Steady-state operator

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

$$\mathbf{g}_k^i = \mathbf{L}_k \hat{\mathbf{q}}_k^i, \quad (3.4.17)$$

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

$$\hat{\mathbf{q}}_k^i = \mathbf{R}_k \mathbf{g}_k^i. \quad (3.4.18)$$

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

$$\mathbf{Q}_k = \mathbf{R}_k \mathbf{G}_k, \quad (3.4.19)$$

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

$$\mathbf{R}_k \approx \mathbf{Q}_k \mathbf{G}_k^+, \quad (3.4.20)$$

where the pseudoinverse is defined $\mathbf{G}_k^+ = (\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} \mathbf{G}_k^* \mathbf{W}$. 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, $\mathbf{Q}_k \mathbf{G}_k^+ = \mathbf{R}_k \mathbf{P}_g$,

where $\mathbf{P}_g$ is an orthogonal projection matrix (in the $\mathbf{W}$ -based norm). This means that $\mathbf{Q}_k \mathbf{G}_k^+$ 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 $\mathbf{M}_k = \Psi_k^* \mathbf{W} \mathbf{R}_k \mathbf{B}$, we define the matrix $\mathbf{E}_k$ as

$$\mathbf{M}_k \approx \mathbf{E}_k = \Psi_k^* \mathbf{W} \mathbf{Q}_k \mathbf{G}_k^+ \mathbf{B}. \quad (3.4.21)$$

The total cost of these operations scales linearly with N_x and quadratically with N_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 [28], may be used.

3.4.4.2 Transient operator

Next, we approximate the operator $\Psi_k^* \mathbf{W} (\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t})^{-1} (\mathbf{I} - e^{\mathbf{A}T})$, which involves cost that scales as $\mathcal{O}(N_x^3)$ to compute directly. This is accomplished using the N_d available SPOD modes at each frequency to form a Galerkin approximation of $\mathbf{A}$ that is then used in the matrix exponentials and inverses. Specifically, the approximation is

$$\Psi_k^* \mathbf{W} (\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t})^{-1} (\mathbf{I} - e^{\mathbf{A}T}) \approx \mathbf{P}_k (\mathbf{I} - e^{(\tilde{\mathbf{A}}_k - i\omega_k \mathbf{I})\Delta t})^{-1} (\mathbf{I} - e^{\tilde{\mathbf{A}}_k T}) \Psi_k^{N_d*} \mathbf{W}, \quad (3.4.22)$$

where $\tilde{\mathbf{A}}_k = \Psi_k^{N_d*} \mathbf{W} \mathbf{A} \Psi_k^{N_d}$ is the Galerkin approximation of $\mathbf{A}$ at the k -th frequency, and the matrix $\mathbf{P}_k = [\mathbf{I}_{r_k} \quad \mathbf{0}] \in \mathbb{R}^{r_k \times N_d}$ selects the first r_k rows of the matrix it multiplies. This approximation converges to the true operator in the limit that $N_d = N_x$. As desired, all matrix exponentials and inverses are of size $N_d \times N_d$, which makes them tractable.

The forcing sum in (3.4.15), 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 use the same approximation of the resolvent operator as before, giving

$$\mathbf{R}_l \mathbf{B} \hat{\mathbf{f}}_l \approx \mathbf{Q}_l \mathbf{G}_l^+ \mathbf{B} \hat{\mathbf{f}}_l. \quad (3.4.23)$$

The equations can, at this point, be written as

$$\mathbf{a}_k = \mathbf{E}_k \hat{\mathbf{f}}_k + \mathbf{F}_k \left(\mathbf{q}_0 - \frac{1}{N_\omega} \sum_l \mathbf{Q}_l \mathbf{G}_l^+ \mathbf{B} \hat{\mathbf{f}}_l \right), \quad (3.4.24)$$

where $\mathbf{F}_k = \left(\mathbf{I} - e^{(\tilde{\mathbf{A}}_k - i\omega_k \mathbf{I})\Delta t}\right)^{-1} \left(\mathbf{I} - e^{\tilde{\mathbf{A}}_k T}\right) \Psi_k^* \mathbf{W} \in \mathbb{C}^{r_k \times N_x}$. 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 (3.4.24) is multiplied on the left by $\mathbf{F}_k$ for every frequency ω_k , leading to $N_x N_\omega r$ scaling. This can be avoided by storing the term in parentheses in (3.4.24) in a rank- p_1 reduced basis $\Phi \in \mathbb{C}^{N_x \times p_1}$ 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

$$\mathbf{a}_k = \mathbf{E}_k \hat{\mathbf{f}}_k + \mathbf{H}_k \left(\Phi^* \mathbf{W} \mathbf{q}_0 - \frac{1}{N_\omega} \sum_l \mathbf{J}_l \hat{\mathbf{f}}_l \right), \quad (3.4.25)$$

where $\mathbf{H}_k = \mathbf{F}_k \Phi \in \mathbb{C}^{r_k \times p_1}$ and $\mathbf{J}_l = \Phi^* \mathbf{W} \mathbf{Q}_l \mathbf{G}_l^+ \mathbf{B} \in \mathbb{C}^{p_1 \times N_f}$. We find that the approximations described in this section produce accurate results for the problems considered in this chapter and next. In Chapter 5, we revisit the approximations.

3.4.5 Computational complexity

Given an initial condition $\mathbf{q}_0$ and forcing $\mathbf{f}(t)$, the online stage of the method consists of taking the Fourier transform of the forcing, then inserting this and the initial condition into (3.4.25) in order to get the SPOD coefficients. In this section, we discuss the scaling of the linear version of the method. Algorithms are presented in the next chapter in Section 4.1.6.1 for the fully nonlinear version of SSOP, a special case of which is the linear version described in this chapter.

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. The complexity of the online algorithm is

$$\mathcal{O}((rp_1 + rN_f + N_f \log N_\omega)N_\omega). \quad (3.4.26)$$

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_1 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).

Obtaining the SPOD modes, e.g., using the techniques described in Chapter 2 or in Refs. [131, 121], scales as $\mathcal{O}(N_d^2 N_x N_\omega)$. If the evaluation of $\mathbf{A}\mathbf{q}$ scales linearly with N_x (i.e., $\mathbf{A}$ is sparse), then the offline scaling is $\mathcal{O}(N_d^2 N_x N_\omega)$. If the evaluation of $\mathbf{A}\mathbf{q}$ scales quadratically with N_x , then the offline scaling is $\mathcal{O}(N_d^2 N_x N_\omega + N_d N_x^2 N_\omega)$.

3.4.6 Data-free method

Here, we briefly outline a data-free version of the method. Ref. [131] 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

$$\mathbf{X}_q \mathbf{R}_k \mathbf{B} \mathbf{X}_f^{-1} = \tilde{\mathbf{U}}_k \Sigma_k \tilde{\mathbf{V}}_k^*. \quad (3.4.27)$$

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 $\|\mathbf{f}\|^2 = \mathbf{f}^* \mathbf{W}_f \mathbf{f}$. 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,

$$\Psi_k = \mathbf{U}_k, \quad (3.4.28)$$

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

$$\Lambda_k = \alpha^2 \Sigma_k^2. \quad (3.4.29)$$

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 [82, 7, 122, 89, 101]. 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 (3.4.15) is cubic in N_x . However, data-free methods [28] 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 3.5.1, 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 [137, 110].

3.5 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 .

3.5.1 Linearized Ginzburg-Landau problem

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

$$\dot{q}(x, t) = \mathcal{A}q(x, t) + f(x, t), \quad (3.5.1)$$

where

$$\mathcal{A} = -\nu \frac{\partial}{\partial x} + \gamma \frac{\partial^2}{\partial x^2} + \mu(x), \quad (3.5.2)$$

and $f(x, t)$ is a forcing. Following Ref. [5], we set $\nu = 2 + 0.4i$ and $\gamma = 1 - i$. The parameter $\mu(x)$ takes the form

$$\mu(x) = (\mu_0 - c_\mu^2) + \frac{\mu_2}{2}x^2, \quad (3.5.3)$$

with $c_\mu = 0.2$, $\mu_2 = -0.01$ [5]. 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 $\mu_0 = 0.229$ [131] 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 (3.4.1). Following Refs. [5, 22], we use a pseudo-spectral Hermite discretization with $N_x = 220$ collocation points [131], and solve the discretized equations using MATLAB's `ode45`, an explicit Runge-Kutta (4,5) integrator. Following Ref. [131], 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 $N_d = 142$ overlapping blocks, each of length $N_\omega = 1024$ time steps.

Figure 3.3(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 Figure 3.3(a) the state is forced with band-limited temporally white noise and a Gaussian spatial correlation while the state in Figure 3.3(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 Figure 3.3(c,d). Note that the forcing occupies the entire domain in this example, i.e., $N_f = N_x$. We thus choose $p_1 = N_x$ 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.

Figure 3.4 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

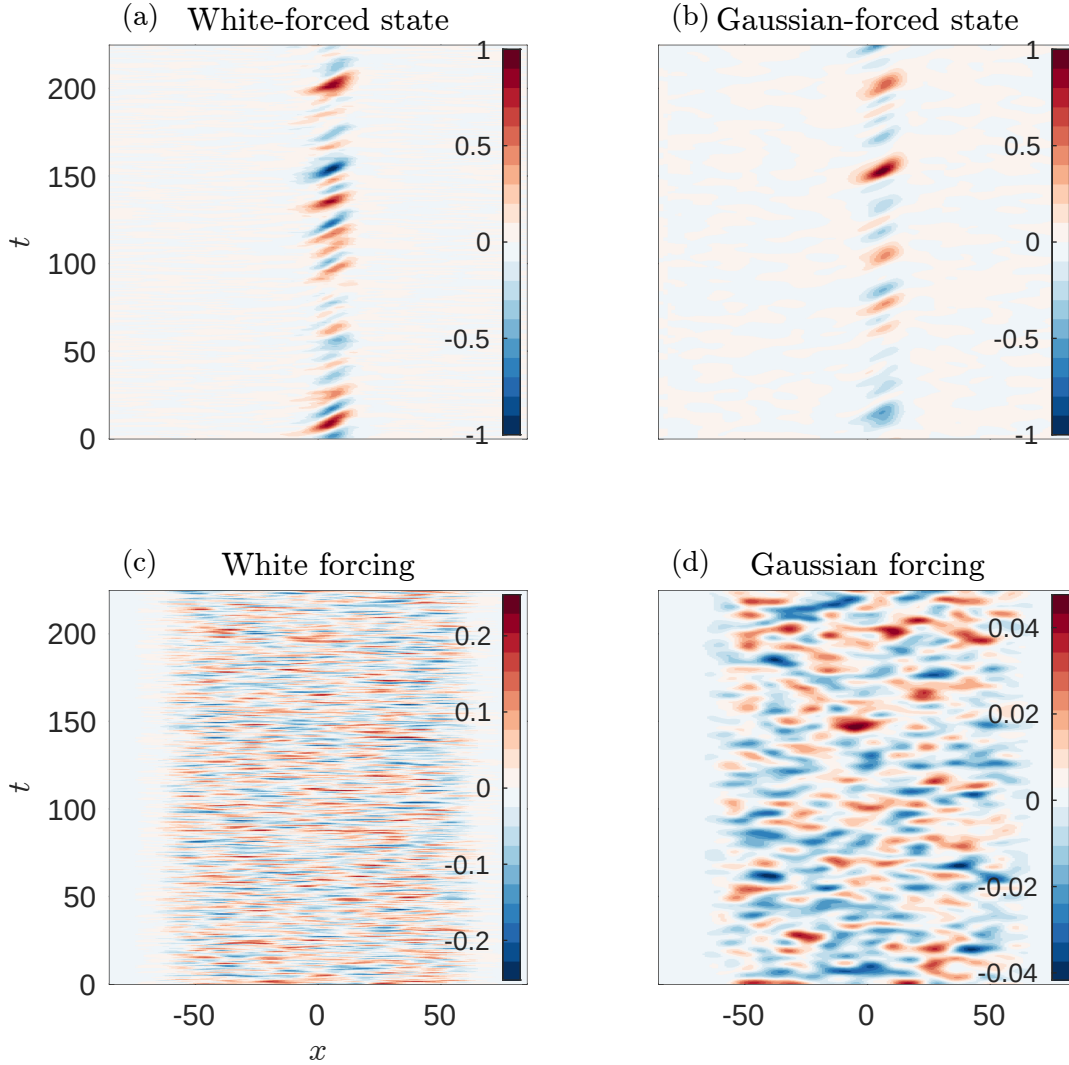


Figure 3.3: Ginzburg-Landau state and forcing trajectories: (a) the state resulting from the white forcing in (c); (b) the state resulting from the Gaussian forcing in (d). Both forcings have the same spatial correlation, but the short temporal correlation in the white forcing leads to more jagged structures in the corresponding state. Both trajectories consist of waves traveling in the positive x -direction that are amplified in a region near $x = 0$.

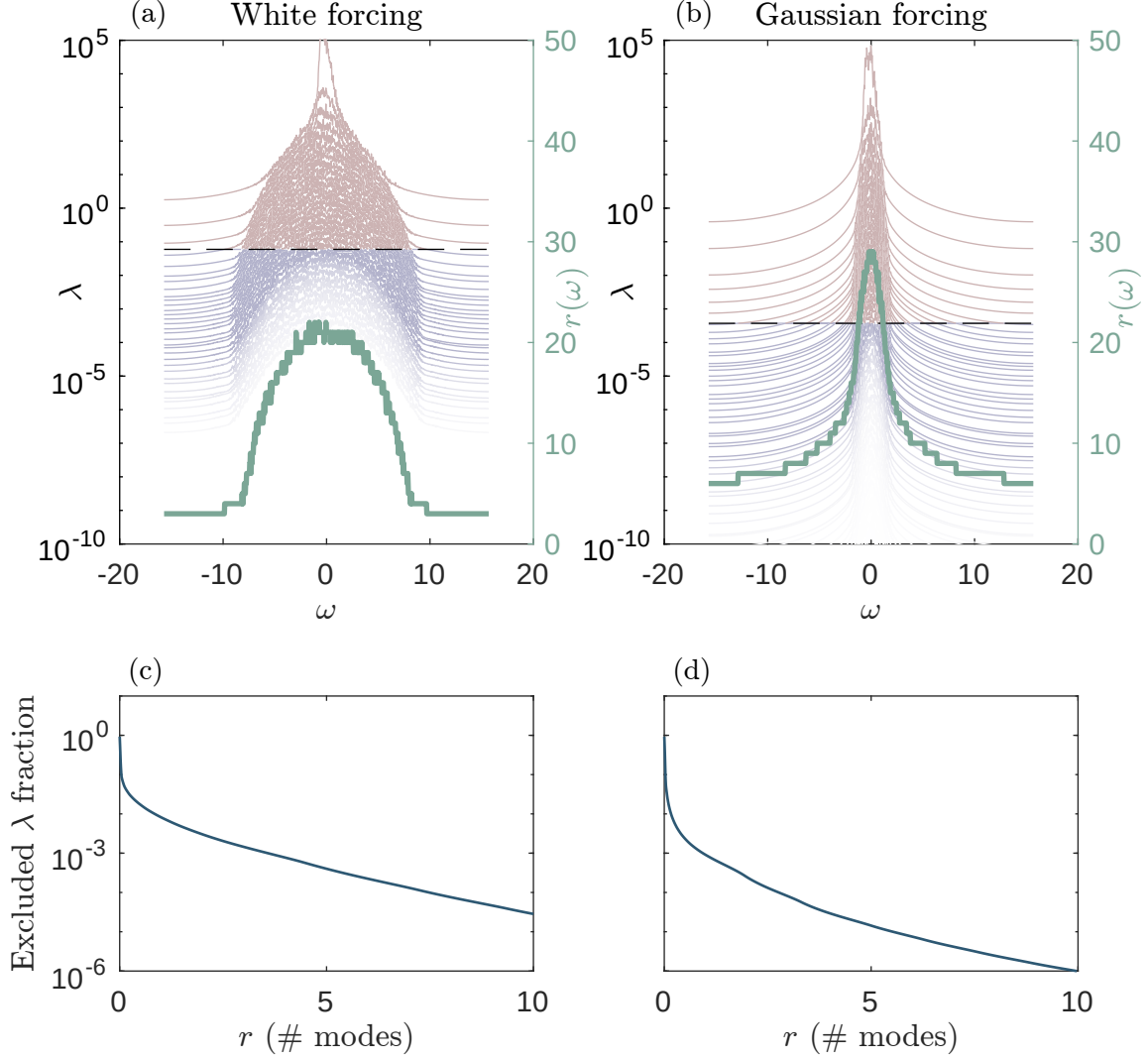


Figure 3.4: SPOD mode energies: (a-b, left axis) energy λ of the retained and unretained modes. The top red curve is the energy of the first SPOD mode as a function of frequency ω . The lower red and blue curves are the energies of the lower mode numbers, as functions of frequency. The retained modes (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 (3.5.4)) as a function of r , the average number of modes per frequency.

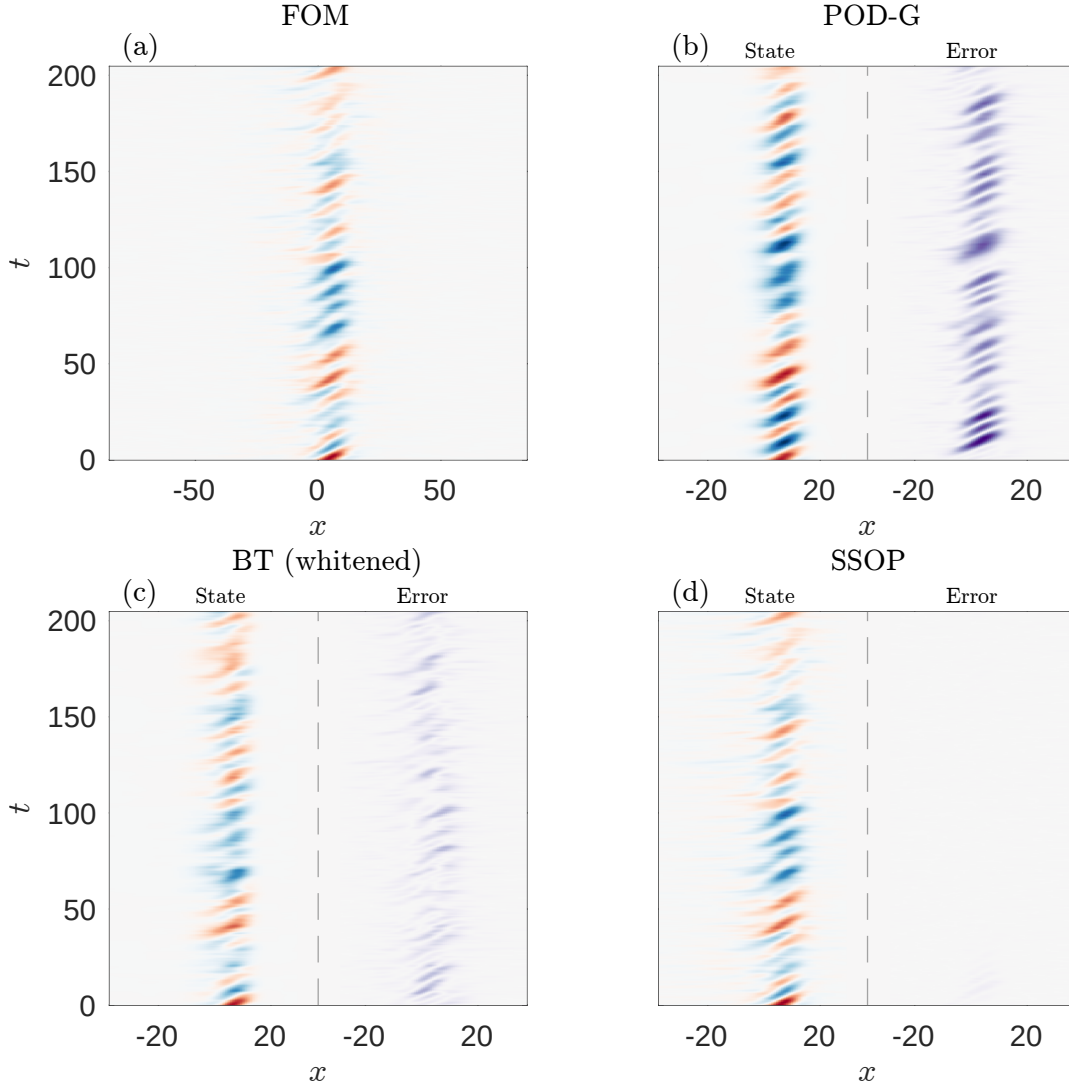


Figure 3.5: FOM trajectory (a) and ROM predictions thereof (b-d) along with the errors for the Ginzburg-Landau system. The error fields shown are the absolute value of the difference of the FOM and ROM trajectories. The peak error value (and the upper limit on the error color scale) is 87% of the peak absolute value of the state.

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

$$\frac{\sum_{k=0}^{N_\omega-1} \sum_{j=r_k+1}^{N_x} \lambda_{kj}}{\sum_{k=0}^{N_\omega-1} \sum_{j=1}^{N_x} \lambda_{kj}}. \quad (3.5.4)$$

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. [112] 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. [86, 70]. 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 Xeon 6128 processor. The cost of building the ROM (with $r = 10$) was 0.7% of the cost of generating the FOM data.

Figure 3.5 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.

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.,

$$e(t) = \frac{\sum_{i=1}^{N_{\text{test}}} \|\tilde{\mathbf{q}}^i(t) - \mathbf{q}^i(t)\|_x^2}{\frac{1}{T} \sum_{i=1}^{N_{\text{test}}} \int_0^T \|\mathbf{q}^i(t')\|_x^2 dt'}. \quad (3.5.5)$$

We also report the mean of this quantity over time.

Figure 3.6 shows the error, defined in (3.5.5), 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 Figure 3.4. 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.

The higher error at the ends of the interval in Figure 3.6 is due to the fact that the SPOD encoding/decoding process acts as a discrete convolution of the true solution with a time-periodic kernel, as shown in (3.2.13). At a time t away from the ends of the interval, data from nearby times is used to reconstruct the solution at t . At the one end of the interval, however, the kernel wraps around and uses data from the other end. Since this data is unrelated to the solution at the first end, the inclusion of it worsens the reconstruction.

Next, we investigate the dependence of the error on the number of modes retained. Figure 3.7 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

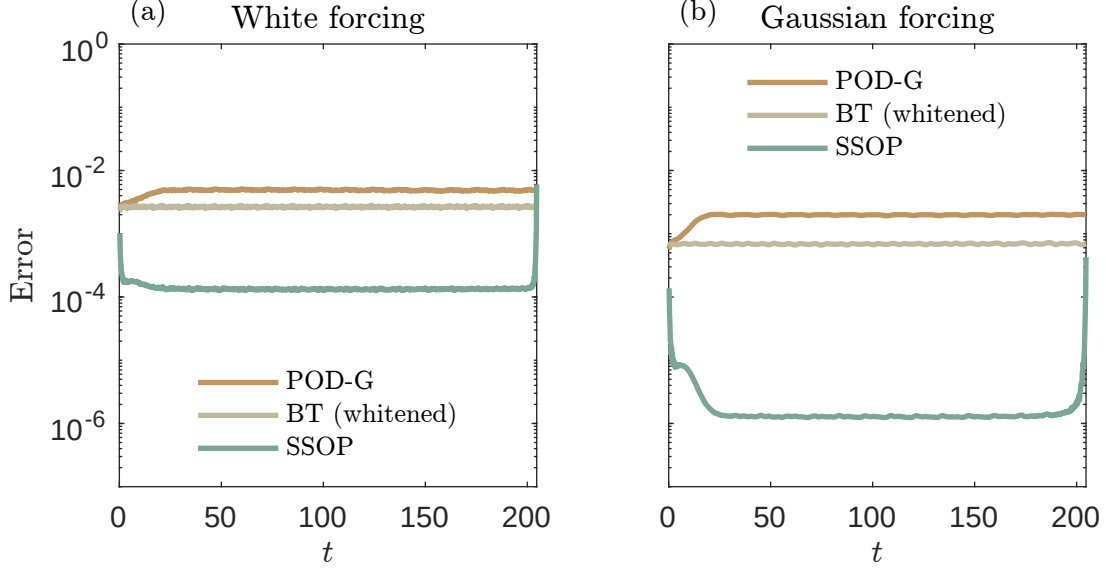


Figure 3.6: Error with $r = 10$ modes relative to the FOM for the Ginzburg-Landau system, averaged over 173 trajectories. The large difference in accuracy is due, in large part, to the ability of the SPOD modes to represent trajectories more accurately than space-only modes. This difference is larger in the Gaussian forcing case.

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, 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 [16], 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 Figure 3.8, 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,

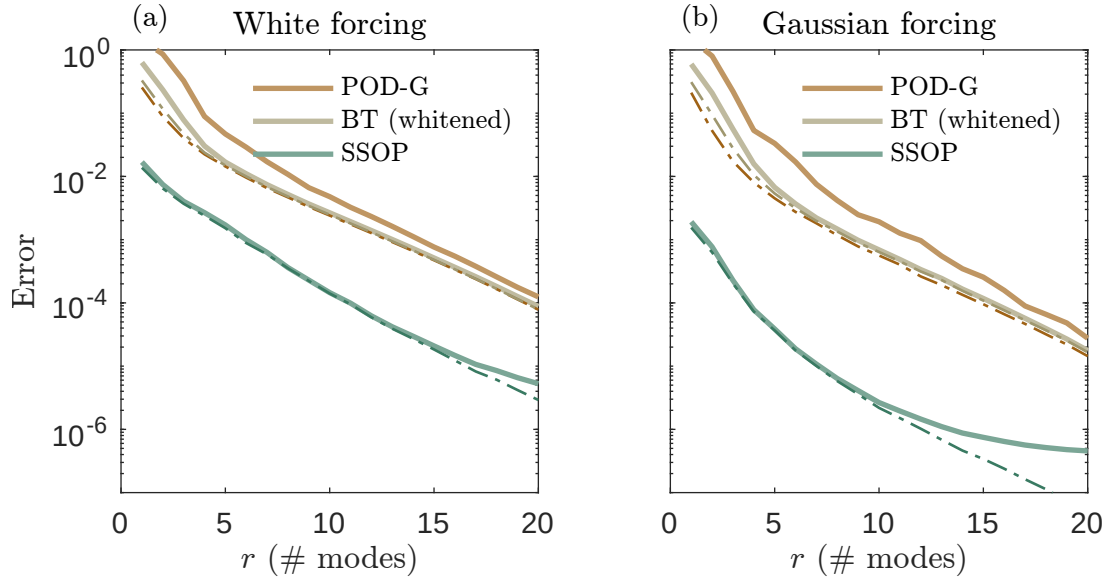


Figure 3.7: Error as a function of the number of modes for the Ginzburg-Landau system. The values reported here are the time averages of the error defined in (3.5.5).

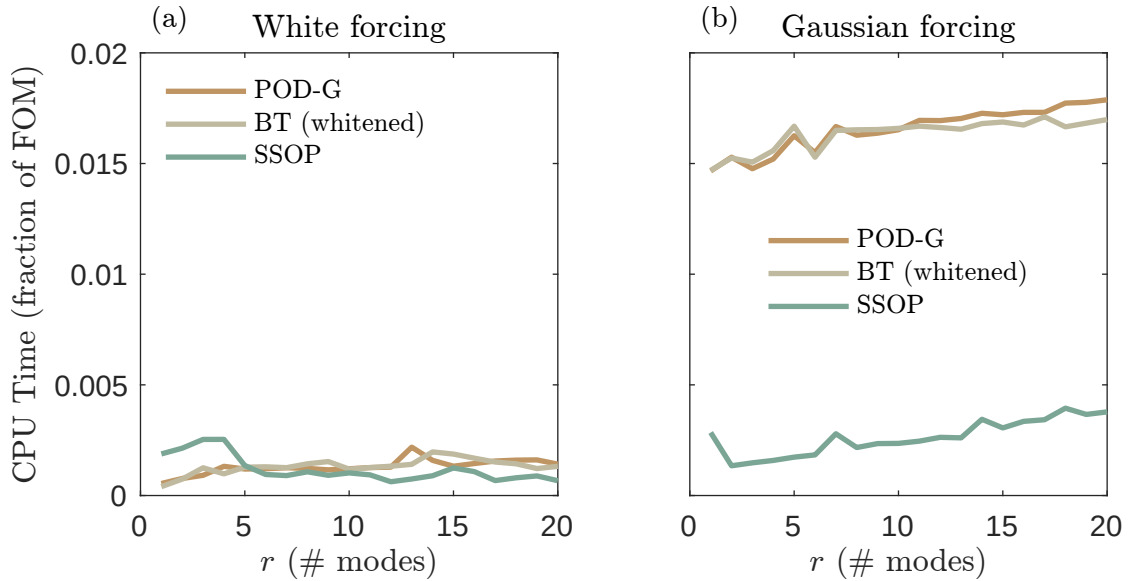


Figure 3.8: Average CPU time as a function of the number of modes for the Ginzburg-Landau system. The values are normalized by the CPU time of the FOM.

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 [5]. 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 [132, 116], 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 Figure 3.9(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 .

Figure 3.9(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

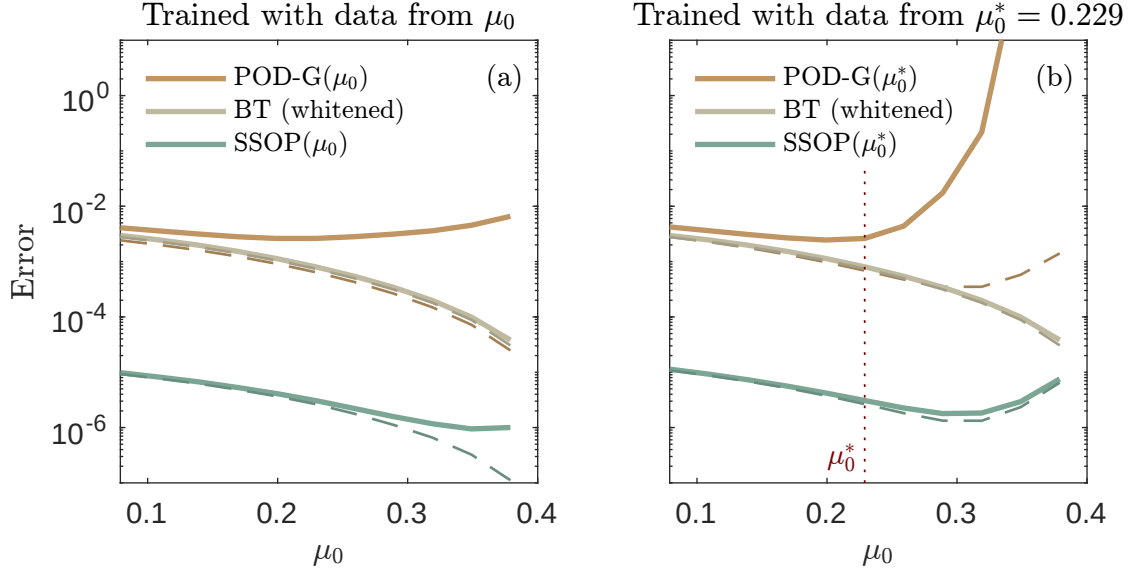


Figure 3.9: Data-driven ROM performance across a range of Ginzburg-Landau systems parameterized by μ_0 : (a) Models are trained and tested on the same system for a range of μ_0 ; (b) Models are trained on a system with $\mu_0 = 0.229$ and tested on a range of μ_0 . All models use $r = 10$. The enhanced 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 $\mu_0 = 0.397$.

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 [5], the SSOP method is remarkably accurate. Indeed, somewhat surprisingly, the method is more accurate at the high μ_0 values, likely because non-modal mechanisms cause the system to be lower ran here. The SSOP method trained on the ‘wrong’ data ((b), solid green) is, in fact, 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 $\mu_0 = 0.229$ 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 $T(t)$.

Figure 3.10 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.

Figure 3.11 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 from that in the training data for the model. Nevertheless, the method gives relatively accurate predictions for this out-of-sample forcing.

Figure 3.12 compares the accuracy of the resolvent SSOP model, described in Section 3.4.6, 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 3.9. 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 Figure 3.12.

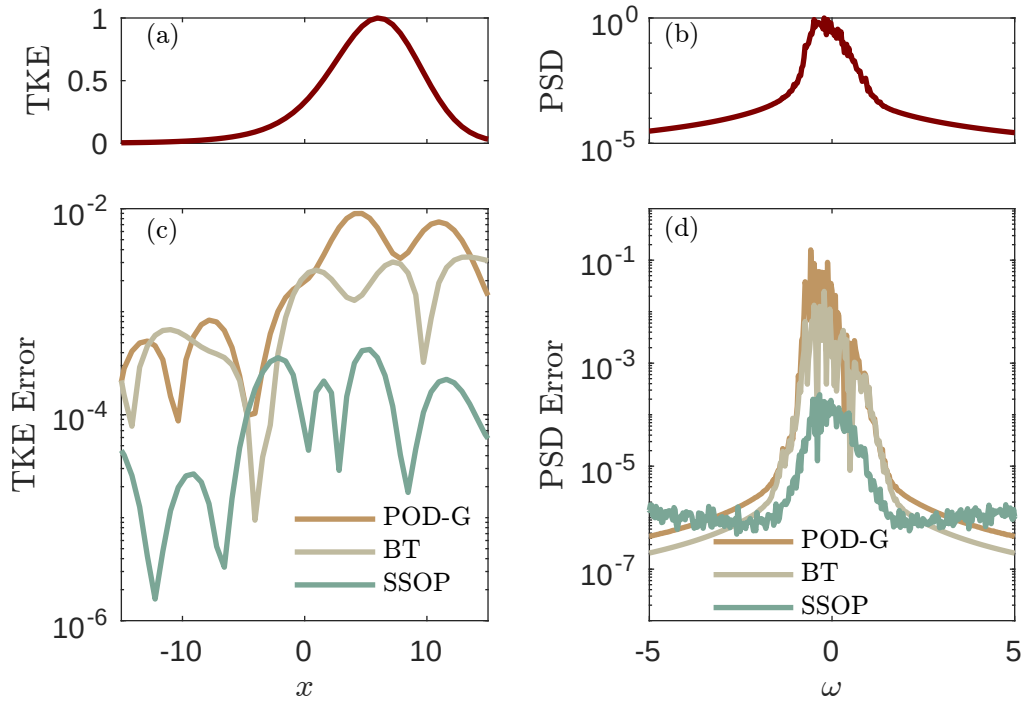


Figure 3.10: Error in the prediction of statistics for the Ginzburg-Landau system: (a) the FOM variance (TKE) as a function of x ; (b) the power spectral density (PSD); (c) the ROM errors in predictions of the TKE; (d) the ROM errors in predictions of the PSD. The ROMs are built using data from the white-noise forcing.

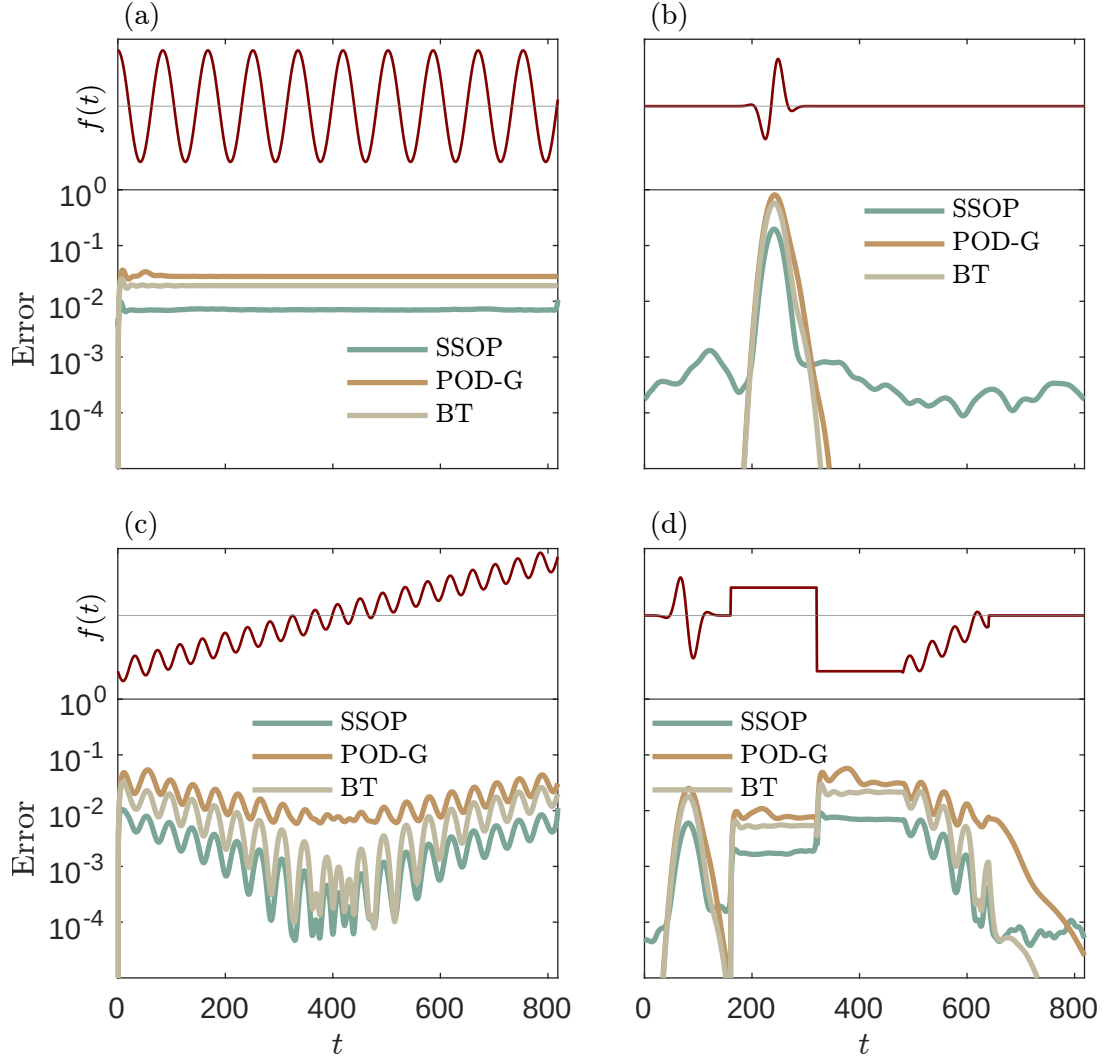


Figure 3.11: Results of the models trained with the white forcing on four out-of-sample non-stationary forcings to the Ginzburg-Landau system. The time-dependence of these forcings is shown in the top panel, and the resulting error for the three ROMs in the bottom panel. Respectively, the mean errors 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.

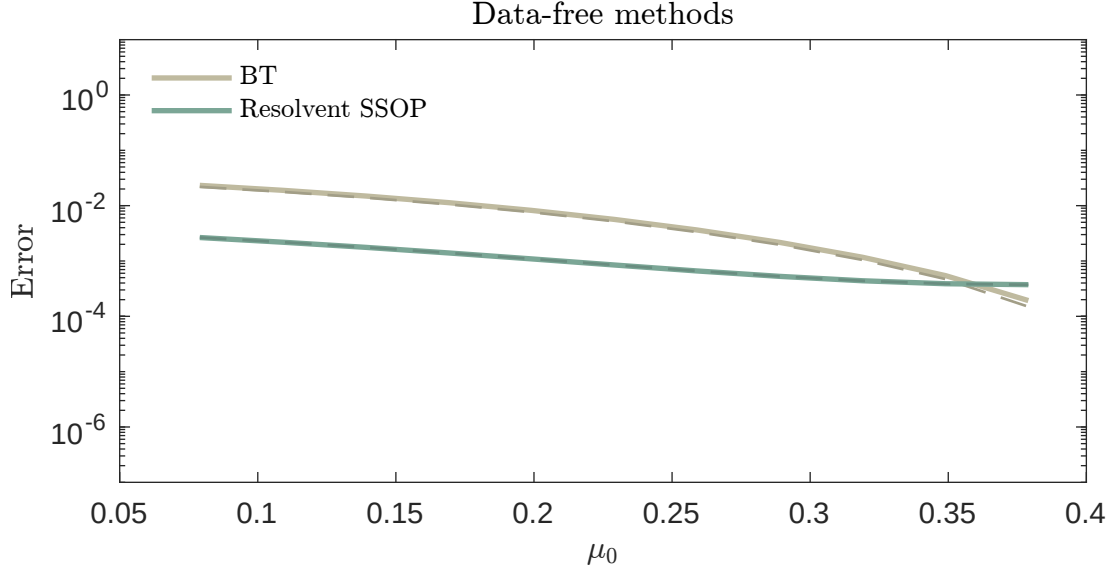


Figure 3.12: The accuracy of the resolvent SSOP model and unwhitened balanced truncation for Ginzburg-Landau systems with a range of μ_0 . Both methods are data-free; the resolvent modes come directly from the system matrices in the governing equation, and ‘unwhitened’ indicates that we did not transform 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.

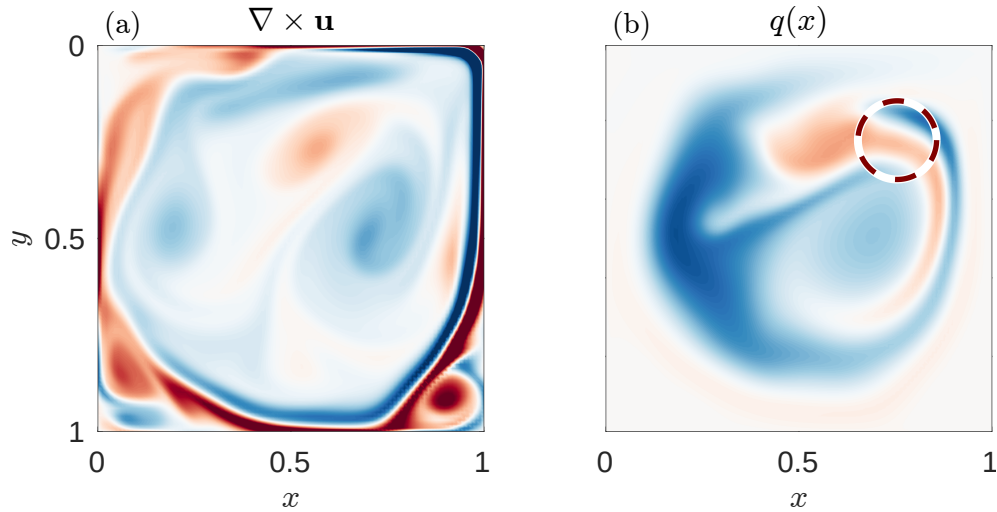


Figure 3.13: Visualization of the scalar transport problem: (a) the vorticity of the steady velocity field that transports the scalar; (b) a snapshot of the scalar field with the forcing region highlighted.

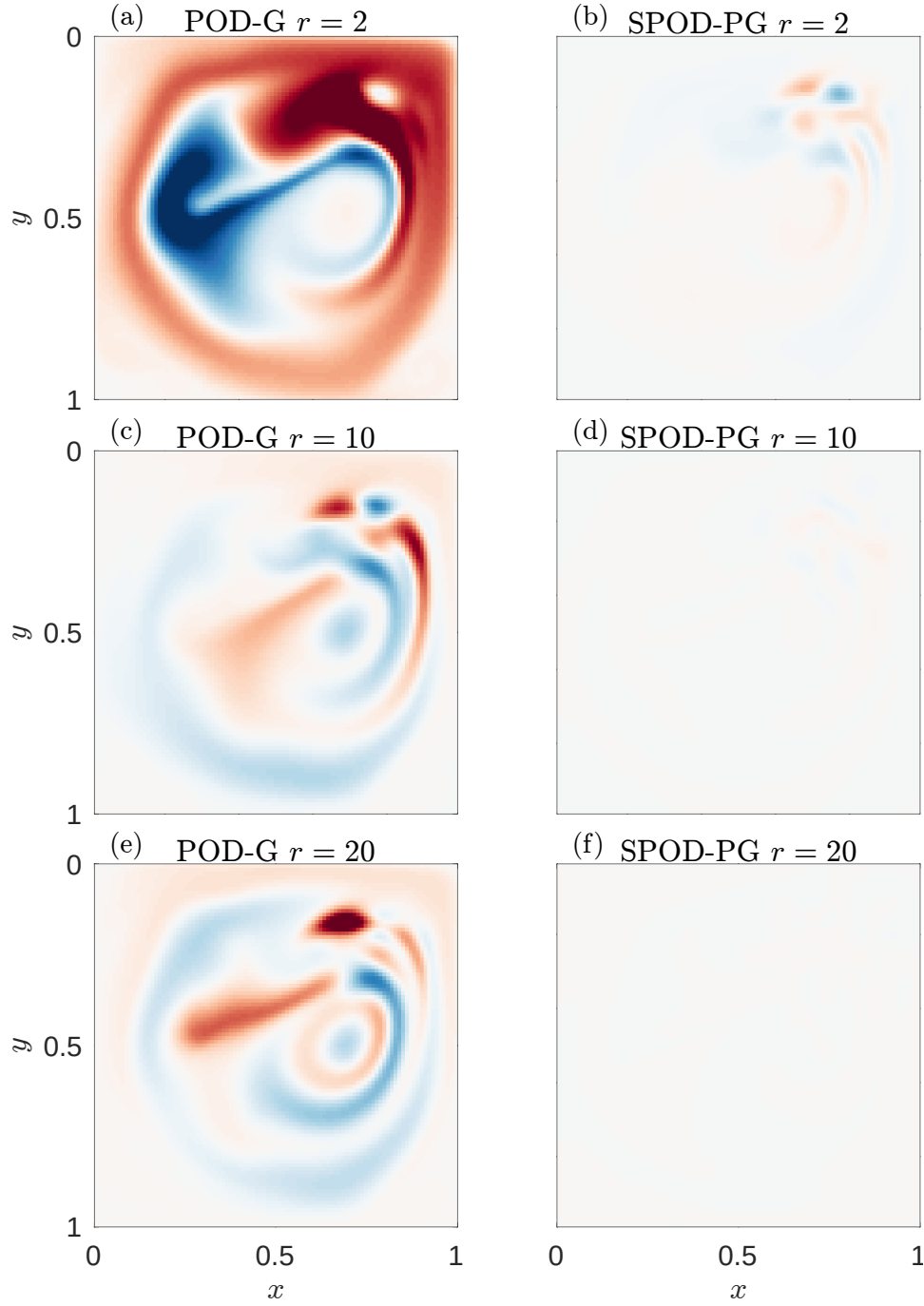


Figure 3.14: Error snapshots for the scalar transport problem: (a, c, e) POD-Galerkin ROM error for 2, 10, and 20 modes; (b, d, f) SSOP ROM error for the same numbers of modes. The extremes of the color scale are 31% of the maximum absolute value of the FOM solution, and the maximum in the 2-mode POD-Galerkin error field is 62% of the maximum absolute value of the FOM solution.

3.5.2 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 [137, 110]. A balanced truncation model must have greater error than the POD representation error, which 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 3.4.4.2.

The continuous governing equations for the scalar transport case may be written in the same form as (3.5.1), where $\mathcal{A}$ is now defined as

$$\mathcal{A} = -\mathbf{u}(\mathbf{x}) \cdot \nabla + \eta \nabla^2, \quad (3.5.6)$$

and where $\mathbf{u}(\mathbf{x})$ 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 $\bar{\mathbf{x}} = [0.75, 0.25]^T$; its statistics are given by

$$C_{ff}(\mathbf{x}_1 \mathbf{x}_2, t_1, t_2) = \exp \left[- \left(\frac{|\mathbf{x}_1 - \bar{\mathbf{x}}|^2 + |\mathbf{x}_2 - \bar{\mathbf{x}}|^2}{l^2} + \frac{|\mathbf{x}_2 - \mathbf{x}_1|^2}{\xi^2} + \frac{(t_2 - t_1)^2}{\tau^2} \right) \right], \quad (3.5.7)$$

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 Figure 3.13. 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 (3.5.7) 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 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 $p_1 = 50$ for the intermediary basis.

Figure 3.14 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.

Figure 3.15 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. Figure 3.16 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.

3.6 Chapter summary

This chapter develops the spectral solution operator projection (SSOP) method for linear problems. In terms of the two steps that define a space-time ROM, the method is as follows:

- (i) Trajectories are encoded using SPOD modes. The SPOD basis at a given frequency is tailored to represent the spatial structures at that frequency, and this is used to encode trajectories by representing each frequency in the corresponding basis of SPOD modes, then taking an inverse DFT. These modes provide an ideal encoding for space-time

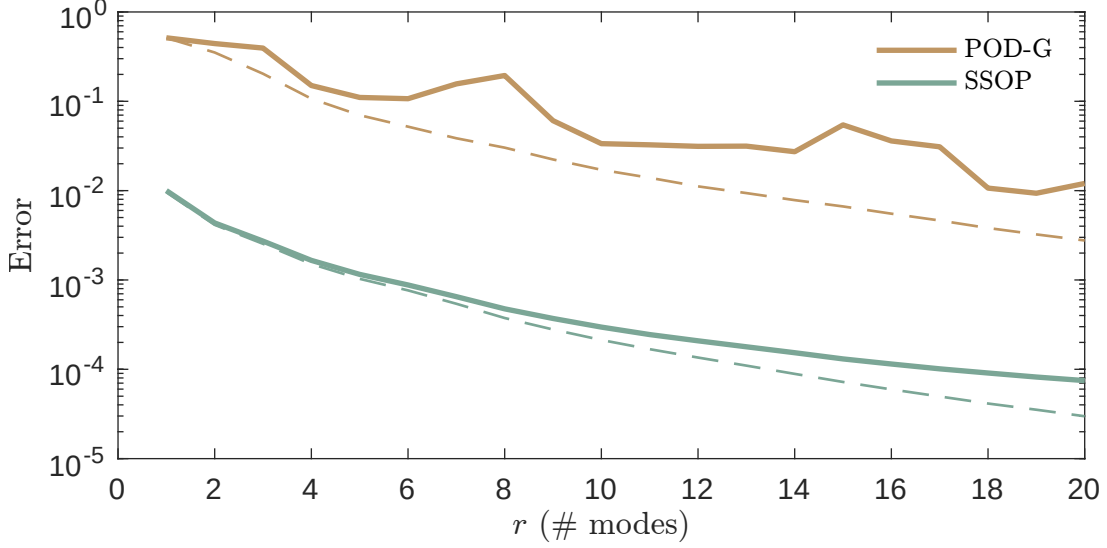


Figure 3.15: Accuracy of the proposed method compared to that of a POD-Galerkin model applied to the scalar transport problem. The dashed lines are the error of the full-order solution projected on the respective bases and are lower bounds for the error of the methods.

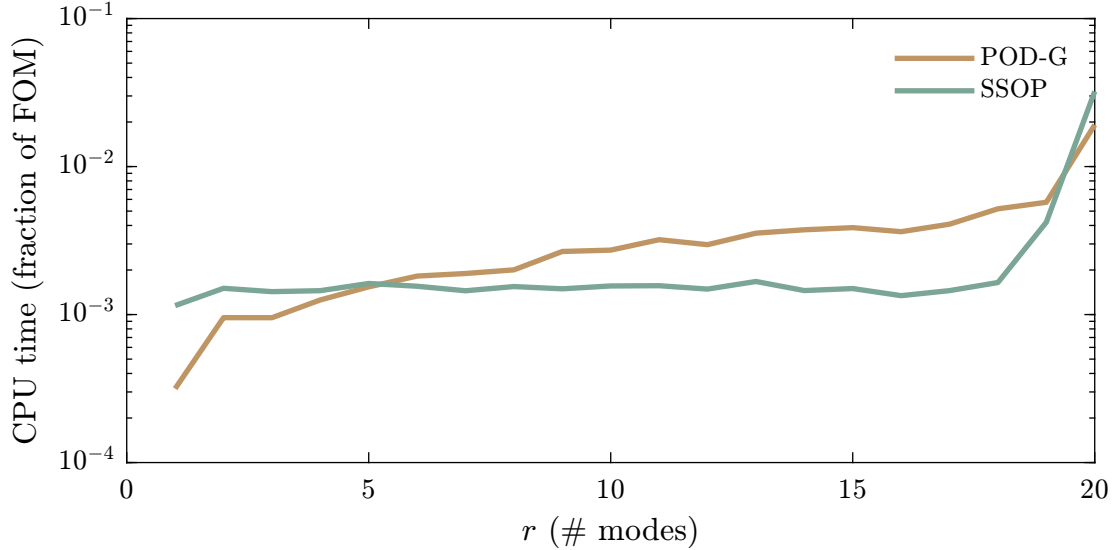


Figure 3.16: Average CPU time to solve the scalar transport problem as a function of the number of space-time modes used as a fraction of the FOM CPU time.

model reduction; as we show theoretically in Chapter 2 and demonstrate numerically in Section 3.2, this encoding is nearly optimal in that the SPOD modes converge to space-time POD modes for large T for statistically stationary data. They also provide an accurate encoding for non-stationary data, as is shown in our numerical results. Moreover, the analytic time-dependence of the SPOD modes allows for the derivation

of the transfer functions that are used in the method.

- (ii) The encoding is obtained online by applying a space-time projection of the analytical solution to the governing equations onto the SPOD modes. This results in transfer functions that map the initial condition and forcing inputs to the SPOD coefficients that result. So long as these transfer functions are approximated accurately offline, there is minimal error in obtaining the encoding online. In addition, the approach overcomes the requirement that the solution be periodic in time, which led to large errors in previous methods based on SPOD modes.

The method is tested on a linearized Ginzburg-Landau system and a scalar transport problem. In both cases, the method produces error that is roughly two orders of magnitude lower than that of POD-Galerkin and balanced truncation at similar computational cost.

CHAPTER 4

Generalizing SSOP to Nonlinear Systems

The goal of this chapter is to generalize SSOP to nonlinear systems. Three new aspects must be addressed. First, the operator must be adjusted to account for the nonlinearity. To do this, we treat the nonlinear term as an additional forcing on the system and use the fundamental solution to the linear system with this additional forcing. This is consistent with an increasingly popular perspective in the fluid dynamics community on the role of nonlinearity [82, 81]. Crucially, this additional forcing depends on the solution itself, and this results in a system of nonlinear equations that must be solved for the SPOD coefficients. Second, the influence of nonlinearity at each frequency must be approximated as a function of the retained SPOD coefficients. We use two approaches, depending on the nature of the nonlinearity. The first is a DEIM-based hyper-reduction approximation of the nonlinearity, and the second, which is only applicable for quadratic nonlinearities, uses a subset of the triadic interactions between the SPOD modes. Third, the system of nonlinear equations that results from the addition of nonlinearity must be efficiently solved. We use a simple fixed-point iteration and find that for most cases, this converges in ~ 10 iterations. For cases where this does not converge, we use a pseudo-time-stepping method, which is somewhat slower but more stable.

We test the method on two Ginzburg-Landau systems, one with the standard cubic nonlinearity and the other with a Burgers'-type (quadratic) nonlinearity used to test the triadic-interaction handling of the nonlinearity. We demonstrate good results for both: at similar cost to POD-G, SSOP results in orders-of-magnitude lower error than POD-G and even the lower bound for space-only methods given by the projection of the solution onto the POD modes. We test the method both on statistically stationary forcings, where the SPOD modes provide the asymptotically optimal encoding, and on non-stochastic forcings, observing similar results for both. We also test the ability of the method to handle affine parameter dependence — we use SPOD modes from one Ginzburg-Landau system to build the ROM, then test it on a parameter sweep of Ginzburg-Landau systems.

4.1 Nonlinear spectral solution operator projection

We now consider systems of the form

$$\dot{\mathbf{q}} = \mathbf{A}\mathbf{q} + \mathbf{B}\mathbf{f} + \mathbf{n}(\mathbf{q}), \quad (4.1.1)$$

where $\mathbf{n} : \mathbb{C}^{N_x} \rightarrow \mathbb{C}^{N_x}$ is some nonlinear function of the state. The parameter dependence in the linear operator $\mathbf{A}$ is temporarily suppressed, and we assume the nonlinear operator is not parameter-dependent. We again assume that the system has been transformed such that the solution has zero mean. An additional transformation may be useful if the linear operator has unstable eigenvalues. The transformation is $\mathbf{A} \rightarrow \mathbf{A} - \alpha \mathbf{I}$, $\mathbf{n}(\mathbf{q}) \rightarrow \mathbf{n}(\mathbf{q}) + \alpha \mathbf{q}$, where α is to the right of the spectrum of $\mathbf{A}$, i.e., $\alpha > \max \Re(\lambda(\mathbf{A}))$. Without this transformation, the terms related to the linear operator may dominate the solution.

Our aim is to generalize SSOP to nonlinear systems of the form (4.1.1). This is accomplished by treating the nonlinear term as an additional forcing on the system, but one that depends on the trajectory. The linear ROM (3.4.25) is amended by adding a term corresponding to the nonlinearity at frequency ω_k that results from the SPOD coefficients at all frequencies. As we show in Section 4.1.1, this results in a system that must be solved for all the SPOD coefficients at once. We then discuss two methods for approximating the effect of nonlinearity in Section 4.1.2. Affine parameter dependence is discussed in Section 4.1.3. We present a method for solving the nonlinear system of equations in Section 4.1.4, and discuss scaling in Section 4.1.6.

The method is shown schematically in Figure 4.1, and the details of the components therein will be discussed in the remainder of Section 4.1. We note here that in the bottom-right box in the figure, an inner product is equated to a vector, and the second argument of the inner product is a matrix. In this shorthand notation, the m -th element of the vector is the inner product of the first argument with the m -th column of the matrix.

4.1.1 Formulation

To handle nonlinearity within the framework described above, we begin with the implicit equation

$$\mathbf{q}_j = \mathbf{L}[\mathbf{q}_0, \mathbf{B}\mathbf{f}(t) + \mathbf{n}(\mathbf{q}(t)); t_j], \quad (4.1.2)$$

where $\mathbf{L}$ is the solution operator given in (3.4.4). This formulation treats the nonlinearity as an additional forcing on the linear part of the system, but, crucially, one that depends on

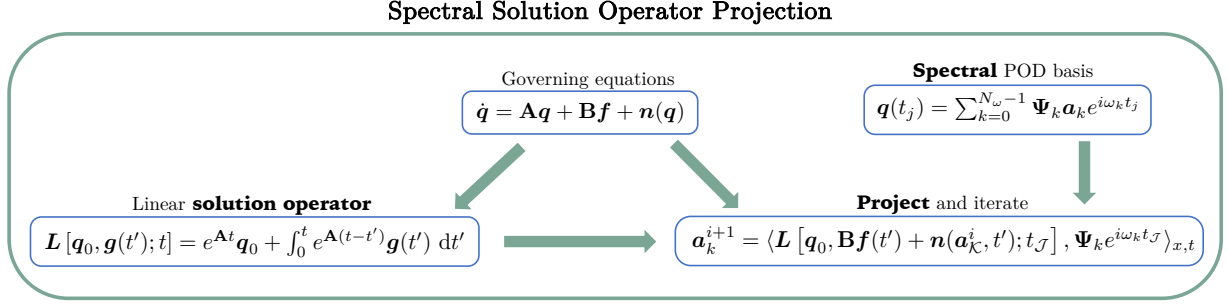


Figure 4.1: Spectral Solution Operator Projection: A system of nonlinear algebraic equations is obtained by performing a space-time projection of a linear solution operator onto the set of SPOD modes used to encode trajectories. The linear solution operator returns the solution to the linear system given the initial condition and forcing. Nonlinearities are handled in this framework as a state-dependent forcing. The system is solved via a fixed-point iteration, which may be viewed as a perturbation series around the linear solution.

the state. The m -th SPOD coefficient at the k -th frequency is now

$$a_{km} = \langle \mathbf{L} [\mathbf{q}_0, \mathbf{B}\mathbf{f}(t) + \mathbf{n}(\mathbf{q}(t)); t_J], \boldsymbol{\psi}_{km} e^{i\omega_k t_J} \rangle_{x,t}. \quad (4.1.3)$$

In order to arrive at a closed system of equations, $\mathbf{n}(\mathbf{q}(t))$ must be approximated using the SPOD coefficients, and we return to this closure in Section 4.1.2. First, as in the linear case, we evaluate the space-time inner product by taking a temporal DFT of the solution operator, followed by a spatial inner product:

$$\hat{\mathbf{q}}_k = \sum_{j=0}^{N_\omega-1} \left(e^{\mathbf{A}j\Delta t} \mathbf{q}_0 + \int_0^{t_j} e^{\mathbf{A}(t_j-t')} (\mathbf{B}\mathbf{f}(t') + \mathbf{n}(\mathbf{q}(t'))) dt' \right) e^{-i\omega_k t_j}. \quad (4.1.4a)$$

$$\mathbf{a}_k = \boldsymbol{\Psi}_k^* \mathbf{W} \hat{\mathbf{q}}_k. \quad (4.1.4b)$$

At this point, the only difference relative to the linear case is the presence of the nonlinear term within the integral, and this term can be handled by following the same approach used for the forcing term in Section 3.4.2. In particular, with the two assumptions from Section 3.4.2, as well as the additional assumption *(iii)* that the nonlinearity can be written in the form $\mathbf{n}(\mathbf{q}(t)) = \sum_{l=0}^{N_\omega-1} \hat{\mathbf{n}}_l e^{i\omega_l t}$ where $\hat{\mathbf{n}}_l \in \mathbb{C}^{N_x}$, the k -th frequency of the solution is given by

$$\begin{aligned} \hat{\mathbf{q}}_k = & \mathbf{R}_k (\mathbf{B}\hat{\mathbf{f}}_k + \hat{\mathbf{n}}_k^q(\hat{\mathbf{q}}_K)) \\ & + \left(\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t} \right)^{-1} \left(\mathbf{I} - e^{\mathbf{A}T} \right) \left(\mathbf{q}_0 - \frac{1}{N_\omega} \sum_{l=0}^{N_\omega-1} \mathbf{R}_l (\mathbf{B}\hat{\mathbf{f}}_l + \hat{\mathbf{n}}_l^q(\mathbf{q}_K)) \right). \end{aligned} \quad (4.1.5)$$

Here, $\hat{\mathbf{n}}_k^q : \mathbb{C}^{N_x N_\omega} \rightarrow \mathbb{C}^{N_x}$ returns the nonlinearity at frequency ω_k that results from $\hat{\mathbf{q}}_{\mathcal{K}}$, the concatenation of all frequencies of the trajectory. The superscript is used to distinguish this function from a similar one introduced in the next subsection. One way to compute $\hat{\mathbf{n}}_k^q$ is to first take the inverse DFT of $\hat{\mathbf{q}}_{\mathcal{K}}$, giving the trajectory in the time domain $\mathbf{q}_{\mathcal{J}}$. The k -th frequency of the nonlinearity $\hat{\mathbf{n}}_k^q$ is given by then computing $\mathbf{n}(\mathbf{q}_{\mathcal{J}})$ and taking the k -th component of the DFT of the result. Similar to the assumption of the form of the forcing, assumption (iii) will likely not hold in practice, but we have found that the error introduced is minimal.

Inserting (4.1.5) into (4.1.4b), we have

$$\begin{aligned} \tilde{\mathbf{a}}_k = & \Psi_k^* \mathbf{W} \mathbf{R}_k \left(\mathbf{B} \hat{\mathbf{f}}_k + \hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}}) \right) \\ & + \Psi_k^* \mathbf{W} \left(\mathbf{I} - e^{(\mathbf{A} - i\omega_k)\Delta t} \right)^{-1} \left(\mathbf{I} - e^{\mathbf{A}T} \right) \left(\mathbf{q}_0 - \frac{1}{N_\omega} \sum_{l=0}^{N_\omega-1} \mathbf{R}_l \left(\mathbf{B} \hat{\mathbf{f}}_l + \hat{\mathbf{n}}_l(\tilde{\mathbf{a}}_{\mathcal{K}}) \right) \right). \end{aligned} \quad (4.1.6)$$

Here, $\tilde{\mathbf{a}}_{\mathcal{K}}$ is the (ROM approximation of the) vector of retained SPOD coefficients at all frequencies concatenated together. With the reduction, the nonlinearity at each frequency must be computed as a function of the vector $\tilde{\mathbf{a}}_{\mathcal{K}}$, not $\hat{\mathbf{q}}$, i.e., $\hat{\mathbf{n}}_k : \mathbb{C}^{rN_\omega} \rightarrow \mathbb{C}^{N_x}$. Two different methods for approximating the nonlinearity at each frequency, given the SPOD coefficients, are described in the following subsection.

The operators in (4.1.6) must be approximated if the system is large. With the same approximations used in the linear case, we have,

$$\tilde{\mathbf{a}}_k = \mathbf{E}_k \hat{\mathbf{f}}_k + \Psi_k^* \mathbf{W} \mathbf{R}_k \hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}}) + \mathbf{H}_k \left(\Phi^* \mathbf{q}_0 - \sum_{l=0}^{N_\omega-1} \mathbf{J}_l \hat{\mathbf{f}}_l + \Phi^* \mathbf{W} \mathbf{R}_k \hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}}) \right). \quad (4.1.7)$$

In this chapter, like in Chapter 3, we assume that the system as written in (4.1.1) has already been transformed such that $\mathbf{q}$ has zero mean. This is especially useful in the nonlinear case because it maximizes the contribution of the linear term, which is handled analytically, to the right-hand side.

4.1.2 Constructing the nonlinearity

To close the reduced-order model equations (4.1.7), we must compute the k -th frequency of the nonlinearity $\hat{\mathbf{n}}_k$ that results from the SPOD coefficients. With no approximation, the nonlinearity is given in terms of the k -th frequency of the nonlinear function acting on the

decoding of the coefficients

$$\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}}) = \text{DFT}_k \left[\mathbf{n}(\mathbf{d}_{\mathcal{J}}^{x,t}(\tilde{\mathbf{a}}_{\mathcal{K}})) \right] = \text{DFT}_k \left[\mathbf{n} \left(\sum_{l=0}^{N_{\omega}-1} \Psi_l \tilde{\mathbf{a}}_l e^{i\omega_l t_{\mathcal{J}}} \right) \right]. \quad (4.1.8)$$

However, computing the nonlinearity this way is expensive since it involves expanding to the N_x -dimensional space. The problem of obtaining $\hat{\mathbf{n}}_k$ without doing computations in this space is entirely analogous to approximating the nonlinearity at a given time step given the POD coefficients at that time step. Accordingly, we use hyper-reduction, specifically, the discrete empirical interpolation method [21] (DEIM) in the case that the nonlinearity is not a quadratic form. If the nonlinearity is a quadratic form, we use (sparsified) triadic interactions. We note that it may be appropriate to apply a dealiasing technique [91] in computing the nonlinearity, such as truncating the sum in (4.1.8). The details and effectiveness of dealiasing depend on the nature of the nonlinearity. We therefore ignore this issue in dealing with general nonlinearities, but implicitly employ a standard 2/3-rule dealiasing for the quadratic nonlinearity. We note here that the function $\mathbf{n}(\tilde{\mathbf{a}}_{\mathcal{K}}, t)$ in Figure 4.1 is $\hat{\mathbf{n}}_{\mathcal{K}}(\tilde{\mathbf{a}}_{\mathcal{K}})$ transformed to the time domain.

4.1.2.1 Hyper-reduction

Here, we describe a DEIM-based approximation of $\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}})$. We assume that the nonlinearity is a local function of the state, i.e., each component of the nonlinearity is a function of only the same component of the state $n_i(\mathbf{q}) = n(q_i)$. The approach can trivially be extended to cases where the nonlinearity is a local function of the state and some number of linear operations on it, such as $n_i(\mathbf{q}) = n(q_i, (\mathbf{D}^1 \mathbf{q})_i)$, where $\mathbf{D}^1$ is a differentiation matrix.

We build the matrices $\mathbf{U}^n \in \mathbb{C}^{N_x \times p_2}$ and $\mathbf{P}^{nT} \in \mathbb{R}^{p_2 \times N_x}$ following the standard DEIM algorithm using snapshots of the nonlinearity and p_2 sample points. The matrix $\mathbf{U}^n$ is the POD modes for the nonlinearity and the matrix $\mathbf{P}^n = [\mathbf{e}_{s_1}, \dots, \mathbf{e}_{s_{p_2}}]^T$ is the sampling matrix, where $\mathbf{e}_j$ is the j -th canonical unit vector and s_i is the i -th sample point given by DEIM. The standard DEIM approximation of the nonlinearity at time t given the sampled state is $\mathbf{n}(\mathbf{q}) \approx \mathbf{U}^n (\mathbf{P}^{nT} \mathbf{U}^n)^{-1} \mathbf{n}(\mathbf{P}^{nT} \mathbf{q})$, where $\mathbf{n}(\mathbf{P}^{nT} \mathbf{q}) \in \mathbb{C}^{p_2}$.

We adapt this methodology to approximating $\hat{\mathbf{n}}_k$ as follows. The nonlinearity at each time t_j is approximated using DEIM and the state at t_j that is implied by the SPOD coefficients. Then, $\hat{\mathbf{n}}_k$ is obtained using the DFT of the nonlinearity at all times as

$$\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}}) \approx \text{DFT}_k \left[\mathbf{U}^n (\mathbf{P}^{nT} \mathbf{U}^n)^{-1} \mathbf{P}^{nT} \mathbf{n} \left(\sum_{l=0}^{N_{\omega}-1} \Psi_l \tilde{\mathbf{a}}_l e^{i\omega_l t_{\mathcal{J}}} \right) \right]. \quad (4.1.9)$$

The right-hand side of (4.1.9) is expensive to evaluate as written. However, leveraging the locality of the nonlinearity, the sampling operator $\mathbf{P}^{nT}$ may be applied directly to the SPOD modes, and the matrix $\mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1}$ may be moved outside the DFT, as it is independent of time. Then, we have

$$\begin{aligned} & \text{DFT}_k \left[\mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1} \mathbf{P}^{nT} \mathbf{n} \left(\sum_{l=0}^{N_\omega-1} \boldsymbol{\Psi}_l \tilde{\mathbf{a}}_l e^{i\omega_l t \mathcal{T}} \right) \right] \\ &= \mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1} \text{DFT}_k \left[\mathbf{n} \left(\sum_{l=0}^{N_\omega-1} \mathbf{P}^{nT} \boldsymbol{\Psi}_l \tilde{\mathbf{a}}_l e^{i\omega_l t \mathcal{T}} \right) \right]. \end{aligned} \quad (4.1.10)$$

With the appropriate precomputed operators, evaluating the expression on the right does not scale with N_x , as desired. Three operators must be precomputed. First, the sampling matrix applied to the SPOD modes, $\mathbf{S}_k = \mathbf{P}^{nT} \boldsymbol{\Psi}_k$, must be precomputed for each frequency. Next, the matrix $\boldsymbol{\Psi}_k^* \mathbf{W} \mathbf{R}_k \mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1}$ must be both precomputed and approximated for each frequency. Using the approximation $\mathbf{R}_k \approx \hat{\mathbf{Q}}_k \mathbf{G}_k^+$ established in Section 3.4.4.1, we have

$$\mathbf{N}_k = \boldsymbol{\Psi}_k^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ \mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1} \approx \boldsymbol{\Psi}_k^* \mathbf{W} \mathbf{R}_k \mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1} \in \mathbb{C}^{r_k \times p_2}. \quad (4.1.11)$$

Similarly, the matrix $\boldsymbol{\Phi}^* \mathbf{W} \mathbf{R}_k \mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1}$ must be precomputed and approximated at each frequency. Using the same approximation of the action of the resolvent, we have

$$\mathbf{M}_k = \boldsymbol{\Phi}^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ \mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1} \approx \boldsymbol{\Phi}^* \mathbf{W} \mathbf{R}_k \mathbf{U}^n(\mathbf{P}^{nT}\mathbf{U}^n)^{-1} \in \mathbb{C}^{p_1 \times p_2}. \quad (4.1.12)$$

With these approximations in place, the nonlinear system with the DEIM approximation of the nonlinearity is

$$\tilde{\mathbf{a}}_k = \mathbf{E}_k \hat{\mathbf{f}}_k + \mathbf{N}_k \hat{\mathbf{n}}_k^s(\tilde{\mathbf{a}}_{\mathcal{K}}) + \mathbf{H}_k \left(\boldsymbol{\Phi}^* \mathbf{q}_0 - \sum_{l=0}^{N_\omega-1} \mathbf{J}_l \hat{\mathbf{f}}_l + \mathbf{M}_l \hat{\mathbf{n}}_l^s(\tilde{\mathbf{a}}_{\mathcal{K}}) \right), \quad (4.1.13)$$

where the sampled nonlinearity $\hat{\mathbf{n}}_k^s \in \mathbb{C}^{p_2}$ is calculated as $\hat{\mathbf{n}}_k^s = \text{DFT}_k \left[\mathbf{n} \left(\sum_{l=0}^{N_\omega-1} \mathbf{S}_l \tilde{\mathbf{a}}_l e^{i\omega_l t \mathcal{T}} \right) \right]$.

4.1.2.2 Sparse triadic interactions

Many systems of interest have quadratic nonlinearities, i.e., ones of the form $\mathbf{n}(\mathbf{q}) = \mathbf{b}(\mathbf{q}, \mathbf{q})$, where $\mathbf{b} : \mathbb{C}^{N_x} \times \mathbb{C}^{N_x} \rightarrow \mathbb{C}^{N_x}$ is a bilinear function. In these systems, $\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}})$ can be computed exactly in the reduced space by summing over triadic interactions. These sums can be costly, so we discuss a strategy for discarding many of these triadic interactions. The resulting approximation of $\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}})$ is likely to be more accurate in many applications than

the DEIM-based approximation described above.

Expressing (4.1.8) in terms of $\mathbf{b}$ gives

$$\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}}) = \text{DFT}_k \left[\mathbf{b} \left(\sum_{l=0}^{N_\omega-1} \Psi_l \tilde{\mathbf{a}}_l e^{i\omega_l t_{\mathcal{T}}}, \sum_{i=0}^{N_\omega-1} \Psi_i \tilde{\mathbf{a}}_i e^{i\omega_i t_{\mathcal{T}}} \right) \right]. \quad (4.1.14)$$

After leveraging the bilinearity, we have

$$\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}}) = \text{DFT}_k \left[\sum_{l=0}^{N_\omega-1} \sum_{i=0}^{N_\omega-1} e^{i(\omega_l+\omega_i)t_{\mathcal{T}}} \mathbf{b}(\Psi_l \tilde{\mathbf{a}}_l, \Psi_i \tilde{\mathbf{a}}_i) \right]. \quad (4.1.15)$$

$\text{DFT}_k[\cdot]$ only depends on pairs of frequencies where $l+i \equiv k \pmod{N_\omega}$, however some of these terms are generated by aliasing. After expanding the SPOD mode sums, making use of the bilinearity, and dropping the aliasing terms, the nonlinear term becomes

$$\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}}) = \sum_{\omega_l+\omega_i=\omega_k} \sum_{m=1}^{r_l} \sum_{n=1}^{r_i} a_{lm} a_{in} \mathbf{b}(\psi_{lm}, \psi_{in}). \quad (4.1.16)$$

The terms involving $\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}})$ in the spatially reduced equation for the SPOD coefficients (4.1.7) are $\Psi_k^* \mathbf{W} \mathbf{R}_k \hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}})$ and $\Phi^* \mathbf{W} \mathbf{R}_k \hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}})$. To evaluate these terms quickly online, we precompute the vectors

$$\mathbf{n}_{klmn} = \Psi_k^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ \mathbf{b}(\psi_{lm}, \psi_{in}) \in \mathbb{C}^{r_k}, \quad (4.1.17a)$$

$$\mathbf{m}_{klmn} = \Phi^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ \mathbf{b}(\psi_{lm}, \psi_{in}) \in \mathbb{C}^{p_1}, \quad (4.1.17b)$$

where $\hat{\mathbf{Q}}_k \mathbf{G}_k^+$ is the approximation of $\mathbf{R}_k$ used elsewhere in the paper. In (4.1.17), we omit an ‘ i ’ subscript because the frequency ω_i is determined by ω_k and ω_l .

In many physical systems, the great majority of the triadic interactions are of negligible magnitude [118]. Accordingly, many of the terms in the sum in (4.1.16) may have little impact on $\hat{\mathbf{n}}_k(\tilde{\mathbf{a}}_{\mathcal{K}})$, due to either $|a_{lm} a_{in}|$ or $\|\mathbf{b}(\psi_{lm}, \psi_{in})\|_x$ being small. In order to accelerate the computation of $\hat{\mathbf{n}}_k$, it is useful to ignore these irrelevant terms in the online stage of the method. We set a threshold ϵ and only retain terms for which

$$\|\mathbf{n}_{klmn}\|_x^2 \lambda_{lm} \lambda_{in} > \epsilon. \quad (4.1.18)$$

The quantity $\|\mathbf{n}_{klmn}\|_x^2 \lambda_{lm} \lambda_{in}$ is an easy-to-evaluate proxy for $\mathbb{E}[\|\mathbf{n}_{klmn} a_{lm} a_{in}\|_x^2]$, and the two are equivalent in the case that $\mathbb{E}[|a_{lm} a_{in}|^2] = \mathbb{E}[|a_{lm}|^2] \mathbb{E}[|a_{in}|^2]$. We denote the set of

tuples (l, m, n) that meet the criterion (4.1.18) by

$$\mathcal{N}_k = \{(l, m, n) : \|\mathbf{n}_{klmn}\|_x^2 \lambda_{lm} \lambda_{in} > \epsilon\}. \quad (4.1.19)$$

The nonlinear system with the nonlinearity approximated using sparse triadic interactions is

$$\tilde{\mathbf{a}}_k = \mathbf{E}_k \hat{\mathbf{f}}_k + \sum_{(l,m,n) \in \mathcal{N}_k} \mathbf{n}_{klmn} a_{lm} a_{in} + \mathbf{H}_k \left(\Phi^* \mathbf{q}_0 - \sum_{l=0}^{N_\omega-1} \mathbf{J}_l \hat{\mathbf{f}}_l + \sum_{(p,m,n) \in \mathcal{N}_l} \mathbf{m}_{lpmn} a_{lm} a_{in} \right), \quad (4.1.20)$$

where the index ‘ i ’ is implied by l via $\omega_i = \omega_k - \omega_l$.

A practical matter bears mentioning: the nonlinear term in a quadratically nonlinear full-order model will likely not be written in quadratic form. That is, the function $\mathbf{n}(\mathbf{q})$ will be available, but not the function $\mathbf{b}(\mathbf{q}_1, \mathbf{q}_2)$. In fact, the latter is not uniquely defined. Defining $\mathbf{b}$ to also be symmetric, i.e., $\mathbf{b}(\mathbf{q}_1, \mathbf{q}_2) = \mathbf{b}(\mathbf{q}_2, \mathbf{q}_1) \forall \mathbf{q}_1, \mathbf{q}_2$, makes it both unique and easy to calculate from $\mathbf{n}$. The unique symmetric bilinear function with $\mathbf{b}(\mathbf{q}, \mathbf{q}) = \mathbf{n}(\mathbf{q})$ is

$$\mathbf{b}(\mathbf{q}_1, \mathbf{q}_2) = \frac{1}{2} [\mathbf{n}(\mathbf{q}_1 + \mathbf{q}_2) - \mathbf{n}(\mathbf{q}_1) - \mathbf{n}(\mathbf{q}_2)]. \quad (4.1.21)$$

4.1.3 Affine parametric dependence

Until now, we have suppressed the dependence of $\mathbf{A}$ on the parameter vector $\boldsymbol{\mu} \in \mathbb{R}^{N_\mu}$. Here, we show that by precomputing certain operators, the matrices in the ROM can quickly be computed given a new parameter vector. We assume this parameter dependence is affine, i.e., $\mathbf{A}(\boldsymbol{\mu})$ can be expressed as a linear combination of M_μ matrices as

$$\mathbf{A}(\boldsymbol{\mu}) = \sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \mathbf{A}_j, \quad (4.1.22)$$

with scalar coefficients ζ_j that depend on $\boldsymbol{\mu}$. The components of the ROM that inherit this parameter dependence are $\mathbf{E}_k$, $\mathbf{J}_k$, $\mathbf{N}_k$, $\mathbf{M}_k$, $\mathbf{n}_{klmn}$, $\mathbf{m}_{klmn}$, $\mathbf{H}_k$, and $\tilde{\mathbf{A}}_k$. The parameter dependence of all but the latter two is through the matrix $\mathbf{G}_k^+$.

We define the matrices

$$\mathbf{G}_{kj} = (\delta_{1j} i \omega_k \mathbf{I} - \mathbf{A}_j) \hat{\mathbf{Q}}_k, \quad (4.1.23)$$

for $j \in \{1, \dots, M_\mu\}$, where δ is the Kronecker delta. The weighted sum of these matrices is $\mathbf{G}_k$, i.e., $\sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \mathbf{G}_{kj} = \mathbf{G}_k$, and they are elements of $\mathbb{C}^{N_x \times N_d}$. We then define the following quantities,

$$\mathbf{G}_{kij}^G = \mathbf{G}_{ki}^* \mathbf{W} \mathbf{G}_{kj} \in \mathbb{C}^{N_d \times N_d}, \quad (4.1.24a)$$

$$\mathbf{B}_{kj}^G = \mathbf{G}_{kj}^* \mathbf{W} \mathbf{B} \in \mathbb{C}^{N_d \times N_f}, \quad (4.1.24b)$$

$$\mathbf{U}_{kj}^G = \mathbf{G}_{kj}^* \mathbf{W} \mathbf{U}^n \in \mathbb{C}^{N_d \times p_2}, \quad (4.1.24c)$$

$$\mathbf{b}_{kjlmin}^G = \mathbf{G}_{kj}^* \mathbf{W} \mathbf{b}(\psi_{lm}, \psi_{in}) \in \mathbb{C}^{N_d}, \quad (4.1.24d)$$

$$\tilde{\mathbf{A}}_{kj} = \mathbf{\Psi}_k^* \mathbf{W} \mathbf{A}_j \mathbf{\Psi}_k. \quad (4.1.24e)$$

Here, the superscript ‘ G ’ and subscript ‘ kj ’ are reminders that the quantity comes from taking inner products with the columns of $\mathbf{G}_{kj}^*$. With these quantities precomputed, the ROM components may be computed quickly online given a parameter vector $\boldsymbol{\mu}$ using the formulae

$$\mathbf{E}_k = \mathbf{\Psi}_k^* \mathbf{W} \hat{\mathbf{Q}}_k (\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} \sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \mathbf{B}_{kj}^G, \quad (4.1.25a)$$

$$\mathbf{J}_k = \mathbf{\Phi}^* \mathbf{W} \hat{\mathbf{Q}}_k (\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} \sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \mathbf{B}_{kj}^G, \quad (4.1.25b)$$

$$\mathbf{N}_k = \mathbf{\Psi}_k^* \mathbf{W} \hat{\mathbf{Q}}_k (\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} \left(\sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \mathbf{U}_{kj}^G \right) (\mathbf{P}^{nT} \mathbf{U}^n)^{-1}, \quad (4.1.25c)$$

$$\mathbf{M}_k = \mathbf{\Phi}^* \mathbf{W} \hat{\mathbf{Q}}_k (\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} \left(\sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \mathbf{U}_{kj}^G \right) (\mathbf{P}^{nT} \mathbf{U}^n)^{-1}, \quad (4.1.25d)$$

$$\mathbf{n}_{klmn} = \mathbf{\Psi}_k^* \mathbf{W} \hat{\mathbf{Q}}_k (\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} \sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \mathbf{b}_{kjlmin}^G, \quad (4.1.25e)$$

$$\mathbf{m}_{klmn} = \mathbf{\Phi}^* \mathbf{W} \hat{\mathbf{Q}}_k (\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} \sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \mathbf{b}_{kjlmin}^G, \quad (4.1.25f)$$

$$\tilde{\mathbf{A}}_k = \sum_{j=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \tilde{\mathbf{A}}_{kj}, \quad (4.1.25g)$$

where $(\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} \in \mathbb{C}^{N_d \times N_d}$ is computed quickly using

$$(\mathbf{G}_k^* \mathbf{W} \mathbf{G}_k)^{-1} = \left(\sum_{j=1}^{M_\mu} \sum_{l=1}^{M_\mu} \zeta_j(\boldsymbol{\mu}) \zeta_l(\boldsymbol{\mu}) \mathbf{G}_{kjl}^G \right)^{-1}. \quad (4.1.26)$$

Note that though $\mathbf{H}_k$ depends on $\boldsymbol{\mu}$, this dependence is only through $\tilde{\mathbf{A}}_k$, so after $\tilde{\mathbf{A}}_k$ is computed with a given parameter, $\mathbf{H}_k$ can be computed quickly using the expression below (3.4.25). With the formulae given in (4.1.25), all ROM components may be computed given a parameter $\boldsymbol{\mu}$ without performing operations that scale with N_x , as desired.

4.1.4 Solving the nonlinear system

Equations (4.1.13) and (4.1.20), pertaining to the DEIM-based and triadic interaction-based approximations of the nonlinearity, respectively, are both systems of rN_ω nonlinear equations in rN_ω unknowns. The online phase of the proposed reduced-order model consists of solving these equations for the coefficients $\tilde{\mathbf{a}}_\mathcal{K}$. For the method to be viable, these equations must be solved as quickly as possible, and to this end, we propose a fixed-point iteration technique. In the majority of our tests, the fixed-point iteration converged in a few (~ 10) iterations. We also describe a slower but more stable pseudo-time-stepping method that we used in the few cases where the fixed-point iteration did not converge. We analyze the convergence of both in Section 4.1.5.

Both systems (4.1.13) and (4.1.20) may be written abstractly as

$$\tilde{\mathbf{a}}_\mathcal{K} = \mathbf{c}_\mathcal{K} + \mathbf{w}_\mathcal{K}(\tilde{\mathbf{a}}_\mathcal{K}), \quad (4.1.27)$$

where $\mathbf{c}_\mathcal{K} \in \mathbb{C}^{rN_\omega}$ is the grouping of terms in the system (either (4.1.13) or (4.1.20)) that do not depend on $\tilde{\mathbf{a}}_\mathcal{K}$, and where $\mathbf{w}_\mathcal{K} : \mathbb{C}^{rN_\omega} \rightarrow \mathbb{C}^{rN_\omega}$ is the grouping of terms that do depend on $\tilde{\mathbf{a}}_\mathcal{K}$. The fixed-point iteration is simply

$$\begin{aligned} \mathbf{a}_\mathcal{K}^0 &= \mathbf{0}, \\ \mathbf{a}_\mathcal{K}^{i+1} &= \mathbf{c}_\mathcal{K} + \mathbf{w}_\mathcal{K}(\mathbf{a}_\mathcal{K}^i). \end{aligned} \quad (4.1.28)$$

With the initial guess of $\mathbf{0}$, the first iterate is $\mathbf{a}_\mathcal{K}^1 = \mathbf{c}_\mathcal{K}$, the solution to the system without the presence of the nonlinearity. The solution $\tilde{\mathbf{a}}_\mathcal{K}$ to (4.1.27) is a fixed point of the iteration above.

A necessary condition for convergence to this fixed point is that the eigenvalues of the Jacobian about it must be within the unit circle. In our numerical examples, we have found that this condition is usually met, and the iteration converges. The exceptions have been in

cases where the nonlinearity is strong, and the solution to the system with no nonlinearity bears little resemblance to that with the nonlinearity. In Section 4.1.5, we present some analysis of the fixed point iteration that is consistent with this observation. We also note that Anderson acceleration [1, 135] may be used to accelerate the convergence and possibly make it more robust.

In cases where the fixed point iteration does not converge, a slower but more stable alternative is the following pseudo-time-stepping method used by Ref. [46] to solve similar problems,

$$\begin{aligned} \mathbf{a}_K(0) &= \mathbf{c}_K, \\ \frac{d}{d\tau} \mathbf{a}_K(\tau) &= \mathbf{c}_K + \mathbf{w}_K(\mathbf{a}_K(\tau)) - \mathbf{a}_K(\tau). \end{aligned} \tag{4.1.29}$$

The stability of this method also depends on the eigenvalues of the same Jacobian. In this case, however, the stability condition is that the real part of the eigenvalues must be less than 1 (with an infinitesimal time step), so the pseudo-time-stepping method is more stable than the fixed point iteration. In our numerical examples, this method never failed to converge. Note that the pseudo-time-stepping method is equivalent to the fixed point iteration if an explicit Euler integrator is used with a time step of 1.

4.1.5 Analysis of the solution procedure

Here, we analyze two limiting cases of the fixed-point iteration used to find the solution of the nonlinear system of equations. While we do not have practical necessary and sufficient conditions for the convergence of the iteration, these limiting cases are consistent with our observation that the cases where the iteration did not converge were ones in which the nonlinearity was strong compared to the other terms.

The nonlinear system of equations to be solved is given in (4.1.27), and, in order to avoid the notational burden in this section, we rewrite it as

$$\mathbf{x} = \mathbf{c} + \mathbf{f}(\mathbf{x}), \tag{4.1.30}$$

where $\mathbf{c} \in \mathbb{C}^N$ is a known constant vector, $\mathbf{f} : \mathbb{C}^N \rightarrow \mathbb{C}^N$ is a known nonlinear function, and $\mathbf{x} \in \mathbb{C}^N$ is the unknown vector to be obtained.

4.1.5.1 Small nonlinearity

Here, we analyze the case where the nonlinear term is small, and show a correspondence between the i -th fixed-point iterate and the i -th term of a perturbation series about $\mathbf{c}$. We

require that the nonlinear term be analytic, so the equations to solve are written

$$\mathbf{x} = \mathbf{c} + \epsilon \sum_{k=2}^{\infty} \overbrace{\mathbf{f}_k(\mathbf{x}, \dots, \mathbf{x})}^{k \text{ arguments}}, \quad (4.1.31)$$

where $\epsilon \in \mathbb{R}$, and $\mathbf{f}_k$ is linear in each of its k arguments. We have excluded the $k = 0$ and $k = 1$ terms of the Taylor series for the nonlinearity since we have assumed that constant and linear terms of the right-hand side in the governing equations are accounted for by $\mathbf{c}$, though the following holds if these terms are included in the sum as well.

Perturbation series

If $\epsilon = 0$, then $\mathbf{x} = \mathbf{c}$. We use the regular perturbation ansatz $\mathbf{x} = \mathbf{x}_0 + \epsilon \mathbf{x}_1 + \epsilon^2 \mathbf{x}_2 + \dots$ to search for solutions near $\mathbf{c}$ when ϵ is small. To find a recurrence relation for $\mathbf{x}_i$, we insert this perturbation ansatz into (4.1.31), giving

$$\sum_{i=0}^{\infty} \epsilon^i \mathbf{x}_i = \mathbf{c} + \epsilon \sum_{k=2}^{\infty} \mathbf{f}_k \left(\sum_{i_1=0}^{\infty} \epsilon^{i_1} \mathbf{x}_{i_1}, \dots, \sum_{i_k=0}^{\infty} \epsilon^{i_k} \mathbf{x}_{i_k} \right). \quad (4.1.32)$$

Using the multilinearity of $\mathbf{f}_k$ and grouping powers of ϵ , this may be rewritten as

$$\sum_{i=0}^{\infty} \epsilon^i \mathbf{x}_i = \mathbf{c} + \sum_{i=1}^{\infty} \epsilon^i \sum_{k=2}^{\infty} \sum_{i_1+\dots+i_k=i-1} \mathbf{f}_k(\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}). \quad (4.1.33)$$

Here, the indices in the inner sum are non-negative integers. Finally, equating like powers of ϵ on the right and left, we have the following recurrence relation

$$\mathbf{x}_0 = \mathbf{c}, \quad (4.1.34a)$$

$$\mathbf{x}_i = \sum_{k=2}^{\infty} \sum_{i_1+\dots+i_k=i-1} \mathbf{f}_k(\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}). \quad (4.1.34b)$$

(4.1.34b) is a recurrence relation since the highest order term on the right is $\mathbf{x}_{i-1}$.

Fixed-point iteration

The fixed-point iteration with the analytic right-hand side is

$$\mathbf{x}^0 = \mathbf{c} \quad (4.1.35a)$$

$$\mathbf{x}^{i+1} = \mathbf{c} + \epsilon \sum_{k=2}^{\infty} \mathbf{f}_k(\mathbf{x}^i, \dots, \mathbf{x}^i). \quad (4.1.35b)$$

Relation between the two

Proposition. The i -th fixed point iterate is equal to the perturbation series truncated at the i -th order to $\mathcal{O}(\epsilon^i)$. That is,

$$\mathbf{x}^i = \sum_{j=0}^i \epsilon^j \mathbf{x}_j + \epsilon^{i+1} \mathbf{r}_i \quad \text{for } i \in \{0, 1, \dots\}, \quad (4.1.36)$$

where $\mathbf{r}_i$ is an error term, and the prefactor of ϵ^{i+1} indicates that this error term is of order ϵ^{i+1} or higher.

Proof. We prove (4.1.36) inductively. The base case is trivial: the zeroth iterate is exactly equal to the zeroth term in the perturbation series, since both are equal to $\mathbf{c}$ (see (4.1.34a) and (4.1.35a)). We assume the statement is true for $i = n$, and show that this implies it for $i = n + 1$. Namely, we assume

$$\mathbf{x}^n = \sum_{j=0}^n \epsilon^j \mathbf{x}_j + \epsilon^{n+1} \mathbf{r}_n. \quad (4.1.37)$$

With this assumption, the $n + 1$ -st fixed point iterate is

$$\mathbf{x}^{n+1} = \mathbf{c} + \epsilon \sum_{k=2}^{\infty} \mathbf{f}_k \left(\sum_{i_1=0}^n \epsilon^{i_1} \mathbf{x}_{i_1} + \epsilon^{n+1} \mathbf{r}_n, \dots, \sum_{i_k=0}^n \epsilon^{i_k} \mathbf{x}_{i_k} + \epsilon^{n+1} \mathbf{r}_n \right). \quad (4.1.38)$$

All terms involving $\mathbf{r}_n$ are of order ϵ^{n+2} or higher, so they may be grouped into the $n + 1$ -th error term. By grouping terms at the same order in ϵ , and combining all terms of order higher than ϵ^{n+1} into the error term, (4.1.38) may be rewritten as

$$\mathbf{x}^{n+1} = \mathbf{c} + \sum_{j=1}^{n+1} \epsilon^j \sum_{k=2}^{\infty} \sum_{i_1+\dots+i_k=j-1} \mathbf{f}_k(\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}) + \epsilon^{n+2} \mathbf{r}_{n+1}. \quad (4.1.39)$$

Since $\mathbf{x}_0 = \mathbf{c}$ and since the inner two sums are $\mathbf{x}_j$ by (4.1.34b), the above equation may be rewritten as

$$\mathbf{x}^{n+1} = \sum_{j=0}^{n+1} \epsilon^j \mathbf{x}_j + \epsilon^{n+2} \mathbf{r}_{n+1}. \quad (4.1.40)$$

This completes the proof.

4.1.5.2 Bilinear right-hand side

Here we analyze the iteration in the case where $\mathbf{c} = \mathbf{0}$ and where $\mathbf{f}(\mathbf{x}) = \mathbf{b}(\mathbf{x}, \mathbf{x})$ where $\mathbf{b} : \mathbb{C}^N \times \mathbb{C}^N \rightarrow \mathbb{C}^N$ is bilinear. This case is relevant where the nonlinear term is quadratic

and dominant, and this case is sometimes discussed in the context of resolvent analysis (see Fig. 1 of Ref. [81], a review of the subject). We show that neither the fixed-point iteration nor the pseudo-time-stepping method can converge to nonzero fixed points due to a linear instability about these points. We first analyze the fixed-point iteration

$$\mathbf{x}^{i+1} = \mathbf{b}(\mathbf{x}^i, \mathbf{x}^i) \quad (4.1.41)$$

near a solution $\bar{\mathbf{x}}$ satisfying $\bar{\mathbf{x}} = \mathbf{b}(\bar{\mathbf{x}}, \bar{\mathbf{x}})$. Writing $\mathbf{x}^i = \bar{\mathbf{x}} + \epsilon \mathbf{y}^i$ and inserting this into the iteration, we have

$$\mathbf{y}^{i+1} = \epsilon \left(\mathbf{b}(\bar{\mathbf{x}}, \mathbf{y}^i) + \mathbf{b}(\mathbf{y}^i, \bar{\mathbf{x}}) \right) + \epsilon^2 \mathbf{b}(\mathbf{y}^i, \mathbf{y}^i). \quad (4.1.42)$$

Sufficiently close to the fixed point (i.e., with ϵ sufficiently small), the dynamics are then given by

$$\mathbf{y}^{i+1} = \mathbf{L} \mathbf{y}^i, \quad (4.1.43)$$

where $\mathbf{L} \mathbf{y}^i = \mathbf{b}(\bar{\mathbf{x}}, \mathbf{y}^i) + \mathbf{b}(\mathbf{y}^i, \bar{\mathbf{x}})$ is the Jacobian about the fixed point $\bar{\mathbf{x}}$. If all the eigenvalues of $\mathbf{L}$ are within the unit circle, $\|\mathbf{y}\|$ decays as the iteration proceeds, and the fixed point $\bar{\mathbf{x}}$ is stable; otherwise, it is unstable.

If $\bar{\mathbf{x}} = \mathbf{0}$, then the fixed point is stable since $\mathbf{L}$ is the zero matrix. However, if $\bar{\mathbf{x}} \neq \mathbf{0}$, the fixed point is unstable because there is an eigenvalue of 2 associated with the eigenvector $\mathbf{y} = \bar{\mathbf{x}}$. This is true because $\mathbf{L} \bar{\mathbf{x}} = \mathbf{b}(\bar{\mathbf{x}}, \bar{\mathbf{x}}) + \mathbf{b}(\bar{\mathbf{x}}, \bar{\mathbf{x}}) = 2\bar{\mathbf{x}}$. This argument can be generalized to the case when the constant is zero and the nonlinearity is any k -linear function. In this case, the origin is again always a stable fixed point (for $k > 1$), and the linear operator about any other fixed point admits an eigenvalue of k associated with the eigenvector $\bar{\mathbf{x}}$.

The above analysis implies that the pseudo-time-stepping method also cannot converge to non-zero fixed points in the case of $\mathbf{c} = \mathbf{0}$ and a k -linear right-hand side since the stability in that case requires that the eigenvalues of the same Jacobian have a real part less than 1.

4.1.6 Algorithms and computational complexity

4.1.6.1 Algorithms

The offline phase of the method is given in Algorithm 1. The first step in the online phase of the method is computing the constant term using Algorithm 2. Then, either the fixed point iteration or the pseudo-time-stepping method where the nonlinear term $\mathbf{w}_K(\mathbf{a}_K)$ is calculated using either Algorithm 3 or Algorithm 4, depending on the nature of the nonlinearity. For readability, we have written Algorithm 4 with for loops, but we note that in our numerical implementation, we used sparse matrices, which will run faster on most systems.

Algorithm 1 SSOP (offline)

```

1:  $\Phi = \text{POD}(\mathbf{Q}^t, \mathbf{W}, p_1)$  ▷ Intermediary basis
2:  $\hat{\mathbf{Q}}_{\mathcal{K}} = \text{WelchBlocks}(\mathbf{Q}^t, N_{\omega})$  ▷ Obtain data matrices of each frequency from snapshot matrix
3: if  $\mathbf{n}$  is non-quadratic then
4:    $[\mathbf{U}^n, \mathbf{P}^{nT}] = \text{DEIM}(\mathbf{n}(\mathbf{Q}^t), p_2)$  ▷ Obtain sample points and basis for nonlinearity
5: end if
6:  $[\Psi_k^{N_d}, \Lambda_k^{N_d}, \Psi_k, \Lambda_k] = \text{SPOD}(\hat{\mathbf{Q}}_{\mathcal{K}}, r)$  ▷ Get SPOD modes and energies
7: for  $k \in \mathcal{K}$  do
8:    $\mathbf{L}_k \leftarrow (i\omega_k \mathbf{I} - \mathbf{A})$ 
9:    $\mathbf{G}_k^+ \leftarrow (\mathbf{L}_k \hat{\mathbf{Q}}_k)^+$  ▷ Weighted pseudoinverse of  $\mathbf{G}_k$  used for approximating resolvent
10:   $\mathbf{E}_k \leftarrow \Psi_k^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ \mathbf{B}$  ▷ Output variable
11:   $\mathbf{J}_k \leftarrow \Phi^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ \mathbf{B}$  ▷ Output variable
12:   $\tilde{\mathbf{A}}_k \leftarrow \Psi_k^{N_d*} \mathbf{W} \mathbf{A} \Psi_k^{N_d}$  ▷ Used for calculating  $\mathbf{H}_k$ 
13:   $\mathbf{P}_k \leftarrow [\mathbf{I}_{r_k \times r_k} \quad \mathbf{0}_{r_k \times (N_d - r_k)}]$  ▷ Used for calculating  $\mathbf{H}_k$ 
14:   $\mathbf{H}_k \leftarrow \mathbf{P}_k (\mathbf{I} - \exp[(\tilde{\mathbf{A}} - i\omega_k \mathbf{I}) \Delta t])^{-1} (\mathbf{I} - \exp[\tilde{\mathbf{A}} T]) (\Psi_k^{N_d*} \mathbf{W} \Phi)$  ▷ Output variable
15:  if  $\mathbf{n}$  is non-quadratic then
16:     $\mathbf{N}_k \leftarrow \Psi_k^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ \mathbf{U}^n (\mathbf{P}^{nT} \mathbf{U}^n)^{-1}$  ▷ Output variable
17:     $\mathbf{M}_k \leftarrow \Phi^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ \mathbf{U}^n (\mathbf{P}^{nT} \mathbf{U}^n)^{-1}$  ▷ Output variable
18:     $\mathbf{S}_k \leftarrow \mathbf{P}^{nT} \Psi_k$  ▷ Output variable
19:  else
20:     $b(q_1, q_2) := \frac{1}{2} [\mathbf{n}(q_1 + q_2) - \mathbf{n}(q_1) - \mathbf{n}(q_2)]$  ▷ Symmetric bilinear form
21:     $\mathcal{N}_k \leftarrow \{\}$ 
22:    for  $(l, i)$  such that  $\omega_l + \omega_i = \omega_k$  do ▷ Looping over triadic interactions
23:      for  $m \in \{1, \dots, r_l\}$  do
24:        for  $n \in \{1, \dots, r_i\}$  do
25:           $\tilde{\mathbf{n}} \leftarrow \Psi_k^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ b(\psi_{lm}, \psi_{in})$ 
26:          if  $\|\tilde{\mathbf{n}}\|_x^2 \lambda_{lm} \lambda_{in} > \epsilon$  then ▷ Checking if interaction meets threshold
27:             $\mathbf{n}_{klmn} \leftarrow \tilde{\mathbf{n}}$  ▷ Output variable
28:             $\mathbf{m}_{klmn} \leftarrow \Phi^* \mathbf{W} \hat{\mathbf{Q}}_k \mathbf{G}_k^+ b(\psi_{lm}, \psi_{in})$  ▷ Output variable
29:             $\mathcal{N}_k \leftarrow \mathcal{N}_k \cup (l, m, n)$  ▷ Updating list of interactions
30:          end if
31:        end for
32:      end for
33:    end for
34:  end if
35: end for

```

Inputs: $\mathbf{Q}^t$, time series data; Δt , time step used in data; N_{ω} , number of time steps desired for solution blocks; $\mathbf{n}$, nonlinear function; p_1 , number of modes to use in intermediary basis; p_2 (if DEIM is used), number of sample points to use in DEIM approximation; r , average number of modes per frequency for ROM; $\mathbf{A}$, $\mathbf{B}$, system matrices; $\mathbf{W}$, weight matrix; ϵ (if triadic interactions are used), tolerance for inclusion.

Outputs: Φ , intermediary basis; $\mathbf{J}_{\mathcal{K}}$, $\mathbf{J}$ operators at all frequencies; $\mathbf{E}_{\mathcal{K}}$, $\mathbf{E}$ operators at all frequencies; $\mathbf{H}_{\mathcal{K}}$, $\mathbf{H}$ operators at all frequencies. If using DEIM, $\mathbf{S}_{\mathcal{K}}$, the sampling operators at all frequencies; $\mathbf{N}_{\mathcal{K}}$, the first set of hyper-reduction matrices; $\mathbf{M}_{\mathcal{K}}$ the second set of hyper-reduction matrices. Otherwise, $\mathcal{N}_{\mathcal{K}}$, the set $\mathcal{N}$ at all frequencies; $\mathbf{n}_{klmn}$ for all $k \in \mathcal{K}$, $(l, m, n) \in \mathcal{N}_k$; $\mathbf{m}_{klmn}$ for all $k \in \mathcal{K}$, $(l, m, n) \in \mathcal{N}_k$.

Algorithm 2 SSOP (online, constant term)

```

1:  $\hat{\mathbf{f}}_{\mathcal{K}} = \text{FFT}_{\mathcal{K}}[\mathbf{f}_{\mathcal{J}}]$  ▷ FFT of forcing
2:  $\mathbf{q}_0^{\Phi} \leftarrow \Phi^* \mathbf{W} \mathbf{q}_0$  ▷ The initial condition in the intermediary basis
3:  $\tilde{\mathbf{q}}_0^{\Phi} \leftarrow \mathbf{0}_{p_1 \times 1}$  ▷ Initializing the forcing sum term
4: for  $k \in \mathcal{K}$  do
5:    $\tilde{\mathbf{q}}_0^{\Phi} \leftarrow \tilde{\mathbf{q}}_0^{\Phi} + \frac{1}{N_{\omega}} \mathbf{J}_k \hat{\mathbf{f}}_k$  ▷ Influence of each frequency on forcing sum
6: end for
7: for  $k \in \mathcal{K}$  do
8:    $\mathbf{c}_k \leftarrow \mathbf{E}_k \hat{\mathbf{f}}_k + \mathbf{H}_k (\mathbf{q}_0^{\Phi} - \tilde{\mathbf{q}}_0^{\Phi})$  ▷ Constructing each frequency of the constant term
9: end for

```

Inputs: $\mathbf{q}_0$, the initial condition; Φ , intermediary basis (POD modes); $\mathbf{W}$, weight matrix; $\mathbf{f}_{\mathcal{J}}$, the forcing on the temporal grid; $\mathbf{J}_{\mathcal{K}}$, $\mathbf{J}$ operators at all frequencies (defined below 3.4.25); $\mathbf{E}_{\mathcal{K}}$, $\mathbf{E}$ operators at all frequencies (defined in (3.4.21)); $\mathbf{H}_{\mathcal{K}}$, $\mathbf{H}$ operators at all frequencies (defined below 3.4.25).

Outputs: $\mathbf{c}_{\mathcal{K}}$, all frequency components of the constant term.

Algorithm 3 SSOP (online, non-quadratic nonlinear term)

```

1: for  $k \in \mathcal{K}$  do
2:    $\tilde{\mathbf{q}}_k^s \leftarrow \mathbf{S}_k \tilde{\mathbf{a}}_k$  ▷ Getting sample points in frequency domain
3: end for
4:  $\mathbf{q}_{\mathcal{J}}^s = \text{IFFT}[\tilde{\mathbf{q}}_{\mathcal{K}}^s]$  ▷ Sample points in time domain
5:  $\mathbf{n}_{\mathcal{J}}^s = \mathbf{n}(\mathbf{q}_{\mathcal{J}}^s)$  ▷ Nonlinearity at sample points in time domain
6:  $\hat{\mathbf{n}}_{\mathcal{K}}^s = \text{FFT}[\mathbf{n}_{\mathcal{J}}^s]$  ▷ Nonlinearity at sample points in frequency domain
7:  $\tilde{\mathbf{q}}_0^{\Phi} \leftarrow \mathbf{0}_{p_1 \times 1}$ 
8: for  $k \in \mathcal{K}$  do
9:    $\tilde{\mathbf{q}}_0^{\Phi} \leftarrow \tilde{\mathbf{q}}_0^{\Phi} + \frac{1}{N_{\omega}} \mathbf{M}_k \hat{\mathbf{n}}_k^s$ 
10:   $\mathbf{w}_k \leftarrow \mathbf{N}_k \hat{\mathbf{n}}_k^s$ 
11: end for
12: for  $k \in \mathcal{K}$  do
13:    $\mathbf{w}_k \leftarrow \mathbf{w}_k - \mathbf{H}_k \tilde{\mathbf{q}}_0^{\Phi}$ 
14: end for

```

Inputs: $\mathbf{a}_{\mathcal{K}}$, the SPOD coefficients at the current iteration; $\mathbf{S}_{\mathcal{K}}$, the sampling operators at all frequencies; $\mathbf{n}$, the nonlinear function; $\mathbf{N}_{\mathcal{K}}$, the first set of matrices hyper-reduction matrices (see (4.1.11)); $\mathbf{M}_{\mathcal{K}}$ the second set of hyper-reduction matrices (see (4.1.12)); $\mathbf{H}_{\mathcal{K}}$, the set of matrices (defined under (3.4.25)).

Output: $\mathbf{w}_{\mathcal{K}}$, the approximate nonlinear term formed by concatenating $\mathbf{w}$ at all frequencies.

Algorithm 4 SSOP (online, quadratic nonlinear term)

```

1:  $\mathbf{w}^\Phi \leftarrow \mathbf{0}_{p_1 \times 1}$ 
2: for  $k \in \mathcal{K}$  do
3:    $\mathbf{w}_k \leftarrow \mathbf{0}_{r_k \times 1}$ 
4:   for  $(l, m, n) \in \mathcal{N}_k$  do
5:      $\mathbf{w}_k \leftarrow \mathbf{w}_k + \mathbf{n}_{klmn} \tilde{a}_{lm} \tilde{a}_{in}$ 
6:      $\mathbf{w}^\Phi \leftarrow \mathbf{w}^\Phi + \mathbf{m}_{klmn} \tilde{a}_{lm} \tilde{a}_{in}$ 
7:   end for
8: end for
9: for  $k \in \mathcal{K}$  do
10:   $\mathbf{w}_k \leftarrow \mathbf{w}_k - \mathbf{H}_k \mathbf{w}^\Phi$ 
11: end for

```

Inputs: $\mathbf{a}_\mathcal{K}$, the SPOD coefficients at the current iteration; $\mathcal{N}_\mathcal{K}$, the set $\mathcal{N}$ at all frequencies (see (4.1.19)); $\mathbf{n}_{klmn}$ for all $k \in \mathcal{K}$, $(l, m, n) \in \mathcal{N}_k$ (see (4.1.17a)); $\mathbf{m}_{klmn}$ for all $k \in \mathcal{K}$, $(l, m, n) \in \mathcal{N}_k$ (see (4.1.17b)); $\mathbf{H}_\mathcal{K}$, the set of matrices (defined under (3.4.25)).

Output: $\mathbf{w}_\mathcal{K}$, the approximate nonlinear term formed by concatenating $\mathbf{w}$ at all frequencies.

4.1.6.2 Computational complexity

Here we give the scalings for both, depending on the handling of the nonlinearity. With either DEIM or the sparse triadic interactions, the dominant online cost comes from calculating the nonlinear term $\mathbf{w}_\mathcal{K}$ at each iteration. In totaling the online scaling of the method in this subsection and timing the method in Section 4.2, we count all operations beginning from the initial condition and forcing and ending at the SPOD coefficients.

The online phase of the method consists of repeating the iteration (4.1.28) until the SPOD coefficients are converged. The constant $\mathbf{c}_\mathcal{K}$ is computed once at the beginning; computing this constant is equivalent to solving for the SPOD coefficients with no nonlinearity present. The nonlinear term is computed for each iteration, and, in practice, it is the dominant cost of the method.

CPU cost due to the constant term scales as

$$\mathcal{O}(N_\omega(N_f \log N_\omega + p_1 N_f + r N_f + r p_1) + p_1 N_x). \quad (4.1.44)$$

The first term is due to the fast Fourier transform (FFT) of the forcing; the second to calculating the effect of the forcing in the intermediary basis; the third to calculating the effect of the forcing directly on each SPOD coefficient; and the fourth to calculating the effect of each coefficient in the intermediary basis on each SPOD coefficient. The last term, which does not scale with the number of frequencies, comes from calculating the initial condition in the intermediary basis.

The cost of the nonlinear term at each iteration, i.e., each evaluation of $\mathbf{w}_{\mathcal{K}}(\tilde{\mathbf{a}}_{\mathcal{K}})$, depends on the handling of the nonlinearity. If DEIM is used, each iteration scales as

$$\mathcal{O}(N_{\omega}(rp_2 + p_2 \log N_{\omega} + p_1 p_2 + rp_1)). \quad (4.1.45)$$

The first term is due to sampling the trajectory at the p_2 sample points at every frequency; the second to taking the inverse FFT (IFFT) of this sampling, computing the nonlinearity, then taking the FFT; the third to computing the effect of the nonlinearity at p_2 sample points in the intermediary basis; and the fourth to calculating the effect of the each coefficient in the intermediary basis on each SPOD coefficient.

If, instead, sparse triadic interactions are used, the scaling of each iteration is

$$\mathcal{O}\left(N_{\omega}p_1r + \sum_{k \in \mathcal{K}} |\mathcal{N}_k|(r_k + p_1)\right). \quad (4.1.46)$$

The first term accounts for the effect of each coefficient of the intermediary basis on each SPOD coefficient. The sum accounts for the number of nonlinear terms that are retained at each frequency.

The total online cost, therefore, scales as

$$\mathcal{O}(N_{\omega}(N_f \log N_{\omega} + p_1 N_f + r N_f + N_{iter}(rp_2 + p_2 \log N_{\omega} + p_1 p_2 + rp_1)) + p_1 N_x) \quad (4.1.47)$$

if DEIM is used and

$$\mathcal{O}\left(N_{\omega}(N_f \log N_{\omega} + p_1 N_f + r N_f) + N_{iter}\left(rp_1 N_{\omega} + \sum_{k \in \mathcal{K}} |\mathcal{N}_k|(r_k + p_1)\right) + p_1 N_x\right) \quad (4.1.48)$$

if the sparse triadic interactions are used.

In practice, in the DEIM case, r , p_1 , and p_2 will likely be of similar magnitudes, and the dominant cost will come from the iteration to solve the system. The scaling will then be

$$\mathcal{O}(N_{\omega}N_{iter}r^2). \quad (4.1.49)$$

If the sparse triadic interactions are used, the dominant cost will likely come from evaluating the nonlinear term at every iteration. A rough estimate of the scaling in this case is

$$\mathcal{O}(N_{iter}N_{int}r), \quad (4.1.50)$$

where $N_{int} = \sum_{k \in \mathcal{K}} |\mathcal{N}_k|$. In practice, we have observed the fixed-point iteration to converge

in ~ 10 iterations or fewer. If the pseudo-time-stepping method is used, N_{iter} is the number of time steps needed for convergence, and in the cases where we used this method, N_{iter} was a few hundred using an adaptive time-stepping method (though we made no attempt to optimize this).

Assuming the governing equations are sparse, i.e., computing the action $\mathbf{A}$ and $\mathbf{n}$ both scale as $\mathcal{O}(N_x)$, the offline scalings are as follows. Obtaining the SPOD modes and computing the operators necessary for the constant term scale as

$$\mathcal{O}(N_\omega N_d^2 N_x). \quad (4.1.51)$$

Constructing the operators needed for the nonlinearity in the DEIM case scales as

$$\mathcal{O}((r + p_1 + p_2)N_d N_x N_\omega). \quad (4.1.52)$$

In the case of sparse triadic interactions, constructing the necessary vectors scales as

$$\mathcal{O}\left(\sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{K}} r_l r_i r_k N_x\right). \quad (4.1.53)$$

So long as the offline cost is feasible, the online cost is the salient quantity. In practice, computing the effect of the nonlinearity for each iteration ((4.1.45) or (4.1.46)) dominates the online cost. In our numerical examples, we find that DEIM is faster (at the same accuracy) than the sparsified triadic interactions, though we do not expect this to hold in general.

4.2 Results

We use the 1-dimensional Ginzburg-Landau equations to test the method. We use the standard form of these equations for testing the DEIM-based handling of nonlinearity and a modified Ginzburg-Landau system with a quadratic nonlinearity to test the triadic-interaction-based treatment of the nonlinearity. For both systems, we compare the relative error and online CPU time of SSOP to those of POD-G. The results are encouraging: for the same number of modes, SSOP gives two orders of magnitude lower error than does POD-G. For the DEIM-based treatment of the nonlinearity, this accuracy improvement comes at slightly lower CPU time than POD-G, whereas the triadic-interaction-based ROM is slower than POD-G. We also test the accuracy of the method on out-of-sample data and find that the method trained on one GL system generalizes well to a new GL system.

4.2.1 Standard Ginzburg-Landau system

The complex Ginzburg-Landau equation is a common test case for model reduction methods [56, 14, 23, 93]. It is

$$\dot{q}(x, t) = \left[-\nu \frac{\partial}{\partial x} + \gamma \frac{\partial^2}{\partial x^2} + \mu_0 - c_\mu^2 + \frac{\mu_2}{2} x^2 \right] q(x, t) - \alpha q(x, t) |q(x, t)|^2 + f(x, t), \quad (4.2.1)$$

where $f(x, t)$ is a forcing and where we set $\nu = 2 + 0.4i$, $\gamma = 1 - i$, $c_\mu = 0.2$, $\mu_2 = -0.01$, and $\alpha = 1$, which are standard values [5, 131]. In most of our tests, we set μ_0 , a bifurcation parameter in the system, to $\mu_0 = 0.229$ [5, 131]. The terms in the system generate advection, diffusion, and local growth or decay depending on the sign of $\mu_0 - c_\mu^2 + \frac{\mu_2}{2} x^2$. For $\mu_0 = 0.229$, $\mu_0 - c_\mu^2 + \frac{\mu_2}{2} x^2 > 0$ for $x \in [-6.15, 6.15]$, so solutions grow in this region. For this value of μ_0 , the system is linearly stable, i.e., the eigenvalues of the linear operator about the steady solution $q(x) = 0$ are all in the stable half-plane. There is a bifurcation at $\mu_0 = 0.397$; above this value, the system is linearly unstable, and the dynamics are characterized by a competition between the amplifying effect of the linear terms and the damping effect of the nonlinear term. While most of our tests use $\mu_0 = 0.229$, we also test the range $\mu_0 \in [0.079, 0.499]$, which envelopes the bifurcation. We refer to (4.2.1) as the ‘standard’ Ginzburg-Landau system to distinguish it from the modified system introduced in the next subsection.

The full-order model consists of a pseudo-spectral Hermite discretization [5, 22] of (4.2.1) with $N_x = 220$ collocation points [131] that we integrate with `ode45` in MATLAB. To generate data, we integrate for 3000 time steps with $\Delta t = 0.8$. We use the method described in Ref. [131] to obtain the modes from the single long trajectory by forming 44 trajectories of length $N_\omega = 256$, each overlapping 75% with the next. The forcing generating this data is spread over the range $x \in [-12, -8]$ and is strongest at $\bar{x} = -10$. It is stochastic with a spatial correlation length of 1, a temporal correlation length of 3.33, and a Gaussian support. The spatiotemporal correlation is given by

$$\mathbb{E}[f(x_1, t_1) f^*(x_2, t_2)] \propto \exp \left[- \left((x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + (x_2 - x_1)^2 + (0.3(t_2 - t_1))^2 \right) \right], \quad (4.2.2)$$

and we take it to be zero outside $x \in [-12, -8]$.

Figure 4.2 shows a space-time diagram of a trajectory of the Ginzburg-Landau system with $\mu_0 = 0.229$. The advective behavior of the system is evident in the diagonally oriented streaks, with the upward inclination indicating advection in the positive direction. There are spatial correlations here, but, crucially, there are strong spatiotemporal correlations as

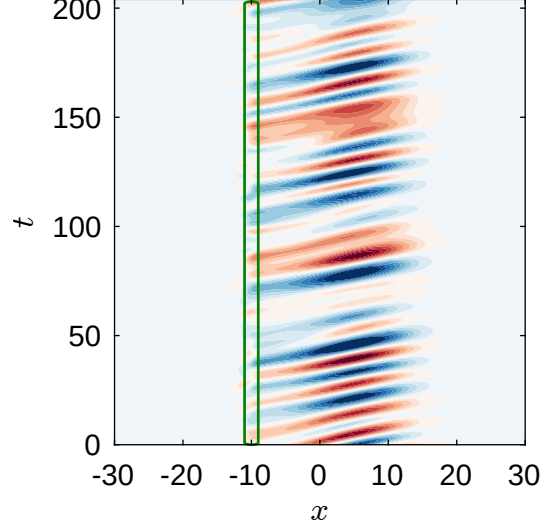


Figure 4.2: A trajectory of the (standard) Ginzburg-Landau system with $\mu_0 = 0.229$. The green lines demarcate the location of the forcing. The spatiotemporal structure enables space-time modes, like SPOD modes, to represent the trajectory far more efficiently than space-only modes.

well. These spatiotemporal correlations — the fact that there is structure in this space-time diagram beyond the spatial structure — are what allow space-time modes to encode the trajectory substantially more efficiently than POD modes.

Figure 4.3 shows the energy of the modes. Specifically, Figure 4.3 (a) shows the fraction of energy that is excluded as a function of r , which is computed by summing the energies of the modes that are not among the rN_ω most energetic, normalized by the sum of all of the energies. This is equivalent to the average relative SPOD reconstruction error over the training data. This gives a lower bound of the relative error of the method applied to the training data — the error of the SSOP ROM cannot be lower than the SPOD reconstruction error. Figure 4.3 (b) shows the SPOD energies of the modes as a function of frequency for $r = 5$. The top curve is $\lambda_{\mathcal{K},1}$, the first SPOD mode as a function of frequency, and the lower curves are higher mode numbers. The red and blue curves represent retained and truncated modes, respectively. The threshold is determined by r as $\tilde{\lambda}_{(rN_\omega)}$ and is shown in orange. The number of modes at each frequency that meet the threshold and are thus retained is shown in green on the right axis. For the least energetic frequencies, as few as 3 modes are retained, whereas for the most energetic frequencies, as many as 8 are retained. The average value is $r = 5$, by definition.

Throughout the remainder of this section, the (relative) errors are defined as follows. The error at time t_j is the square norm of the difference between the ROM and FOM solutions normalized by the mean square norm of the FOM solution averaged over time and over test

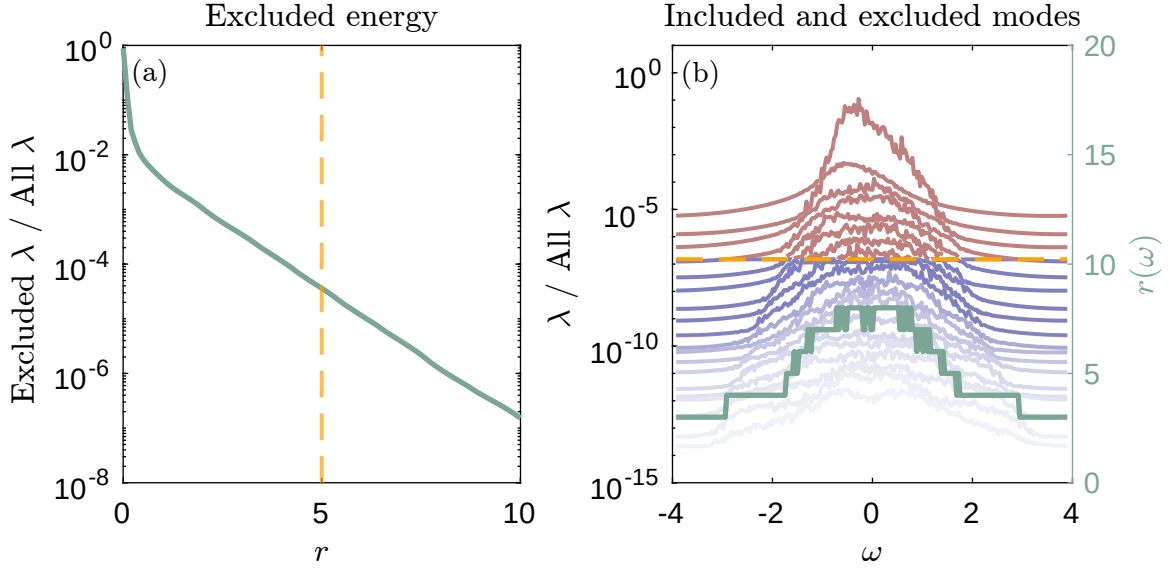


Figure 4.3: SPOD energies. (a) The energy not captured by the first rN_ω modes, calculated as the ratio of the sum of the energies of the excluded modes to the sum of the energies of all the modes. The dashed line corresponds to the r value used to determine the cutoff in (b). (b) The energy of the included (red) and excluded (blue) modes as a function of ω for $r = 5$. Inclusion of a given mode ψ_{km} is determined by whether the associated energy is among the rN_ω largest energies over all frequencies i.e., whether $\lambda_{km} \geq \tilde{\lambda}_{(rN_\omega)}$ (see (3.2.10)). The cutoff energy $\tilde{\lambda}_{(rN_\omega)}$ is shown in orange. The number of modes at each frequency $r(\omega)$ is shown in green on the right axis; the mean is $r = 5$.

trajectories. This may be written as

$$e_j = \frac{\|\tilde{\mathbf{q}}_j - \mathbf{q}_j\|_x^2}{\frac{1}{N_\omega N_{test}} \sum_{i=1}^{N_{test}} \|\mathbf{q}_{\mathcal{J}}^i\|_{x,t}^2}, \quad (4.2.3)$$

where $\mathbf{q}$ is the FOM solution, $\tilde{\mathbf{q}}$ is the ROM approximation thereof, and $\mathbf{q}_{\mathcal{J}}^i$ is the i -th trajectory in the test data. This error averaged over time is

$$e = \frac{\|\tilde{\mathbf{q}}_{\mathcal{J}} - \mathbf{q}_{\mathcal{J}}\|_{x,t}^2}{\frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \|\mathbf{q}_{\mathcal{J}}^i\|_{x,t}^2}. \quad (4.2.4)$$

For the most part, we report these quantities averaged over the test trajectories below.

We first test the method on 30 new trajectories, i.e., trajectories not included in the training data. The forcing is stochastic in each case and is drawn from the same distribution as the stochastic forcing used to generate the training data (but the realizations are different). The 30 initial conditions in the test data are taken from a single long run of the system and are also not in the data used to compute the SPOD modes, but come from the same statistical distribution as the states in the training data. These conditions together mean that the SPOD modes from the training data are good at representing the trajectories in the test data.

In Figure 4.4, we show the relative error of the proposed method with $r = 5$ modes along with that of POD-G, also with 5 modes. The dashed curves are the projection errors of the respective bases. In other words, they represent the discrepancy between the full-order solution and the projection thereof onto the POD and SPOD bases. The projection error is a lower bound for the model error; if the model recovers the exact coefficients, then it achieves the projection error. All errors are calculated as the mean over the 30 realizations on which we test the method, and the shaded regions represent data within one standard deviation of the mean. Most notably, SSOP outperforms POD-G by nearly three orders of magnitude for most of the interval. It also significantly outperforms the projection of the FOM solution onto the POD modes (dashed brown). Since POD modes are the optimal linear encoder, the POD projection error is a lower bound for any method based on a space-only linear encoding, such as a space-only Petrov-Galerkin method or operator inference. That SSOP and the SPOD projection (solid and dashed green, respectively) nearly overlap indicates that SSOP solves nearly exactly for the SPOD coefficients. A notable feature of Figure 4.4 is that the error is largest at the ends of the interval, and it bears mentioning that this feature is not due to Gibbs phenomena — indeed, it is too small. Instead, it is due to a cyclic property of the SPOD encoder/decoder pair, and as more modes are included, the effect diminishes.

Figure 4.5 shows the relative error, averaged over time, as a function of the number of

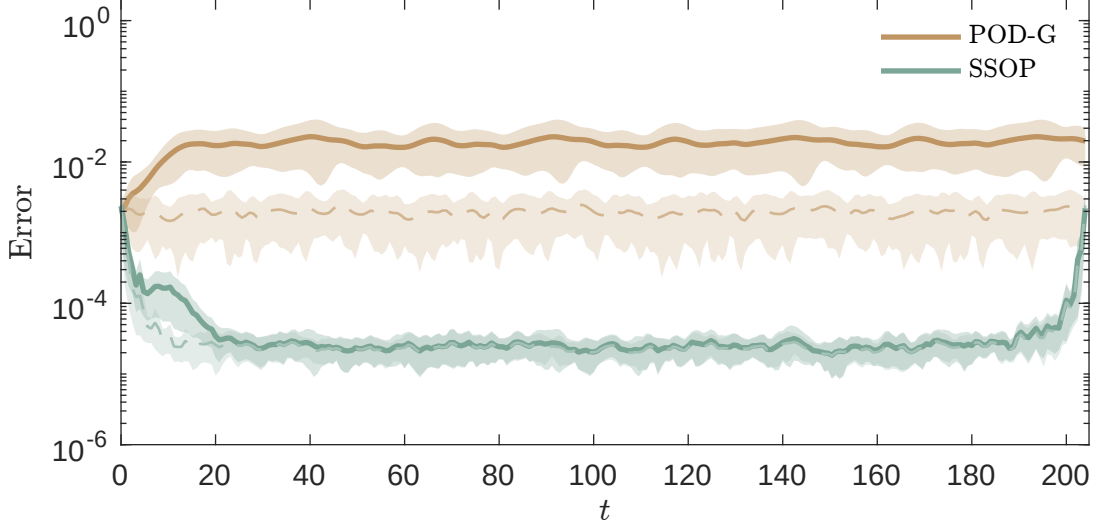


Figure 4.4: Error as a function of time for the proposed method and for POD-G, both with $r = 5$. The curves represent the mean error over the 30 test trajectories, and the shaded regions indicate the standard deviation of these errors. The dashed curves show the respective projection errors, which serve as lower bounds for the two methods.

modes retained. For all the values shown, SSOP again gives multiple orders of magnitude lower error than POD-G. The dashed curves are again the projection error. Another notable feature of the method is that it nearly achieves the SPOD mode projection error for relatively few modes. The error asymptote is likely due to the fact that the forcing and nonlinearity are not exactly represented by a finite Fourier series, as was assumed in deriving the method.

Figure 4.6 shows the CPU time of the proposed method in comparison to that of POD-G. The substantial error reduction shown does not come at an increase in computational cost; indeed, the method is faster for most of the range considered. In what follows, we use $r = 5$ for all tests.

Next, we investigate how the method performs for non-stationary forcings. We again use the SPOD modes from data generated by a stochastic, statistically stationary forcing, but test the resulting model on a variety of non-stationary forcings. The forcing is active in the same spatial region as before, and is given in this region by

$$f(x, t) = f(t) \sqrt{\exp[-(x - \bar{x})^2]}, \quad (4.2.5)$$

The forcing is again strongest at $\bar{x} = -10$. We test four choices of the function $f(t)$: a periodic function, a pulse function, a quasiperiodic function, and a series of pulses, steps, and quasiperiodic functions. The initial condition is zero in each case.

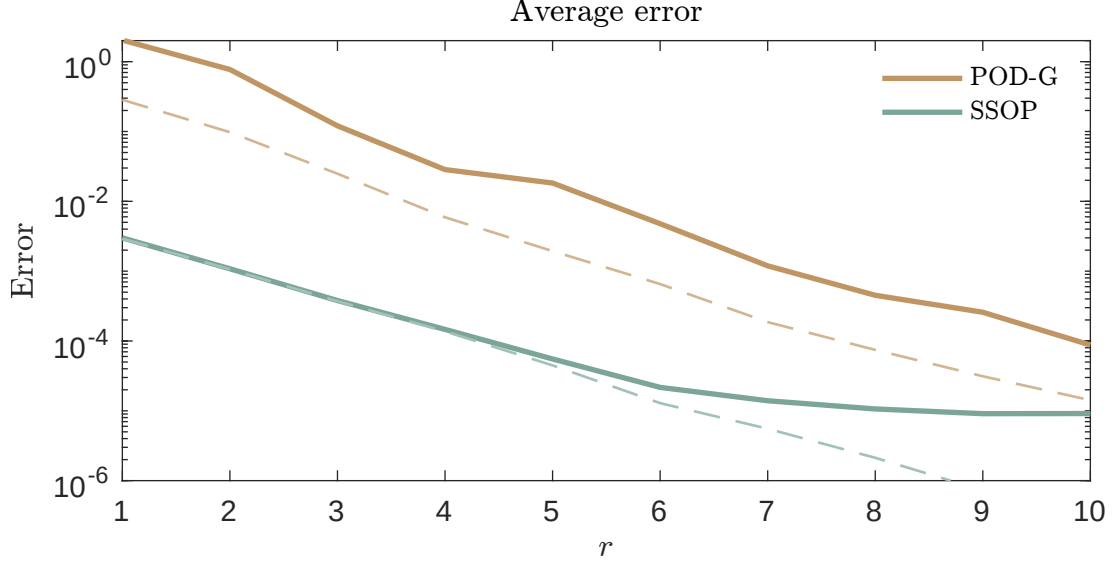


Figure 4.5: Error, averaged over time, for SSOP and for POD-G as a function of r , the number of modes used in the ROMs. The dashed curves again represent the projection error onto the POD and SPOD modes.

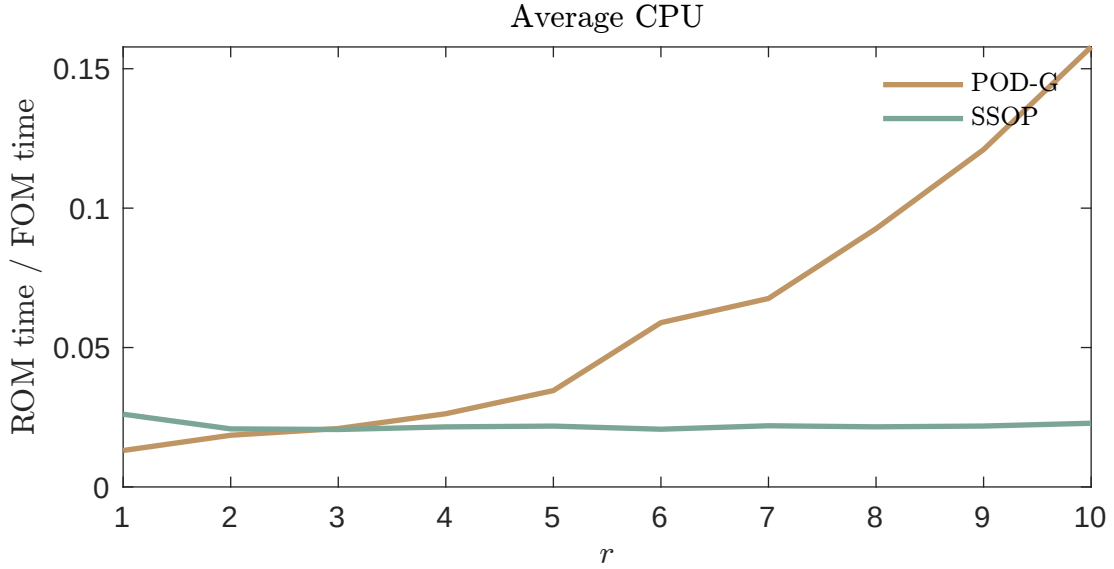


Figure 4.6: CPU time for SSOP in comparison to that of POD-G as a function of the number of modes used in the ROMs.

These functions, the relative errors of SSOP and POD-G, and the respective projection errors are shown in Figure 4.7. The salient takeaways are the same as in the stochastic forcing case: SSOP achieves substantially lower error than POD-G and than the POD projection error, and nearly achieves the SPOD projection error. With the periodic (a) and quasiperiodic (c) forcings, the solution at the beginning of the interval is different than the

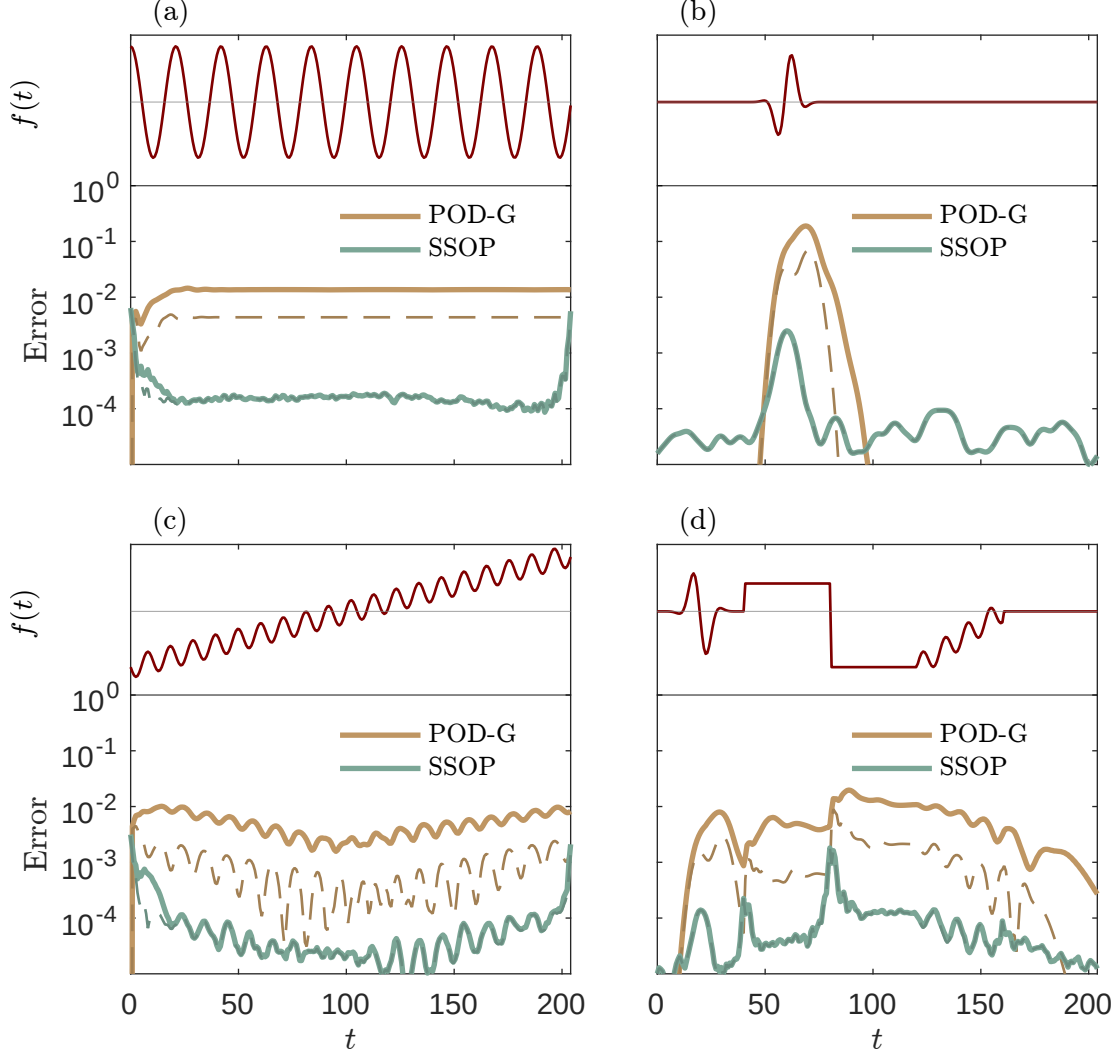


Figure 4.7: The error as a function of time for a variety of non-stochastic forcings for both SSOP and POD-G with $r = 5$ modes, along with the respective projection errors (dashed). The SSOP performance is robust for these out-of-sample forcings.

solution at the end of the interval, so the SPOD projection error, and thus SSOP, is high at the beginning and end of the interval. Conversely, because the pulse (b) and series (d) forcing are both zero for long enough at the end of the interval for the solution to decay, the error of the SPOD projection and SSOP remains low at the beginning and end of the interval. Note that because the error is normalized by the average square norm of the solution over the interval, the pulse error is substantially higher than the others. We also note that none of the forcings here (or in the stochastic forcing case) meet assumption (ii), but this has little effect on the accuracy of the method.

Finally, we test the method on a range of Ginzburg-Landau systems by varying μ_0 over the range $\mu_0 \in [0.079, 0.499]$ in increments of 0.03. The behavior of the Ginzburg-Landau

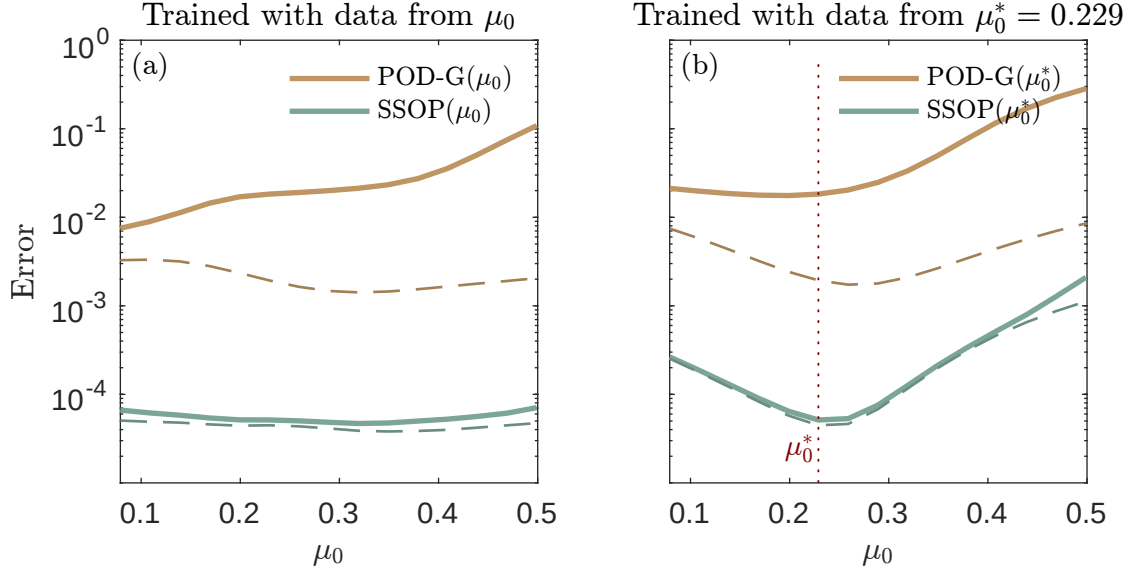


Figure 4.8: The methods tested on a range of Ginzburg-Landau systems parameterized by μ_0 with $r = 5$. In (a), the model is trained using data from the same GL system. In (b), the models are trained using data from $\mu_0^* = 0.229$. The dashed curves correspond to the projection error for the modes from the training system.

system changes significantly over this range. At the lower end, the system is linearly stable, i.e., all of the eigenvalues of $\mathbf{A}$ are in the stable half plane, and the behavior is entirely modal, i.e., the eigenvectors are nearly orthogonal. At $\mu_0 = 0.229$, the value we have used in the tests so far, the system is slightly non-modal — one measure of this is the optimal transient growth [133, 116], which is nearly 5. At $\mu_0 = 0.379$, the system is linearly stable but is strongly non-modal, with an optimal transient growth of nearly 200. At $\mu_0 > 0.397$, the system is linearly unstable, and the dynamics feature a competition between the growth caused by the linear instability and the damping of the nonlinear term.

We perform two tests over this range of μ_0 . First, we build an SSOP model with SPOD modes from training data at each μ_0 with the stochastic forcing (4.2.2), then test the model at the same μ_0 with different realizations from the same distribution of forcings. This test verifies that the method can perform well across a range of Ginzburg-Landau systems. Second, we take SPOD modes from $\mu_0^* = 0.229$, then use them to build an SSOP model for each μ_0 . This second test is designed to see if the method can be used to take data from one system and use it to predict behavior of a different one.

Figure 4.8(a) shows the results of the first test using $r = 5$ modes — the average relative error over both time and the 30 trajectories is shown as a function of μ_0 . Again, SSOP substantially outperforms POD-G and the projection error of the POD basis, and nearly achieves its own projection error. The SSOP error is remarkably uniform over the range of

parameters, in contrast to the POD-G error. The SSOP performance past the bifurcation at $\mu_0 = 0.397$ shows the efficacy of the strategy described in Section 4.1.1 of diverting some of the linear term to the nonlinear term.

Figure 4.8(b) shows the second test with $r = 5$ modes. Again, the relative error averaged over time and trajectories is shown, but, in contrast to (a), the model is built using modes educed using data from a system with $\mu_0^* = 0.229$. The projection errors (dashed) are also computed using the modes from μ_0^* . That the projection error is substantially lower near μ_0^* than at other values indicates that the SPOD modes change significantly as μ_0 changes. Despite this, the model error results (solid) show SSOP can generate accurate predictions for new systems. In fact, the SSOP model built using data from a different Ginzburg-Landau system ((b), solid green) produces lower error in this example than the projection error onto the POD modes of the test system ((a), dashed brown). We view this as a strong result showcasing the robustness of the proposed method.

4.2.2 Quadratic Ginzburg-Landau system

Here, we test the method described in Subsection 4.1.2.2 for handling quadratic nonlinearities. We do this by replacing the standard nonlinear term with a Burgers'-type (quadratic) nonlinearity and find that most of the triadic interactions have a negligible impact. We exclude roughly 99% of the interactions, which leads to substantial speedup with no meaningful increase in the error relative to retaining all of them. Even with excluding these interactions, however, the method is slower than using DEIM on the same problem. Our outlook is that using the triadic-interaction-sum method of handling nonlinearities will be more effective than DEIM for more complex problems based on the findings in Ref. [125].

The quadratically nonlinear Ginzburg-Landau equation is

$$\frac{\partial}{\partial t} q(x, t) = \left[-\nu \frac{\partial}{\partial x} + \gamma \frac{\partial^2}{\partial x^2} + \mu_0 - c_\mu^2 + \frac{\mu_2}{2} x^2 \right] q(x, t) + \kappa q(x, t) \frac{\partial}{\partial x} q(x, t). \quad (4.2.6)$$

We set $\kappa = 5$ and otherwise use the same parameters from before (including $\mu_0 = 0.229$).

We generate data following the same procedure as before — using the stochastic forcing (4.2.2) to generate 3000 time steps of training data with $\Delta t = 0.8$, forming this data into 44 overlapping blocks of length $N_\omega = 256$, and computing the SPOD modes and operators. We then again use different realizations from the same forcing distribution to generate the data used for testing.

The criterion (described in Section 4.1.2.2) for including the triadic interaction between mode m at ω_l and mode n at frequency $\omega_i = \omega_k - \omega_l$ is $\|\mathbf{n}_{klmn}\|_x^2 \lambda_{lm} \lambda_{in} \geq \epsilon$ for a user-defined

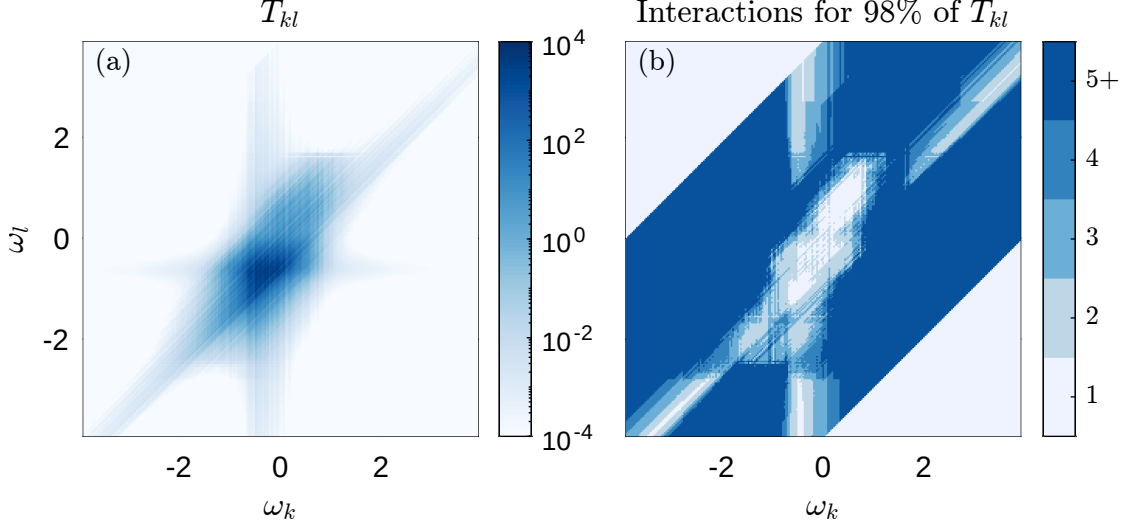


Figure 4.9: The impact of ω_l on ω_k as defined by (4.2.7) via triadic interaction with $\omega_i = \omega_k - \omega_l$. (a) shows T_{kl} , a proxy for the strength of all interactions between the SPOD modes at ω_l with those at ω_i . (b) shows how many of these interactions are needed to account for 98% of T_{kl} .

value of ϵ . We define the matrix $\mathbf{T} \in \mathbb{R}^{N_\omega \times N_\omega}$ as

$$T_{kl} = \sum_{m=1}^{r_l} \sum_{n=1}^{r_i} \|\mathbf{n}_{klmn}\|_x^2 \lambda_{lm} \lambda_{in}. \quad (4.2.7)$$

T_{kl} is a proxy for the total impact of ω_l on ω_k via triadic interactions. We plot T_{kl} for all pairs of frequencies in Figure 4.9(a). In Figure 4.9(b), we show the number of interactions at a given frequency pair ω_k, ω_l that is required to account for 98% of T_{kl} . We see that for many of the frequency pairs, the total impact at a frequency pair is dominated by just a few mode pairs within the two frequencies. Noting the overlap in the high-value region of (a) and the low-value region of (b), we see that this is particularly true at the more energetic mode pairs. The implication of (a) and (b) together is that the great majority of frequencies may be discarded without having a meaningful effect on the approximation of the nonlinearity.

Figure 4.10(a) shows the number of interactions at each frequency pair with $r = 5$. These values are only a function of r_K — the number at (ω_k, ω_l) is $r_l r_i$, where $\omega_i = \omega_k - \omega_l$. Figure 4.10(b) shows the number at each pair that are retained when the threshold in (4.1.18) is set to $\epsilon = 10^{-1.8}$ (we show this one because it leads to a good balance between accuracy and speed in the results). With this choice of ϵ , 1.7% of the interactions are retained.

We test the accuracy and speed of the method on the 30 trajectories as a function of the number of interactions included. In Figure 4.11(a), we plot the relative error averaged over time and trajectories against the percentage of the total (1384103) interactions. As one

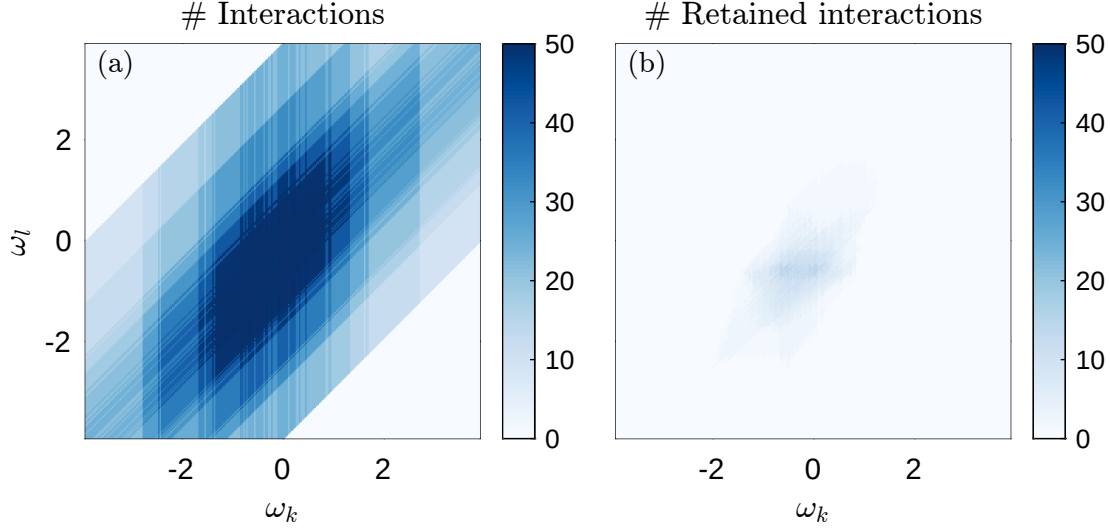


Figure 4.10: (a) The total number of interactions between SPOD modes at ω_l and $\omega_k - \omega_l$ with $r = 5$. (b) The number of these interactions that are retained when $\epsilon = 10^{-1.8}$. Only 1.7% of the interactions in (a) are retained with this threshold.

would expect, increasing the number of retained interactions causes the error to decrease and the CPU time to increase. Figure 4.11(b) shows that the CPU increase is linear in the number of interactions retained, as is predicted by the scaling analysis in Section 4.1.6. The error quickly reaches a plateau as the newly retained interactions become less energetic (the plateau is above the SPOD projection error, though it is quite close).

The red ticks in Figure 4.11 indicate the error and timing results using the DEIM-based approximation of the nonlinearity. For this problem, DEIM is able to achieve lower error at lower CPU time than the triadic-interaction-based approximation, but we do not expect that this is true in general.

4.3 Chapter summary

This chapter extends the SSOP method to nonlinear problems. Three aspects are addressed to handle nonlinearity.

- (i) The linear SSOP formulation is based on the analytic solution to linear systems, however, there is no analogous analytical solution to nonlinear systems. We incorporate nonlinearity into the method by treating it as an additional forcing on the LTI system. This amounts to projecting the following implicit equation for $\mathbf{q}(t)$ onto the SPOD modes,

$$\mathbf{q}(t) = e^{\mathbf{A}t} \mathbf{q}_0 + \int_0^t e^{\mathbf{A}(t-t')} (\mathbf{f}(t') + \mathbf{n}(\mathbf{q}(t'))) dt'. \quad (4.3.1)$$

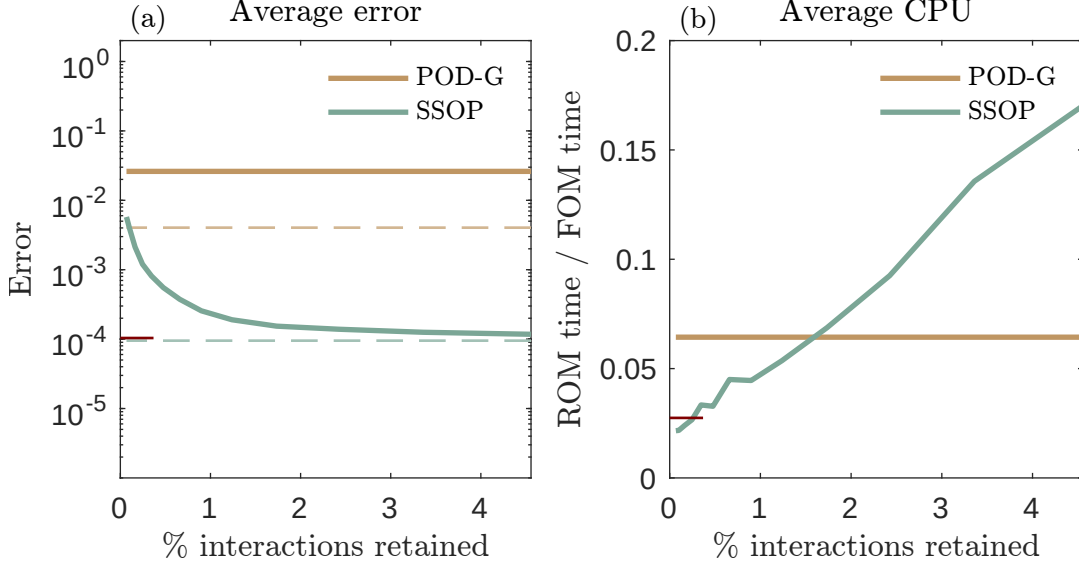


Figure 4.11: The relative error (a) and CPU time (b) for the method with $r = 5$ using the sparse-triadic-interaction approximation of the nonlinearity as a function of the number of triadic interactions retained. The x -axis shows the percentage of the total triadic interactions used. The red ticks indicate results using the DEIM approximation of the nonlinearity.

- (ii) After projection, the nonlinear term must be modeled as a function of the SPOD coefficients to arrive at a closed system for these coefficients. In the case of quadratic nonlinearities, this is done by summing over a subset of the triadic interactions between SPOD modes. In our numerical tests, we find that this set can be vastly smaller ($\sim 1\%$) than the total number of interactions without a major impact on the error. In the case of more general nonlinearities, we use DEIM to approximate the nonlinear term.
- (iii) The result of the above two steps is a system of nonlinear equations that must be solved simultaneously for the SPOD coefficients at all frequencies. We use a fixed-point iteration to solve this system, which we find converges in ~ 10 iterations or fewer for most of the cases we tested. With this solution procedure, the nonlinearity can be thought of as an iterative correction on the linear equations.

The nonlinear version of SSOP is then tested on two Ginzburg-Landau systems, one with the standard nonlinearity and the other with a quadratic nonlinearity. The performance is similar to the linear case: for the same number of modes, SSOP gives roughly two orders of magnitude lower error than POD-Galerkin projection at similar cost. The method also nearly recovers the SPOD encoding.

CHAPTER 5

Preliminary Work on Application to the Navier-Stokes Equations

This chapter describes preliminary work on applying SSOP to the incompressible Navier-Stokes equations (INS). The implementation of the method is not yet complete, but we describe our approach to two issues that arise in the application to this system. The first is that the INS, with the divergence-free constraint, are not naturally in the form of a dynamical system, where there is an equation that gives the time-derivative of the state as a function of itself and time. We instead work with a Leray-projection form of the equations and derive the forms of the linear, nonlinear and forcing terms. Second, we found that the ROM operator approximations described in Chapters 3 and 4 did not give sufficient accuracy. We instead develop time-stepping approximations of these operators, which give more accuracy at the cost of offline time.

The implementation uses the code Nek5000 [95], which is based on the spectral element method [94], as the underlying full-order model. The offline phase of the method is carried out within the Nek5000 environment. To obtain POD and SPOD modes, Nek5000 is used to generate time-series data, Fourier transforms are performed using the FORTRAN library FFTW [36], and a Nek5000-defined inner product is then used to obtain the modes. These modes are used within Nek5000 in subsequent steps: as initial conditions for the linear runs used to construct the transient operator, as described in Section 5.2.2, as forcing for the adjoint runs used to approximate the action of the resolvent, as described in Section 5.2.3, and as inputs to the bilinear operator for the triadic interaction terms. The Nek5000 routines for applying the differential operators and linear solves described in Section 5.1 are used to compute the action of the bilinear operator. The output from the offline phase is a set of matrices and tensors that can then be used in a separate scripting environment for the online phase of SSOP.

5.1 Formulation of the incompressible Navier-Stokes equations

5.1.1 Incompatibility with the standard form

The incompressible Navier-Stokes equations are given by

$$\nabla \cdot \mathbf{u} = 0 \quad (5.1.1a)$$

$$\dot{\mathbf{u}} = -\mathbf{u} \cdot \nabla \mathbf{u} - \nabla p + \frac{1}{Re} \nabla^2 \mathbf{u} + \mathbf{g}(t). \quad (5.1.1b)$$

The starting point for the SSOP method is a governing equation giving the time derivative of the state as a function of it, and of time, i.e., $\dot{\mathbf{q}} = \mathbf{h}(\mathbf{q}, t)$. However, the INS are not in this form as written in (5.1.1). If the state is the velocity field $\mathbf{u}$, then its time derivative is not given by a function of $\mathbf{u}$ alone, but also depends on pressure p . If, instead, the state is the vector $\mathbf{q} = [\mathbf{u}, p]^T$, then (5.1.1) is not of the right form since there is no evolution equation for p .

This issue is present in space-only projection-based methods, like POD-Galerkin projection, but the remedy for those cases is not directly applicable here. Space-only projection-based methods are formulated in terms of inner products between spatial modes $\boldsymbol{\phi}$ and the governing equations. The pressure contribution to the inner product between (5.1.1b) and a mode satisfying $\nabla \cdot \boldsymbol{\phi} = 0$ either is equal to zero or a constant that can be evaluated without retaining pressure as a dynamic variable [51, 109]. SSOP, however, does not involve inner products between modes and the governing equations, so we approach the issue differently.

5.1.2 Leray projection form

We instead work with the following form of the INS,

$$\dot{\mathbf{u}} = \mathcal{P} \left[-\mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{Re} \nabla^2 \mathbf{u} + \mathbf{g}(t) \right] \quad (5.1.2)$$

where $\mathcal{P}$ is the Leray projection operator [72, 31], which returns the solenoidal component of its argument. The INS in this formulation are clearly in the form to which the SSOP method may be applied – the state is the velocity field $\mathbf{u}$, and its time derivative is given straightforwardly in (5.1.2).

A familiar method to apply the Leray projection to a field $\mathbf{w}$ is to solve a Poisson equation with the divergence of $\mathbf{w}$ as the source term, then subtract the gradient of the result from $\mathbf{w}$. This can be written as $\mathcal{P} = (\mathcal{I} - \nabla \Delta^{-1} \nabla \cdot)$, where Δ^{-1} applied to a scalar field f returns

the solution p to the Poisson equation $\nabla^2 p = f$. An alternative expression for the Leray projection is $\mathcal{P} = -(\nabla \times \Delta^{-1} \nabla \times)$, where the Poisson solve is applied to each component separately. We use this latter formulation due to implementational convenience in Nek5000.

Before separating the terms in (5.1.2) into linear and nonlinear parts, as is required to apply SSOP, we first decompose the velocity field into a time-independent baseflow and a fluctuation $\mathbf{u} = \mathbf{U} + \mathbf{u}'$, in order to maximize the portion of the right-hand side given by the linear and constant terms. With this decomposition, (5.1.2) becomes

$$\dot{\mathbf{u}}' = \mathcal{P} \left[-(\mathbf{U} + \mathbf{u}') \cdot \nabla (\mathbf{U} + \mathbf{u}') + \frac{1}{Re} \nabla^2 (\mathbf{U} + \mathbf{u}') + \mathbf{g}(t) \right]. \quad (5.1.3)$$

This results in the following linear, nonlinear, and forcing terms,

$$\mathcal{A}(\mathbf{u}') = \mathcal{P} \left[-\mathbf{U} \cdot \nabla \mathbf{u}' - \mathbf{u}' \cdot \nabla \mathbf{U} + \frac{1}{Re} \nabla^2 \mathbf{u}' \right], \quad (5.1.4a)$$

$$\mathcal{N}(\mathbf{u}') = \mathcal{P} [-\mathbf{u}' \cdot \nabla \mathbf{u}'], \quad (5.1.4b)$$

$$\mathbf{f}(t) = \mathcal{P} \left[-\mathbf{U} \cdot \nabla \mathbf{U} + \frac{1}{Re} \nabla^2 \mathbf{U} + \mathbf{g}(t) \right]. \quad (5.1.4c)$$

These operators are defined for continuous fields, and the full-order model components in (4.1.1) that form the starting point for SSOP are given by discretizations of their continuous analogs above. In particular, the discretized state $\mathbf{q}$ in the FOM is a discretization of the fluctuating part of the velocity field with respect to the mean flow; the matrix $\mathbf{A}$ is a discretization of the linearized Navier-Stokes operator $\mathcal{A}$ defined about the mean flow; the nonlinear term $\mathbf{n}(\mathbf{q})$ is a bilinear operator and may be written $\mathbf{n}(\mathbf{q}) = \mathbf{b}(\mathbf{q}, \mathbf{q})$, and it represents the fluctuating part of the velocity field convecting itself; and the forcing $\mathbf{f}(t)$ includes time-independent component that arises from the mean flow not satisfying the time-independent Navier-Stokes equations (and any constant body forces) and a time-dependent component representing any time-dependent forcing in the system.

5.2 Time-stepping approximation of ROM operators

Initial attempts at applying SSOP to the Navier-Stokes equations have shown that the methods for approximating the ROM operators described in Chapters 3 and 4 do not result in accurate operators for this more challenging case. In this section, we describe more sophisticated methods, based on time stepping the linearized system and using the results to compute approximations of the operators. These are expected to lead to more accurate operators. The drawbacks of these approximations relative to those given in the previous

chapters are 1) that these approximations require access to a linearized FOM, 2) that they require additional simulations to be run, and 3) that they cannot be modified quickly to new parameters in the case of affine parameter dependence.

The intuitive reason that time stepping may be an effective approach for obtaining the operators that arise in the SSOP method is that each element of these operators comes from a space-time inner product between an SPOD mode and a trajectory to the linearized system with a particular initial condition or forcing.

5.2.1 Test case

We show preliminary results for both approximations on two-dimensional flow past a cylinder at Reynolds number $Re = 100$. The mesh contains 1472 elements, each with 36 points, resulting in a state dimension of $N_x = 105984$. The system is integrated forward through 200 vortex-shedding oscillations, which occurs over a time interval of roughly 1200 time units. Time is nondimensionalized by the cylinder diameter and free-stream velocity. SPOD modes are constructed using this data with a window length of 59.6 time units, corresponding to 10 shedding periods, and a 50% overlap between windows.

5.2.2 Transient operator

5.2.2.1 Formulation

In Chapters 3 and 4, we approximated the operator

$$\mathbf{Z}_k = \mathbf{\Psi}_k^* \mathbf{W} \left(\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t} \right)^{-1} \left(\mathbf{I} - e^{\mathbf{A}T} \right) \mathbf{\Phi} \in \mathbb{C}^{r_k \times p_1} \quad (5.2.1)$$

by using the SPOD modes to form a Galerkin-type approximation of $\mathbf{A}$. Here, we detail a means of forming this operator exactly (up to FOM errors) by performing p_1 runs of length T of the linearized FOM.

Equation (3.4.8) implies

$$\left(\mathbf{I} - e^{(\mathbf{A} - i\omega_k \mathbf{I})\Delta t} \right)^{-1} \left(\mathbf{I} - e^{\mathbf{A}T} \right) \mathbf{v}_0 = \text{DFT}_k \left[e^{\mathbf{A}t_{\mathcal{J}}} \mathbf{v}_0 \right]. \quad (5.2.2)$$

That is, the effect of the operator on the left-hand side above applied to a vector $\mathbf{v}_0 \in \mathbb{C}^{N_x}$ is to take the k -th frequency of the DFT of the time series generated by using $\mathbf{v}_0$ as the initial condition to the linear system $\dot{\mathbf{v}} = \mathbf{A}\mathbf{v}$.

Using this equivalence, the l -th column of $\mathbf{Z}_k$ is $\mathbf{\Psi}_k^* \mathbf{W} \hat{\phi}_k^l$, where $\hat{\phi}_k^l = \text{DFT}_k[e^{\mathbf{A}t_{\mathcal{J}}} \phi^l]$ is the k -th component of the DFT of the time series created by using the l -th column of $\mathbf{\Phi}$ as

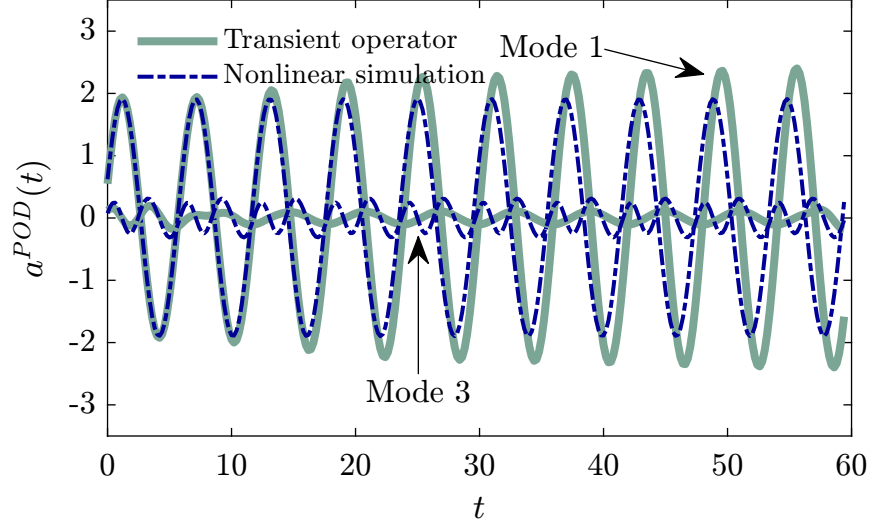


Figure 5.1: Comparison of a nonlinear trajectory and the linear SSOP prediction using the transient operator. The trajectory is projected onto the first and third POD modes. Results for the second mode are qualitatively similar to those for the first.

an initial condition to the unforced linear system. Thus, by initializing this linear system with each column of Φ , and taking the DFT of each result, $\mathbf{Z}_k$ can be computed for all k up to the time-stepping errors in integrating the linear system. Each of the p_1 linear runs required is of length T , so the cost of this approach is the same as integrating the system for a duration of $p_1 T$.

5.2.2.2 Preliminary results

It is difficult to test the accuracy of this approximation directly, though since the only source of error is the error involved in integrating the linear system, we expect the approximation to be accurate. Instead, we test its ability to serve as a model for the fully nonlinear system. We take an initial condition $\mathbf{q}_0$ from the cylinder flow case described above, and compute SPOD coefficients using only the transient term as

$$\mathbf{a}_k = \mathbf{Z}_k \Phi^* \mathbf{W} \mathbf{q}_0. \quad (5.2.3)$$

Doing so ignores the effect of the nonlinear term, as well as the (weak) effect of the steady forcing that results from the mean flow as given in (5.1.4c), the latter of which is quite weak. The resulting SPOD coefficients are not expected to give a quantitatively accurate model, and comparing the decoded trajectory from these coefficients to the FOM tests the importance of nonlinearity to the dynamics.

Figure 5.1, shows this comparison, projecting the results onto the first and third POD

mode for visualization (mode 2 is very similar to mode 1). The trajectory chosen is unforced and is on the attractor, and is thus periodic. Notably, the two are qualitatively fairly similar, especially for the first mode. This indicates that, while important for smaller structures, nonlinearity has only a minor influence on the large-scale dynamics of the flow. Nevertheless, there are important differences between the two trajectories. First, the first mode grows over the interval in the linear case, indicating that $\mathbf{A}$ is slightly unstable. Nonlinearity therefore must remove energy from this oscillation. Second, the frequency in the linear dynamics is slightly lower than that of the fully nonlinear dynamics. Finally, the third mode in the nonlinear trajectory oscillates with roughly twice the frequency relative to linear prediction and to the first mode. This is due to triadic interactions, which are not accounted for in the linear model.

It is expected that adding nonlinearity to the model will account for these differences. To do so, the triadic interactions must be multiplied by the resolvent operator, and we discuss a computational approach for this next.

5.2.3 Resolvent operator

In Chapters 3 and 4, we approximated the action of the matrix $\Psi_k^* \mathbf{W} \mathbf{R}_k$ on a vector $\mathbf{b}$ by computing the action of the inverse of the resolvent on the data matrix used to construct the SPOD modes. This resulting approximation of the action of the resolvent was exact when applied to a certain subspace. Here, we detail a means of obtaining $\Psi_k^* \mathbf{W} \mathbf{R}_k$ on a vector $\mathbf{b}$ based on Ref. [28].

In the precomputations for SSOP, this matrix is applied to the columns of the forcing matrix $\mathbf{B}$ and to the triadic interactions that couple to frequency ω_k . The m -th element of the resulting vector is given by the inner product

$$(\Psi_k^* \mathbf{W} \mathbf{R}_k \mathbf{b})_m = \langle \mathbf{R}_k \mathbf{b}, \psi_{km} \rangle_x. \quad (5.2.4)$$

By definition, the right-hand side of the above equation can be written with the adjoint of the resolvent as

$$\langle \mathbf{R}_k \mathbf{b}, \psi_{km} \rangle_x = \langle \mathbf{b}, \mathbf{R}_k^\dagger \psi_{km} \rangle_x. \quad (5.2.5)$$

In practice, it is likely that the number of vectors $\mathbf{b}$ for which this inner product must be computed is greater than r_k , the number of SPOD modes at the k -th frequency. Therefore, to obtain the necessary inner products, and thus the action of $\Psi_k^* \mathbf{W} \mathbf{R}_k$ on the necessary vectors, it is more efficient to compute $\mathbf{R}_k^\dagger \psi_{km}$ for each m and then compute the inner products with the various vectors $\mathbf{b}$ than to compute the action of the resolvent on each $\mathbf{b}$.

The adjoint of the resolvent is given by $\mathbf{R}_k^\dagger = (-i\omega_k \mathbf{I} - \mathbf{A}^\dagger)^{-1}$. The basic idea behind the time-stepping approximation is that the steady-state response at frequency ω_k to the system

$$\dot{\mathbf{q}} = \mathbf{A}^\dagger \mathbf{q} + \boldsymbol{\psi} e^{-i\omega_k t} \quad (5.2.6)$$

is given by $\mathbf{R}_k^\dagger \boldsymbol{\psi}$ [28].

This holds only for stable systems, i.e., ones for which the eigenvalues of $\mathbf{A}$ all have negative real part. This potential issue can usually be overcome; the linearized Navier-Stokes equations are marginally stable for many flows [126], and for these marginally stable systems, a small multiple of the identity can be subtracted from $\mathbf{A}$ without having a large impact on the modes [60, 139]. We assume that this stabilization has already been applied.

A naive approach would involve running an adjoint simulation for each SPOD at all frequencies separately, a total of $N_\omega r$ runs. Since the system is linear, however, all frequencies can be superimposed in a single run, and with this strategy, only $\max_k r_k$ runs are required. The run used to obtain $\mathbf{R}_k^\dagger \boldsymbol{\psi}_{km}$ for all k for a given m is

$$\mathbf{q}^m(0) = \mathbf{0}, \quad (5.2.7a)$$

$$\dot{\mathbf{q}}^m = \mathbf{A}^\dagger \mathbf{q}^m + \sum_k \boldsymbol{\psi}_{km} e^{-i\omega_k t} \mathbf{1}_{\{r_k \geq m\}}. \quad (5.2.7b)$$

This system is integrated for as many periods as are required for the transient to decay, which can be determined *a priori* based on the eigenvalue of $\mathbf{A}$ with the largest real part. After this decay, the DFT of the result is taken, and we define

$$\boldsymbol{\chi}_{km} = \text{DFT}_k[\mathbf{q}_{\mathcal{J}}^m], \quad (5.2.8)$$

where $\mathbf{q}_{\mathcal{J}}^m$ is the N_ω -time-step-long trajectory from $t \in [sT, (s+1)T)$ for some integer s such that the transient has decayed by time sT . The vectors $\boldsymbol{\chi}_{km}$ are the desired approximations of $\mathbf{R}_k^\dagger \boldsymbol{\psi}_{km}$. These vectors $\boldsymbol{\chi}_{km} \in \mathbb{C}^{N_x}$ are the time-stepping approximations of $\mathbf{R}_k^\dagger \boldsymbol{\psi}_{km}$, and the desired inner products are given by

$$\langle \mathbf{R}_k \mathbf{b}, \boldsymbol{\psi}_{km} \rangle_x = \langle \mathbf{b}, \boldsymbol{\chi}_{km} \rangle_x. \quad (5.2.9)$$

5.2.3.1 Preliminary results

The result of this approximation can be tested more directly than those of the transient operator approximation: in this case, we can compute a residual. The vector $\boldsymbol{\chi}_{km}$ is an approximate solution to the linear system $\mathbf{L}_k^\dagger \mathbf{x} = \boldsymbol{\psi}_{km}$, where $\mathbf{L}_k^\dagger = -i\omega_k \mathbf{I} - \mathbf{A}^\dagger$. The error

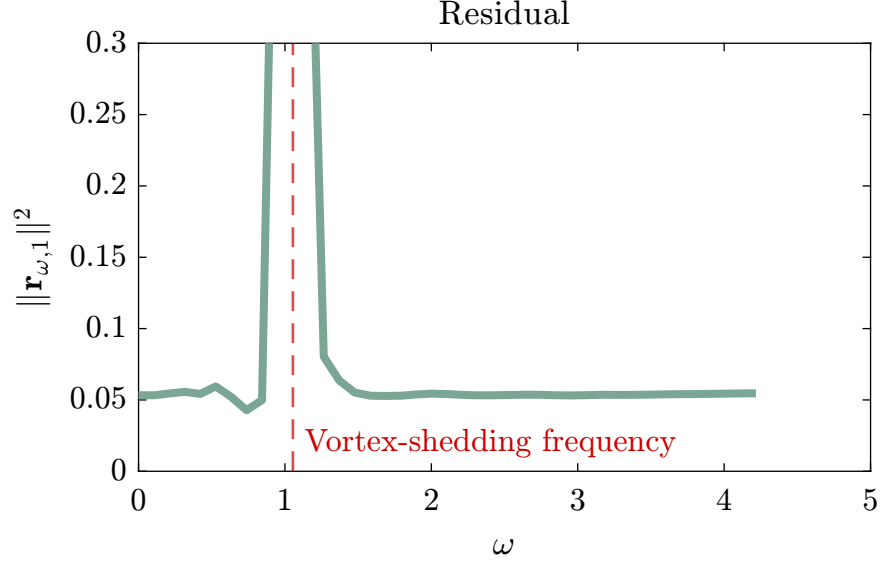


Figure 5.2: Residual for the time-stepping approximation of the resolvent adjoint applied to the first SPOD mode, as a function of frequency. The spike at the vortex-shedding frequency goes to a value of roughly 80.

is given by $\mathbf{e}_{km} = \boldsymbol{\chi}_{km} - \mathbf{R}_k^\dagger \boldsymbol{\psi}_{km}$, and the residual is given by

$$\mathbf{r}_{km} = \mathbf{L}_k^\dagger \boldsymbol{\chi}_{km} - \boldsymbol{\psi}_{km}. \quad (5.2.10)$$

While computing the error involves computing the action of the operator we are attempting to approximate, computing the residual is feasible — it only involves computing the action of $\mathbf{A}^\dagger$, which can be obtained by taking a few time steps of the adjoint system and using a finite difference approximation of the time derivative. This residual can serve as a proxy for the error, and the two are related by $\mathbf{r}_{km} = \mathbf{L}_k^\dagger \mathbf{e}_{km}$.

In Figure 5.2, we show the residual for the same cylinder flow test case described in Section 5.2.2. This figure plots the square norm of this residual for the resolvent adjoint applied to the first SPOD mode as a function of frequency. For most frequencies, the residual is fairly low with a spike at the peak frequency.

It is difficult to draw conclusions about the error from this plot alone — the error bound based on the condition number of $\mathbf{L}_k^\dagger$ is both hard to estimate and likely far too pessimistic. We conducted a similar test for the Ginzburg-Landau system described in Chapter 3, and found that the residual from the time-stepping approximation of the resolvent also spiked at the fundamental frequency for that problem. In that case, which is small enough to compute the error directly, we found that the error was nearly constant across frequencies. Moreover, this error was comparable (and slightly smaller) to the residual at the low-gain frequencies.

If these observations hold for the Navier-Stokes case, it would indicate that the square norm of the error is smaller than 0.05 for all frequencies.

5.3 Steps remaining for a complete model

The Nek5000 implementation of the model currently computes SPOD modes and the operator approximations described above. The computation of the triadic interactions of the form $\mathbf{b}(\psi_{km}, \psi_{ln})$ have also been implemented. However, including these results in the model, by using the inner products with the resolvent adjoint applied to the SPOD modes, as described in Section 5.2.3, does not yet give accurate results.

There are two candidate explanations for this. The first is that the action of the resolvent operator is still insufficiently accurate. While the spike in the residual at the vortex-shedding frequency shown in Figure 5.2 is not necessarily indicative of large error, as argued above, large error at this frequency is certainly possible. It is also possible that a value of 0.05 for the square norm of the residual is too large at all frequencies. Errors in computing the action of the resolvent have a direct impact on the way that nonlinearity is handled in the model, since terms representing nonlinearity in the model come from inner products of SPOD modes and the action of the resolvent on triadic interactions. A second possibility is that the triadic interactions are not being handled correctly, either in the offline calculations or in the online phase. The bookkeeping involved in computing the correct interactions in the offline phase, and using the correct ones in the online phase, is nontrivial, and errors are hard to detect directly.

We do not expect that resolving the issues described above will require fundamental changes to the methods or an overhaul of the implementation. The time stepping approximation of the resolvent has been used successfully in much more challenging contexts [28], and any issues in the implementation of the triadic interactions will likely only require minor tweaks to the code. Once the fully nonlinear model is working, the natural next step is to test it against POD-Galerkin projection on cylinder flow, which is already implemented.

CHAPTER 6

Conclusion

This brief chapter summarizes the thesis and outlines a few interesting directions for future research.

6.1 Summary

The through line of this thesis is space-time methods for model reduction, and this is explored from two angles. The first is an investigation into the connection between space-time POD and the singular value decomposition of the Hankel matrix that is used in many time-delay embedding methods. The second, which is the main contribution of the thesis, is the development of a new space-time model reduction method based on SPOD modes.

Chapter 2 demonstrates that the singular modes of the Hankel matrix are a discrete approximation of the continuous space-time POD modes. This connection is made by recognizing that the Hankel matrix multiplied by its conjugate transpose produces an approximation of a space-time correlation matrix, and the eigendecomposition of this matrix gives discrete space-time POD modes. We are able to gain useful insights into the Hankel modes by analyzing this correlation matrix rather than the Hankel matrix itself. This analysis makes clear the sense in which the modes are optimal: they are optimal if the weight matrix is uniform, which is often not desirable due to a variety of reasons including grid non-uniformity, the need to weight certain flow variables differently, or preference for certain regions in space. We derive a formula for the case of a non-uniform weight. The analysis of the correlation matrix also allows us to distinguish between the meaning of the time step along rows vs. along columns: the time step along columns corresponds to the temporal discretization of the space-time integral eigenvalue problem (2.3.6), while the time step along rows corresponds to the time between successive flow realizations used to approximate the correlation matrix. Viewed in this way, it is clear that these time steps need not be the same. We leverage this insight to ease computation, omitting many of the columns of the Hankel matrix then taking the SVD of this new data matrix with uncorrelated columns. This produces modes with the

same meaning at lower cost. We also show that two popular versions of POD, space-only POD and spectral POD, are recovered in the limits that the columns of the Hankel matrix represent short and long times, respectively. Finally, we improve the Hankel modes by fully exploiting the assumption of ergodicity to form a more accurate correlation matrix with the same data. This correlation matrix has Toeplitz structure owing to the fact that temporal correlations should only depend on the time separation, not on the times explicitly. We show that, especially if data is limited, the modes obtained as the eigenvectors of this Toeplitz matrix can be substantially more accurate than the Hankel modes.

In Chapters 3 and 4, we develop a novel space-time ROM, called spectral solution operator projection, based on spectral POD modes. One important motivation for the approach is that space-time encodings allow for a substantially more accurate representation of a trajectory than do space-only encodings, and that among space-time encodings, SPOD modes are near optimal. The objective of the method, therefore, is to solve for the SPOD encoding quickly and accurately.

The conceptual starting point of the method in the linear case is the familiar fundamental solution to an LTI system. This is viewed as a solution operator acting on the initial condition and forcing that returns the corresponding trajectory. The method obtains the SPOD coefficients by taking space-time inner products between this solution operator and the SPOD modes, which is equivalent to projecting the operator onto the SPOD modes. The benefit of this approach is that if these inner products are done exactly, there is no error in obtaining the SPOD coefficients, meaning the only error in the method is representation error. While computing the inner products exactly would result in poor online scaling, we show that with certain approximations and precomputations, the inner products can be evaluated both quickly and accurately.

Nonlinearity is accommodated within this solution operator projection framework by treating the nonlinearity as an additional forcing on the linear system. Crucially, though, this forcing depends on the solution itself. This results in a system of nonlinear equations to be solved simultaneously for the space-time encoding. In the majority of cases, we solve this system using a fixed-point iteration method. This solution procedure may be viewed as an iterative correction to the linear solution: the first iteration is the linear solution, and at each subsequent iteration, the forcing to the linear system is computed by evaluating the nonlinear term using the solution from the previous iteration. To evaluate the nonlinear term, we use a hyper-reduction technique for non-quadratic nonlinearities, and use a greatly sparsified set of triadic interactions in the case of quadratic nonlinearities.

In the linear case, we demonstrated the method on a linearized Ginzburg-Landau system, and a two-dimensional advection diffusion problem for a cavity flow. For the nonlinear

case, we demonstrated the method on two Ginzburg-Landau systems, one with the standard cubic nonlinearity, and one with a quadratic nonlinearity. The method, in all cases, delivered much lower error than the benchmarks at similar cost. Specifically, the method was able to recover the space-time encoding, i.e., the SPOD coefficients, with high accuracy. Since this encoding affords a much more accurate representation of the trajectory than any space-only encoding, the method was able to achieve two orders of magnitude lower error than even the projection onto POD modes, which serves as a loose lower bound for the error for space-only methods (based on linear encodings). We also found that the method generalized well to out-of-sample conditions; it was tested on forcings that were qualitatively different from those in the training data, as well as Ginzburg-Landau systems with different parameters. In all cases, the method achieved substantially (two-orders of magnitude) lower error than the benchmarks.

In Chapter 5, we describe preliminary work toward applying the method to the incompressible Navier-Stokes equation within Nek5000. Specifically, we address two challenges with the application to this system. The first is that the incompressible Navier-Stokes, in their standard form, are a differential-algebraic system. We instead work with a Leray projection formulation of the equations, and derive the discrete full-order model operators for this formulation. Second, we describe time-stepping approximations for the transient and steady-state operators in the ROM. While these approximations are more time-consuming offline, they are expected to provide better accuracy for large-scale problems than the approximations given in Chapters 3 and 4. We close Chapter 5 with a discussion of the remaining steps to an implementation of the model within Nek5000.

A number of drawbacks of SSOP, and the solution operator projection approach more broadly, bear mentioning. First, the initial step in the method is to take the Fourier transform of the forcing, meaning that if the forcing is not known on the entire interval before commencing the online stage, the method cannot be applied. Second, the method is bound to the time interval $[0, T]$, which is set by the block length used in calculating the SPOD modes. If longer-term forecasts are desired, the method may be repeated using the final (or near the final, as the error increases at the end of the interval) state as the new initial condition. This is somewhat cumbersome in comparison to space-only methods, which preserve the Markovian property of the governing equations. Perhaps the most serious drawback for the application to complex problems is that the method requires careful precomputation of the operators that define it, as discussed in Chapter 5.

However, the extra precomputation cost leads to a key benefit of the method. The operators that define SSOP are, in a sense, ‘doing more’ than the operators that define a space-only ROM: they contain information about the dynamics that result from the linearized

right-hand side, whereas the analogous space-only operators contain information merely about the right-hand side itself. Therefore, it is natural that the precomputation steps in SSOP are more costly than those in most space-only ROMs. The benefit of containing this information within the precomputed operators is that, as we show in the results, it enables accurate recovery of the SPOD encoding coefficients. Indeed, the two key aspects of the method, as we view them, are:

- (i) Achieving an efficient trajectory encoding using SPOD modes, which is optimal among all linear space-time encodings in the limit $T \rightarrow \infty$.
- (ii) Solving accurately for the encoding using the solution operator projection approach.

6.2 Future work

Here we outline four interesting directions for future work, the first and third of which are currently underway.

- The most immediate direction for future work is to continue the development, described in Chapter 5, on the application of the SSOP method to the Navier-Stokes equations in Nek5000. Preliminary results on the time-stepping approximations suggest that the challenges encountered so far related to the operator approximations are surmountable. A working implementation within Nek5000 could also be made available to the community, which may be useful for both practitioners of Nek5000 and researchers in model reduction.
- With a working implementation in Nek5000, a natural next step is to apply the method to more complex problems. An interesting first test beyond the cylinder case described in Chapter 5 is the laminar flow past an airfoil. This is more challenging, and has served as a benchmark for control studies [139, 62], making it well suited for testing SSOP as a model for control design. More investigation is needed to determine how the method will perform on complex problems such as turbulent flows. While we do not foresee any fundamental barriers to obtaining sufficiently accurate operator approximation via time-stepping, the turbulent case may require closure modeling to account for the effect of the truncated modes.
- The SSOP method may be viewed as a special case of a more general solution operator projection framework: with a chosen space-time basis, one can construct a ROM by projecting the same solution operator onto the elements of the basis. Nonlinearity can

be handled in the same way, as an additional forcing on the system that depends on the solution, and the fixed-point iteration solution procedure can also be used to solve the system that results. While unlike the SPOD case, it is not possible in general to derive analytic expressions for the ROM operators that result from the projection onto a chosen space-time basis, time stepping approximations of the operators remove the requirement for analytic expressions. Using the solution operator projection framework with a different space-time basis may be useful for shorter intervals, cyclostationary systems, or for non-statistically stationary systems more broadly.

- The SSOP method makes use of a decomposition of the right-hand side of the governing equations into a linear part and a nonlinear part, and the response of the linear part is obtained with the analytic solution to linear systems, with the nonlinearity treated as an iterative correction. One would expect this to work best when the linear solution closely resembles the fully nonlinear one, both in that the nonlinear system would then be easier to solve, and in that the solution to the nonlinear system would be closer to the true SPOD coefficients. The decomposition into linear and nonlinear parts is not unique, with the only requirement being that their sum is the full right-hand side of the governing equations. In fact the linear operator can be modified to any $\hat{\mathbf{A}}$ so long as the nonlinear part is modified to $\mathbf{n} + (\mathbf{A} - \hat{\mathbf{A}})$. This raises the question of how to choose the operator such that the linear dynamics most closely resemble the fully nonlinear ones. One clear approach is to obtain $\hat{\mathbf{A}}$ from data, e.g., using dynamic mode decomposition or a similar regression method. Another benefit of this approach is that it is a step towards a non-intrusive version of SSOP.

BIBLIOGRAPHY

- [1] D. G. Anderson. Iterative procedures for nonlinear integral equations. *J. ACM*, 12(4):547–560, Oct. 1965.
- [2] H. Arbabi and I. Mezić. Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the koopman operator. *SIAM J. Appl. Dyn. Syst.*, 16(4):2096–2126, 2017.
- [3] N. Aubry. On the hidden beauty of the proper orthogonal decomposition. *Theor. Comput. Fluid Dyn.*, 2(5-6):339–352, Aug. 1991.
- [4] N. Aubry, P. Holmes, J. L. Lumley, and E. Stone. The dynamics of coherent structures in the wall region of a turbulent boundary layer. *J. Fluid Mech.*, 192:115–173, July 1988.
- [5] S. Bagheri, D. S. Henningson, J. Hoepffner, and P. J. Schmid. Input-output analysis and control design applied to a linear model of spatially developing flows. *Appl. Mech. Rev.*, 62(2), Feb. 2009.
- [6] J. Barnett and C. Farhat. Quadratic approximation manifold for mitigating the kolmogorov barrier in nonlinear projection-based model order reduction. *J. Comput. Phys.*, 464:111348, Sept. 2022.
- [7] S. Beneddine, D. Sipp, A. Arnault, J. Dandois, and L. Lesshafft. Conditions for validity of mean flow stability analysis. *J. Fluid Mech.*, 798:485–504, June 2016.
- [8] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Rev.*, 57(4):483–531, Jan. 2015.
- [9] M. Bergmann, C.-H. Bruneau, and A. Iollo. Enablers for robust pod models. *J. Comput. Phys.*, 228(2):516–538, Feb. 2009.
- [10] H. Bijl, D. Lucor, S. Mishra, and C. Schwab. *Uncertainty Quantification in Computational Fluid Dynamics*. Springer International Publishing, 2013.
- [11] E. Bozzo, R. Carniel, and D. Fasino. Relationship between singular spectrum analysis and fourier analysis: Theory and application to the monitoring of volcanic activity. *Comput. Math. Appl.*, 60(3):812–820, 2010.
- [12] D. Broomhead and G. P. King. Extracting qualitative dynamics from experimental data. *Phys. D*, 20(2):217–236, 1986.
- [13] S. Brunton, B. Brunton, J. Proctor, E. Kaiser, and J. Kutz. Chaos as an intermittently forced linear system. *Nat. Commun.*, 8, 08 2016.
- [14] S. L. Brunton, J. H. Tu, I. Bright, and J. N. Kutz. Compressive sensing and low-rank

- libraries for classification of bifurcation regimes in nonlinear dynamical systems. *SIAM J. Appl. Dyn. Syst.*, 13(4):1716–1732, Jan. 2014.
- [15] K. Carlberg, M. Barone, and H. Antil. Galerkin v. least-squares petrov–galerkin projection in nonlinear model reduction. *J. Comput. Phys.*, 330:693–734, Feb. 2017.
 - [16] K. Carlberg, C. Bou-Mosleh, and 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):155–181, Oct. 2010.
 - [17] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem. The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows. *J. Comput. Phys.*, 242:623–647, June 2013.
 - [18] A. V. G. Cavalieri, P. Jordan, and L. Lesshafft. Wave-Packet Models for Jet Dynamics and Sound Radiation. *Appl. Mech. Rev.*, 71(2), 03 2019. 020802.
 - [19] W. Cazemier, R. W. C. P. Verstappen, and A. E. P. Veldman. Proper orthogonal decomposition and low-dimensional models for driven cavity flows. *Phys. Fluids*, 10(7):1685–1699, 1998.
 - [20] M. Cenedese, J. Axås, B. Bäuerlein, K. Avila, and G. Haller. Data-driven modeling and prediction of non-linearizable dynamics via spectral submanifolds. *Nat. Commun.*, 13(1), Feb. 2022.
 - [21] S. Chaturantabut and D. C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM J. Sci. Comput.*, 32(5):2737–2764, Jan. 2010.
 - [22] K. K. Chen and C. W. Rowley. Optimal actuator and sensor placement in the linearised complex Ginzburg–Landau system. *J. Fluid Mech.*, 681:241–260, June 2011.
 - [23] Y. Chen and C. Liu. Data driven robust control of complex ginzburg-landau equations for spatial developing flow. In *2021 IEEE 4th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, page 490–500. IEEE, Nov. 2021.
 - [24] Y. Choi, P. Brown, W. Arrighi, R. Anderson, and K. Huynh. Space–time reduced order model for large-scale linear dynamical systems with application to Boltzmann transport problems. *J. Comput. Phys.*, 424:109845, 2021.
 - [25] Y. Choi and K. Carlberg. Space–time least-squares petrov–galerkin projection for nonlinear model reduction. *SIAM J. Sci. Comput.*, 41(1):A26–A58, 2019.
 - [26] S. T. M. Dawson and B. J. McKeon. On the shape of resolvent modes in wall-bounded turbulence. *J. Fluid Mech.*, 877:682–716, Aug. 2019.
 - [27] D. Dylewsky, E. Kaiser, S. L. Brunton, and J. N. Kutz. Principal component trajectories for modeling spectrally continuous dynamics as forced linear systems. *Phys. Rev. E*, 105:015312, Jan 2022.
 - [28] A. Farghadan, E. Martini, and A. Towne. Scalable resolvent analysis for three-dimensional flows. *J. Comput. Phys.*, 524:113695, mar 2025.
 - [29] B. F. Farrell and P. J. Ioannou. Generalized stability theory part i: autonomous operators. pages 2025–2040, 1996.
 - [30] M. Fazel, T. K. Pong, D. Sun, and P. Tseng. Hankel matrix rank minimization with applications to system identification and realization. *SIAM J. Matrix Anal. Appl.*, 34(3):946–977, 2013.

- [31] C. Foias, O. Manley, R. Rosa, and R. Temam. *Navier-Stokes Equations and Turbulence*. Cambridge University Press, Aug. 2001.
- [32] P. Frame, C. Lin, O. T. Schmidt, and A. Towne. Linear model reduction using spectral proper orthogonal decomposition. *Comput. Methods Appl. Mech. Eng.*, 447:196–215, Dec. 2025.
- [33] P. Frame and A. Towne. Space-time POD and the Hankel matrix. *PLOS ONE*, 18(8):e0289637, Aug. 2023.
- [34] P. Frame and A. Towne. Beyond optimal disturbances: a statistical framework for transient growth. *J. Fluid Mech.*, 983, Mar. 2024.
- [35] P. Frame and A. Towne. Nonlinear space-time model reduction in the frequency domain, 2025.
- [36] M. Frigo and S. G. Johnson. *The Fastest Fourier Transform in the West*. Sept. 1997.
- [37] R. Geelen, L. Balzano, S. Wright, and K. Willcox. Learning physics-based reduced-order models from data using nonlinear manifolds. *Chaos*, 34(3), Mar. 2024.
- [38] R. Geelen, S. Wright, and K. Willcox. Operator inference for non-intrusive model reduction with quadratic manifolds. *Comput. Methods Appl. Mech. Eng.*, 403:115717, Jan. 2023.
- [39] J. F. Gibson, J. Doyne Farmer, M. Casdagli, and S. Eubank. An analytic approach to practical state space reconstruction. *Phys. D*, 57(1):1–30, 1992.
- [40] F. J. Gonzalez and M. Balajewicz. Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems. *arXiv preprint arXiv:1808.01346*, 2018.
- [41] S. Gordeyev and F. Thomas. A temporal proper decomposition (tpod) for closed-loop flow control. *Exp. Fluids*, 54, 03 2013.
- [42] S. Grimberg, C. Farhat, R. Tezaur, and C. Bou-Mosleh. Mesh sampling and weighting for the hyperreduction of nonlinear petrov–galerkin reduced-order models with local reduced-order bases. *Int. J. Numer. Methods Eng.*, 122(7):1846–1874, Jan. 2021.
- [43] C. Guervilly, P. Cardin, and N. Schaeffer. Turbulent convective length scale in planetary cores. *Nature*, 570(7761):368–371, June 2019.
- [44] S. Gupta, J. Paulson, P. Frame, and A. Towne. Statistical modeling of energy amplification of inflow perturbations in boundary layer flows. In *AIAA SCITECH 2026 Forum*. American Institute of Aeronautics and Astronautics, Jan. 2026.
- [45] K. C. Hall, K. Ekici, J. P. Thomas, and E. H. Dowell. Harmonic balance methods applied to computational fluid dynamics problems. *Int. J. Comput. Fluid Dyn.*, 27(2):52–67, Jan. 2013.
- [46] K. C. Hall, J. P. Thomas, and W. S. Clark. Computation of unsteady nonlinear flows in cascades using a harmonic balance technique. *AIAA J.*, 40(5):879–886, May 2002.
- [47] G. Haller and S. Ponsioen. Nonlinear normal modes and spectral submanifolds: existence, uniqueness and use in model reduction. *Nonlinear Dyn.*, 86(3):1493–1534, Aug. 2016.
- [48] D. Hartman and L. K. Mestha. A deep learning framework for model reduction of

- dynamical systems. In *2017 IEEE Conference on Control Technology and Applications (CCTA)*, page 1917–1922. IEEE, Aug. 2017.
- [49] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.
 - [50] C. Hoang, K. Chowdhary, K. Lee, and J. Ray. Projection-based model reduction of dynamical systems using space–time subspace and machine learning. *Comput. Methods Appl. Mech. Eng.*, 389:114341, Feb. 2022.
 - [51] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge Monographs on Mechanics. Cambridge University Press, 2 edition, 2012.
 - [52] H. Hotelling. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.*, 24(6):417–441, Sept. 1933.
 - [53] S. Hovland, J. T. Gravdahl, and K. E. Willcox. Explicit model predictive control for large-scale systems via model reduction. *J. Guid. Control Dyn.*, 31(4):918–926, July 2008.
 - [54] C. Huang and K. Duraisamy. Predictive reduced order modeling of chaotic multi-scale problems using adaptively sampled projections. *J. Comput. Phys.*, 491:112356, Oct. 2023.
 - [55] L. Huisman and S. Weiland. Identification and model predictive control of an industrial glass-feeder. *IFAC Proc. Vol.*, 36(16):1645–1649, Sept. 2003.
 - [56] M. Ilak, S. Bagheri, L. Brandt, C. W. Rowley, and D. S. Henningson. Model reduction of the nonlinear complex ginzburg–landau equation. *SIAM J. Appl. Dyn. Syst.*, 9(4):1284–1302, Jan. 2010.
 - [57] T. Iliescu and Z. Wang. Variational multiscale proper orthogonal decomposition: Navier-stokes equations. *Numer. Methods Partial Differ. Equ.*, 30(2):641–663, Dec. 2013.
 - [58] S. Jain, P. Tiso, J. B. Rutzmoser, and D. J. Rixen. A quadratic manifold for model order reduction of nonlinear structural dynamics. *Comput. Struct.*, 188:80–94, Aug. 2017.
 - [59] A. Jameson. Aerodynamic design via control theory. *J. Sci. Comput.*, 3(3):233–260, Sept. 1988.
 - [60] M. R. Jovanović. *Modeling, analysis, and control of spatially distributed systems*. PhD thesis, University of California, Santa Barbara, 2004.
 - [61] J.-N. Juang and R. S. Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *J. Guid. Control Dyn.*, 8(5):620–627, 1985.
 - [62] J. Jung, R. Bhagwat, and A. Towne. Resolvent-based estimation and control of a laminar airfoil wake. *J. Fluid Mech.*, 1016, Aug. 2025.
 - [63] E. Kaiser, J. N. Kutz, and S. L. Brunton. *Bull. Am. Phys. Soc.*, 2020.
 - [64] I. Kalashnikova, B. van Bloemen Waanders, S. Arunajatesan, and M. Barone. Stabilization of projection-based reduced order models for linear time-invariant systems via optimization-based eigenvalue reassignment. *Comput. Methods Appl. Mech. Eng.*, 272:251–270, Apr. 2014.

- [65] M. Kamb, E. Kaiser, S. L. Brunton, and J. N. Kutz. Time-delay observables for koopman: Theory and applications. *SIAM J. Appl. Dyn. Syst.*, 19(2):886–917, 2020.
- [66] K. Kashima. Nonlinear model reduction by deep autoencoder of noise response data. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, page 5750–5755. IEEE, Dec. 2016.
- [67] B. Kaszás and G. Haller. Globalizing manifold-based reduced models for equations and data. *Nat. Commun.*, 16(1), July 2025.
- [68] Y. Kim, K. Wang, and Y. Choi. Efficient space–time reduced order model for linear dynamical systems in python using less than 120 lines of code. *Mathematics*, 9(14), 2021.
- [69] B. Kramer, B. Peherstorfer, and K. E. Willcox. Learning nonlinear reduced models from data with operator inference. *Annu. Rev. Fluid Mech.*, 56(1):521–548, Jan. 2024.
- [70] A. Laub, M. Heath, C. Paige, and R. Ward. Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms. *IEEE Trans. Autom. Control*, 32(2):115–122, Feb. 1987.
- [71] K. Lee and K. T. Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *J. Comput. Phys.*, 404:108973, Mar. 2020.
- [72] J. Leray. Étude de diverses équations intégrales non linéaires et de quelques problèmes que pose l’hydrodynamique. *J. Math. Pures Appl.*, 12:1–82, 1933.
- [73] X. Li and D. Lasagna. Space-time nonlinear reduced-order modelling for unsteady flows, 2025.
- [74] C. Lin. Model order reduction in the frequency domain via spectral proper orthogonal decomposition. *Master’s thesis, University of Illinois at Urbana-Champaign*, 2019.
- [75] M. Luhar, A. S. Sharma, and B. J. McKeon. On the structure and origin of pressure fluctuations in wall turbulence: predictions based on the resolvent analysis. *J. Fluid Mech.*, 751:38–70, June 2014.
- [76] J. L. LUMLEY. The structure of inhomogeneous turbulent flows. *Atmospheric Turbulence and Radio Wave Propagation*, 1967.
- [77] J. L. Lumley. Stochastic tools in turbulence. 1970.
- [78] E. Martini, A. V. G. Cavalieri, P. Jordan, and L. Lesshafft. Accurate frequency domain identification of ODEs with arbitrary signals, 2019.
- [79] J. R. Martins. Aerodynamic design optimization: Challenges and perspectives. *Comput. Fluids*, 239:105391, May 2022.
- [80] A. R. McClellan, J. Lorenzetti, M. Pavone, and C. Farhat. Projection-based model order reduction for flight dynamics and model predictive control. In *AIAA Scitech 2020 Forum*. American Institute of Aeronautics and Astronautics, Jan. 2020.
- [81] B. J. McKeon. The engine behind (wall) turbulence: perspectives on scale interactions. *J. Fluid Mech.*, 817:P1, 2017.
- [82] B. J. McKeon and A. S. Sharma. A critical-layer framework for turbulent pipe flow. *J. Fluid Mech.*, 658:336–382, July 2010.
- [83] michio. Cfd101: 2d lid driven cavity flow.

- [84] M. Milano and P. Koumoutsakos. Neural network modeling for near wall turbulent flow. *J. Comput. Phys.*, 182(1):1–26, Oct. 2002.
- [85] P. Moin and R. D. Moser. Characteristic-eddy decomposition of turbulence in a channel. *J. Fluid Mech.*, 200:471–509, 1989.
- [86] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Trans. Autom. Control*, 26(1):17–32, Feb. 1981.
- [87] P. Morra, O. Semeraro, D. S. Henningson, and C. Cossu. On the relevance of reynolds stresses in resolvent analyses of turbulent wall-bounded flows. *J. Fluid Mech.*, 867:969–984, Apr. 2019.
- [88] B. R. Noack, K. Afanasiev, M. Morzyński, G. Tadmor, and F. Thiele. A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *J. Fluid Mech.*, 497:335–363, Dec. 2003.
- [89] P. A. S. Nogueira, A. V. G. Cavalieri, P. Jordan, and V. Jaunet. Large-scale streaky structures in turbulent jets. *J. Fluid Mech.*, 873:211–237, June 2019.
- [90] M. Ohlberger and S. Rave. Reduced basis methods: Success, limitations and future challenges. *Proceedings of the Conference Algoritmy*, pages 1–12, 2016.
- [91] S. A. Orszag. On the elimination of aliasing in finite-difference schemes by filtering high-wavenumber components. *J. Atmos. Sci.*, 28(6):1074–1074, Sept. 1971.
- [92] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw. Geometry from a time series. *Phys. Rev. Lett.*, 45:712–716, Sep 1980.
- [93] A. Padovan, B. Vollmer, and D. J. Bodony. Data-driven model reduction via non-intrusive optimization of projection operators and reduced-order dynamics, 2024.
- [94] A. T. Patera. A spectral element method for fluid dynamics: Laminar flow in a channel expansion. *J. Comput. Phys.*, 54(3):468–488, June 1984.
- [95] J. W. L. Paul F. Fischer and S. G. Kerkemeier. nek5000 Web page, 2008. <http://nek5000.mcs.anl.gov>.
- [96] K. Pearson. Liii. on lines and planes of closest fit to systems of points in space. *Philos. Mag.*, 2(11):559–572, Nov. 1901.
- [97] B. Peherstorfer. Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling. *SIAM J. Sci. Comput.*, 42(5):A2803–A2836, Jan. 2020.
- [98] B. Peherstorfer. Breaking the kolmogorov barrier with nonlinear model reduction. *Notices Am. Math. Soc.*, 69(05):1, May 2022.
- [99] B. Peherstorfer, D. Butnaru, K. Willcox, and H.-J. Bungartz. Localized discrete empirical interpolation method. *SIAM J. Sci. Comput.*, 36(1):A168–A192, Jan. 2014.
- [100] B. Peherstorfer and K. Willcox. Data-driven operator inference for nonintrusive projection-based model reduction. *Comput. Methods Appl. Mech. Eng.*, 306:196–215, July 2016.
- [101] E. Pickering, G. Rigas, P. A. S. Nogueira, A. V. G. Cavalieri, O. T. Schmidt, and T. Colonius. Lift-up, kelvin–helmholtz and orr mechanisms in turbulent jets. *J. Fluid Mech.*, 896, May 2020.
- [102] A. Pollard, L. Castillo, L. Danaila, and M. Glauser. *Whither turbulence and big data in the 21st century?* 08 2016.

- [103] M.-L. Rapún and J. M. Vega. Reduced order models based on local pod plus galerkin projection. *J. Comput. Phys.*, 229(8):3046–3063, Apr. 2010.
- [104] D. Rempfer. On low-dimensional galerkin models for fluid flow. *Theor. Comput. Fluid Dyn.*, 14(2):75–88, June 2000.
- [105] E. Rezaian and M. Wei. A hybrid stabilization approach for reduced-order models of compressible flows with shock-vortex interaction. *Int. J. Numer. Methods Eng.*, 121(8):1629–1646, Dec. 2019.
- [106] G. Rigas, D. Sipp, and T. Colonius. Nonlinear input/output analysis: application to boundary layer transition. *J. Fluid Mech.*, 911, 2021.
- [107] F. Rizzi, P. J. Blonigan, E. J. Parish, and K. T. Carlberg. Pressio: Enabling projection-based model reduction for large-scale nonlinear dynamical systems, 2020.
- [108] Z. Rosario, A. Towne, and G. Iaccarino. Dimension reduction for shape design insight. 01 2018.
- [109] C. W. Rowley. *Modeling, Simulation, and Control of Cavity Flow Oscillations*. PhD thesis, California Institute of Technology, 2002.
- [110] C. W. Rowley. Model reduction for fluids, using balanced proper orthogonal decomposition. *Int. J. Bifurcation Chaos*, 15(03):997–1013, Mar. 2005.
- [111] C. W. Rowley, T. Colonius, and R. M. Murray. Model reduction for compressible flows using pod and galerkin projection. *Phys. D*, 189(1):115–129, 2004.
- [112] C. W. Rowley and S. T. Dawson. Model reduction for flow analysis and control. *Annu. Rev. Fluid Mech.*, 49(1):387–417, 2017.
- [113] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, Oct. 1986.
- [114] J. Rutzmoser, D. Rixen, P. Tiso, and S. Jain. Generalization of quadratic manifolds for reduced order modeling of nonlinear structural dynamics. *Comput. Struct.*, 192:196–209, Nov. 2017.
- [115] O. San and T. Iliescu. A stabilized proper orthogonal decomposition reduced-order model for large scale quasigeostrophic ocean circulation. *Adv. Comput. Math.*, 41(5):1289–1319, May 2015.
- [116] P. J. Schmid. Nonmodal stability theory. *Annu. Rev. Fluid Mech.*, 39(1):129–162, Jan. 2007.
- [117] P. J. SCHMID. Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.*, 656:5–28, 2010.
- [118] O. T. Schmidt. Bispectral mode decomposition of nonlinear flows. *Nonlinear Dyn.*, 102(4):2479–2501, Nov. 2020.
- [119] O. T. Schmidt and 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, Apr. 2019.
- [120] O. T. Schmidt, P. J. Schmid, A. Towne, and S. K. Lele. Statistical description of intermittency and rare events via conditional space-time pod: Example of acoustic bursts in turbulent jets. Technical report, Proceedings of the Center for Turbulence Research Summer Program, 2018.

- [121] O. T. Schmidt and A. Towne. An efficient streaming algorithm for spectral proper orthogonal decomposition. *Comput. Phys. Commun.*, 237:98–109, Apr. 2019.
- [122] O. T. Schmidt, A. Towne, G. Rigas, T. Colonius, and G. A. Brès. Spectral analysis of jet turbulence. *J. Fluid Mech.*, 855:953–982, Sept. 2018.
- [123] A. S. Sharma and B. J. McKeon. On coherent structure in wall turbulence. *J. Fluid Mech.*, 728:196–238, July 2013.
- [124] L. Silverman. Realization of linear dynamical systems. *IEEE Trans. Automat. Contr.*, 16(6):554–567, 1971.
- [125] D. Sipp, M. Fosas de Pando, and P. J. Schmid. Nonlinear model reduction: A comparison between pod-galerkin and pod-deim methods. *Comput. Fluids*, 208:104628, Aug. 2020.
- [126] D. SIPP and A. LEBEDEV. Global stability of base and mean flows: a general approach and its applications to cylinder and open cavity flows. *J. Fluid Mech.*, 593:333–358, Nov. 2007.
- [127] L. Sirovich. Turbulence and the dynamics of coherent structures. ii. symmetries and transformations. *Q. Appl. Math.*, 45(3):573–582, 1987.
- [128] S. Symon, S. J. Illingworth, and I. Marusic. Energy transfer in turbulent channel flows and implications for resolvent modelling. *J. Fluid Mech.*, 911:A3, 2021.
- [129] F. Takens. Detecting strange attractors in turbulence. In D. Rand and L.-S. Young, editors, *Dynamical Systems and Turbulence, Warwick 1980*, pages 366–381, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg.
- [130] 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, Jan. 2021.
- [131] A. Towne, O. T. Schmidt, and T. Colonius. Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis. *J. Fluid Mech.*, 847:821–867, May 2018.
- [132] L. N. Trefethen. *Spectra and Pseudospectra: The Behaviour of Non-Normal Matrices and Operators*, page 217–250. Springer Berlin Heidelberg, 1999.
- [133] L. N. Trefethen, A. E. Trefethen, S. C. Reddy, and T. A. Driscoll. Hydrodynamic stability without eigenvalues. *Science*, 261(5121):578–584, 1993.
- [134] R. Vautard and M. Ghil. Singular spectrum analysis in nonlinear dynamics, with applications to paleoclimatic time series. *Phys. D*, 35(3):395–424, 1989.
- [135] H. F. Walker and P. Ni. Anderson acceleration for fixed-point iterations. *SIAM J. Numer. Anal.*, 49(4):1715–1735, Jan. 2011.
- [136] 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):70–73, 1967.
- [137] K. Willcox and J. Peraire. Balanced model reduction via the proper orthogonal decomposition. *AIAA J.*, 40(11):2323–2330, Nov. 2002.
- [138] J. Xu and K. Duraisamy. Multi-level convolutional autoencoder networks for para-

- metric prediction of spatio-temporal dynamics. *Comput. Methods Appl. Mech. Eng.*, 372:113379, Dec. 2020.
- [139] C.-A. Yeh and K. Taira. Resolvent-analysis-based design of airfoil separation control. *J. Fluid Mech.*, 867:572–610, Mar. 2019.
- [140] J. Zhang. Modern monte carlo methods for efficient uncertainty quantification and propagation: A survey. *WIREs Comput. Stat.*, 13(5), Dec. 2020.